

RM03,02

DISKLESS DIAGNOSTIC
CZRMJB0

AH-B019B-MC

COPYRIGHT © 1977

FICHE 1 OF 2

JAN 1978

digital

MADE IN USA

This image shows a microfiche card with a grid of 14 columns and 12 rows of frames. Each frame contains a small, illegible image of a document page, likely a diagnostic manual or technical document. The frames are arranged in a regular grid pattern across the card.

RM03,02

DISKLESS DIAGNOSTIC
CZRMJB0

AH-B019B-MC
COPYRIGHT © 1977
FICHE 2 OF 2

JAN 1978
digital
MADE IN USA

This microfiche card contains a grid of 14 columns and 20 rows of frames. Each frame contains a small portion of a larger document, likely a diagnostic manual or report. The text within the frames is extremely small and difficult to read, but it appears to be organized into columns and rows, possibly representing a table or a list of data points. The overall layout is a dense grid of information.

CONTENTS

- 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

1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100

001

CZRMJBO RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 3

SEQ 0003

PAGE 2

109
110
111
112
113
114
115
116
117
118

- 6. XXDP COMPATIBILITY
- 7. OPERATING SYSTEM COMPATIBILITY
- 6. TEST DESCRIPTION

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
172
173
174

1.0 INTRODUCTION

1.1 ABSTRACT

THE RMO3 DISKLESS DIAGNOSTIC IS A STAND ALONE PROGRAM WHICH USES FUNCTIONAL AND DIAGNOSTIC MEANS TO VERIFY THE OPERABILITY OF THE RMO3 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 RH70 MASSBUS CONTROLLER;

TO DETECT ERRORS AND FAULTS IN THE RMO3 MASSBUS ADAPTER;

TO RESOLVE HARDWARE FAILURES IN THE RH70 AND RMO3 TO A FIELD REPLACEABLE MODULE OR MODULES.

1.2 UNIT UNDER TEST

THE UNIT UNDER TEST (UUT) IS THE RMO3 DISK SUBSYSTEM, EXCLUDING THE STORAGE MODULE DISK DRIVE AND THE RH70 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 RMO3 DISKLESS DIAGNOSTIC:

PDP-11 PROCESSOR
24K MEMORY
KW11-L OR KW11-P CLOCK
PROGRAM LOADING DEVICE
TERMINAL
RH CONTROLLER

UNIT UNDER TEST,
WHERE THE UNIT UNDER TEST CONSISTS OF ONE TO EIGHT RMO3 ADAPTERS.

NOTE

A STORAGE MODULE DISK DRIVE IS NOT REQUIRED FOR EXECUTING THE RMO3 DISKLESS DIAGNOSTIC AS LONG AS THE "DEVICE ADDRESS" AND "PLUG ENABLE" LINES ARE PROVIDED BY SOME OTHER MEANS SUCH AS JUMERS.

F01

SEG 0005

CZRMJBO RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 5

THE DEVICE ADDRESS LINES MAY FLOAT TO
ADDRESS 7 IF THERE IS NO OTHER DEVICE

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
229

WITH ADDRESS 7 ON THE SUBSYSTEM. THE
"PLUG ENABLE" LINE MUST BE PROVIDED TO
ALLOW MASSBUS DIALOGUE BY THE RMO3.

2.2 MEDIA REQUIREMENTS

NONE

2.3 PREREQUISITE DIAGNOSTIC PROGRAMS

NONE

3.0 OPERATING PROCEDURE

3.1 LOADING

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
A PARTICULAR TEST WHICH THE PROGRAM WILL LOOP ON.

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

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 OCCURS EACH TIME THE PROGRAM IS EXECUTED FOR ALL ADAPTERS IN THE TEST QUEUE. 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.

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 RM03 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 NUMBER OF THE UNIT BEING TESTED, 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

CZRMJBO RM03/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 8

IO1

SEG 0008

296

RESULTS.

30087
30088
30089
30090
30091
30092
30093
30094
30095
30096
30097
30098
30099
30100
30101
30102
30103
30104
30105
30106
30107
30108
30109
30110
30111
30112
30113
30114
30115
30116
30117
30118
30119
30120
30121
30122
30123
30124
30125
30126
30127
30128
30129
30130
30131
30132
30133
30134
30135
30136
30137
30138
30139
30140
30141

5.0 ENVIRONMENTAL SUPPORT

5.1 PROCESSOR COMPATIBILITY

THE RMO3 DISKLESS DIAGNOSTIC IS EXECUTABLE ON ANY PDP-11 PROCESSOR, PROVIDING PREVIOUSLY MENTIONED HARDWARE REQUIREMENTS ARE MET.

5.2 DUAL PORT CONFIGURATIONS

THE RMO3 DISKLESS DIAGNOSTIC IS NOT EXECUTABLE ON RMO3 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 RMO3 DISKLESS DIAGNOSTIC.

5.4 MEMORY MANAGEMENT HARDWARE

MEMORY MANAGEMENT HARDWARE WILL NOT BE USED DURING THE RMO3 DISKLESS DIAGNOSTIC.

5.5 ACT11, APT11 COMPATIBILITY

THE RMO3 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 RMO3 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.

342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397

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 "RM03 DISKLES DIAGNOSTIC" CAN BE EXECUTED USING AN RM70 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

PURPOSE:

TO VERIFY THAT THE RM03 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.

398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452

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

CTOD TEST

PURPOSE:

TO VERIFY THAT DATA CAN BE TRANSFERRED TO AND FROM THE RMO3 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

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 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.

453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508

C. 1JBO RM3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 13

NO1

SEQ 0013

509

510
511
512
513
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

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:

- 1. IF MODULE
- 2. ASYNCHRONOUS MASSBUS MODULE

565
566
567
568
569
570
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

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 PORT OR DUAL PORT RMD3 DRIVE.

PROBABLE FAULT:

- 1. IF MODULE

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

HOLDING REGISTER TRANSFER TEST

PURPOSE:

TO VERIFY THAT THE HOLDING REGISTER IS NOT STUCK AT ONE,
STUCK AT ZERO, AND THAT THERE IS NO BIT INTERFERENCE.

PROCEDURE:

THE PROGRAM TRANSFERS ONES, THEN ZEROS TO THE HOLDING
REGISTER AND VERIFIES THAT NONE OF THE BITS ARE STUCK AT ONE.

THE PROGRAM TRANSFERS ZEROS, THEN ONES TO THE HOLDING
REGISTER AND VERIFIES THAT NONE OF THE BITS ARE STUCK AT ZERO.

FINALLY, THE TEST TRANSFERS A SHIFTING ONE BIT PATTERN AND VERIFIES THAT EACH BIT IS INDEPENDENT.

PROBABLE FAULT:

- 1. IF MODULE

CONTROL STATUS #1 TRANSFER TEST

PURPOSE:

TO VERIFY THAT BITS 01 THROUGH 05 OF CONTROL STATUS REGISTER 1 ARE NOT STUCK AT ONE OR ZERO, AND THAT THERE IS NOT BIT INTERFERENCE.

PROCEDURE:

THIS TEST WRITES ONES IN CONTROL STATUS REGISTER 1, RMCS1, THEN WRITES ZEROS AND VERIFIES THAT THE BITS ARE NOT STUCK AT ONE. THE GO BIT IS NOT TESTED IN THIS TEST.

NEXT, THE TEST CLEARS THE CONTROL STATUS REGISTER, RMCS1, WRITES ONES IN BITS 01 THROUGH 05 AND VERIFIES THAT THE BITS ARE NOT STUCK AT ZERO. THE GO BIT IS NOT TESTED.

THE TEST TRANSFERS A SHIFTING ONE BIT DATA PATTERN TO AND FROM RMCS1 AND CHECKS FOR ADJACENT BIT INTERFERENCE.

PROBABLE FAULT:

- 1. IF MODULE

ERROR REGISTER #1 TRANSFER TEST

PURPOSE:

TO VERIFY THAT ERROR REGISTER 1 IS NOT STUCK AT ONE OR ZERO, AND THAT THERE IS NOT BIT INTERFERENCE.

PROCEDURE:

THIS TEST WRITES ONES IN ERROR REGISTER 1, RMER1, THEN WRITES ZEROS AND VERIFIES THAT THE REGISTER IS NOT STUCK AT ONE.

63
64
65
66
67
68
69
70
71
72
73
74
75
76

E02

CZRMJBO RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 17

SEQ 0017

677
678

"UNSAFE" IS NOT TESTED DURING THIS TEST. IN ORDER TO LIMIT THE
PROBABLE FAULTS TO ONE OR TWO MODULES, THE TEST IS EXECUTED IN 3

679
680
681
682
683
684
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

PARTS WITH EACH PART TESTING THOSE BITS WHOSE PRESET FUNCTIONS ARE DERIVED FROM THE SAME MODULE.

THE TEST WRITES ZEROS IN ERROR REGISTER 1, RMER1, THEN WRITES ONES AND VERIFIES THAT THE REGISTER IS NOT STUCK AT ZERO.

FINALLY, THE TEST WRITES A SHIFTING ONE BIT PATTERN IN RMER1 AND CHECKS FOR ADJACENT BIT INTERFERENCE.

PROBABLE FAULT:

1. IF MODULE
2. CS MODULE
3. DS MODULE

ERROR REGISTER #2 TRANSFER TEST

PURPOSE:

TO VERIFY THAT ERROR REGISTER 2, RMER2, IS NOT STUCK AT ONE, STUCK AT ZERO, AND THAT THERE IS NOT BIT INTERFERENCE.

PROCEDURE:

THE TEST WRITES ONES THEN WRITES ZEROS IN RMER2 AND VERIFIES THAT NONE OF THE BITS ARE STUCK AT ONE. "SKI" AND "DVC" ARE NOT TESTED. IN ORDER TO LIMIT THE NUMBER OF PROBABLE FAULTS TO ONE OR TWO MODULES, THE TEST IS EXECUTED IN 3 PARTS WITH EACH PART TESTING THOSE BITS WHOSE PRESET FUNCTIONS ARE DERIVED FROM THE SAME MODULE.

THEN THE TEST WRITES ZEROS IN ERROR REGISTER 2 AND WRITES ONES VERIFYING THAT THE REGISTER IS NOT STUCK AT ZERO.

FINALLY, THE TEST WRITES A SHIFTING ONE BIT PATTERN IN THE REGISTER AND VERIFIES THAT ALL BIT POSITIONS ARE INDEPENDENT.

PROBABLE FAULT:

1. IF MODULE
2. CS MODULE
3. DS MODULE

734
735
736
737
738
739
740
741
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

CLEAR OFFSET STUCK ACTIVE TEST

PURPOSE:

TO VERIFY THAT THE SIGNAL WHICH CLEARS OFFSET MODE IS NOT STUCK IN ACTIVE STATE.

PROCEDURE:

THE TEST WRITES A ONE IN THE OFFSET DIRECTION BIT WHICH IS CLEARED BY THE SIGNAL AND VERIFIES THAT A ONE CAN BE READ BACK.

PROBABLE FAULT:

1. IF MODULE
2. OS MODULE

OFFSET REGISTER TRANSFER TEST

PURPOSE:

TO VERIFY THAT THE OFFSET REGISTER IS NOT STUCK AT ONE, STUCK AT ZERO, AND THAT THERE IS NO ADJACENT BIT INTERFERENCE.

PROCEDURE:

THE OFFSET REGISTER, RMOF, IS WRITTEN WITH ONES, THEN WRITTEN WITH ZEROS AND READ TO VERIFY THAT NONE OF THE BITS ARE STUCK AT ONE.

THEN THE OFFSET REGISTER IS WRITTEN WITH ZEROS AND WRITTEN WITH ONES TO VERIFY THAT THE REGISTER IS NOT STUCK AT ZERO.

FINALLY, THE OFFSET REGISTER IS TESTED WITH A SHIFTING ONE BIT PATTERN.

PROBABLE FAULT:

1. IF MODULE

788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839

SERIAL NUMBER TEST

PURPOSE:

TO VERIFY THAT THE SERIAL NUMBER CAN BE READ.

PROCEDURE:

THE TEST READS THE SERIAL NUMBER REGISTER SEVERAL TIMES AND VERIFIES THAT THE NUMBER IS THE SAME EACH TIME.

PROBABLE FAULT:

1. CS MODULE

CONTROL BUS PARITY DETECTION TEST

PURPOSE:

TO TEST THE RMO3'S PARITY CHECKING LOGIC FOR THE MASSBUS ASYNCHRONOUS CONTROL BUS.

PROCEDURE:

THIS TEST WRITES A SHIFTING ONE BIT DATA PATTERN IN THE DISK ADDRESS REGISTER USING "PAT" TO CONTROL THE STATE OF THE PARITY BIT. "PAR" STATUS, BIT 03 OF RMR1, IS CHECKED AFTER EACH PATTERN IS TRANSFERRED. NOTE THE FOLLOWING TABLE SHOWS A SET OF TEST PATTERNS THAT COULD BE USED INSTEAD OF A SHIFTING ONE BIT PATTERN.

DATA PATTERN	PAT	PAR
000000	1	1
075266	0	0
163753	0	0
116535	1	1

PROBABLE FAULT:

1. IF MODULE
2. ASYNCHRONOUS MASSBUS MODULE

840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895

CONTROL BUS PARITY GENERATION TEST

PURPOSE:

TO TEST THE RMO3'S PARITY GENERATING LOGIC FOR THE MASSBUS ASYNCHRONOUS CONTROL BUS.

PROCEDURE:

THE TEST TRANSFERS A SHIFTING ONE BIT DATA PATTERN TO THE DISK ADDRESS REGISTER. AFTER EACH PATTERN IS READ BACK, "MASSBUS CONTROL BUS PARITY ERROR" IS TESTED AND SHOULD BE ZERO. NOTE THE FOLLOWING SET OF TEST PATTERNS COULD BE USED INSTEAD OF THE SHIFTING ONE BIT PATTERN.

DATA PATTERN	MCPE
000000	0
056747	0
135672	0
153135	0

PROBABLE FAULT:

1. IF MODULE
2. ASYNCHRONOUS MASSBUS MODULE

RMDA, RMDC FAULT TEST

PURPOSE:

TO VERIFY THAT THERE ARE NOT FAULTS WHICH INHIBIT THE PROGRAM FROM WRITING RMDC AND RMDA. SPECIFICALLY, THESE FAULTS INCLUDE:

"GO H" STUCK HIGH, WHICH WOULD INHIBIT THE REGISTER LOAD FUNCTION.

"RIP L" STUCK LOW, WHICH WOULD CONSTANTLY CLEAR THE REGISTER

. "EBL" STUCK, WHICH WOULD INHIBIT THE CLOCK FUNCTION.

PROCEDURE:

THE TEST WRITES AND READS BOTH RMDC, AND RMDA. WITH ZEROS, THEN ONES. THE TEST PASSES IF EITHER REGISTER CAN BE WRITTEN

J02

CZRMJBO RMO3'2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 22

SEQ 0022

896

WITH ONES.

897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952

PROBABLE FAULT:

- 1. DS MODULE
- 2. IF MODULE
- 3. CS MODULE

DISK ADDRESS TRANSFER TEST

PURPOSE:

TO VERIFY THAT THE DISK ADDRESS REGISTER IS NOT STUCK AT ONE OR ZERO, AND THAT THERE IS NOT BIT INTERFERENCE.

PROCEDURE:

THIS TEST PRESETS THE DISK ADDRESS TO A NONZERO VALUE, THEN USES A MOVE TO CLEAR THE REGISTER. THE TEST THEN READS RMDA AND VERIFIES THAT NONE OF THE BITS ARE STUCK AT ONE.

THEN THE TEST PRECLEARS THE MASSBUS ADAPTER DISK ADDRESS REGISTER (RMDA), LOADS IT TO ALL ONES, AND VERIFIES THAT NONE OF THE BITS ARE STUCK AT ZERO.

A SHIFTING ONE BIT PATTERN IS TRANSFERRED TO AND FROM THE DISK ADDRESS REGISTER, RMDA, AND THE TEST VERIFIES THAT EACH BIT IS INDEPENDENT.

PROBABLE FAULT:

- 1. DS MODULE
- 2. IF MODULE

DESIRED CYLINDER TRANSFER TEST

PURPOSE:

TO VERIFY THAT THE DESIRED CYLINDER ADDRESS REGISTER, RMDC, IS NOT STUCK AT ONE OR ZERO, AND THAT THERE IS NOT BIT INTERFERENCE.

CZRMJBO RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

L02

MACY11 30(1046) 23-NOV-77 12:14 PAGE 24

SEQ 0024

953

PROCEDURE:

954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002

THIS TEST WRITES ONES IN THE DESIRED CYLINDER REGISTER RMDC, THEN WRITES ZEROS AND VERIFIES THAT THE REGISTER IS NOT STUCK AT ONE.

THEN THE TEST WRITES ZEROS IN THE DESIRED CYLINDER REGISTER, RMDC, WRITES ONES AND VERIFIES THAT THE REGISTER IS NOT STUCK AT ZERO.

FINALLY, A SHIFTING 1 BIT PATTERN IS TRANSFERRED TO AND FROM RMDC AND THE PROGRAM CHECKS FOR BIT INTERFERENCE.

PROBABLE FAULT:

- 1. DS MODULE
- 2. IF MODULE

ILLEGAL REGISTER TEST

PURPOSE:

TO TEST ILLEGAL REGISTER ERROR DETECTION IN THE RM03.

PROCEDURE:

THIS TEST READS ALL LEGAL REGISTERS AND VERIFIES THAT "ILR" BIT 2 OF RMR1 DOES NOT SET. THEN, TO THE EXTENT ALLOWED BY THE MASSBUS CONTROLLER, IT READS ILLEGAL REGISTERS AND VERIFIES THAT "ILR" IS SET.

PROBABLE FAULT:

- 1. IF MODULE
- 2. ASSYNCHRONOUS MASSBUS MODULE

1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
J58

RESET GO BY INIT TEST

PURPOSE:

TO VERIFY THAT GO CAN BE RESET BY INITIALIZE.

PROCEDURE:

THE TEST SETS GO THEN CLEARS GO USING MASSBUS INITIALIZE, I.E., CONTROLLER CLEAR.

PROBABLE FAULT:

1. CS MODULE
2. IF MODULE

DIAGNOSTIC MODE TEST

PURPOSE:

TO VERIFY THAT "DIAGNOSTIC MODE", BIT 0 OF RMMR1, IS NOT STUCK AT ONE OR ZERO.

PROCEDURE:

THE RMO3 IS INITIALIZED AND "DMD" IS CHECKED FOR ZERO. "DMD" IS WRITTEN WITH ONE AND READ TO VERIFY THAT IT IS NOT STUCK AT ZERO, THEN WRITTEN WITH ZERO AND READ TO VERIFY THAT IT IS NOT STUCK AT ONE.

PROBABLE FAULT:

1. CS MODULE
2. IF MODULE

MOL TEST

PURPOSE:

TO VERIFY THAT "MEDIUM ON LINE" STATUS CAN BE SET AND RESET USING MAINTENANCE UNIT READY.

B03

CZRMJBO RMD3 2 DSKLS DIAG
CZRMJB.F11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 27

SEQ 0027

1059
1060

PROCEDURE:

1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116

AFTER INITIALIZING THE SUBSYSTEM, THE TEST SETS "DIAGNOSTIC MODE:" AND READS THE DRIVE STATUS REGISTER, RMD5, EXPECTING MOL BIT 12 TO BE ZERO. "MAINTENANCE UNIT READY", BIT 9 OF RMMR1, IS SET AND MOL SHOULD BE ONE. THE TEST THEN WRITES A ZERO IN MUR AND READS RMD5, VERIFYING THAT "MEDIUM ON LINE" IS ZERO.

PROBABLE FAULT:

1. CS MODULE
2. IF MODULE

WRITE LOCK TEST

PURPOSE:

TO VERIFY THAT "WRITE LOCK" STATUS, WRL, CAN BE SET AND RESET USING "MAINTENANCE WRITE PROTECT", MWP.

PROCEDURE:

WITH DIAGNOSTIC MODE SET, THE PROGRAM SETS MWP, BIT 03 OF RMMR1 AND READS RMD5 TO VERIFY THAT WRL, BIT 11 IS SET. THEN MWP IS RESET AND WRL SHOULD BE ZERO.

PROBABLE FAULT:

1. CS MODULE
2. IF MODULE

DRIVE FAULT TEST

PURPOSE:

TO VERIFY THAT "DEVICE CHECK", DVC, AND "UNSAFE", UNS, CAN BE SET AND RESET USING "MAINTENANCE DRIVE FAULT", MDF.

PROCEDURE:

WITH DIAGNOSTIC MODE SET, THE PROGRAM SETS MDF, BIT 06 OF RMMR1 AND READS RMR3 TO VERIFY THAT DVC, BIT 07 IS SET RMR1 IS ALSO READ AND UNS, BIT 14 SHOULD ALSO BE SET. THEN MDF IS RESET

D03

CZRMJBO RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 29

SEQ 0029

1117

AND DVC AND UNS SHOULD BE RESET.

1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173

PROBABLE FAULT:

- 1. CS MODULE
- 2. IF MODULE

SEEK ERROR TEST

PURPOSE:

TO VERIFY THAT "SEEK ERROR", SKI, CAN BE SET AND RESET USING "MAINTENANCE SEEK ERROR", MSER.

PROCEDURE:

WITH DIAGNOSTIC MODE SET, THE TEST SETS MSER, BIT 07 OF RMMR1 AND READS RMER3 TO VERIFY THAT SKI, BIT 14 IS SET. MSER IS RESET AND SKI SHOULD RESET.

PROBABLE FAULT:

- 1. CS MODULE
- 2. IF MODULE

PIP TEST

PURPOSE:

TO VERIFY THAT "POSITIONING IN PROGRESS", PIP, CAN BE SET AND RESET USING "MAINTENANCE ON CYLINDER", MOC.

PROCEDURE:

DIAGNOSTIC MODE IS SET THEN MOC, BIT 08 OF RMMR1 IS SET AND PIP, BIT 13 OF RMD5, SHOULD BE ZERO. MOC IS THEN RESET AND PIP SHOULD BE ONE.

PROBABLE FAULT:

- 1. CS MODULE
- 2. IF MODULE

CZRMJBO RMO3-2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 3:

F03

SEQ 0031

1174

1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230

EBL TEST

PURPOSE:

TO VERIFY THAT END OF BLOCK STATUS "EBL" CAN BE SET AND RESET USING DIAGNOSTIC END OF BLOCK "DEBL".

PROCEDURE:

THE PROGRAM SETS DIAGNOSTIC MODE AND VERIFIES THAT EBL IS RESET. THEN IT SETS DEBL AND VERIFIES THAT EBL IS SET. FINALLY THE TEST TRANSFERS A SHIFTING ONE BIT TO RMMR1, AND CHECKS FOR DEBL BEING SET BY AN ADJACENT BIT.

PROBABLE FAULT:

- 1. CS MODULE

LAST SECTOR, LAST TRACK TEST

PURPOSE:

TO VERIFY THE DESIRED TRACK/SECTOR PLA ON THE DS MODULE USING RMMR1, BITS 01 AND 02.

PROCEDURE:

THE TEST WRITES ALL POSSIBLE PATTERNS IN THE DISK ADDRESS REGISTER, RMDA, AND VERIFIES "LS" AND "LS T" STATUS FOR EACH PATTERN. THE PROCEDURE IS DONE ONCE FOR 16 BIT FORMAT AND ONCE FOR 18 BIT FORMAT.

PROBABLE FAULT:

- 1. DS MODULE
- 2. CS MODULE

RMDA COUNT TEST

PURPOSE

TO VERIFY THAT THE DISK ADDRESS REGISTER (RMDA) INCREMENTS

CZRMJBO RMD3-2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

1231

MACY11 30(1046) 23-NOV-77 12:14 PAGE 33

H03

SEQ 0033

PROPERLY.

1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287

PROCEDURE:

THE TEST INCREMENTS RMDA USING DIAGNOSTIC END OF BLOCK "DEBL" AND VERIFIES THE RESULT IN BOTH 16 AND 18 BIT FORMAT.

PROBABLE FAULT:

- 1. DS MODULE

RMDC COUNT TEST

PURPOSE:

TO VERIFY THAT THE DESIRED CYLINDER REGISTER, RMDC, INCREMENTS PROPERLY.

PROCEDURE:

THE PROGRAM INCREMENTS RMDC USING DIAGNOSTIC END OF BLOCK, "DEBL", VERIFYING THAT RMDC INCREMENTS THROUGH A COMPLETE CYCLE.

PROBABLE FAULT:

- 1. DS MODULE

LBT TEST

PURPOSE:

TO INSURE THAT LAST BLOCK TAKEN STATUS, "LBT" CLEARS WHEN RMDA IS WRITTEN, AND SETS WHEN THE LAST SECTOR IS TRANSFERRED.

PROCEDURE:

THE TEST USES DIAGNOSTIC EBL TO SET LBT, AND TRANSFERS TO RMDA TO RESET LBT.

PROBABLE FAULT:

- 1. DS MODULE
- 2. IF MODLLE

1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343

COMPOSITE ERROR TEST

PURPOSE:

TO TEST "COMPOSITE ERROR", BIT 14 OF RMD5.

PROCEDURE:

THE TEST USES INITIALIZE AND DIAGNOSTIC MODE TO FORCE ALL ERRORS TO ZERO THEN VERIFIES THAT "ERR" IS ZERO. EACH ERROR IS INDIVIDUALLY SET AND "ERR" SHOULD BE ONE FOR EVERY ERROR TESTED. ADDRESSES #2 AND #17 OF THE COMPOSITE ERROR PLA ARE NOT TESTED. "ABORT" AND "EXCEPTION" OUTPUTS OF THE PLA ARE NOT TESTED. THE TEST FAILS IF ERR IS NOT ZERO WITH ALL SET ARGUMENTS ZERO OR IF ERR IS NOT ONE WITH ANY SET ARGUMENT ONE.

PROBABLE FAULT:

- 1. IF MODULE

WRITE GO TEST

PURPOSE:

TO VERIFY THAT GO CAN BE SET.

PROCEDURE:

THE TEST ENABLES THE DEBUG CLOCK, THEN TRANSFERS A NOP FUNCTION CODE AND GO BIT TO RMCSI, VERIFYING THAT GO SETS. ALL FUNCTION CODES ARE TESTED.

PROBABLE FAULT:

- 1. IF MODULE
- 2. CS MODULE

BRANCH MULTIPLEXOR TEST

PURPOSE:

TO VERIFY THAT THE OUTPUT OF THE COMMAND SEQUENCER BRANCH

K03

CZRMJ80 RM03/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 36

SEQ 0036

1344

MULTIPLEXOR DOES NOT HAVE A FAULT.

PROCEDURE:

WITH DEBUG CLOCK ENABLED, THE TEST USES VARIOUS FUNCTION CODES AND REGISTER CONDITIONS TO ADDRESS THE TEST BIT MULTIPLEXOR SUCH THAT THE TEST BIT, BIT12 OF RMMR2, CAN BE CHECKED FOR A STUCK FAULT.

PROBABLE FAULT:

1. CS MODULE

SET/RESET GO TEST

PURPOSE:

TO VERIFY THAT GO CAN BE SET AND RESET.

PROCEDURE:

THE SUBSYSTEM IS INITIALIZED AND PUT IN DIAGNOSTIC MODE WITH "DEBUG CLOCK ENABLE" BIT 14 OF RMMR1 SET. CERTAIN FUNCTION CODES ARE WRITTEN IN RMCS1 AND THE PROGRAM READS RMCS1 TO VERIFY THAT GO IS SET. RMDS IS ALSO READ TO VERIFY THAT "DRY" IS RESET. THEN THE PROGRAM STEPS THE DEBUG CLOCK USING BIT 15 OF RMMR1 AND VERIFIES THAT "GO" RESETS AND "DRY" SETS. USING A FUNCTION CODE THAT RESETS GO AT A DIFFERENT PROM ADDRESS. THE TEST FAILS IF GO DOES NOT SET OR CANNOT BE RESET BY THE COMMAND SEQUENCER. THE TEST ALSO FAILS IF "DRIVE READY" IS NOT THE COMPLIMENT OF GO.

PROBABLE FAULT:

1. CS MODULE
2. IF MODULE

END 1 RESET GO TEST

PURPOSE:

TO VERIFY THAT THE COMMAND SEQUENCER CAN RESET GO AT THE END1 LOCATION.

PROCEDURE:

1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400

THE TEST EXECUTES RELEASE, SEARCH AND ILLEGAL FUNCTION CODE
32 IN DIAGNOSTIC MODE AND VERIFIES THAT GO RESETS ON THE
SPECIFIED CLOCK CYCLE.

PROBABLE FAULT:

- 1. CS MODULE

SET PULSE TEST

PURPOSE:

TO VERIFY THAT THE COMMAND SEQUENCER CAN GENERATE SET PULSE.

PROCEDURE:

WHICH DEBUG CLOCK ENABLED, THE TEST STEPS THE COMMAND
SEQUENCER THROUGH PARTS OF VARIOUS FUNCTION CODES AND CHECKS
CONTINUE BIT 06 OF RMMR1 TO DETERMINE IF SET PULSE IS BEING
GENERATED.

PROBABLE FAULT:

- 1. CS MODULE

SET/RESET IVC TEST

PURPOSE:

TO TEST "INVALID COMMAND" ST S FOR EACH FUNCTION CODE.

PROCEDURE:

THE PROGRAM RESETS VOLUME VALID USING "MAINTENANCE UNIT
REQDY" BIT09 OF RMMR1, THEN LOADS THE FUNCTION CODE AND GO IN
FACS1. EACH FUNCTION CODE IS TESTED AND "IVC" BIT 12 OF RMR2
IS CHECKED.

PROBABLE FAULT:

- 1. CS MODULE
- 2. IF MODULE

1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456

CZRMJBO RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 39

NO3

SEQ 0039

1457

1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512

SET LSC TEST

PURPOSE:

TO VERIFY THAT "LOSS OF SYSTEM CLOCK" CAN SET AND RESET.

PROCEDURE:

THE TEST ENABLES THE DEBUG CLOCK AND SETS THE GO BIT. AFTER WAITING ENOUGH TIME FOR THE ONE SHOT TO SET, THE TEST DISABLES THE DEBUG CLOCK AND VERIFIES THAT LSC SETS.

PROBABLE FAULT:

- 1. CS MODULE
- 2. IF MODULE

DECODE TEST

PURPOSE:

TO VERIFY THAT THE "DECODE" FLOP ON THE IF MODULE SETS WITH THE LEADING EDGE OF "SET PULSE" EXCEPT WHEN "COMPOSITE ERROR" IS ACTIVE.

PROCEDURE:

THE TEST USES "VOLUME VALID" AND "OCCUPIED" TO DETERMINE IF THE DECODE FLOP IS SET OR RESET. INITIALLY, VV AND OCCUPIED ARE RESET AND THE TEST EXECUTES THOSE COMMANDS WHICH SET VV OR OCC AND VERIFIES THAT ONE OR BOTH BITS SET. THE SAME COMMANDS ARE EXECUTED AGAIN WITH COMPOSITE ERROR SET, AND THE TEST VERIFIES THAT NEITHER BIT SETS.

PROBABLE FAULT:

- 1. IF MODULE

SET/RESET VOLUME VALID TEST

PURPOSE:

1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568

TO VERIFY THAT "VOLUME VALID" RESETS WITH THE LEADING EDGE OF UNIT READY, AND SETS WITH PACK ACKNOWLEDGE AND READ IN PRESET COMMANDS.

PROCEDURE:

USING "MAINTENANCE UNIT READY", BIT 9 OF RMMR1, THIS TEST FORCES A ZERO TO ONE TRANSITION OF UNIT READY AND VERIFIES THAT VOLUME VALID, BIT 6 OF RMDS IS ZERO. THEN THE TEST EXECUTES A PACK ACKNOWLEDGE COMMAND, VERIFYING THAT VV SETS. THE PROCEDURE IS REPEATED WITH A READ IN PRESET COMMAND.

PROBABLE FAULT:

1. IF MODULE

ILLEGAL FUNCTION TEST

PURPOSE:

TO TEST ILLEGAL FUNCTION ERROR IN THE RM03.

PROCEDURE:

WITH DIAGNOSTIC CLOCK ENABLED TO INHIBIT THE COMMAND SEQUENCER, THIS TEST VERIFIES THAT "ILF", BIT 0 OF RMER1, IS OFF FOR LEGAL FUNCTION CODES AND ON FOR ILLEGAL FUNCTIONCODES. THE STATUS OF THE "GO" BIT IS IGNORED.

PROBABLE FAULT:

1. IF MODULE

OCCUPIED TEST

PURPOSE:

TO VERIFY THAT "OCCUPIED" IS SET DURING DATA TRANSFERS AND IS RESET FOR ALL OTHER COMMANDS.

PROCEDURE:

FOR EACH DATA TRANSFER COMMAND, "OCC", BIT 15 OF RMMP1

004

CZRMJBO RMD3/2 DSKLS DIAG
CZRMJB.F11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 42

SEQ 0042

1569
1570

SHOULD BE ONE DEBUG CLOCK IS ENABLED TO PREVENT GO FROM
RESETTING BEFORE STATUS IS SAMPLED.

1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625

PROBABLE FAULT:

- 1. IF MODULE
- 2. CS MODULE

READ IN PRESET TEST

PURPOSE:

TO VERIFY THAT "READ IN PRESET" COMMAND IS DECODED, AND IN PARTICULAR, TO VERIFY THAT "IFS READ IN CMD L" IS NOT STUCK AT ONE.

PROCEDURE:

EACH VISIBLE STATUS OR REGISTER BIT WHICH IS CLEARED BY "READ IN PRESET" IS SET. THEN THE RIP COMMAND IS EXECUTED AND THE TEST VERIFIES THAT ONE OR MORE BITS ARE CLEARED. THE FOLLOWING ARE USED DURING THE TEST.

. BITS 10-12 OF RMOF ARE SET BY A MOVE INSTRUCTION AND THE TEST PASSES IF THESE BITS ARE ZERO AFTER THE RIP COMMAND.

. THE DESIRED CYLINDER REGISTER, RMDC, IS SET WITH A MOVE INSTRUCTION AND THE TEST PASSES IF BITS 00-03, OR BITS 04-07, OR BITS 08-09 ARE ZERO AFTER THE RIP COMMAND.

. THE DISK ADDRESS REGISTER, RMDA, IS SET WITH A MOVE INSTRUCTION AND THE TEST PASSES IF BITS 00-03, OR BITS 04-07, OR BITS 08-11, OR BITS 12-15 ARE ZERO AFTER THE RIP COMMAND.

THE TEST FAILS IF NONE OF THE PRESET TERMS ARE ZERO AFTER THE RIP COMMAND.

PROBABLE FAULT:

- 1. IF MODULE

RIP/RMOF TEST

PURPOSE:

1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681

TO VERIFY THAT "READ IN PRESET" RESETS FMT16, ECI AND HCI,
BITS 10,11 AND 12 OF RMOF.

PROCEDURE:

FMT16, ECI AND HCI ARE SET, THEN A RIP COMMAND IS EXECUTED
AND EACH BIT SHOULD BE ZERO.

PROBABLE FAULT:

- 1. IF MODULE

RMDA/RMDC/RIP TEST

PURPOSE:

TO VERIFY THAT "READ IN PRESET" RESETS THE DESIRED CYLINDER
ADDRESS, RMDC, AND THE DISK ADDRESS, RMDA.

PROCEDURE:

RMDA AND RMDC ARE PRESET THEN TESTED FOR ZERO AFTER THE RIP
COMMAND.

PROBABLE FAULT:

- 1. DS MODULE

OFFSET COMMAND TEST

PURPOSE:

TO VERIFY THAT "OFFSET MODE" SETS WITH OFFSET COMMAND.

PROCEDURE:

THE TEST EXECUTES OFFSET COMMAND AND VERIFIES THAT "OM", BIT
00 OF RMD5 IS ONE.

PROBABLE FAULT:

- 1. IF MODULE

1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737

RETURN TO CENTER TEST

PURPOSE:

TO VERIFY THAT "RETURN TO CENTER" RESETS OFFSET MODE.

PROCEDURE:

OFFSET MODE, BIT 00 OF RMD5, IS SET WITH OFFSET COMMAND, THEN THE TEST EXECUTES A RETURN TO CENTER COMMAND AND VERIFIES THAT OFFSET MODE RESETS. OFFSET DIRECTION IS ALSO SET AND CHECKED FOR ZERO AFTER THE COMMAND.

PROBABLE FAULT:

- 1. IF MODULE

RMDC CLEAR OFFSET TEST

PURPOSE:

TO VERIFY THAT CLEAR OFFSET IS ACTIVE WHEN THE DESIRED CYLINDER ADDRESS IS WRITTEN.

PROCEDURE:

THE TEST EXECUTES AN OFFSET COMMAND, WRITES RMDC, AND VERIFIES THAT OM, BIT 00 OF RMD5 IS ZERO.

PROBABLE FAULT:

- 1. DS MODULE
- 2. IF MODULE

EBL CLEAR OFFSET TEST

PURPOSE:

TO VERIFY THAT OFFSET MODE CLEARS WHEN HEAD SWITCHING OCCURS.

PROCEDURE:

1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793

THE TEST EXECUTES AN OFFSET COMMAND TO SET OFFSET MODE. AFTER SETTING THE FORMAT BIT AND LOADING THE LAST SECTOR/TRACK ADDRESS IN RMDA, THE TEST FORCES AN EBL AND VERIFIES THAT OFFSET MODE RESETS.

PROBABLE FAULT:

1. DS MODULE

RUN AND GO TEST

PURPOSE:

TO VERIFY THAT "RUN AND GO" FLOP SETS DURING READ AND WRITE COMMANDS.

PROCEDURE:

THE RMO3 IS INITIALIZED AND A DATA TRANSFER COMMAND WITH GO SET IS WRITTEN IN RMCS1. "RUN AND GO", BIT 14 OF RMMR1 SHOULD BE ONE FOR EACH DATA COMMAND. THE DEBUG CLOCK IS ENABLED SO THAT GO DOES NOT RESET BEFORE STATUS IS TESTED.

PROBABLE FAULT:

1. CS MODULE
2. SYNCHRONOUS MASSBUS MODULE

SET IAE TEST

PURPOSE:

TO VERIFY THAT INVALID ADDRESS ERROR CAN SET.

PROCEDURE:

THE TEST LOADS INVALID SECTOR, TRACK AND CYLINDER ADDRESSES AND EXECUTES A SEARCH COMMAND, VERIFYING THAT "IAE" SETS. THE PROCESS IS REPEATED WITH A DIFFERENT COMMAND IF THE IAE DOES NOT SET, AND THE TEST FAILS IF IAE CANNOT BE SET.

PROBABLE FAULT:

1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845

1. DS MODULE
2. IF MODULE

SEARCH, SEEK, READ, WRITE TEST

PURPOSE:

TO VERIFY THAT THE "SCH SK R OR W" DECODE ON THE IF MODULE IS CORRECT FOR ALL FUNCTION CODES.

PROCEDURE:

THE TEST LOADS INVALID SECTOR, TRACK AND CYLINDER ADDRESSES AND EXECUTES EACH COMMAND TO WHERE SET PULSE IS ACTIVE AND VERIFIES THE DECODE BY CHECKING "IAE".

PROBABLE FAULT:

1. IF MODULE

INVALID SECTOR/TRACK TEST

PURPOSE:

TO VERIFY THAT INVALID SECTOR AND TRACK ADDRESSES ARE DETECTED.

PROCEDURE:

THE TEST LOADS THE TEST PATTERN IN RMDA AND EXECUTES A SEARCH COMMAND, VERIFYING THAT "IAE" SETS.

PROBABLE FAULT:

1. DS MODULE
2. TRACK ADDRESS OPTION JUMPER

1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901

INVALID CYLINDER TEST

PURPOSE:

TO VERIFY THAT INVALID CYLINDER ADDRESSES ARE DETECTED.

PROCEDURE:

THE TEST LOADS THE TEST PATTERN IN RMDC AND EXECUTES A SEARCH COMMAND, VERIFYING THAT "IAE" SETS.

PROBABLE FAULTS:

1. DS MODULE
2. CYLINDER ADDRESS OPTION JUMPER

SET AOE TEST

PURPOSE:

TO VERIFY THAT ADDRESS OVERFLOW ERROR IS DETECTED.

PROCEDURE:

THE TEST LOADS THE ADDRESS OF THE LAST SECTOR IN RMDA AND RMDC, THEN INITIATES A DATA COMMAND WITH DEBUG CLOCK ENABLED. END OF BLOCK IS FORCED TO INCREMENT THE SECTOR ADDRESS, AND THE TEST VERIFIES THAT "AOE" IS SET.

PROBABLE FAULT:

1. DS MODULE

SET RMR TEST

PURPOSE:

TO VERIFY THAT "REGISTER MODIFICATION REFUSED" SETS WHEN A REGISTER IS WRITTEN WHILE GO IS SET, EXCEPT WHEN THE ATTENTION OR MAINTENANCE REGISTER IS WRITTEN.

PROCEDURE:

1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954

"DEBUG CLOCK ENABLE" IS SET TO INHIBIT THE COMMAND SEQUENCER. THEN A NOP COMMAND AND GO BIT IS WRITTEN IN RMCS1. WITHOUT STEPPING THE DEBUG CLOCK, THE TEST WRITES RMMR AND RMA5 WHICH SHOULD NOT SET RMR STATUS. THEN RMDA IS WRITTEN AND RMA STATUS, BIT 02 OF RMR1, SHOULD BE ONE.

PROBABLE FAULT:

- 1. IF MODULE

PGM STATUS CHECK

PURPOSE:

TO VERIFY THAT THE PROGRAMMABLE STATUS BIT AND THE DRIVE REQUEST STATUS BIT ARE COMPATABLE.

PROCEDURE:

THE TEST REPORTS AN ERROR IF PGM IS ON AND DRQ IS OFF. PGM IS NOT PREDICTABLE IN THE CASE WHERE DRQ IS ON BECAUSE OF THE PORT SELECT SWITCH.

PROBABLE FAULT:

- 1. IF MODULE

DVA/DPR STATUS CHECK PURPOSE:

TO VERIFY THAT DRIVE PRESENT STATUS AND DEVICE AVAILABLE STATUS ARE SET.

PROCEDURE:

DPR AND DVA ARE TESTED AND BOTH SHOULD BE ON.

PROBABLE FAULT:

- 1. IF MODULE

1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010

PORT REQUEST TEST, PART 1/3

PURPOSE:

TO VERIFY THAT THE PORT REQUEST FLOPS ON THE IF MODULE SET WHEN THE PROGRAM READS RMCS1.

PROCEDURE:

THE TEST EXECUTES A RELEASE COMMAND, THEN, ASSUMING THE PORT IS RELEASED, IT READS RMCS1, THEN READS RMMR2 AND VERIFIES THAT ONE OF THE PORT REQUEST FLOPS IS SET.

PROBABLE FAULT:

1. IF MODULE
2. CS MODULE

PORT REQUEST TEST, PART 2/3

PURPOSE:

TO VERIFY THAT THE PORT REQUEST FLOPS ON THE IF MODULE SET WHEN THE PROGRAM WRITES RMAS.

PROCEDURE:

THE TEST EXECUTES A RELEASE COMMAND THEN WRITES RMAS AND READS RMMR2, VERIFYING THAT ONE OF THE REQUEST FLOPS IS SET.

PROBABLE FAULT:

1. IF MODULE
2. CS MODULE

DATA COMMAND TESTS

PURPOSE:

TO VERIFY THE COMMAND SEQUENCER DURING DATA COMMANDS.

CZRMJBO RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 51

M04

SEQ 0051

2011

PROCEDURE:

THIS TEST, LIKE RECALIBRATE, SEEK, AND SEARCH TESTS, USES THE MAINTENANCE REGISTER TO SIMULATE DRIVE CONDITIONS AND FORCE THE COMMAND SEQUENCER THROUGH EACH BRANCH PATH. ADDITIONAL ITEMS WHICH ARE TESTED INCLUDE OFFSET PLUS AND MINUS ON THE TAG BUS AND "ENABLE SEARCH", BIT 11 OF RMMR1.

PROBABLE FAULT:

1. CS MODULE

DATA SEQUENCER READ TEST

PURPOSE:

TO TEST THE CS AND DS MODULES FOR EXECUTION OF A READ COMMAND.

PROCEDURE:

THE RMO3 IS INITIALIZED AND, USING DIAGNOSTIC MODE, A READ COMMAND IS STEPPED TO THE POINT WHERE THE COMMAND SEQUENCER ENABLES SEARCH. THE PROGRAM THEN STARTS STEPPING THE DATA SEQUENCER ON THE CS MODULE AND MONITORS HARDWARE OPERATION USING THE FOLLOWING STATUS:

.PLFS, BIT 10 OF RMMR1, SHOWS THE PERIOD WHEN THE DATA PROM HAS ENABLED SYNC BYTE DETECTION;

.PDA, BIT 08 OF RMMR1, SHOWS WHEN THE DATA PROM IS IN THE DATA AREA OF THE SECTOR;

.PHA, BIT 07 OF RMMR1, SHOWS WHEN THE DATA PROM IS IN THE HEADER AREA OF THE SECTOR;

.BBO0, BIT 00 OF RMMR2, SHOWS THE STATE OF WRITE GATE PROVIDING TAG IS HIGH;

.BBO1, BIT 01 OF RMMR2, SHOWS THE STATE OF READ GATE PROVIDING TAG IS HIGH;

.WC, BIT 05 OF RMMR1, SHOWS THE STATE OF WORD CLOCK;

.EECC, BIT 04 OF RMMR1, SHOWS WHEN ECC CHARACTER IS ENABLED DURING WRITE;

.ECRC, BIT 09 OF RMMR1, SHOWS WHEN CRC CHARACTER IS ENABLED DURING WRITE;

2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066

2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121

.WD, BIT 03 OF RMMPI, SHOWS THE STATE OF WRITE DATA.

ADDITIONALLY, STATUS BITS IN RMER1 ARE USED TO TEST COMMAND EXECUTION. THE WRITE ONLY BITS OF RMMPI ARE USED TO SIMULATE DRIVE STATUS AND CONTROL SIGNALS, AND TO SPECIFY READ DATA.

PROBABLE FAULT:

- 1. CS MODULE
- 2. DS MODULE

DATA SEQUENCER WRITE TEST

PURPOSE:

TO TEST THE CS AND DS MODULES FOR EXECUTION OF A WRITE COMMAND.

PROCEDURE:

THE PROCEDURE IS THE SAME AS THE DATA SEQUENCER READ TEST.

PROBABLE FAULT:

- 1. CS MODULE
- 2. DS MODULE

DATA SEQUENCER FORMAT TEST

PURPOSE:

TO TEST CS AND DS MODULES FOR EXECUTION OF A FORMAT COMMAND.

PROCEDURE:

THE PROCEDURE IS THE SAME AS THE DATA SEQUENCER READ TEST.

PROBABLE FAULT:

- 1. CS MODULE

2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175

2. DS MODULE

18 BIT WORD CLOCK TEST

PURPOSE:

TO TEST THE GENERATION OF WORD CLOCK IN 18 BIT FORMAT.

PROCEDURE:

DIAGNOSTIC MODE IS USED TO EXECUTE A READ COMMAND. THE PROGRAM VERIFIES THAT WORD CLOCK SWITCHES TO 18 BIT FORMAT AT THE DATA AREA AND SWITCHES BACK TO 16 BIT FORMAT DURING ECC. THE TEST IS REPEATED FOR A WRITE COMMAND.

PROBABLE FAULT:

- 1. CS MODULE
- 2. DS MODULE

HCRC SET TEST

PURPOSE:

TO VERIFY THAT "HCRC" STATUS SETS WHEN THE HEADER CRC WORD IS IN ERROR.

PROCEDURE:

USING DIAGNOSTIC MODE TO EXECUTE A READ COMMAND, THE PROGRAM STEPS THE DATA SEQUENCER THROUGH THE HEADER AND PROVIDES READ DATA WHICH SHOULD CAUSE A CRC ERROR. THE TEST FAILS IF "HCRC", BIT 08 OF RMER1 DOES NOT SET.

PROBABLE FAULT:

- 1. DS MODULE
- 2. IF MODULE

2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227

HCE SET TEST

PURPOSE:

TO VERIFY THAT "HCE" STATUS SETS IF THE HEADER DOES NOT COMPARE WITH RMDC AND RMDA.

PROCEDURE:

THE PROGRAM EXECUTES A READ COMMAND USING DIAGNOSTIC MODE. READ DATA IS SUPPLIED WHICH CAUSES A HEADER COMPARE ERROR IN BITS 00-03 OF HEADER WORD ONE AND THE PROGRAM VERIFIES THAT "HCE", BIT 07 OF RMER1 IS ONE. THIS PROCESS IS REPEATED FOR BITS 04-07 AND 08-11.

PROBABLE FAULT:

- 1. DS MODULE
- 2. CS MODULE
- 3. IF MODULE

FER SET TEST

PURPOSE:

TO VERIFY THAT "FER" STATUS SETS WHEN THE FORMAT BIT DOES NOT COMPARE WITH THE HEADER.

PROCEDURE:

THIS PROGRAM EXECUTES A READ COMMAND USING DIAGNOSTIC MODE AND FORCES A FORMAT ERROR IN THE FIRST HEADER WORD. THE TEST FAILS IF "FER", BIT 04 OF RMER1 DOES NOT SET.

PROBABLE FAULT:

- 1. DS MODULE
- 2. CS MODULE
- 3. IF MODULE

2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277

DCK SET TEST

PURPOSE:

TO VERIFY THAT "DCK" STATUS SETS WHEN A CORRECTABLE ECC ERROR OCCURS.

PROCEDURE:

THE PROGRAM SIMULATES A READ OPERATION AND FORCES AN ECC ERROR WHICH IS CORRECTABLE. "DCK", BIT 15 OF RMER1 SHOULD BE SET AND "ECH" BIT 06 OF RMER1 SHOULD BE RESET. RMEC1 AND RMEC2 SHOULD PERMIT DATA CORRECTION.

PROBABLE FAULT:

- 1. DS MODULE
- 2. CS MODULE
- 3. IF MODULE

ECH SET TEST

PURPOSE:

TO VERIFY THAT "ECH" STATUS SETS WHEN A NONCORRECTABLE ECC ERROR OCCURS DURING READ.

PROCEDURE:

THIS TEST IS LIKE THE DCK SET TEST EXCEPT THE PROGRAM USES AN UNCORRECTABLE ERROR BURST.

PROBABLE FAULT:

- 1. DS MODULE
- 2. CS MODULE
- 3. IF MODULE

2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323

DVA DPR

STATUS CHECK PURPOSE:

TO VERIFY THAT DRIVE PRESENT STATUS AND DEVICE AVAILABLE STATUS ARE SET.

PROCEDURE:

DPR AND DVA ARE TESTED AND BOTH SHOULD BE ON.

PROBABLE FAULT:

- 1. IF MODULE

PGM STATUS CHECK

PURPOSE:

TO VERIFY THAT THE PROGRAMMABLE STATUS BIT AND THE DRIVE REQUEST STATUS BIT ARE COMPATABLE.

PROCEDURE:

THE TEST REPORTS AN ERROR IF PGM IS ON AND DRQ IS OFF. PGM IS NOT PREDICTABLE IN THE CASE WHERE DRQ IS ON BECAUSE OF THE PORT SELECT SWITCH.

PROBABLE FAULT:

- 1. IF MODULE

SET AOE TEST

PRUPOSE:

TO VERIFY THAT ADDRESS OVERFLOW ERROR IS DETECTED.

PROCEDURE:

THE TEST LOADS THE ADDRESS OF THE LAST SECTOR IN RMDA AND RMDC, THEN INITIATES A DATA COMMAND WITH DEBUG CLOCK ENABLED.

G05

CZRMJ80 RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 58

SEG 0058

2334
2335

END OF BLOCK IS FORCED TO INCREMENT THE SECTOR ADDRESS, AND THE
TEST VERIFIES THAT "AOE" IS SET.

2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388

PROBABLE FAULT:

- 1. DS MODULE

CLEAR OFFSET STUCK ACTIVE TEST

PURPOSE:

TO VERIFY THAT THE SIGNAL WHICH CLEARS OFFSET MODE IS NOT STUCK IN ACTIVE STATE.

PROCEDURE:

THE TEST WRITES A ONE IN THE OFFSET DIRECTION BIT WHICH IS CLEARED BY THE SIGNAL AND VERIFIES THAT A ONE CAN BE READ BACK.

PROBABLE FAULT:

- 1. IF MODULE
- 2. DS MODULE

INVALID SECTOR/TRACK TEST

PURPOSE:

TO VERIFY THAT INVALID SECTOR AND TRACK ADDRESSES ARE DETECTED.

PROCEDURE:

THE TEST LOADS THE TEST PATTERN IN RMDA AND EXECUTES A SEARCH COMMAND, VERIFYING THAT "IAE" SETS.

PROBABLE FAULT:

- 1. DS MODULE
- 2. TRACK ADDRESS OPTION JUMPER

INVALID CYLINDER TEST

PURPOSE:

TO VERIFY THAT INVALID CYLINDER ADDRESSES ARE DETECTED.

PROCEDURE:

THE TEST LOADS THE TEST PATTERN IN RMDC AND EXECUTES A SEARCH COMMAND, VERIFYING THAT "IAE" SETS.

PROBABLE FAULTS:

- 1. DS MODULE
- 2. CYLINDER ADDRESS OPTION JUMPER,

SEARCH, SEEK, READ, WRITE TEST

PURPOSE:

TO VERIFY THAT THE "SCH SK R OR W" DECODE ON THE IF MODULE IS CORRECT FOR ALL FUNCTION CODES.

PROCEDURE:

THE TEST LOADS INVALID SECTOR, TRACK AND CYLINDER ADDRESSES AND EXECUTES EACH COMMAND TO WHERE SET PULSE IS ACTIVE AND VERIFIES THE DECODE BY CHECKING "IAE".

PROBABLE FAULT:

- 1. IF MODULE

EBL CLEAR OFFSET TEST

PURPOSE:

TO VERIFY THAT OFFSET MODE CLEARS WHEN HEAD SWITCHING OCCURS.

PROCEDURE:

Vertical column of characters on the left margin, likely a page number or index marker.

K05

CZRMJ80 RM03/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 62

SEQ 0062

2500
2501

PROBABLE FAULT:

?

2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553

1. IF MODULE

SET IAE TEST

PURPOSE:

TO VERIFY THAT INVALID ADDRESS ERROR CAN SET.

PROCEDURE:

THE TEST LOADS INVALID SECTOR, TRACK AND CYLINDER ADDRESSES AND EXECUTES A SEARCH COMMAND, VERIFYING THAT "IAE" SETS. THE PROCESS IS REPEATED WITH A DIFFERENT COMMAND IF THE IAE DOES NOT SET, AND THE TEST FAILS IF IAE CANNOT BE SET.

PROBABLE FAULT:

1. DS MODULE
2. IF MODULE

PORT REQUEST TEST, PART 3/3

PURPOSE:

TO VERIFY THAT PORT REQUEST SETS WHEN ANY REGISTER EXCEPT RMAS IS WRITTEN.

PROCEDURE:

THE TEST WRITES THE DISK ADDRESS REGISTER AND VERIFIES THAT THE PORT REQUEST FLOP IS ON.

PROBABLE FAULT:

1. IF MODULE
2. CS MODULE

2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609

WRITE ATA TEST

PURPOSE:

TO VERIFY THAT ATTENTION CAN BE CLEARED BY WRITING THE ATTENTION SUMMARY REGISTER.

PROCEDURE:

THE PROGRAM RESETS AND SETS UNIT READY WHICH SHOULD CAUSE AN ATTENTION, THEN WRITES THE ATTENTION SUMMARY REGISTER AND VERIFIES THAT ATTENTION IS RESET.

PROBABLE FAULT:

- 1. IF MODULE
- 2. CS MODULE

RESET ATA BY GO TEST

PURPOSE:

TO VERIFY THAT ATA RESETS WHEN GO IS ON AND COMPOSITE ERROR IS OFF.

PROCEDURE:

THE PROGRAM SETS MAINTENANCE UNIT READY WHICH SHOULD CAUSE AN ATTENTION. THEN, WITH DEBUG CLOCK ENABLED, GO IS SET, AND ATA SHOULD BE ZERO.

PROBABLE FAULT:

- 1. IF MODULE

UNIT READY ATA TEST

PURPOSE:

TO VERIFY THAT ONE-ZERO AND ZERO-ONE TRANSITIONS OF UNIT READY SET ATTENTION.

CZRMJBO RM03/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 65

N05

SEQ 0065

2610

PROCEDURE:

2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2660
2661
2662
2663
2664

THE TEST USES DIAGNOSTIC MODE TO FORCE BOTH TRANSITIONS OF UNIT READY AND VERIFIES THAT ATA SETS WITH EACH TRANSITION.

PROBABLE FAULT:

- 1. IF MODULE

ERROR ATA TEST

PURPOSE:

TO VERIFY THAT ATTENTION SETS WHEN COMPOSITE ERROR OCCURS WHILE GO IS OFF.

PROCEDURE:

THE PROGRAM CLEARS THE DEVICE AND SETS AN ERROR, THEN VERIFIES ATA IS ON.

PROBABLE FAULT:

- 1. IF MODULE

REGISTER TRANSFER ATA TEST

TO VERIFY THAT ATTENTION SETS WHEN ANY REGISTER, EXCEPT FOR RMA5 AND RMCS, IS WRITTEN WHILE COMP ERROR IS SET.

PROCEDURE:

THE PROGRAM FORCES AN ERROR THEN RESETS ATTENTION FROM THE ERROR. THE PROGRAM THEN WRITES RMA5 AND RMCS AND VERIFIES THAT NO ATTENTION OCCURS, AND WRITES RMDC AND VERIFIES THAT ATTENTION DOES OCCUR.

PROBABLE FAULT:

- 1. IF MODULE

2665
2666
2667
2668
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2699
2700
2701
2702
2703
2704
2705
2706
2707
2708
2709
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2720

000001

001100

000011

000012

000015

000200

177776

177774

177772

177570

177570

000000

000001

000002

000003

```

;PROGRAM REVISION #001
.TITLE CZRMJBO RMO3/2 DSKLS DIAG
*COPYRIGHT (C) 1977
*DIGITAL EQUIPMENT CORP.
*MAYNARD, MASS. 01754
*
*PROGRAM BY DOUG RIIKONEN
*
*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
*
$TN=1
.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
.SBTTL BASIC DEFINITIONS
*
*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
STACK= 1100
.EQUIV EMT,ERROR ;;BASIC DEFINITION OF ERROR CALL
.EQUIV IOT,SCOPE ;;BASIC DEFINITION OF SCOPE CALL
*
*MISCELLANEOUS DEFINITIONS
AT= 11 ;;CODE FOR HORIZONTAL TAB
LF= 12 ;;CODE FOR LINE FEED
CR= 15 ;;CODE FOR CARRIAGE RETURN
CRLF= 200 ;;CODE FOR CARRIAGE RETURN-LINE FEED
PS= 177776 ;;PROCESSOR STATUS WORD
.EQUIV PS,PSW
STKLMT= 177774 ;;STACK LIMIT REGISTER
PIRQ= 177772 ;;PROGRAM INTERRUPT REQUEST REGISTER
DSWR= 177570 ;;HARDWARE SWITCH REGISTER
DDISP= 177570 ;;HARDWARE DISPLAY REGISTER
*
*GENERAL PURPOSE REGISTER DEFINITIONS
R0= %0 ;;GENERAL REGISTER
R1= %1 ;;GENERAL REGISTER
R2= %2 ;;GENERAL REGISTER
R3= %3 ;;GENERAL REGISTER

```

BASIC DEFINITIONS

2721	000004	R4=	%4	::	GENERAL REGISTER
2722	000005	R5=	%5	::	GENERAL REGISTER
2723	000006	R6=	%6	::	GENERAL REGISTER
2724	000007	R7=	%7	::	GENERAL REGISTER
2725	000006	SP=	%6	::	STACK POINTER
2726	000007	PC=	%7	::	PROGRAM COUNTER

.*PRIORITY LEVEL DEFINITIONS

2728		PR0=	0	::	PRIORITY LEVEL 0
2729	000000	PR1=	40	::	PRIORITY LEVEL 1
2730	000040	PR2=	100	::	PRIORITY LEVEL 2
2731	000100	PR3=	140	::	PRIORITY LEVEL 3
2732	000140	PR4=	200	::	PRIORITY LEVEL 4
2733	000200	PR5=	240	::	PRIORITY LEVEL 5
2734	000240	PR6=	300	::	PRIORITY LEVEL 6
2735	000300	PR7=	340	::	PRIORITY LEVEL 7
2736	000340				

.*"SWITCH REGISTER" SWITCH DEFINITIONS

2738		SW15=	100000		
2739	100000	SW14=	40000		
2740	040000	SW13=	20000		
2741	020000	SW12=	10000		
2742	010000	SW11=	4000		
2743	004000	SW10=	2000		
2744	002000	SW09=	1000		
2745	001000	SW08=	400		
2746	000400	SW07=	200		
2747	000200	SW06=	100		
2748	000100	SW05=	40		
2749	000040	SW04=	20		
2750	000020	SW03=	10		
2751	000010	SW02=	4		
2752	000004	SW01=	2		
2753	000002	SW00=	1		
2754	000001				
2755		.EQUIV	SW09, SW9		
2756		.EQUIV	SW08, SW8		
2757		.EQUIV	SW07, SW7		
2758		.EQUIV	SW06, SW6		
2759		.EQUIV	SW05, SW5		
2760		.EQUIV	SW04, SW4		
2761		.EQUIV	SW03, SW3		
2762		.EQUIV	SW02, SW2		
2763		.EQUIV	SW01, SW1		
2764		.EQUIV	SW00, SW0		
2765					

.*DATA BIT DEFINITIONS (BIT00 TO BIT15)

2766		BIT15=	100000		
2767	100000	BIT14=	40000		
2768	040000	BIT13=	20000		
2769	020000	BIT12=	10000		
2770	010000	BIT11=	4000		
2771	004000	BIT10=	2000		
2772	002000	BIT09=	1000		
2773	001000	BIT08=	400		
2774	000400	BIT07=	200		
2775	000200	BIT06=	100		
2776	000100				

BASIC DEFINITIONS

```

2777      000040      BIT05= 40
2778      000020      BIT04= 20
2779      000010      BIT03= 10
2780      000004      BIT02= 4
2781      000002      BIT01= 2
2782      000001      BIT00= 1
2783      .EQUIV      BIT09, BIT9
2784      .EQUIV      BIT08, BIT8
2785      .EQUIV      BIT07, BIT7
2786      .EQUIV      BIT06, BIT6
2787      .EQUIV      BIT05, BIT5
2788      .EQUIV      BIT04, BIT4
2789      .EQUIV      BIT03, BIT3
2790      .EQUIV      BIT02, BIT2
2791      .EQUIV      BIT01, BIT1
2792      .EQUIV      BIT00, BIT0
2793
2794      ; *BASIC "CPU" TRAP VECTOR ADDRESSES
2795      ERRVEC= 4      ; TIME OUT AND OTHER ERRORS
2796      RESVEC= 10     ; RESERVED AND ILLEGAL INSTRUCTIONS
2797      TBITVEC= 14    ; "T" BIT
2798      TRTVEC= 14     ; TRACE TRAP
2799      BPTVEC= 14     ; BREAKPOINT TRAP (BPT)
2800      IOTVEC= 20     ; INPUT/OUTPUT TRAP (IOT) **SCOPE**
2801      PWRVEC= 24     ; POWER FAIL
2802      EMTVEC= 30     ; EMULATOR TRAP (EMT) **ERROR**
2803      TRAPVEC= 34    ; "TRAP" TRAP
2804      TKVEC= 60      ; TTY KEYBOARD VECTOR
2805      TPVEC= 64      ; TTY PRINTER VECTOR
2806      PIRQVEC= 240   ; PROGRAM INTERRUPT REQUEST VECTOR
2807
2808      .SBTTL RM03 REGISTER BIT DEFINITIONS
2809
2810      ; RMCS1 CONTROL STATUS REGISTER
2811
2812      DVA = BIT11     ; DEVICE AVAILABLE-READ ONLY
2813      F4 = BIT05     ; FUNCTION CODE
2814      F3 = BIT04     ; FUNCTION CODE
2815      F2 = BIT03     ; FUNCTION CODE
2816      F1 = BIT02     ; FUNCTION CODE
2817      F0 = BIT01     ; FUNCTION CODE
2818      GO = BIT00     ; GO BIT
2819      FNCMSK = 000077 ; FUNCTION CODE MASK
2820
2821      ; FUNCTION CODES (BITS 01-05 OF RMCS1)
2822
2823      NOP = 000000    ; NOP COMMAND
2824      ILFO2 = 000002 ; ILLEGAL COMMAND
2825      SEEK = 000004  ; SEEK COMMAND
2826      RECAL = 000006 ; RECALIBRATE COMMAND
2827      DRVCLR = 000010 ; DRIVE CLEAR COMMAND
2828      RELEASE = 000012 ; RELEASE COMMAND
2829      OFFSET = 000014 ; OFFSET COMMAND
2830      RTC = 000016  ; RETURN TO CENTERLINE COMMAND
2831      RIP = 000020  ; READ IN PRESET COMMAND
2832      PAKACK = 000022 ; PACK ACKNOWLEDGE COMMAND

```

2833	000022	PACACK	=	PAKACK	
2834	000024	ILF24	=	000024	; ILLEGAL COMMAND
2835	000026	ILF26	=	000026	; ILLEGAL COMMAND
2836	000030	SEARCH	=	000030	; SEARCH COMMAND
2837	000030	ILF30	=	000030	; ILLEGAL COMMAND
2838	000032	ILF32	=	000032	; ILLEGAL COMMAND
2839	000034	ILF34	=	000034	; ILLEGAL COMMAND
2840	000036	ILF36	=	000036	; ILLEGAL COMMAND
2841	000040	ILF40	=	000040	; ILLEGAL COMMAND
2842	000042	ILF42	=	000042	; ILLEGAL COMMAND
2843	000044	ILF44	=	000044	; ILLEGAL COMMAND
2844	000046	ILF46	=	000046	; ILLEGAL COMMAND
2845	000050	WCD	=	000050	; WRITE CHECK DATA COMMAND
2846	000052	WCH	=	000052	; WRITE CHECK HEADER AND DATA
2847	000054	ILF54	=	000054	; ILLEGAL COMMAND
2848	000056	ILF56	=	000056	; ILLEGAL COMMAND
2849	000060	WD	=	000060	; WRITE DATA COMMAND
2850	000062	WH	=	000062	; WRITE HEADER AND DATA COMMAND
2851	000064	ILF64	=	000064	; ILLEGAL COMMAND
2852	000066	ILF66	=	000066	; ILLEGAL COMMAND
2853	000070	RD	=	000070	; READ DATA COMMAND
2854	000072	RH	=	000072	; READ HEADER AND DATA COMMAND
2855	000074	ILF74	=	000074	; ILLEGAL COMMAND
2856	000076	ILF76	=	000076	; ILLEGAL COMMAND
2857					
2858		;RMDA		DISK ADDRESS REGISTER	
2859					
2860	002000	TA4	=	BIT10	; TRACK ADDRESS 4
2861	001000	TA2	=	BIT09	; TRACK ADDRESS 2
2862	000400	TA1	=	BIT08	; TRACK ADDRESS 1
2863	000020	SA16	=	BIT04	; SECTOR ADDRESS 16
2864	000010	SA8	=	BIT03	; SECTOR ADDRESS 8
2865	000004	SA4	=	BIT02	; SECTOR ADDRESS 4
2866	000002	SA2	=	BIT01	; SECTOR ADDRESS 2
2867	000001	SA1	=	BIT00	; SECTOR ADDRESS 1
2868					
2869		; TRACK, SECTOR MASKS			
2870					
2871	003400	TADMSK	=	003400	; TRACK ADDRESS MASK
2872	000037	SA0MSK	=	000037	; SECTOR ADDRESS MASK
2873					
2874		;RMD5		DRIVE STATUS REGISTER	
2875					
2876	100000	ATA	=	BIT15	; ATTENTION ACTIVE
2877	040000	ERR	=	BIT14	; COMPOSITE ERROR
2878	020000	PIP	=	BIT13	; POSITIONING IN PROGRESS
2879	010000	MOL	=	BIT12	; MEDIUM ON LINE
2880	004000	WRL	=	BIT11	; WRITE LOCK
2881	002000	LBT	=	BIT10	; LAST BLOCK TRANSFERRED
2882	001000	PGM	=	BIT09	; PROGRAMMABLE
2883	000400	DPR	=	BIT08	; DRIVE PRESENT
2884	000200	DRY	=	BIT07	; DRIVE READY
2885	000100	VV	=	BIT06	; VOLUME VALID
2886	000001	OM	=	BIT00	; OFFSET MODE ACTIVE
2887					
2888		;RMR1		ERROR REGISTER #1	


```

2889
2890      100000      DCK      =      BIT15      ;DATA CHECK ERROR
2891      040000      UNS      =      BIT14      ;DRIVE UNSAFE
2892      020000      OPI      =      BIT13      ;OPERATION INCOMPLETE
2893      010000      DTE      =      BIT12      ;DRIVE TIMING ERROR
2894      004000      WLE      =      BIT11      ;WRITE LOCK ERROR
2895      002000      IAE      =      BIT10      ;INVALID ADDRESS ERROR
2896      001000      AOE      =      BIT09      ;ADDRESS OVERFLOW ERROR
2897      000400      HCRC     =      BIT08      ;HEADER CRC ERROR
2898      000200      HCE      =      BIT07      ;HEADER COMPARE ERROR
2899      000100      ECH      =      BIT06      ;ECC "HARD" ERROR
2900      000040      WCF      =      BIT05      ;WRITE CLOCK FAILURE
2901      000020      FER      =      BIT04      ;FORMAT ERROR
2902      000010      PAR      =      BIT03      ;PARITY ERROR
2903      000004      RMR      =      BIT02      ;REGISTER MODIFICATION REFUSED
2904      000002      ILR      =      BIT01      ;ILLEGAL REGISTER
2905      000001      ILF      =      BIT00      ;ILLEGAL FUNCTION
2906
2907      115760      NDTMSK   =      DCK!DTE!WLE!AOE!HCRC!HCE!ECH!WCF!FER
2908      ;"NDTMSK" IS USED TO MASK ERROR REGISTER 1 DURING NON - DATA
2909      ;COMMANDS, I.E., HOUSEKEEPING AND POSITIONING COMMANDS
2910
2911      ;RMS      ATTENTION SUMMARY REGISTER
2912
2913      000377      ATNMSK   =      377      ;MASK FOR ATTENTION BITS
2914
2915      ;RMLA     LOOK AHEAD REGISTER
2916
2917      002000      SC4      =      BIT10      ;SECTOR COUNT = 16
2918      001000      SC3      =      BIT09      ;SECTOR COUNT = 8
2919      000400      SC2      =      BIT08      ;SECTOR COUNT = 4
2920      000200      SC1      =      BIT07      ;SECTOR COUNT = 2
2921      000100      SC0      =      BIT06      ;SECTOR COUNT = 1
2922
2923      003700      SCTMSK   =      003700   ;SECTOR COUNT MASK
2924
2925      ;RMMR     MAINTENANCE REGISTER
2926
2927      ;          WRITE ONLY BITS
2928
2929      100000      DBCK     =      BIT15      ;DEBUG CLOCK
2930      040000      DBEN     =      BIT14      ;DEBUG CLOCK ENABLE

```

RM03 REGISTER BIT DEFINITIONS

2931	020000	DEBL	=	BIT13	;DIAGNOSTIC END OF BLOCK
2932	010000	OTO	=	BIT12	;DIAGNOSTIC TIMEOUT
2933	004000	MCLK	=	BIT11	;MAINTENANCE CLOCK
2934	002000	MRD	=	BIT10	;READ DATA
2935	001000	MUR	=	BIT09	;UNIT READY
2936	000400	MOC	=	BIT08	;ON CYLINDER
2937	000200	MSER	=	BIT07	;SEEK ERROR
2938	000100	MDF	=	BIT06	;DRIVE FAULT
2939	000040	MS	=	BIT05	;SECTOR PULSE
2940	000010	MWP	=	BIT03	;WRITE PROTECT
2941	000004	MI	=	BIT02	;INDEX PULSE
2942	000002	MSC	=	BIT01	;SECTOR COMPARE
2943	000001	DMD	=	BIT00	;DIAGNOSTIC MODE
2944					
2945		:		READ ONLY BITS	
2946					
2947	100000	OCC	=	BIT15	;OCCUPIED
2948	040000	RG	=	BIT14	;RUN AND GO
2949	020000	EBL	=	BIT13	;END OF BLOCK
2950	010000	REX	=	BIT12	;EXCEPTION
2951	004000	ESRC	=	BIT11	;ENABLE SEARCH
2952	002000	PLFS	=	BIT10	;LOOKING FOR SYNC
2953	001000	ECRC	=	BIT09	;ENABLE CRC OUT
2954	000400	PDA	=	BIT08	;DATA AREA
2955	000200	PHA	=	BIT07	;HEADER AREA
2956	000100	CONT	=	BIT06	;CONTINUE
2957	000040	WC	=	BIT05	;WORD CLOCK
2958	000020	EECC	=	BIT04	;ENABLE ECC OUT
2959	000010	MWD	=	BIT03	;WRITE DATA BIT
2960	000004	LS	=	BIT02	;LAST SECTOR
2961	000002	LST	=	BIT01	;LAST SECTOR AND TRACK
2962	000001	DMD	=	BIT00	;DIAGNOSTIC MODE
2963					
2964		;RMDT		DRIVE TYPE REGISTER	
2965					
2966	100000	NSA	=	BIT15	;NOT SECTOR ADDRESSED=0
2967	040000	TAP	=	BIT14	;TAPE DRIVE = 0
2968	020000	MOH	=	BIT13	;MOVING HEAD = 1
2969	004000	DRQ	=	BIT11	;DRIVE REQUEST REQUIRED
2970					
2971	020024	SNGPRT	=	020024	;SINGLE PORT DRIVE TYPE
2972	024024	DULPRT	=	024024	;DUAL PORT DRIVE TYPE
2973					
2974		;RMOF		OFFSET REGISTER	
2975					
2976	010000	FMT16	=	BIT12	;16 BIT WORD FORMAT
2977	004000	ECI	=	BIT11	;ECC INHIBIT
2978	002000	HCI	=	BIT10	;HEADER COMPARE INHIBIT
2979	000200	OFO	=	BIT07	;OFFSET FORWARD
2980	161577	XNUOF	=	161577	;UNUSED BITS OF RMOF
2981					
2982		;RMDC		DESIRED CYLINDER ADDRESS REGISTER	
2983					
2984	001777	CYLMSK	=	1777	;MASK FOR CYLINDER ADDRESS
2985	176000	XNUDC	=	176000	;UNUSED BITS OF RMDC
2986					

```

2987 ;RMMR2 MAINTENANCE REGISTER #2
2988
2989 ; READ ONLY BITS
2990 100000 RQA = BIT15 ; PORT A REQUEST
2991 040000 RQB = BIT14 ; PORT B REQUEST
2992 020000 TAG = BIT13 ; TAG CONTROL
2993 010000 TST = BIT12 ; COMMAND SEQUENCE TEST BIT
2994 004000 CC = BIT11 ; CONTROL OR CYLINDER TAG
2995 002000 CH = BIT10 ; CONTROL OR HEAD TAG
2996 001000 BB09 = BIT09 ; TAG BUS
2997 000400 BB08 = BIT08 ; TAG BUS
2998 000200 BB07 = BIT07 ; TAG BUS
2999 000100 BB06 = BIT06 ; TAG BUS
3000 000040 BB05 = BIT05 ; TAG BUS
3001 000020 BB04 = BIT04 ; TAG BUS
3002 000010 BB03 = BIT03 ; TAG BUS
3003 000004 BB02 = BIT02 ; TAG BUS
3004 000002 BB01 = BIT01 ; TAG BUS
3005 000001 BB00 = BIT00 ; TAG BUS
3006
3007
3008 ;RMR2 ERROR REGISTER 2
3009
3010 100000 BSE = BIT15 ; BAD SECTOR ERROR
3011 040000 SKI = BIT14 ; SEEK INCOMPLETE
3012 020000 OPE = BIT13 ; OPERATOR PLUG ERROR
3013 010000 IVC = BIT12 ; INVALID COMMAND ERROR
3014 004000 LSC = BIT11 ; LOSS OF SYSTEM CLOCK
3015 002000 LBC = BIT10 ; LOSS OF BIT CLOCK
3016 000200 DVC = BIT07 ; DEVICE CHECK
3017 000010 OPE = BIT03 ; DATA PARITY ERROR
3018 001567 XNUER2 = 001567 ; UNUSED BITS OF RMR2
3019
3020 .SBTTL PROGRAM MNEMONICS
3021
3022 100000 MSE = BIT15 ; MANUFACTURING DETECTED SECTOR ERROR
3023 040000 USE = BIT14 ; USER DETECTED SECTOR ERROR
3024
3025 .SBTTL RM03 REGISTER INDEX VALUES
3026
3027 000000 RMC51 = 00 ; CONTROL STATUS REGISTER
3028 000006 RMDA = 06 ; DISK ADDRESS REGISTER
3029 000012 RMD5 = 12 ; DRIVE STATUS REGISTER
3030 000014 RMR1 = 14 ; ERROR REGISTER 1
3031 000016 RMA5 = 16 ; ATTENTION SUMMARY REGISTER
3032 000020 RMLA = 20 ; LOOK AHEAD REGISTER
3033 000024 RMMR1 = 24 ; MAINTENANCE REGISTER
3034 000026 RMDT = 26 ; DRIVE TYPE REGISTER
3035 000030 RMSN = 30 ; SERIAL NUMBER REGISTER
3036 000032 RMOF = 32 ; OFFSET REGISTER
3037 000034 RMDC = 34 ; DESIRED CYLINDER REGISTER
3038 000036 RMHR = 36 ; HOLDING REGISTER
3039 000040 RMMR2 = 40 ; MAINTENANCE REGISTER 2
3040 000042 RMR2 = 42 ; ERROR REGISTER 2
3041 000044 RMEC1 = 44 ; ECC POSITION REGISTER
3042 000046 RMEC2 = 46 ; ECC PATTERN REGISTER

```

```

3043      000050      ILRG50 =      50      ; ILLEGAL REGISTER 50
3044      000052      ILRG52 =      52      ; ILLEGAL REGISTER 52
3045      000054      ILRG54 =      54      ; ILLEGAL REGISTER 54
3046      000056      ILRG56 =      56      ; ILLEGAL REGISTER 56
3047      000060      ILRG60 =      60      ; ILLEGAL REGISTER 60
3048      000062      ILRG62 =      62      ; ILLEGAL REGISTER 62
3049      000064      ILRG64 =      64      ; ILLEGAL REGISTER 64
3050      000066      ILRG66 =      66      ; ILLEGAL REGISTER 66
3051      000070      ILRG70 =      70      ; ILLEGAL REGISTER 70
3052      000072      ILRG72 =      72      ; ILLEGAL REGISTER 72
3053      000074      ILRG74 =      74      ; ILLEGAL REGISTER 74
3054      000076      ILRG76 =      76      ; ILLEGAL REGISTER 76
3055
3056
3057      000077      IDXMSK =      77      ; MASK FOR REGISTER INDEX NUMBER
3058
3059      .SBTTL      RH CONTROLLER REGISTER BIT DEFINITIONS
3060
3061      ;RMCS1      CONTROL STATUS REGISTER #1
3062
3063      100000      SC      =      BIT15      ; SPECIAL CONDITION-READ ONLY
3064      040000      TRE      =      BIT14      ; TRANSFER ERROR
3065      020000      MCPE     =      BIT13      ; MASSBUS CONTROL BUS PARITY
3066
3067      002000      PSEL     =      BIT10      ; ERROR-READ ONLY
3068      001000      A17      =      BIT09      ; PORT B SELECT
3069      000400      A16      =      BIT08      ; ADDRESS EXTENSION
3070      000200      RDY      =      BIT07      ; ADDRESS EXTENSION
3071      000100      IE       =      BIT06      ; READY-READ ONLY
3072
3073      ;RMCS2      RH CONTROL STATUS REGISTER #2
3074
3075      100000      DLT       =      BIT15      ; DATA LATE-READ ONLY
3076      040000      WCE       =      BIT14      ; WRITE CHECK ERROR-READ ONLY
3077      020000      UPE       =      BIT13      ; UNIBUS PARITY ERROR
3078      010000      NED       =      BIT12      ; NONEXISTANT DRIVE-READ ONLY
3079      004000      NEM       =      BIT11      ; NONEXISTANT MEMORY-READ ONLY
3080      002000      PGE       =      BIT10      ; PROGRAM ERROR-READ ONLY
3081      001000      MXF       =      BIT09      ; MISSED TRANSFER
3082      000400      MDPE     =      BIT08      ; MASSBUS DATA BUS PARITY
3083
3084      000200      OR        =      BIT07      ; ERROR-READ ONLY
3085      000100      IR        =      BIT06      ; OUTPUT READY-READ ONLY
3086      000040      CLR       =      BIT05      ; INPUT READY-READ ONLY
3087      000020      PAT       =      BIT04      ; CONTROLLER CLEAR
3088      000010      BAI       =      BIT03      ; PARITY TEST
3089
3090      000004      U2        =      BIT02      ; UNIBUS ADDRESS INCREMENT
3091      000002      U1        =      BIT01      ; INHIBIT
3092      000001      U0        =      BIT00      ; UNIT SELECT
3093
3094      ;UNIT SELECT MASK
3095      000007      UNTMSK =      7      ; UNIT SELECT MASK
3096
3097      ;RMCS3      RH70 CONTROL STATUS REGISTER #3
3098      100000      APE       =      BIT15      ; ADDRESS PARITY ERROR

```

```

3099      040000      DPEHI   =      BIT14      ; DATA PARITY ERROR HIGH WORD
3100      020000      DPELO   =      BIT13      ; DATA PARITY ERROR LOW WORD
3101      010000      WCEHI   =      BIT12      ; WRITE CHECK ERROR HIGH WORD
3102      004000      WCELO   =      BIT11      ; WRITE CHECK ERROR LOW WORD
3103      002000      DBL     =      BIT10      ; DOUBLE WORD TRANSFER
3104      000100      IE      =      BIT06      ; INTERRUPT ENABLE
3105      000010      IPCK3   =      BIT03      ; INVERT PARITY CHECK
3106      000004      IPCK2   =      BIT02      ; INVERT PARITY CHECK
3107      000002      IPCK1   =      BIT01      ; INVERT PARITY CHECK
3108      000001      IPCK0   =      BIT00      ; INVERT PARITY CHECK
3109      .SBTTL     RH CONTROLLER REGISTER INDEX VALUES
3110
3111      000000      RMCS1   =      00      ; CONTROL STATUS REGISTER
3112      000002      RMWC    =      02      ; WORD COUNT REGISTER
3113      000004      RMBA    =      04      ; BUS ADDRESS REGISTER
3114      000010      RMCS2   =      10      ; CONTROLLER STATUS REGISTER
3115      000022      RMDB    =      22      ; DATA BUFFER
3116      000050      RMBAE   =      50      ; BUS ADDRESS EXTENSION
3117      000052      RMCS3   =      52      ; CONTROL STATUS REGISTER #3
3118
3119      176700      ABASE   =      176700    ; UNIBUS ADDRESS
3120      120254      AVECT1  =      120254    ; UNIBUS VECTOR ADDRESS AND PRIORITY
3121
3122      .SBTTL     TRAP CATCHER
3123
3124
3125      000000      .=0
3126      ; *ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A "+2,HALT"
3127      ; *SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
3128      ; *LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
3129      .=174
3130      000174      000000      DISPREG: .WORD 0      ; SOFTWARE DISPLAY REGISTER
3131      000176      000000      SWREG:   .WORD 0      ; SOFTWARE SWITCH REGISTER
3132
3133      .SBTTL     ACT11 HOOKS
3134
3135      ; *****
3136      ; HOOKS REQUIRED BY ACT11
3137      $SVPC=.      ; SAVE PC
3138      .=46
3139      $ENDAD      ; ;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
3140      000046      057422
3141      .=52
3142      000052      000000      .WORD 0      ; ;2)SET LOC.52 TO ZERO
3143      000200      .=$SVPC      ; ; RESTORE PC
3144
3145      .SBTTL     STARTING ADDRESS
3146
3147      ; THE PROGRAM STARTS AT LOCATION 200
3148      = 200
3149      000200      000137      004542      JMP      START      ; JUMP TO START OF PROGRAM
3150
3151      .=1100
3152      .SBTTL     APT PARAMETER BLOCK
3153
3154      ; *****

```

```

3155
3156
3157      001100
3158      000024
3159 000024 000200
3160      000044
3161 000044 001100
3162      001100
3163
3164
3165
3166
3167 001100
3168 001100 000000
3169 001102 001222
3170 001104 000001
3171 001106 000002
3172 001110 000002
3173 001112 000042
3174      001114

```

```

;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
;*****
.SX=      ;;SAVE CURRENT LOCATION
=24      ;;SET POWER FAIL TO POINT TO START OF PROGRAM
200      ;;FOR APT START UP
=44      ;;POINT TO APT INDIRECT ADDRESS PNTR.
$APTHOR  ;;POINT TO APT HEADER BLOCK
=.SX     ;;RESET LOCATION COUNTER
;*****
;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
;INTERFACE SPEC.

$APTHD:
$HIBTS: .WORD 0      ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
$MADR:  .WORD $MAIL  ;;ADDRESS OF APT MAILBOX (BITS 0-15)
$STMT:  .WORD 1      ;;RUN TIME OF LONGEST TEST
$PASTM: .WORD 2      ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
$UNITM: .WORD 2      ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
          .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
TAGADR =

```

3175
3176
3177
3178
3179
3180
3181 001114
3182 001114
3183 001114 000000
3184 001116 000
3185 001117 000
3186 001120 000000
3187 001122 000000
3188 001124 000000
3189 001126 000000
3190 001130 000
3191 001131 001
3192 001132 000000
3193 001134 000000
3194 001136 000000
3195 001140 000000
3196 001142 000000
3197 001144 000000
3198 001146 000000
3199 001150 000
3200 001151 000
3201 001152 000000
3202 001154 177570
3203 001156 177570
3204 001160 177560
3205 001162 177562
3206 001164 177564
3207 001166 177566
3208 001170 000
3209 001171 002
3210 001172 012
3211 001173 000
3212 001174 000000
3213 001176 000000
3214 001200 000000
3215 001202 000000
3216 001204 000000
3217 001206 000000
3218 001210 000000
3219 001212 177607 000377
3220 001216 077
3221 001217 015
3222 001220 000012
3223
3224
3225
3226
3227
3228 001222
3229 001222 000000
3230 001224 000000

.SBTTL COMMON TAGS

; THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
; USED IN THE PROGRAM

.=TAGADR

\$CMTAG: .WORD 0

;; START OF COMMON TAGS

\$STNM: .BYTE 00

;; CONTAINS THE TEST NUMBER

\$ERFLG: .BYTE 00

;; CONTAINS ERROR FLAG

\$ICNT: .WORD 00

;; CONTAINS SUBTEST ITERATION COUNT

\$LPADR: .WORD 00

;; CONTAINS SCOPE LOOP ADDRESS

\$LPERR: .WORD 00

;; CONTAINS SCOPE RETURN FOR ERRORS

\$ERTTL: .WORD 00

;; CONTAINS TOTAL ERRORS DETECTED

\$ITEMB: .BYTE 00

;; CONTAINS ITEM CONTROL BYTE

\$ERMAX: .BYTE 1

;; CONTAINS MAX. ERRORS PER TEST

\$ERRPC: .WORD 00

;; CONTAINS PC OF LAST ERROR INSTRUCTION

\$GDADR: .WORD 00

;; CONTAINS ADDRESS OF 'GOOD' DATA

\$BDADR: .WORD 00

;; CONTAINS ADDRESS OF 'BAD' DATA

\$GDAT: .WORD 00

;; CONTAINS 'GOOD' DATA

\$BDAT: .WORD 00

;; CONTAINS 'BAD' DATA

.WORD 00

;; RESERVED--NOT TO BE USED

\$AUTOB: .BYTE 00

;; AUTOMATIC MODE INDICATOR

\$INTAG: .BYTE 00

;; INTERRUPT MODE INDICATOR

\$SWR: .WORD 0

;; ADDRESS OF SWITCH REGISTER

\$DISPLAY: .WORD 0

;; ADDRESS OF DISPLAY REGISTER

\$TKS: 177560

;; TTY KBD STATUS

\$TKB: 177562

;; TTY KBD BUFFER

\$TPS: 177564

;; TTY PRINTER STATUS REG. ADDRESS

\$TPB: 177566

;; TTY PRINTER BUFFER REG. ADDRESS

\$NULL: .BYTE 0

;; CONTAINS NULL CHARACTER FOR FILLS

\$FILLS: .BYTE 2

;; CONTAINS # OF FILLER CHARACTERS REQUIRED

\$FILLC: .BYTE 12

;; INSERT FILL CHARS. AFTER A "LINE FEED"

\$TPFLG: .BYTE 0

;; "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)

\$TMP0: .WORD 00

;; USER DEFINED

\$TMP1: .WORD 00

;; USER DEFINED

\$TMP2: .WORD 00

;; USER DEFINED

\$TMP3: .WORD 00

;; USER DEFINED

\$TMP4: .WORD 0

;; USER DEFINED

\$TIMES: 0

;; MAX. NUMBER OF ITERATIONS

\$ESCAPE: 0

;; ESCAPE ON ERROR ADDRESS

\$BELL: .ASCIZ <207><377><377>

;; CODE FOR BELL

\$QUES: .ASCII /?

;; QUESTION MARK

\$CRLF: .ASCII <15>

;; CARRIAGE RETURN

\$LF: .ASCIZ <12>

;; LINE FEED

.SBTTL APT MAILBOX-ETABLE

.EVEN

\$MAIL: .WORD 00

;; APT MAILBOX

\$MSGTY: .WORD 00

;; MESSAGE TYPE CODE

\$FATAL: .WORD 00

;; FATAL ERROR NUMBER

3231	001226	000000	\$TESTN: .WORD	ATESTN	:: TEST NUMBER
3232	001230	000000	\$PASS: .WORD	APASS	:: PASS COUNT
3233	001232	000000	\$DEVCT: .WORD	ADEVCT	:: DEVICE COUNT
3234	001234	000000	\$UNIT: .WORD	AUNIT	:: I/O UNIT NUMBER
3235	001236	000000	\$MSGAD: .WORD	AMSGAD	:: MESSAGE ADDRESS
3236	001240	000000	\$MSGLG: .WORD	AMSGLG	:: MESSAGE LENGTH
3237	001242		\$ETABLE:		:: APT ENVIRONMENT TABLE
3238	001242	000	\$ENV: .BYTE	AENV	:: ENVIRONMENT BYTE
3239	001243	000	\$ENVM: .BYTE	AENVM	:: ENVIRONMENT MODE BITS
3240	001244	000000	\$SWREG: .WORD	ASWREG	:: APT SWITCH REGISTER
3241	001246	000000	\$USWR: .WORD	AUSWR	:: USER SWITCHES
3242	001250	000000	\$CPUOP: .WORD	ACPUOP	:: CPU TYPE, OPTIONS
3243			*		BITS 15-11=CPU TYPE
3244			*		11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
3245			*		11/70=06, PQ=07, Q=10
3246			*		BIT 10=REAL TIME CLOCK
3247			*		BIT 9=FLOATING POINT PROCESSOR
3248			*		BIT 8=MEMORY MANAGEMENT
3249	001252	000	\$MAMS1: .BYTE	AMAMS1	:: HIGH ADDRESS, M.S. BYTE
3250	001253	000	\$MTYP1: .BYTE	AMTYP1	:: MEM. TYPE, BLK#1
3251			*		MEM. TYPE BYTE -- (HIGH BYTE)
3252			*		900 NSEC CORE=001
3253			*		300 NSEC BIPOLAR=002
3254			*		500 NSEC MOS=003
3255	001254	000000	\$MADR1: .WORD	AMADR1	:: HIGH ADDRESS, BLK#1
3256			*		MEM. LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
3257	001256	000	\$MAMS2: .BYTE	AMAMS2	:: HIGH ADDRESS, M.S. BYTE
3258	001257	000	\$MTYP2: .BYTE	AMTYP2	:: MEM. TYPE, BLK#2
3259	001260	000000	\$MADR2: .WORD	AMADR2	:: MEM. LAST ADDRESS, BLK#2
3260	001262	000	\$MAMS3: .BYTE	AMAMS3	:: HIGH ADDRESS, M.S. BYTE
3261	001263	000	\$MTYP3: .BYTE	AMTYP3	:: MEM. TYPE, BLK#3
3262	001264	000000	\$MADR3: .WORD	AMADR3	:: MEM. LAST ADDRESS, BLK#3
3263	001266	000	\$MAMS4: .BYTE	AMAMS4	:: HIGH ADDRESS, M.S. BYTE
3264	001267	000	\$MTYP4: .BYTE	AMTYP4	:: MEM. TYPE, BLK#4
3265	001270	000000	\$MADR4: .WORD	AMADR4	:: MEM. LAST ADDRESS, BLK#4
3266	001272	120254	\$VECT1: .WORD	AVECT1	:: INTERRUPT VECTOR#1 BUS PRIORITY#1
3267	001274	000000	\$VECT2: .WORD	AVECT2	:: INTERRUPT VECTOR#2 BUS PRIORITY#2
3268	001276	176700	\$BASE: .WORD	ABASE	:: BASE ADDRESS OF EQUIPMENT UNDER TEST
3269	001300	000000	\$DEVN: .WORD	ADEVN	:: DEVICE MAP
3270	001302	000000	\$CDW1: .WORD	ACDW1	:: CONTROLLER DESCRIPTION WORD#1
3271	001304	000000	\$CDW2: .WORD	ACDW2	:: CONTROLLER DESCRIPTION WORD#2
3272	001306	000000	\$DDW0: .WORD	ADDW0	:: DEVICE DESCRIPTOR WORD#0
3273	001310	000000	\$DDW1: .WORD	ADDW1	:: DEVICE DESCRIPTOR WORD#1
3274	001312	000000	\$DDW2: .WORD	ADDW2	:: DEVICE DESCRIPTOR WORD#2
3275	001314	000000	\$DDW3: .WORD	ADDW3	:: DEVICE DESCRIPTOR WORD#3
3276	001316	000000	\$DDW4: .WORD	ADDW4	:: DEVICE DESCRIPTOR WORD#4
3277	001320	000000	\$DDW5: .WORD	ADDW5	:: DEVICE DESCRIPTOR WORD#5
3278	001322	000000	\$DDW6: .WORD	ADDW6	:: DEVICE DESCRIPTOR WORD#6
3279	001324	000000	\$DDW7: .WORD	ADDW7	:: DEVICE DESCRIPTOR WORD#7
3280	001326		\$ETEND:		
3281			.MEXIT		
3282					
3283					
3284					
3285			.SBTTL REGISTER INPUT BUFFER		
3286					

3287
3288
3289
3290 001326 000000
3291 001330 000000
3292 001332 000000
3293 001334 000000
3294 001336 000000
3295 001340 000000
3296 001342 000000
3297 001344 000000
3298 001346 000000
3299 001350 000000
3300 001352 000000
3301 001354 000000
3302 001356 000000
3303 001360 000000
3304 001362 000000
3305 001364 000000
3306 001366 000000
3307 001370 000000
3308 001372 000000
3309 001374 000000
3310 001376 000000
3311 001400 000000
3312
3313
3314
3315
3316
3317
3318
3319
3320 001402 000000
3321 001404 000000
3322 001406 000000
3323 001410 000000
3324 001412 000000
3325 001414 000000
3326 001416 000000
3327 001420 000000
3328 001422 000000
3329 001424 000000
3330 001426 000000
3331 001430 000000
3332 001432 000000
3333 001434 000000
3334 001436 000000
3335 001440 000000
3336 001442 000000
3337 001444 000000
3338 001446 000000
3339 001450 000000
3340 001452 000000
3341 001454 000000
3342

: THE REGISTER INPUT BUFFER IS USED FOR STORING REGISTER
;CONTENTS AS THEY ARE READ FROM THE DEVICE.

RMCS1I: .WORD	: CONTROL, STATUS REGISTER #1
RMWC1: .WORD	: WORD COUNT REGISTER
RMBAI: .WORD	: BUS ADDRESS REGISTER
RMDAI: .WORD	: DISK ADDRESS REGISTER
RMCS2I: .WORD	: CONTROL, STATUS REGISTER #2
RMDS1: .WORD	: DRIVE STATUS REGISTER
RMER1I: .WORD	: ERROR REGISTER 1
RMASI: .WORD	: ATTENTION SUMMARY REGISTER
RMLAI: .WORD	: LOOK AHEAD REGISTER
RMOBI: .WORD	: DATA BUFFER
RMMR1I: .WORD	: MAINTENANCE REGISTER #1
RMOTI: .WORD	: DRIVE TYPE REGISTER
RMSNI: .WORD	: SERIAL NUMBER REGISTER
RMOFI: .WORD	: OFFSET REGISTER
RMDCI: .WORD	: DESIRED CYLINDER REGISTER
RMHRI: .WORD	: HOLDING REGISTER
RMMR2I: .WORD	: MAINTENANCE REGISTER #2
RMER2I: .WORD	: ERROR REGISTER 2
RMEC1I: .WORD	: ECC POSITION REGISTER
RMEC2I: .WORD	: ECC PATTERN REGISTER
RMBAEI: .WORD	: BUS ADDRESS EXTENSION REGISTER
RMCS3I: .WORD	: CONTROL, STATUS REGISTER #3

.SBTTL REGISTER OUTPUT BUFFER

: THE REGISTER OUTPUT BUFFER IS USED FOR ASSEMBLING DATA TO
;BE WRITTEN TO THE DEVICE.

RMCS10: .WORD	: CONTROL, STATUS REGISTER #1
RMWC0: .WORD	: WORD COUNT REGISTER
RMBAO: .WORD	: BUS ADDRESS REGISTER
RMDAO: .WORD	: DISK ADDRESS REGISTER
RMCS20: .WORD	: CONTROL, STATUS REGISTER #2
RMDS0: .WORD	: DRIVE STATUS REGISTER
RMER10: .WORD	: ERROR REGISTER 1
RMAS0: .WORD	: ATTENTION SUMMARY REGISTER
RMLA0: .WORD	: LOOK AHEAD REGISTER
RMOB0: .WORD	: DATA BUFFER
RMMR10: .WORD	: MAINTENANCE REGISTER #1
RMOT0: .WORD	: DRIVE TYPE REGISTER
RMSNO: .WORD	: SERIAL NUMBER REGISTER
RMOF0: .WORD	: OFFSET REGISTER
RMOC0: .WORD	: DESIRED CYLINDER REGISTER
RMHR0: .WORD	: HOLDING REGISTER
RMMR20: .WORD	: MAINTENANCE REGISTER #2
RMER20: .WORD	: ERROR REGISTER 2
RMEC10: .WORD	: ECC POSITION REGISTER
RMEC20: .WORD	: ECC PATTERN REGISTER
RMBAE0: .WORD	: BUS ADDRESS EXTENSION REGISTER
RMCS30: .WORD	: CONTROL, STATUS REGISTER #3

```

3343 .SBTTL TEST QUE
3344 ;
3345 ; EACH WORD OF THE TEST QUE CONTAINS THE DEVICE NUMBER IN
3346 ; THE LOW BYTE AND THE ATTENTION BIT IN THE HIGH BYTE. A ZERO
3347 ; WORD IS A BLANK AND REPRESENTS THE END OF THE QUE.
3348
3349 001456 000012 TSTQUE: .BLKW 10. ;TEST QUE
3350
3351 001502 172540 $LPCSR: .WORD 172540 ;KW11-P CONTROL + STATUS REGISTER
3352 001504 172542 $LPCSB: .WORD 172542 ;KW11-P COUNT SET BUFFER
3353 001506 000104 $LPVEC: .WORD 104 ;KW11-P INTERRUPT VECTOR
3354 001510 000106 .WORD 106
3355 001512 177546 $LLCSR: .WORD 177546 ;KW11-L CONTROL + STATUS REGISTER
3356 001514 000100 $LLVEC: .WORD 100 ;KW11-L INTERRUPT VECTOR
3357 001516 000102 .WORD 102
3358 001520 000000 $PSW: .WORD ;STORAGE FOR PRIORITY
3359 001522 000000 TIME: .WORD ;STORAGE FOR ELAPSED TIME
3360 001524 000000 WATCH: .WORD ;STORAGE FOR REMAINING TIME
3361 001526 000000 CLOCK: .WORD ;ADDRESS OF START CLOCK SUB
3362 001530 000000 STOP: .WORD ;ADDRESS OF STOP CLOCK SUB
3363
3364 ;PUT TAGS HERE
3365

```

3366
3367
3368
3369
3370
3371
3372
3373
3374
3375
3376
3377
3378
3379
3380
3381
3382

001532

.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

\$ERRTB:

3383				
3384			;ERROR 1	CANNOT CLEAR NED STATUS
3385				
3386	001532	066700		EMT1
3387	001534	074556		EHT1
3388	001536	074652		EDT1
3389	001540	074700		EFT1
3390				
3391				
3392			;ERROR 2	CANNOT READ OR WRITE ANY DEVICE REG WITHOUT NED
3393				
3394	001542	066706		EMT2
3395	001544	074562		EHT2
3396	001546	074654		EDT2
3397	001550	074702		EFT2
3398				
3399				
3400			;ERROR 3	CANNOT WRITE/READ ONES TO ANY DEVICE REGISTER
3401				
3402	001552	066734		EMT3
3403	001554	000000		0
3404	001556	000000		0
3405	001560	000000		0
3406				
3407				
3408			;ERROR 4	CANNOT CLEAR ANY DEVICE REGISTER BITS W/MASSBUS INIT
3409				
3410	001562	066754		EMT4
3411	001564	000000		0
3412	001566	000000		0
3413	001570	000000		0
3414				
3415				
3416			;ERROR 5	CANNOT WRITE/READ ZEROS TO ALL BIT POSITIONS
3417				
3418	001572	066776		EMT5
3419	001574	074566		EHT5
3420	001576	074656		EDT5
3421	001600	074704		EFT5
3422				
3423				
3424			;ERROR 6	CANNOT WRITE/READ ONES TO ALL BIT POSITIONS
3425				
3426	001602	067022		EMT6
3427	001604	074566		EHT5
3428	001606	074656		EDT5
3429	001610	074704		EFT5
3430				
3431				
3432			;ERROR 7	CANNOT WRITE/READ SHIFTING ONE BIT TO ALL BIT POSITIONS
3433			:	OF DEVICE REGISTERS
3434				
3435	001612	067044		EMT7
3436	001614	074572		EHT7
3437	001616	074656		EDT5
3438	001620	074704		EFT5

3439				
3440				
3441			;ERROR	10 REGISTER SELECT 1 APPEARS S-A-0
3442				
3443	001622	067066		EMT10
3444	001624	000000		0
3445	001626	000000		0
3446	001630	000000		0
3447				
3448				
3449			;ERROR	11 REGISTER SELECT 1 APPEARS S-A-1
3450				
3451	001632	067104		EMT11
3452	001634	000000		0
3453	001636	000000		0
3454	001640	000000		0
3455				
3456				
3457			;ERROR	12 REGISTER SELECT 2 APPEARS S-A-0
3458				
3459	001642	067122		EMT12
3460	001644	000000		0
3461	001646	000000		0
3462	001650	000000		0
3463				
3464				
3465			;ERROR	13 REGISTER SELECT 2 APPEARS S-A-1
3466				
3467	001652	067140		EMT13
3468	001654	000000		0
3469	001656	000000		0
3470	001660	000000		0
3471				
3472				
3473			;ERROR	14 REGISTER SELECT 4 APPEARS S-A-0
3474				
3475	001662	067156		EMT14
3476	001664	000000		0
3477	001666	000000		0
3478	001670	000000		0
3479				
3480				
3481			;ERROR	15 REGISTER SELECT 4 APPEARS S-A-1
3482				
3483	001672	067174		EMT15
3484	001674	000000		0
3485	001676	000000		0
3486	001700	000000		0
3487				
3488				
3489			;ERROR	16 REGISTER SELECT 8 APPEARS S-A-0
3490				
3491	001702	067212		EMT16
3492	001704	000000		0
3493	001706	000000		0
3494	001710	000000		0

3495				
3496				
3497				
3498				
3499	001712	067230		
3500	001714	000000		
3501	001716	000000		
3502	001720	000000		
3503				
3504				
3505				
3506				
3507	001722	067246		
3508	001724	074556		
3509	001726	074652		
3510	001730	074700		
3511				
3512				
3513				
3514				
3515	001732	067266		
3516	001734	074556		
3517	001736	074652		
3518	001740	074700		
3519				
3520				
3521				
3522				
3523	001742	067306		
3524	001744	074556		
3525	001746	074652		
3526	001750	074700		
3527				
3528				
3529				
3530				
3531	001752	067322		
3532	001754	074556		
3533	001756	074652		
3534	001760	074700		
3535				
3536				
3537				
3538				
3539	001762	067342		
3540	001764	074556		
3541	001766	074652		
3542	001770	074700		
3543				
3544				
3545				
3546				
3547	001772	067362		
3548	001774	074556		
3549	001776	074652		
3550	002000	074700		

;ERROR 17 REGISTER SELECT 8 APPEARS S-A-1

EMT17
0
0
0

;ERROR 20 CANT WRITE ZEROS RMDA

EMT20
EHT1
EDT1
EFT1

;ERROR 21 CANT WRITE ONES RMDA

EMT21
EHT1
EDT1
EFT1

;ERROR 22 BIT INTERFERENCE IN WRITING/READING RMDA

EMT22
EHT1
EDT1
EFT1

;ERROR 23 CANT WRITE ZEROS RMCS1

EMT23
EHT1
EDT1
EFT1

;ERROR 24 CANT WRITE ONES RMCS1

EMT24
EHT1
EDT1
EFT1

;ERROP 25 BIT INTERFERENCE IN WRITING/READING RMCS1

EMT25
EHT1
EDT1
EFT1

3551				
3552				
3553			;ERROR 26	MBA CLR L IS STUCK ACTIVE
3554				
3555	002002	067376	EMT26	
3556	002004	000000	0	
3557	002006	000000	0	
3558	002010	000000	0	
3559				
3560				
3561			;ERROR 27	CANNOT CLEAR RMER1-PAR,RMR,ILF,ILR
3562				
3563	002012	067430	EMT27	
3564	002014	074556	EHT1	
3565	002016	074652	EDT1	
3566	002020	074700	EFT1	
3567				
3568				
3569			;ERROR 30	CANNOT CLEAR RMER1-DCK,IAE,AOE,HCRC,HCE,ECH,WCF,FER
3570				
3571	002022	067442	EMT30	
3572	002024	074556	EHT1	
3573	002026	074652	EDT1	
3574	002030	074700	EFT1	
3575				
3576				
3577			;ERROR 31	CANNOT CLEAR RMER1-OPI,DTE
3578				
3579	002032	067456	EMT31	
3580	002034	074556	EHT1	
3581	002036	074652	EDT1	
3582	002040	074700	EFT1	
3583				
3584				
3585			;ERROR 32	CANNOT WRITE 0 IN RMER1-PAR,RMR,ILF,ILR
3586				
3587	002042	067472	EMT32	
3588	002044	074556	EHT1	
3589	002046	074652	EDT1	
3590	002050	074700	EFT1	
3591				
3592				
3593			;ERROR 33	CANNOT WRITE 0 IN RMER1-DCK,IAE,AOE,HCRC,HCE,ECH,WCF,FER
3594				
3595	002052	067506	EMT33	
3596	002054	074556	EHT1	
3597	002056	074652	EDT1	
3598	002060	074700	EFT1	
3599				
3600				
3601			;ERROR 34	CANNOT WRITE 0 IN RMER1-OPI,DTE
3602				
3603	002062	067524	EMT34	
3604	002064	074556	EHT1	
3605	002066	074652	EDT1	
3606	002070	074700	EFT1	

3607				
3608				
3609			;ERROR 35	CANNOT WRITE 1 IN RMER1
3610				
3611	002072	067542	EMT35	
3612	002074	074556	EHT1	
3613	002076	074652	EDT1	
3614	002100	074700	EFT1	
3615				
3616				
3617			;ERROR 36	CANNOT WRITE SHIFTING 1 IN RMER1
3618				
3619	002102	067556	EMT36	
3620	002104	074556	EHT1	
3621	002106	074652	EDT1	
3622	002110	074700	EFT1	
3623				
3624				
3625			;ERROR 37	CANNOT WRITE ZEROS IN RMDC
3626				
3627	002112	067572	EMT37	
3628	002114	074556	EHT1	
3629	002116	074652	EDT1	
3630	002120	074700	EFT1	
3631				
3632				
3633			;ERROR 40	CANNOT WRITE ONES IN RMDC
3634				
3635	002122	067606	EMT40	
3636	002124	074556	EHT1	
3637	002126	074652	EDT1	
3638	002130	074700	EFT1	
3639				
3640				
3641			;ERROR 41	BIT INTERFERENCE IN WRITING/READING RMDC
3642				
3643	002132	067626	EMT41	
3644	002134	074556	EHT1	
3645	002136	074652	EDT1	
3646	002140	074700	EFT1	
3647				
3648				
3649			;ERROR 42	CANNOT WRITE 1'S IN RMDC OR RMDA
3650				
3651	002142	067642	EMT42	
3652	002144	000000	0	
3653	002146	000000	0	
3654	002150	000000	0	
3655				
3656				
3657			;ERROR 43	CANNOT CLEAR RMCS1-FUNCTION CODE
3658				
3659	002152	067666	EMT43	
3660	002154	074556	EHT1	
3661	002156	074652	EDT1	
3662	002160	074700	EFT1	

3663				
3664				
3665			;ERROR 44	UNUSED BITS OF RMER2 NOT ZERO
3666				
3667	002162	067700		EMT44
3668	002164	074556		EHT1
3669	002166	074652		EDT1
3670	002170	074700		EFT1
3671				
3672				
3673			;ERROR 45	CANNOT CLEAR RMER2-OPE,IVC,LSC
3674				
3675	002172	067714		EMT45
3676	002174	074556		EHT1
3677	002176	074652		EDT1
3678	002200	074700		EFT1
3679				
3680				
3681			;ERROR 46	CANNOT CLEAR RMER2-LBC,DPE
3682				
3683	002202	067730		EMT46
3684	002204	074556		EHT1
3685	002206	074652		EDT1
3686	002210	074700		EFT1
3687				
3688				
3689			;ERROR 47	CANNOT WRITE ZEROS RMER2-OPE,IVC,LSC
3690				
3691	002212	067744		EMT47
3692	002214	074556		EHT1
3693	002216	074652		EDT1
3694	002220	074700		EFT1
3695				
3696				
3697			;ERROR 50	CANNOT WRITE ZEROS RMER2-LBC,DPE
3698				
3699	002222	067766		EMT50
3700	002224	074556		EHT1
3701	002226	074652		EDT1
3702	002230	074700		EFT1
3703				
3704				
3705			;ERROR 51	CANNOT WRITE ONES RMER2
3706				
3707	002232	070010		EMT51
3708	002234	074556		EHT1
3709	002236	074652		EDT1
3710	002240	074700		EFT1
3711				
3712				
3713			;ERROR 52	CANNOT WRITE SHIFTING ONES RMER2
3714				
3715	002242	070030		EMT52
3716	002244	074556		EHT1
3717	002246	074652		EDT1
3718	002250	074700		EFT1

3719				
3720				
3721			;ERROR 53	UNUSED BITS OF RMOF ARE NOT ZERO
3722				
3723	002252	070044		EMT53
3724	002254	074556		EHT1
3725	002256	074652		EDT1
3726	002260	074700		EFT1
3727				
3728				
3729			;ERROR 54	CANNOT WRITE ZEROS RMOF-FMT,ECI,HCI,OFD
3730				
3731	002262	070060		EMT54
3732	002264	074556		EHT1
3733	002266	074652		EDT1
3734	002270	074700		EFT1
3735				
3736				
3737			;ERROR 55	CANNOT WRITE ONES RMOF-FMT,ECI,HCI,OFD
3738				
3739	002272	070100		EMT55
3740	002274	074556		EHT1
3741	002276	074652		EDT1
3742	002300	074700		EFT1
3743				
3744				
3745			;ERROR 56	CANNOT WRITE SHIFTING ONES RMOF
3746				
3747	002302	070120		EMT56
3748	002304	074556		EHT1
3749	002306	074652		EDT1
3750	002310	074700		EFT1
3751				
3752				
3753			;ERROR 57	DEVICE IS NOT AN RMO3
3754				
3755	002312	070134		EMT57
3756	002314	074576		EHT57
3757	002316	074660		EDT57
3758	002320	074706		EFT57
3759				
3760				
3761			;ERROR 60	DEVICE AVAILABLE IS NOT SET
3762				
3763	002322	070150		EMT60
3764	002324	074556		EHT1
3765	002326	074652		EDT1
3766	002330	074700		EFT1
3767				
3768				
3769			;ERROR 61	CANNOT WRITE ZEROS RMHR
3770				
3771	002332	070164		EMT61
3772	002334	074556		EHT1
3773	002336	074652		EDT1
3774	002340	074700		EFT1

3775				
3776				
3777				
3778				
3779	002342	070204		
3780	002344	074556		
3781	002346	074652		
3782	002350	074700		
3783				
3784				
3785				
3786				
3787	002352	070224		
3788	002354	074556		
3789	002356	074652		
3790	002360	074700		
3791				
3792				
3793				
3794				
3795	002362	070240		
3796	002364	074556		
3797	002366	074652		
3798	002370	074700		
3799				
3800				
3801				
3802				
3803	002372	070252		
3804	002374	074602		
3805	002376	074662		
3806	002400	074710		
3807				
3808				
3809				
3810				
3811	002402	070266		
3812	002404	074602		
3813	002406	074662		
3814	002410	074710		
3815				
3816				
3817				
3818				
3819	002412	070302		
3820	002414	074556		
3821	002416	074652		
3822	002420	074700		
3823				
3824				
3825				
3826				
3827	002422	070320		
3828	002424	074556		
3829	002426	074652		
3830	002430	074700		

;ERROR 62 CANNOT WRITE ONES RMHR

EMT62
EHT1
EDT1
EFT1

;ERROR 63 CANNOT WRITE SHIFTING ONES RMHR

EMT63
EHT1
EDT1
EFT1

;ERROR 64 CANNOT CLEAR ILR STATUS

EMT64
EHT1
EDT1
EFT1

;ERROR 65 ILR ERROR SHOULD NOT BE SET

EMT65
EHT65
EDT65
EFT65

;ERROR 66 ILR ERROR SHOULD BE SET

EMT66
EHT65
EDT65
EFT65

;ERROR 67 CANNOT CLEAR PAR STATUS-DPE IS RESET

EMT67
EHT1
EDT1
EFT1

;ERROR 70 CANNOT CLEAR PAR AND DPE STATUS

EMT70
EHT1
EDT1
EFT1

3831				
3832				
3833			;ERROR 71	"PAR" ERROR SHOULD NOT BE SET-"PAT" IS OFF
3834				
3835	002432	070340	EMT71	
3836	002434	074606	EHT71	
3837	002436	074664	EDT71	
3838	002440	074712	EFT71	
3839				
3840				
3841			;ERROR 72	"PAR" ERROR SHOULD BE SET-"PAT" IS ON
3842				
3843	002442	070364	EMT72	
3844	002444	074606	EHT71	
3845	002446	074664	EDT71	
3846	002450	074712	EFT71	
3847				
3848				
3849			;ERROR 73	"MCPE" ERROR SHOULD NOT BE SET
3850				
3851	002452	070410	EMT73	
3852	002454	074606	EHT71	
3853	002456	074664	EDT71	
3854	002460	074712	EFT71	
3855				
3856				
3857			;ERROR 74	UNEXPECTED BUS TIMEOUT
3858				
3859	002462	070430	EMT74	
3860	002464	074612	EHT74	
3861	002466	074666	EDT74	
3862	002470	074714	EFT74	
3863				
3864				
3865			;ERROR 75	CANT CLEAR "DMD"
3866				
3867	002472	070440	EMT75	
3868	002474	074556	EHT1	
3869	002476	074652	EDT1	
3870	002500	074700	EFT1	
3871				
3872				
3873			;ERROR 76	CANT WRITE ZERO "DMD"
3874				
3875	002502	070452	EMT76	
3876	002504	074556	EHT1	
3877	002506	074652	EDT1	
3878	002510	074700	EFT1	
3879				
3880				
3881			;ERROR 77	CANT WRITE ONE "DMD"
3882				
3883	002512	070466	EMT77	
3884	002514	074556	EHT1	
3885	002516	074652	EDT1	
3886	002520	074700	EFT1	

3887			
3888			
3889			;ERROR 100 DMD SET BY WRONG BIT
3890			
3891	002522	070502	EMT100
3892	002524	074606	EHT71
3893	002526	074662	EDT65
3894	002530	074710	EFT65
3895			
3896			
3897			;ERROR 101 CANT CLEAR "MOL" IN DIAGNOSTIC MODE
3898			
3899	002532	070522	EMT101
3900	002534	074556	EHT1
3901	002536	074652	EDT1
3902	002540	074700	EFT1
3903			
3904			
3905			;ERROR 102 CANT SET "MOL" IN DIAGNOSTIC MODE
3906			
3907	002542	070542	EMT102
3908	002544	074556	EHT1
3909	002546	074652	EDT1
3910	002550	074700	EFT1
3911			
3912			
3913			;ERROR 103 "MUR" SET BY WRONG BIT
3914			
3915	002552	070562	EMT103
3916	002554	074606	EHT71
3917	002556	074662	EDT65
3918	002560	074710	EFT65
3919			
3920			
3921			;ERROR 104 CANT RESET "WRL" IN DIAGNOSTIC MODE
3922			
3923	002562	070606	EMT104
3924	002564	074556	EHT1
3925	002566	074652	EDT1
3926	002570	074700	EFT1
3927			
3928			
3929			;ERROR 105 CANT SET "WRL" IN DIAGNOSTIC MODE
3930			
3931	002572	070626	EMT105
3932	002574	074556	EHT1
3933	002576	074652	EDT1
3934	002600	074700	EFT1
3935			
3936			
3937			;ERROR 106 "MWP" SET BY WRONG BIT
3938			
3939	002602	070646	EMT106
3940	002604	074606	EHT71
3941	002606	074662	EDT65
3942	002610	074710	EFT65

3943				
3944				
3945			;ERROR 107	CANT RESET "DVC" USING "MDVC"
3946				
3947	002612	070672	EMT107	
3948	002614	074556	EHT1	
3949	002616	074652	EDT1	
3950	002620	074700	EFT1	
3951				
3952				
3953			;ERROR 110	"DVC" IS RESET BUT "UNS" IS SET
3954				
3955	002622	070712	EMT110	
3956	002624	074556	EHT1	
3957	002626	074652	EDT1	
3958	002630	074700	EFT1	
3959				
3960				
3961			;ERROR 111	"DVC" IS SET BUT "UNS" IS NOT SET
3962				
3963	002632	070734	EMT111	
3964	002634	074556	EHT1	
3965	002636	074652	EDT1	
3966	002640	074700	EFT1	
3967				
3968				
3969			;ERROR 112	CANT SET "DVC" USING MDVC"
3970				
3971	002642	070754	EMT112	
3972	002644	074556	EHT1	
3973	002646	074652	EDT1	
3974	002650	074700	EFT1	
3975				
3976				
3977			;ERROR 113	"DVC" IS RESET BUT "UNS" IS SET
3978				
3979	002652	070774	EMT113	
3980	002654	074556	EHT1	
3981	002656	074652	EDT1	
3982	002660	074700	EFT1	
3983				
3984				
3985			;ERROR 114	"DVC" IS SET BUT "UNS" IS NOT SET
3986				
3987	002662	071020	EMT114	
3988	002664	074556	EHT1	
3989	002666	074652	EDT1	
3990	002670	074700	EFT1	
3991				
3992				
3993			;ERROR 115	"MDF" IS SET BY WRONG BIT
3994				
3995	002672	071042	EMT115	
3996	002674	074616	EHT115	
3997	002676	074670	EDT115	
3998	002700	074716	EFT115	

3999				
4000				
4001			;ERROR 116	CANT RESET "SKI" USING "MSER"
4002				
4003	002702	071066	EMT116	
4004	002704	074556	EHT1	
4005	002706	074652	EDT1	
4006	002710	074700	EFT1	
4007				
4008				
4009			;ERROR 117	CANT SET "SKI" USING "MSER"
4010				
4011	002712	071106	EMT117	
4012	002714	074556	EHT1	
4013	002716	074652	EDT1	
4014	002720	074700	EFT1	
4015				
4016				
4017			;ERROR 120	"SKI" SET BY WRONG BIT
4018				
4019	002722	071126	EMT120	
4020	002724	074616	EHT115	
4021	002726	074670	EDT115	
4022	002730	074716	EFT115	
4023				
4024				
4025			;ERROR 121	CANT RESET "PIP" USING "MOC"
4026				
4027	002732	071152	EMT121	
4028	002734	074556	EHT1	
4029	002736	074652	EDT1	
4030	002740	074700	EFT1	
4031				
4032				
4033			;ERROR 122	CANT SET "PIP" USING "MOC"
4034				
4035	002742	071172	EMT122	
4036	002744	074556	EHT1	
4037	002746	074652	EDT1	
4038	002750	074700	EFT1	
4039				
4040				
4041			;ERROR 123	"MOC" SET BY WRONG BIT
4042				
4043	002752	071212	EMT123	
4044	002754	074616	EHT115	
4045	002756	074670	EDT115	
4046	002760	074716	EFT115	
4047				
4048				
4049			;ERROR 124	CANT CLEAR "EBL"
4050				
4051	002762	071236	EMT124	
4052	002764	074556	EHT1	
4053	002766	074652	EDT1	
4054	002770	074700	EFT1	

4055				
4056				
4057			;ERROR 125	"EBL" NOT ZERO IN DIAGNOSTIC MODE
4058				
4059	002772	071254	EMT125	
4060	002774	074556	EHT1	
4061	002776	074652	EDT1	
4062	003000	074700	EFT1	
4063				
4064				
4065			;ERROR 126	CANT SET "EBL" USING "DEBL"
4066				
4067	003002	071274	EMT126	
4068	003004	074556	EHT1	
4069	003006	074652	EDT1	
4070	003010	074700	EFT1	
4071				
4072				
4073			;ERROR 127	"DEBL" SET BY WRONG BIT
4074				
4075	003012	071312	EMT127	
4076	003014	074616	EHT115	
4077	003016	074670	EDT115	
4078	003020	074716	EFT115	
4079				
4080				
4081			;ERROR 130	"LS" NOT CORRECT ACCORDING TO RMDA
4082				
4083	003022	071336	EMT130	
4084	003024	074622	EHT130	
4085	003026	074672	EDT130	
4086	003030	074720	EFT130	
4087				
4088				
4089			;ERROR 131	"LST" NOT CORRECT ACCORDING TO RMDA
4090				
4091	003032	071354	EMT131	
4092	003034	074622	EHT130	
4093	003036	074672	EDT130	
4094	003040	074720	EFT130	
4095				
4096				
4097			;ERROR 132	CANNOT INCREMENT SECTOR ADDRESS USING "DEBL"
4098				
4099	003042	071372	EMT132	
4100	003044	074626	EHT132	
4101	003046	074674	EDT132	
4102	003050	074722	EFT132	
4103				
4104				
4105			;ERROR 133	CANNOT INCREMENT TRACK ADDRESS USING "DEBL"
4106				
4107	003052	071412	EMT133	
4108	003054	074626	EHT132	
4109	003056	074674	EDT132	
4110	003060	074722	EFT132	

4111				
4112				
4113			;ERROR 134	UNUSED BITS OF RMDC NOT ZERO
4114				
4115	003062	071432		EMT134
4116	003064	074556		EHT1
4117	003066	074652		EDT1
4118	003070	074700		EFT1
4119				
4120				
4121			;ERROR 135	"VV" NOT RESET BY UNIT READY
4122				
4123	003072	071446		EMT135
4124	003074	074556		EHT1
4125	003076	074652		EDT1
4126	003100	074700		EFT1
4127				
4128				
4129			;ERROR 136	SERIAL NUMBER IS INCONSISTENT
4130				
4131	003102	071464		EMT136
4132	003104	074556		EHT1
4133	003106	074652		EDT1
4134	003110	074700		EFT1
4135				
4136				
4137			;ERROR 137	CANT CLEAR "GO" BIT
4138				
4139	003112	071476		EMT137
4140	003114	074556		EHT1
4141	003116	074652		EDT1
4142	003120	074700		EFT1
4143				
4144				
4145			;ERROR 140	CANT INCREMENT CYLINDER USING "DEBL"
4146				
4147	003122	071514		EMT140
4148	003124	074626		EHT132
4149	003126	074674		EDT132
4150	003130	074722		EFT132
4151				
4152				
4153			;ERROR 141	CANT RESET "LBT" BY WRITING RMDA
4154				
4155	003132	071534		EMT141
4156	003134	074556		EHT1
4157	003136	074652		EDT1
4158	003140	074700		EFT1
4159				
4160				
4161			;ERROR 142	CANT SET "LBT" USING "DEBL"
4162				
4163	003142	071550		EMT142
4164	003144	074556		EHT1
4165	003146	074652		EDT1
4166	003150	074700		EFT1

4167				
4168				
4169			:ERROR 143	CANT READ ZERO FROM COMP ERROR
4170				
4171	003152	071566	EMT143	
4172	003154	074556	EHT1	
4173	003156	074652	EDT1	
4174	003160	074700	EFT1	
4175				
4176				
4177			;ERROR 144	CANT SET COMP ERROR WITH RMER1 OR RMER2
4178				
4179	003162	071602	EMT144	
4180	003164	074556	EHT1	
4181	003166	074652	EDT1	
4182	003170	074700	EFT1	
4183				
4184				
4185			;ERROR 145	COMP ERROR DID NOT SET
4186				
4187	003172	071624	EMT145	
4188	003174	074632	EHT145	
4189	003176	074672	EDT130	
4190	003200	074720	EFT130	
4191				
4192				
4193			;ERROR 146	CANT SET "GO" BIT
4194				
4195	003202	071646	EMT146	
4196	003204	074556	EHT1	
4197	003206	074652	EDT1	
4198	003210	074700	EFT1	
4199				
4200				
4201			:ERROR 147	CANT READ A ONE FROM "TST"
4202				
4203	003212	071664	EMT147	
4204	003214	074556	EHT1	
4205	003216	074652	EDT1	
4206	003220	074700	EFT1	
4207				
4208				
4209			;ERROR 150	"TST" IS INCORRECT FOR THE FUNCTION CODE
4210				
4211	003222	071676	EMT150	
4212	003224	074636	EHT150	
4213	003226	074670	EDT115	
4214	003230	074716	EFT115	
4215				
4216				
4217			;ERROR 151	CANT SET THE "GO" BIT
4218				
4219	003232	071720	EMT151	
4220	003234	074556	EHT1	
4221	003236	074652	EDT1	
4222	003240	074700	EFT1	

4223				
4224				
4225			;ERROR 152	"DRY" NOT THE COMPLEMENT OF "GO"
4226				
4227	003242	071732	EMT152	
4228	003244	074556	EHT1	
4229	003246	074652	EDT1	
4230	003250	074700	EFT1	
4231				
4232				
4233			;ERROR 153	"GO" RESET EARLY
4234				
4235	003252	071746	EMT153	
4236	003254	074556	EHT1	
4237	003256	074652	EDT1	
4238	003260	074700	EFT1	
4239				
4240				
4241			;ERROR 154	"GO" DIDNT RESET ON TIME
4242				
4243	003262	071766	EMT154	
4244	003264	074556	EHT1	
4245	003266	074652	EDT1	
4246	003270	074700	EFT1	
4247				
4248				
4249			;ERROR 155	CANT CLEAR CONTINUE
4250				
4251	003272	072006	EMT155	
4252	003274	074556	EHT1	
4253	003276	074652	EDT1	
4254	003300	074700	EFT1	
4255				
4256				
4257			;ERROR 156	CONTINUE IS INCORRECT FOR THE FUNCTION CODE
4258				
4259	003302	072024	EMT156	
4260	003304	074636	EHT150	
4261	003306	074670	EDT115	
4262	003310	074716	EFT115	
4263				
4264				
4265			;ERROR 157	CANT CLEAR IVC
4266				
4267	003312	072046	EMT157	
4268	003314	074556	EHT1	
4269	003316	074652	EDT1	
4270	003320	074700	EFT1	
4271				
4272				
4273			;ERROR 160	IVC IS INCORRECT FOR THE FUNCTION CODE
4274				
4275	003322	072064	EMT160	
4276	003324	074636	EHT150	
4277	003326	074670	EDT115	
4278	003330	074716	EFT115	

4279				
4280				
4281				
4282				
4283	003332	072114		
4284	003334	074556		
4285	003336	074652		
4286	003340	074700		
4287				
4288				
4289				
4290				
4291	003342	072132		
4292	003344	074556		
4293	003346	074652		
4294	003350	074700		
4295				
4296				
4297				
4298				
4299	003352	072150		
4300	003354	074556		
4301	003356	074652		
4302	003360	074700		
4303				
4304				
4305				
4306				
4307	003362	072172		
4308	003364	074556		
4309	003366	074652		
4310	003370	074700		
4311				
4312				
4313				
4314				
4315	003372	072214		
4316	003374	000000		
4317	003376	000000		
4318	003400	000000		
4319				
4320				
4321				
4322				
4323	003402	072240		
4324	003404	074556		
4325	003406	074652		
4326	003410	074700		
4327				
4328				
4329				
4330				
4331	003412	072260		
4332	003414	074556		
4333	003416	074652		
4334	003420	074700		

;ERROR 161 CANT CLEAR LSC

EMT161
EHT1
EDT1
EFT1

;ERROR 162 CANT SET LSC

EMT162
EHT1
EDT1
EFT1

;ERROR 163 COMMAND DECODE WAS ENABLED WITH COMP ERROR SET

EMT163
EHT1
EDT1
EFT1

;ERROR 164 COMMAND DECODE WAS ENABLED WITH COMP ERROR SET

EMT164
EHT1
EDT1
EFT1

;ERROR 165 DECODE DOES NOT SET

EMT165
0
0
0

;ERROR 166 CANT CLEAR OCCUPIED

EMT166
EHT1
EDT1
EFT1

;ERROR 167 ILF SET WITHOUT GO BIT

EMT167
EHT1
EDT1
EFT1

4335			
4336			
4337			;ERROR 170 CANT SET VOLUME VALID
4338			
4339	003422	072302	EMT170
4340	003424	074636	EHT150
4341	003426	074670	EDT115
4342	003430	074716	EFT115
4343			
4344			
4345			;ERROR 171 ILF IS INCORRECT
4346			
4347	003432	072314	EMT171
4348	003434	074636	EHT150
4349	003436	074670	EDT115
4350	003440	074716	EFT115
4351			
4352			
4353			;ERROR 172 CANT SET OFFSET DIRECTION BIT
4354			
4355	003442	072330	EMT172
4356	003444	074556	EHT1
4357	003446	074652	EDT1
4358	003450	074700	EFT1
4359			
4360			
4361			;ERROR 173 OCCUPIED IS INCORRECT FOR FUNCTION CODE
4362			
4363	003452	072346	EMT173
4364	003454	074636	EHT150
4365	003456	074670	EDT115
4366	003460	074716	EFT115
4367			
4368			
4369			;ERROR 174 READ IN PRESET DIDNT CLEAR RMDA, RMDC OR RMOF
4370			
4371	003462	072370	EMT174
4372	003464	000000	0
4373	003466	000000	0
4374	003470	000000	0
4375			
4376			
4377			;ERROR 175 READ IN PRESET DIDNT CLEAR RMOF
4378			
4379	003472	072416	EMT175
4380	003474	074556	EHT1
4381	003476	074652	EDT1
4382	003500	074700	EFT1
4383			
4384			
4385			;ERROR 176 READ IN PRESET DIDNT CLEAR RMDA
4386			
4387	003502	072434	EMT176
4388	003504	074556	EHT1
4389	003506	074652	EDT1
4390	003510	074700	EFT1

4391			
4392			
4393			;ERROR 177 READ IN PRESET DIDNT CLEAR RMDC
4394			
4395	003512	072452	EMT177
4396	003514	074556	EHT1
4397	003516	074652	EDT1
4398	003520	074700	EFT1
4399			
4400			
4401			;ERROR 200 CANT SET OFFSET MODE BY OFFSET COMMAND
4402			
4403	003522	072470	EMT200
4404	003524	074556	EHT1
4405	003526	074652	EDT1
4406	003530	074700	EFT1
4407			
4408			
4409			;ERROR 201 CANT RESET OFFSET MODE BY RTC COMMAND
4410			
4411	003532	072506	EMT201
4412	003534	074556	EHT1
4413	003536	074652	EDT1
4414	003540	074700	EFT1
4415			
4416			
4417			;ERROR 202 CANT RESET OFD BY RTC COMMAND
4418			
4419	003542	072524	EMT202
4420	003544	074556	EHT1
4421	003546	074652	EDT1
4422	003550	074700	EFT1
4423			
4424			
4425			;ERROR 203 CANT RESET OM BY RMDC
4426			
4427	003552	072542	EMT203
4428	003554	074556	EHT1
4429	003556	074652	EDT1
4430	003560	074700	EFT1
4431			
4432			
4433			;ERROR 204 CANT RESET OM BY EBL
4434			
4435	003562	072564	EMT204
4436	003564	074556	EHT1
4437	003566	074652	EDT1
4438	003570	074700	EFT1
4439			
4440			
4441			;ERROR 205 RUN AND GO NOT CORRECT FOR FUNCTION CODE
4442			
4443	003572	072604	EMT205
4444	003574	074636	EHT150
4445	003576	074670	EDT115
4446	003600	074716	EFT115

4447			
4448			
4449			;ERROR 206 CANT SET IAE ERROR
4450			
4451	003602	072624	EMT206
4452	003604	000000	0
4453	003606	000000	0
4454	003610	000000	0
4455			
4456			
4457			;ERROR 207 IAE IS INCORRECT FOR FUNCTION CODE
4458			
4459	003612	072654	EMT207
4460	003614	074636	EHT150
4461	003616	074670	EDT115
4462	003620	074716	EFT115
4463			
4464			
4465			;ERROR 210 IAE IS INCORRECT FOR RMDA
4466			
4467	003622	072670	EMT210
4468	003624	074616	EHT115
4469	003626	074670	EDT115
4470	003630	074716	EFT115
4471			
4472			
4473			;ERROR 211 IAE IS INCORRECT FOR RMDC
4474			
4475	003632	072706	EMT211
4476	003634	074616	EHT115
4477	003636	074670	EDT115
4478	003640	074716	EFT115
4479			
4480			
4481			;ERROR 212 CANT SET AOE
4482			
4483	003642	072724	EMT212
4484	003644	074556	EHT1
4485	003646	074652	EDT1
4486	003650	074700	EFT1
4487			
4488			
4489			;ERROR 213 RMR SET WHEN WRITING RMAS OR RMCS
4490			
4491	003652	072736	EMT213
4492	003654	074642	EHT213
4493	003656	074670	EDT115
4494	003660	074716	EFT115
4495			
4496			
4497			;ERROR 214 CANT SET RMR
4498			
4499	003662	072766	EMT214
4500	003664	074642	EHT213
4501	003666	074670	EDT115
4502	003670	074716	EFT115

4503			
4504			
4505			;ERROR 215 DRQ IS 0 AND PGM IS 1
4506			
4507	003672	073000	EMT215
4508	003674	074556	EHT1
4509	003676	074652	EDT1
4510	003700	074700	EFT1
4511			
4512			
4513			;ERROR 216 DVA IS NOT SET
4514			
4515	003702	073020	EMT216
4516	003704	074556	EHT1
4517	003706	074652	EDT1
4518	003710	074700	EFT1
4519			
4520			
4521			;ERROR 217 DPR IS NOT SET
4522			
4523	003712	073034	EMT217
4524	003714	074556	EHT1
4525	003716	074652	EDT1
4526	003720	074700	EFT1
4527			
4528			
4529			;ERROR 220 CANT SET PORT REQUEST BY READING RMCS1
4530			
4531	003722	073050	EMT220
4532	003724	074646	EHT220
4533	003726	074676	EDT220
4534	003730	074724	EFT220
4535			
4536			
4537			;ERROR 221 CANT SET PORT REQUEST BY WRITING RMAS
4538			
4539	003732	073066	EMT221
4540	003734	074646	EHT220
4541	003736	074676	EDT220
4542	003740	074724	EFT220
4543			
4544			
4545			;ERROR 222 CANT SET PORT REQUEST BY WRITING RMDA
4546			
4547	003742	073104	EMT222
4548	003744	074646	EHT220
4549	003746	074676	EDT220
4550	003750	074724	EFT220
4551			
4552			
4553			;ERROR 223 CANT RESET PORT REQUEST BY RELEASE COMMAND
4554			
4555	003752	073122	EMT223
4556	003754	074646	EHT220
4557	003756	074676	EDT220
4558	003760	074724	EFT220

4559			
4560			
4561			;ERROR 224 CANT CLEAR ATA BY RMAS
4562			
4563	003762	073140	EMT224
4564	003764	074556	EHT1
4565	003766	074652	EDT1
4566	003770	074700	EFT1
4567			
4568			
4569			;ERROR 225 ATA IS RESET BUT RMAS NOT ZERO
4570			
4571	003772	073160	EMT225
4572	003774	074556	EHT1
4573	003776	074652	EDT1
4574	004000	074700	EFT1
4575			
4576			
4577			;ERROR 226 CANT RESET ATA BY GO
4578			
4579	004002	073202	EMT226
4580	004004	074556	EHT1
4581	004006	074652	EDT1
4582	004010	074700	EFT1
4583			
4584			
4585			;ERROR 227 ATA NOT SET BY UNIT READY
4586			
4587	004012	073220	EMT227
4588	004014	074556	EHT1
4589	004016	074652	EDT1
4590	004020	074700	EFT1
4591			
4592			
4593			;ERROR 230 ATA NOT SET BY UNIT READY
4594			
4595	004022	073236	EMT230
4596	004024	074556	EHT1
4597	004026	074652	EDT1
4598	004030	074700	EFT1
4599			
4600			
4601			;ERROR 231 ATA NOT SET BY COMP ERROR
4602			
4603	004032	073254	EMT231
4604	004034	074556	EHT1
4605	004036	074652	EDT1
4606	004040	074700	EFT1
4607			
4608			
4609			;ERROR 232 ATA SET/DID NOT SET WHEN REGISTER WRITTEN
4610			WHILE COMP ERROR WAS SET
4611			
4612	004042	073270	EMT232
4613	004044	074642	EHT213
4614	004046	074670	EDT115

4615	004050	074716	EFT115
4616			
4617			
4618			;ERROR 233 ATA NOT SET BY COMMAND SEQUENCER
4619			
4620	004052	073312	EMT233
4621	004054	074636	EHT150
4622	004056	074670	EDT115
4623	004060	074716	EFT115
4624			
4625			
4626			;ERROR 234 WLE INCORRECT ACCORDING TO FUNCTION CODE
4627			
4628	004062	073334	EMT234
4629	004064	074636	EHT150
4630	004066	074670	EDT115
4631	004070	074716	EFT115
4632			
4633			
4634			;ERROR 235 CANT CLEAR EXCEPTION
4635			
4636	004072	073362	EMT235
4637	004074	074556	EHT1
4638	004076	074652	EDT1
4639	004100	074700	EFT1
4640			
4641			
4642			;ERROR 236 CANT SET EXCEPTION
4643			
4644	004102	073402	EMT236
4645	004104	074616	EHT115
4646	004106	074670	EDT115
4647	004110	074716	EFT115
4648			
4649			
4650			;ERROR 237 CANT CLEAR OPI
4651			
4652	004112	073434	EMT237
4653	004114	074556	EHT1
4654	004116	074652	EDT1
4655	004120	074700	EFT1
4656			
4657			
4658			;ERROR 240 CANT SET OPI
4659			
4660	004122	073434	EMT237
4661	004124	074636	EHT150
4662	004126	074670	EDT115
4663	004130	074716	EFT115
4664			
4665			
4666			;ERROR 241 OPI NOT SET DURING RECALIBRATE
4667			
4668	004132	073500	EMT241
4669	004134	074556	EHT1
4670	004136	074652	EDT1

4671	004140	074700	EFT1
4672			
4673			
4674			;ERROR 242 RECALIBRATE DID NOT ABORT WHEN DRIVE FAULT SET
4675			
4676	004142	073524	EMT242
4677	004144	074556	EHT1
4678	004146	074652	EDT1
4679	004150	074700	EFT1
4680			
4681			
4682			;ERROR 243 OPI SHOULD HAVE SET BECAUSE ON CYLINDER NEVER
4683			DROPPED DURING RECALIBRATE
4684			
4685	004152	073550	EMT243
4686	004154	074556	EHT1
4687	004156	074652	EDT1
4688	004160	074700	EFT1
4689			
4690			
4691			;ERROR 244 ATA NOT SET DURING RECALIBRATE
4692			
4693	004162	073574	EMT244
4694	004164	074556	EHT1
4695	004166	074652	EDT1
4696	004170	074700	EFT1
4697			
4698			
4699			;ERROR 245 GO RESET EARLY DURING RECALIBRATE
4700			
4701	004172	073612	EMT245
4702	004174	074556	EHT1
4703	004176	074652	EDT1
4704	004200	074700	EFT1
4705			
4706			
4707			;ERROR 246 GO NOT RESET DURING RECALIBRATE
4708			
4709	004202	073630	EMT246
4710	004204	074556	EHT1
4711	004206	074652	EDT1
4712	004210	074700	EFT1
4713			
4714			
4715			;ERROR 247 INCORRECT TAG BUS DURING RECALIBRATE
4716			
4717	004212	073654	EMT247
4718	004214	074556	EHT1
4719	004216	074652	EDT1
4720	004220	074700	EFT1
4721			
4722			
4723			;ERROR 250 OPI SHOULD HAVE SET DURING SEEK BECAUSE UNIT
4724			READY DROPPED
4725			
4726	004222	073672	EMT250

4727	004224	074556	EHT1
4728	004226	074652	EDT1
4729	004230	074700	EFT1
4730			
4731			
4732			;ERROR 251 SEEK DID NOT ABORT WHEN DRIVE FAULT SET
4733			
4734	004232	073716	EMT251
4735	004234	074556	EHT1
4736	004236	074652	EDT1
4737	004240	074700	EFT1
4738			
4739			
4740			;ERROR 252 OPI SHOULD HAVE SET BECAUSE ON CYLINDER NEVER
4741			; DROPPED DURING SEEK
4742			
4743	004242	073742	EMT252
4744	004244	074556	EHT1
4745	004246	074652	EDT1
4746	004250	074700	EFT1
4747			
4748			
4749			;ERROR 253 ATA NOT SET DURING SEEK
4750			
4751	004252	073766	EMT253
4752	004254	074556	EHT1
4753	004256	074652	EDT1
4754	004260	074700	EFT1
4755			
4756			
4757			;ERROR 254 GO RESFT EARLY DURING SEEK
4758			
4759	004262	074004	EMT254
4760	004264	074556	EHT1
4761	004266	074652	EDT1
4762	004270	074700	EFT1
4763			
4764			
4765			;ERROR 255 GO DID NOT RESET DURING SEEK
4766			
4767	004272	074022	EMT255
4768	004274	074556	EHT1
4769	004276	074652	EDT1
4770	004300	074700	EFT1
4771			
4772			
4773			;ERROR 256 INCORRECT TAG BUS DURING SEEK
4774			
4775	004302	074046	EMT256
4776	004304	074556	EHT1
4777	004306	074652	EDT1
4778	004310	074700	EFT1
4779			
4780			
4781			;ERROR 257 OPI NOT SET DURING SEARCH
4782			

4783	004312	074064	EMT257
4784	004314	074556	EHT1
4785	004316	074652	EDT1
4786	004320	074700	EFT1
4787			
4788			
4789			;ERROR 260 SEARCH DID NOT ABORT WHEN DRIVE FAULT SET
4790			
4791	004322	074110	EMT260
4792	004324	074556	EHT1
4793	004326	074652	EDT1
4794	004330	074700	EFT1
4795			
4796			
4797			;ERROR 261 OPI SHOULD HAVE SET BECAUSE ON CYLINDER NEVER
4798			; DROPPED DURING SEARCH
4799			
4800	004332	074134	EMT261
4801	004334	074556	EHT1
4802	004336	074652	EDT1
4803	004340	074700	EFT1
4804			
4805			
4806			;ERROR 262 ATA NOT SET DURING SEARCH
4807			
4808	004342	074160	EMT262
4809	004344	074556	EHT1
4810	004346	074652	EDT1
4811	004350	074700	EFT1
4812			
4813			
4814			;ERROR 263 GO RESET EARLY DURING SEARCH
4815			
4816	004352	074176	EMT263
4817	004354	074556	EHT1
4818	004356	074652	EDT1
4819	004360	074700	EFT1
4820			
4821			
4822			;ERROR 264 GO DID NOT RESET DURING SEARCH
4823			
4824	004362	074214	EMT264
4825	004364	074556	EHT1
4826	004366	074652	EDT1
4827	004370	074700	EFT1
4828			
4829			
4830			;ERROR 265 SEARCH ENABLE DIDNT SET DURING SEARCH
4831			
4832	004372	074240	EMT265
4833	004374	074556	EHT1
4834	004376	074652	EDT1
4835	004400	074700	EFT1
4836			
4837			
4838			;ERROR 266 INCORRECT TAG BUS DURING SEARCH

4839			
4840	004402	074256	EMT266
4841	004404	074556	EHT1
4842	004406	074652	EDT1
4843	004410	074700	EFT1
4844			
4845			
4846			;ERROR 267 OPI NOT SET BY SEARCH TIMEOUT
4847			
4848	004412	074274	EMT267
4849	004414	074556	EHT1
4850	004416	074652	EDT1
4851	004420	074700	EFT1
4852			
4853			
4854			;ERROR 270 OPI NOT SET DURING DATA COMMAND
4855			
4856	004422	074310	EMT270
4857	004424	074556	EHT1
4858	004426	074652	EDT1
4859	004430	074700	EFT1
4860			
4861			
4862			;ERROR 271 DATA COMMAND DID NOT ABORT WHEN DRIVE FAULT SET
4863			
4864	004432	074334	EMT271
4865	004434	074556	EHT1
4866	004436	074652	EDT1
4867	004440	074700	EFT1
4868			
4869			
4870			;ERROR 272 EBL RESET EARLY DURING DATA COMMAND
4871			
4872	004442	074360	EMT272
4873	004444	074556	EHT1
4874	004446	074652	EDT1
4875	004450	074700	EFT1
4876			
4877			
4878			;ERROR 273 EBL DIDNT RESET ON TIME DURING DATA COMMAND
4879			
4880	004452	074376	EMT273
4881	004454	074556	EHT1
4882	004456	074652	EDT1
4883	004460	074700	EFT1
4884			
4885			
4886			;ERROR 274 GO NOT RESET DURING DATA COMMAND
4887			
4888	004462	074414	EMT274
4889	004464	074556	EHT1
4890	004466	074652	EDT1
4891	004470	074700	EFT1
4892			
4893			
4894			;ERROR 275 RUN AND GO NOT SET DURING DATA COMMAND

```

4895
4896 004472 074432          EMT275
4897 004474 074556          EHT1
4898 004476 074652          EDT1
4899 004500 074700          EFT1
4900
4901
4902          ;ERROR 276      INCORRECT TAG BUS DURING DATA COMMAND
4903
4904 004502 074452          EMT276
4905 004504 074556          EHT1
4906 004506 074652          EDT1
4907 004510 074700          EFT1
4908
4909
4910          ;ERROR 277      OPI NOT SET DURING DATA COMMAND WHEN ON
4911          ;                          CYLINDER DIDNT DROP
4912
4913 004512 074470          EMT277
4914 004514 074556          EHT1
4915 004516 074652          EDT1
4916 004520 074700          EFT1
4917
4918
4919          ;ERROR 300      DATA COMMAND DID NOT ABORT WHEN SEEK ERROR SET
4920
4921 004522 074514          EMT300
4922 004524 074556          EHT1
4923 004526 074652          EDT1
4924 004530 074700          EFT1
4925
4926
4927          ;ERROR 301      SEARCH NOT ENABLED DURING DATA COMMAND
4928
4929 004532 074540          EMT301
4930 004534 074556          EHT1
4931 004536 074652          EDT1
4932 004540 074700          EFT1
4933
4934
4935          ;PUT EPROR TABLE HERE
4936

```

```

4937
4938
4939
4940 004542
4941
4942
4943 004542 000240
4944 004544 000005
4945
4946
4947
4948 004546 012706 001114
4949 004552 005026
4950 004554 022706 001154
4951 004560 001374
4952 004562 012706 001100
4953
4954 004566 012737 062212 000020
4955 004574 012737 000340 000022
4956 004602 012737 062766 000030
4957 004610 012737 000340 000032
4958 004616 012737 064526 000034
4959 004624 012737 000340 000036
4960 004632 012737 064634 000024
4961 004640 012737 000340 000026
4962 004646 013737 057266 057260
4963 004654 005037 001206
4964 004660 005037 001210
4965 004664 112737 000001 001131
4966 004672 012737 004672 001122
4967 004700 012737 004700 001124
4968
4969
4970 004706 013746 000004
4971 004712 012737 004746 000004
4972 004720 012737 177570 001154
4973 004726 012737 177570 001156
4974 004734 022777 177777 174212
4975 004742 001012
4976
4977 004744 000403
4978 004746 012716 004754 64$:
4979 004752 000002
4980 004754 012737 000176 001154 65$:
4981 004762 012737 000174 001156
4982 004770 012637 000004 66$:
4983
4984 004774 005037 001230
4985 005000 132737 000200 001243
4986 005006 001403
4987 005010 012737 001244 001154
4988 005016 67$:
4989 005016 012746 000140
4990 005022 012746 005030
4991 005026 000002
4992 005030 68$:

```

```

.SBTTL START OF PROGRAM

START:

;CLEAR AND SETUP FOR TEST EXECUTION
NOP
RESET ;INITIALIZE THE SYSTEM

.SBTTL INITIALIZE THE COMMON TAGS
;;CLEAR THE COMMON TAGS ($CMTAG) AREA
MOV #CMTAG,R6 ;:FIRST LOCATION TO BE CLEARED
CLR (R6)+ ;:CLEAR MEMORY LOCATION
CMP #SWR,R6 ;:DONE?
BNE -6 ;:LOOP BACK IF NO
MOV #STACK,SP ;:SETUP THE STACK POINTER
;;INITIALIZE A FEW VECTORS
MOV #SCOPE,#IOTVEC ;:IOT VECTOR FOR SCOPE ROUTINE
MOV #340,#IOTVEC+2 ;:LEVEL 7
MOV #ERROR,#EMTVEC ;:EMT VECTOR FOR ERROR ROUTINE
MOV #340,#EMTVEC+2 ;:LEVEL 7
MOV #TRAP,#TRAPVEC ;:TRAP VECTOR FOR TRAP CALLS
MOV #340,#TRAPVEC+2 ;:LEVEL 7
MOV #SPWRON,#PWAVEC ;:POWER FAILURE VECTOR
MOV #340,#PWAVEC+2 ;:LEVEL 7
MOV #ENDCT,#EOPCT ;:SETUP END-OF-PROGRAM COUNTER
CLR $TIMES ;:INITIALIZE NUMBER OF ITERATIONS
CLR $ESCAPE ;:CLEAR THE ESCAPE ON ERROR ADDRESS
MOV #1,$ERMAX ;:ALLOW ONE ERROR PER TEST
MOV #,$SLPADR ;:INITIALIZE THE LOOP ADDRESS FOR SCOPE
MOV #,$SLPERR ;:SETUP THE ERROR LOOP ADDRESS
;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
;;EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
MOV #ERRVEC,-(SP) ;:SAVE ERROR VECTOR
MOV #64,$ERRVEC ;:SET UP ERROR VECTOR
MOV #DSWR,$SWR ;:SETUP FOR A HARDWARE SWICH REGISTER
MOV #DISP,$DISPLAY ;:AND A HARDWARE DISPLAY REGISTER
CMP #-1,$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
MOV #65,$(SP) ;:SET UP FOR TRAP RETURN
RTI
65$: MOV #SWREG,$SWR ;:POINT TO SOFTWARE SWR
MOV #DISPREG,$DISPLAY
66$: MOV (SP)+,#ERRVEC ;:RESTORE ERROR VECTOR
CLR $PASS ;:CLEAR PASS COUNT
BITB #APTSIZE,$ENVM ;:TEST USER SIZE UNDER APT
BEQ 67$ ;:YES,USE NON-APT SWITCH
MOV #SSWREG,$SWR ;:NO,USE APT SWITCH REGISTER
67$: MOV #PR3,-(SP) ;:PUT NEW PS ON STACK
MOV #68$,-(SP) ;:PUT NEW PC ON STACK
RTI ;:POP NEW PC AND PS
68$:

```



```

4993      :SBTTL TYPE PROGRAM NAME
4994      ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
4995 005030 005227 177777      INC      #1          ;;FIRST TIME?
4996 005034 001060          BNE      69$         ;;BRANCH IF NO
4997 005036 022737 057422 000042  CMP      #SENDAD,2#42  ;;ACT-11?
4998 005044 001454          BEQ      69$         ;;BRANCH IF YES
4999 005046 104401 005114      TYPE      70$        ;;TYPE ASCIZ STRING
5000      .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
5001 005052 005737 000042      TST      2#42        ;;ARE WE RUNNING UNDER XXDP/ACT?
5002 005056 001012          BNE      71$         ;;BRANCH IF YES
5003 005060 123727 001242 000001  CMPB    $ENV,#1      ;;ARE WE RUNNING UNDER APT?
5004 005066 001406          BEQ      71$         ;;BRANCH IF YES
5005 005070 023727 001154 000176  CMP      SWR,#SWREG  ;;SOFTWARE SWITCH REG SELECTED?
5006 005076 001005          BNE      72$         ;;BRANCH IF NO
5007 005100 104407          GTSWR          ;;GET SOFT-SWR SETTINGS
5008 005102 000403          BR       72$
5009 005104 112737 003001 001150 71$:   MOVB    #1,$AUTOB    ;;SET AUTO-MODE INDICATOR
5010 005112 72$:
5011 005112 000431          BR       69$         ;;GET OVER THE ASCIZ
5012      ;;70$: .ASCIZ <CRLF>@CZRMJ80, RMO3/RMO2 DISKLESS DIAGNOSTIC PROGRAM 2<CRLF>
5013      69$:
5014
5015
5016      :FIND OUT IF PROGRAM IS RUNNING IN STANDALONE MODE
5017 005176 005737 000042      TST      42          ;;IS LOC 42 ZERO ??
5018 005202 001003          BNE      10$        ;;NO - NOT IN STANDALONE
5019 005204 105737 001242      TSTB    $ENV        ;;IS APT ENVIRONMENT ZERO ??
5020 005210 001451          BEQ      STANDALONE ;;YES - PROGRAM IN STANDALONE
5021 005212 10$:
5022
5023      :PROGRAM NOT RUNNING IN STANDALONE - SEE IF SIZING IS ALLOWED
5024 005212 132737 000200 001243 XSIZ:  BITB    #BIT7,$ENVM  ;;SIZING ALLOWED ??
5025 005220 001043          BNE      20$        ;;NO
5026      ;
5027 005222 005037 001300      MOV      #377,$DEV  ;;YES - SET DEVICE MAP FOR ALL DEVICES
5028 005226 005001          CLR     $DEV        ;;CLEAR THE BIT MAP
5029 005230 012704 000001      CLR     R1          ;;START FROM THE DRIVE 0
5030 005234 013700 001276      MOV      #BIT0,R4   ;;BIT MAP DOR DRIVE 0
5031 005240 012760 000040 000010 15$:  MOV      $BASE,R0   ;;BASE ADDRESS
5032 005246 010160 000010      MOV      #CLR,AMCS2(R0) ;;LOAD THE DRIVE ADDRESS
5033 005252 016003 000010      MOV      R1,AMCS2(R0) ;;LOAD THE DRIVE ADDRESS
5034 005256 032703 010000      MOV      AMCS2(R0),R3 ;;NEED BIT SET ?
5035 005262 001010          BIT      #NED,R3    ;;BRANCH IF SO
5036 005264 016003 000000          BNE      16$       ;;TO NEXT DRIVE
5037 005270 032703 004000      MOV      #DVA,R3   ;;CHECK THE DVA BIT
5038 005274 001403          BEQ     16$        ;;DRIVE AVAILABLE ?
5039 005276 050437 001300      BIS     R4,$DEV    ;;BRANCH IF DRIVE NOT AVAILABLE
5040 005302 000405          BR     17$        ;;SET THE BIT MAP FOR ALL AVAILABLE DRIVE
5041 005304 16$:
5042 005304 104401 066115      TYPE    NOTEX      ;;TYPE NOT EXIT MESSAGE
5043 005310 010146      MOV     R1,-(SP)  ;;DRIVE NUMBER
5044 005312 104403      TYPOS
5045 005314 006          .BYTE 6
5046 005315 000          .BYTE 0
5047 005316 005201 17$:  INC     R1          ;;INCREMENT THE DRIVE ADDRESS
5048 005320 006304      ASL    R4          ;;SET UP THE BIT MAP FOR NEXT DRIVE

```

CZRMJBO RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 112
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0112

5049 005322 022701 000007
5050 005326 103344
5051 005330
5052
5053
5054 005330 000137 000162

CMP #7,R1 ;ALL DRIVE ARE CHECKED ?
BHIS 15\$;BRANCH IF NOT
20\$:
;GO TO COMMON START CODE
JMP CMNSTART

```

5055 005334
5056
5057 005334 004737 063176
5058
5059 005340 005327 000001
5060 005344 100024
5061
5062
5063
5064 005346 104401 065424
5065 005352 104411
5066 005354 012637 001176
5067 005360 104401 001176
5068 005364 123727 001176 000131
5069 005372 001002
5070 005374 000137 006322
5071 005400 123727 001176 000116
5072 005406 001420
5073 005410 104401 065267
5074 005414 000754
5075 005416
5076
5077
5078 005416 104401 065271
5079 005422 104411
5080 005424 012637 001176
5081 005430 104401 001176
5082 005434 123727 001176 000131
5083 005442 001002
5084 005444 104401 107030
5085 005450
5086
5087
5088 005450 104401 065325
5089 005454 104411
5090 005456 012637 001176
5091 005462 104401 001176
5092 005466 104411
5093 005470 005726
5094 005472 123727 001176 000131
5095 005500 001137
5096 005502
5097
5098
5099 005502 104401 065463
5100 005506 013746 001276
5101 005512 104402
5102 005514 104401 065260
5103 005520 104413
5104 005522 012637 001176
5105 005526 001412
5106 005530 022737 160000 001176
5107 005536 101403
5108 005540 104401 065510
5109 005544 000756
5110 005546 013737 001176 001276

```

```

STANDALONE:
;SEE IF THIS IS THE FIRST START AFTER PROGRAM WAS LOADED
JSR PC,$TKINT
DEC #1 ;FIRST START ??
BPL 10$ ;YES !!

;SEE IF THE USER WANTS TO KEEP SAME DEVICES FOR TESTING
5$: TYPE ,CNSLOO ;MAINTAIN PREVIOUS PARAMETERS??
RDCHR ;GET RESPONSE
MOV (SP)+,$TMP1 ;ECHO RESPONSE
TYPE $TMP1
CMPB $TMP1,#'Y ;YES RESPONSE??
BNE 6$ ;NO!!
JMP READY ;KEEP PREVIOUS PARAMETERS
6$: CMPB $TMP1,#'N ;NO RESPONSE??
BEQ 20$ ;GET NEW PARAMETERS
TYPE ,QSTMRK ;NOT YES OR NO, TYPE ""?
BR 5$ ;RETRY
10$:

;SEE IF OPERATOR WANTS HELP FILE
TYPE ,HELPOST ;WANT HELP ??
RDCHR ;GET RESPONSE
MOV (SP)+,$TMP1 ;SAVE AND ECHO RESPONSE
TYPE $TMP1
CMPB $TMP1,#'Y ;WAS IT A YES RESPONSE ??
BNE 20$ ;NO - DONT TYPE HELP
TYPE ,HELP ;YES - TYPE HELP TEXT
20$:

;SEE IF USER WANTS TO CHANGE RMO3 UNIBUS ADDRESS
TYPE ,UBUSQST ;WANT TO CHANGE ADDRESS ??
RDCHR ;GET RESPONSE
MOV (SP)+,$TMP1 ;SAVE AND ECHO RESPONSE
TYPE $TMP1
RDCHR ;READ S CR
TST (SP)+ ;CLEAR THE STACK
CMPB $TMP1,#'Y ;WAS IT A YES RESPONSE ??
BNE 90$ ;NO !!
30$:

;DIALOGUE TO CHANGE THE UNIBUS ADDRESS, AND INTERRUPT VECTOR
TYPE ,CNSLO1 ;TYPE CURRENT BUS ADDRESS
MOV $BASE,-(SP) ;SAVE $BASE FOR TYPEOUT
TYPOC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
TYPE ,CLSPRM
RDOCT ;GET NEW BUS ADDRESS
MOV (SP)+,$TMP1 ;CARRIAGE RETURN??
BEQ 50$ ;YES-SKIP TO NEXT ENTRY
CMP #160000,$TMP1 ;BASE ADDRESS IN I/O PAGE??
BLOS 40$ ;YES
TYPE ,CNSLO2 ;TYPE WARNING MESSAGE
BR 30$ ;RETRY
40$: MOV $TMP1,$BASE ;STORE NEW BUS ADDRESS

```

```

5111 005554 113737 001272 001176 50$:   MOVB   $VECT1,$TMP1      ;TYPE CURRENT VECTOR ADDRESS
5112 005562 105037 001177          CLRB   $TMP1+1
5113 005566 104401 065571          TYPE  ,CNSLO3
5114 005572 013746 001176          MOV    $TMP1,-(SP)      ;;SAVE $TMP1 FOR TYPEOUT
5115 005576 104403          TYPOS          ;;GO TYPE--OCTAL ASCII
5116 005600          .BYTE 3              ;;TYPE 3 DIGIT(S)
5117 005601          .BYTE 0              ;;SUPPRESS LEADING ZEROS
5118 005602 104401 065260          TYPE  ,CLSPRN
5119 005606 104413          RDOCT          ;GET NEW VECTOR ADDRESS
5120 005610 012637 001176          MOV    (SP)+,$TMP1     ;CARRIAGE RETURN??
5121 005614 001412          BEQ    70$            ;YES-SKIP TO NEXT ENTRY
5122 005616 022737 001000 001176          CMP    #1000,$TMP1     ;VECTOR ADDRESS < 1000??
5123 005624 101003          BHI    60$            ;YES!!
5124 005626 104401 065621          TYPE  ,CNSLO4         ;TYPE WARNING MESSAGE
5125 005632 000750          BR     50$            ;RETRY
5126 005634 113737 001176 001272 60$:   MOVB   $TMP1,$VECT1     ;STORE NEW VECTOR ADDRESS
5127 005642 113737 001273 001176 70$:   MOVB   $VECT1+1,$TMP1  ;TYPE CURRENT PRIORITY
5128 005650 006237 001176          ASR    $TMP1
5129 005654 006237 001176          ASR    $TMP1
5130 005660 006237 001176          ASR    $TMP1
5131 005664 006237 001176          ASR    $TMP1
5132 005670 006237 001176          ASR    $TMP1
5133 005674 105037 001177          CLRB   $TMP1+1
5134 005700 104401 065675          TYPE  ,CNSLO5
5135 005704 013746 001176          MOV    $TMP1,-(SP)     ;;SAVE $TMP1 FOR TYPEOUT
5136 005710 104403          TYPOS          ;;GO TYPE--OCTAL ASCII
5137 005712          .BYTE 1              ;;TYPE 1 DIGIT(S)
5138 005713          .BYTE 0              ;;SUPPRESS LEADING ZEROS
5139 005714 104401 065260          TYPE  ,CLSPRN
5140 005720 104413          RDOCT          ;GET NEW PRIORITY
5141 005722 012637 001176          MOV    (SP)+,$TMP1     ;CARRIAGE RETURN??
5142 005726 001424          BEQ    90$            ;YES-SKIP TO NEXT ENTRY
5143 005730 023727 001176 000007          CMP    $TMP1,#7        ;LEGAL PRIORITY??
5144 005736 002403          BLT    80$            ;YES!!
5145 005740 104401 065731          TYPE  ,CNSLO6         ;TYPE WARNING MESSAGE
5146 005744 000736          BR     70$            ;RETRY
5147 005746          80$:   BR     ;STORE NEW PRIORITY
5148 005746 006337 001176          ASL    $TMP1
5149 005752 006337 001176          ASL    $TMP1
5150 005756 006337 001176          ASL    $TMP1
5151 005762 006337 001176          ASL    $TMP1
5152 005766 006337 001176          ASL    $TMP1
5153 005772 113737 001176 001273          MOVB   $TMP1,$VECT1+1
5154 006000          90$:
5155
5156          ;DIALOGUE TO INPUT DEVICE NUMBERS
5157 006000 005037 001300          CLR    $DEVN           ;CLEAR DEVICE MAP
5158 006004 104401 065756          TYPE  ,CNSLO7         ;TYPE INPUT INSTRUCTIONS
5159 006010 104401 065263          TYPE  ,PROMPT         ;TYPE PROMPTING CHARACTER
5160 006014 104411          RDCHR          ;GET RESPONSE
5161 006016 012637 001176          MOV    (SP)+,$TMP1     ;ECHO RESPONSE
5162 006022 104401 001176          TYPE  $TMP1
5163 006026 023727 001176 000101          CMP    $TMP1,#'A       ;TEST ALL DRIVES??
5164 006034 001021          BNE    110$           ;NO
5165 006036 000137 005212          JMP    XSIZ           ;TO SIZE ALL POWER UP DRIVES
5166 006042 012737 000377 001300          MOV    #377,$DEVN     ;TEST ALL DEVICES

```

5167	006050	000444			BR	140\$; SKIP TO NEXT ENTRY
5168	006052	104401	065263	100\$:	TYPE	, PROMPT		; TYPE PROMPTING CHARACTER
5169	006056	104411			RDCHR			; GET RESPONSE
5170	006060	012637	001176		MOV	(SP)+, \$TMP1		; ECHO RESPONSE
5171	006064	104401	001176		TYPE	\$TMP1		
5172	006070	023727	001176	000015	CMP	\$TMP1, #CR		; CARRIAGE RETURN??
5173	006076	001431			BEG	140\$		
5174	006100	023727	001176	000060	110\$:	CMP	\$TMP1, #'0	; NUMBER < 0??
5175	006106	002404			BLT	120\$; YES!!
5176	006110	023727	001176	000067	CMP	\$TMP1, #'7		; NUMBER > 7??
5177	006116	003403			BLE	130\$; NO!!
5178	006120	104401	065267	120\$:	TYPE	Q\$TMRK		; TYPE "??"
5179	006124	000752			BR	100\$; RETRY
5180	006126	013701	001176	130\$:	MOV	\$TMP1, R1		; R1=DRIVE NUMBER
5181	006132	042701	177770		BIC	#C7, R1		
5182	006136	116102	066242		MOVB	ATNTBL(R1), R2		; DECODE DEVICE NUMBER
5183	006142	042702	177400		BIC	#C377, R2		; CLEAR UNUSED BITS
5184	006146	050237	001300		BIS	R2, \$DEV#		; SET DEVICE # IN MAP
5185	006152	122737	000377	001300	CMPB	#377, \$DEV#		; DONE ??
5186	006160	101334			BHI	100\$; NO
5187	006162			140\$:				
5188								

```

5189 006162
5190
5191
5192 006162 013700 001300
5193 006166 012701 001460
5194 006172 010137 001456
5195 006176 012702 000001
5196 006202 005003
5197 006204 030200
5198 006206 001406
5199 006210 010311
5200 006212 116361 066242 000001
5201 006220 062701 000002
5202 006224 006302
5203 006226 105702
5204 006230 001402
5205 006232 005203
5206 006234 000763
5207 006236 005011
5208
5209
5210 006240 004737 060244
5211 006244 000403
5212 006246 104000
5213 006250 000000
5214 006252 000775
5215 006254
5216
5217 006254 005737 000042
5218 006260 001012
5219 006262 123727 001242 000001
5220 006270 001406
5221 006272 023727 001154 000176
5222 006300 001005
5223 006302 104407
5224 006304 000403
5225 006306 112737 000001 001150
5226 006314
5227 006314 012737 000140 000032
5228 006322 000240
5229 006324 004737 063176
5230 006330 117737 173122 001234

```

```

CMNSTART:
;ASSEMBLE TEST QUE FROM DEVICE MAP
MOV $DEVN,R0 ;R0 = DEVICE MAP
MOV #TSTQUE+2,R1 ;R1 = ADDRESS OF FIRST ENTRY IN QUE
MOV R1,TSTQUE ;INITIALIZE ENTRY POINTER
MOV #1,R2 ;R2 = DEVICE POINTER
MOV R3,R2 ;R3 = DEVICE NUMBER
CLR R3
10$: BIT R2,R0 ;IS THIS DEVICE IN MAP ??
BEQ 20$ ;NO !!
MOV R3,(R1) ;YES - ENTER DEVICE NUMBER IN QUE
MOVB ATNTBL(R3),1(R1) ;ENTER ATTENTION BIT IN QUE
ADD #2,R1 ;ADVANCE ENTRY POINTER
20$: ASL R2 ;ADVANCE DEVICE POINTER
TSTB R2 ;DONE ALL DEVICES ??
BEQ 25$ ;YES
INC R3 ;ADVANCE DEVICE NUMBER
BR 10$ ;ENTER NEXT DEVICE
25$: CLR (R1) ;TERMINATE TEST QUE

;SIZE FOR CLOCK
JSR PC,SIZCLK ;SEE IF CLOCK PRESENT
BR 40$ ;YES - CLOCK IS PRESENT
30$: ERROR ;NO CLOCK
HALT
BR 30$
40$:
.SBTL GET VALUE FOR SOFTWARE SWITCH REGISTER
TST #42 ;ARE WE RUNNING UNDER XXDP/ACT?
BNE 64$ ;BRANCH IF YES
CMPB $ENV,#1 ;ARE WE RUNNING UNDER APT?
BEQ 64$ ;BRANCH IF YES
CMP SWR,#SWREG ;SOFTWARE SWITCH REG SELECTED?
BNE 65$ ;BRANCH IF NO
GTSWR ;GET SOFT-SWR SETTINGS
BR 65$
64$: MOVB #1,$AUTOB ;SET AUTO-MODE INDICATOR
65$:
MOV #PR3,$EMTVEC+2 ;DROP PRIORITY DURING ERROR TYPEOUT
READY: NOP ;READY TO START TEST
JSR PC,$TKINT ;INITIALIZE TTY
MOVB #TSTQUE,$UNIT ;LOAD UNIT NUMBER

```

5231
5232
5233
5234
5235
5236
5237
5238
5239
5240
5241
5242
5243
5244
5245
5246
5247
5248
5249
5250
5251
5252
5253
5254
5255
5256
5257
5258
5259

```
.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
:
```

```

5260 ;*****
5261 ;TEST 1 TRANSFER TEST
5262 ;*****
5263
5264 006336 000004
5265 006340 012737 000001 001226
5266
5267 006346 000240
5268 006350 012737 000024 001120
5269 006356 112737 000001 001131
5270 006364 012737 006400 001122
5271 006372 012737 006400 001124
5272 006400
5273 006400 012706 001100
5274 006404 013700 001276
5275 006410 013701 001456
5276 006414 012702 000000
5277 006420
5278
5279 006420 012760 000040 000010
5280 006426 111160 000010
5281
5282 006432 016037 000010 001142
5283 006440 032737 010000 001142
5284 006446 001417
5285 006450 111137 001140
5286 006454 042737 177770 001140
5287 006462 052737 000100 001140
5288 006470 010037 001136
5289 006474 062737 000010 001136
5290 006502 104001
5291 006504 000500
5292 006506
5293
5294
5295 006506 010003
5296 006510 060203
5297 006512 011304
5298 006514 032760 010000 000010
5299 006522 001473
5300 006524 012760 000040 000010
5301 006532 111160 000010
5302
5303 006536 016037 000010 001142
5304 006544 032737 010000 001142
5305 006552 001417
5306 006554 111137 001140
5307 006560 042737 177770 001140
5308 006566 052737 000100 001140
5309 006574 010037 001136
5310 006600 062737 000010 001136
5311 006606 104001
5312 006610 000436
5313 006612
5314
5315

;*****
;TEST 1 TRANSFER TEST
;*****
T1: SCOPE
MOV #1,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX

NOP
MOV #20,$ICNT ;20 ITERATIONS
MOVB #1,$ERMAX ;ONE ERROR ALLOWED
MOV #T1,$LPADR ;LOAD LOOP ON TEST ADDRESS
MOV #T1,$LPERR ;LOAD LOOP ON ERROR ADDRESS

T1:
MOV #STACK,SP ;LOAD THE STACK POINTER
MOV $BASE,R0 ;R0 = UNIBUS ADDRESS OF I/O
MOV $TQUE,R1 ;R1 = POINTER TO DEVICE
MOV #0,R2 ;R2 = REGISTER INDEX

10$:
;CLEAR THE MASSBUS AND VERIFY THAT NONEXISTANT DEVICE ERROR IS RESET
MOV #CLR, RMCS2(R0) ;CLEAR THE MASSBUS
MOVB (R1), RMCS2(R0) ;SELECT UNIT

MOV RMCS2(R0), $BDDAT ;STORE RMCS2 AT $BDDAT
BIT #NED, $BDDAT
BEQ 20$
MOVB (R1), $GDDAT
BIC #ICNTMSK, $GDDAT
BIS #IR, $GDDAT
MOV R0, $BDAOR
ADD #RMCS2, $BDAOR
ERROR 1
BR 60$

20$:
;READ THE REGISTER WHOSE INDEX IS IN R2 AND EXIT TEST IF THE READ
;DOES NOT SET "NED" ERROR
MOV R0, R3 ;R3 = REGISTER ADDRESS
ADD R2, R3
MOV (R3), R4 ;READ REGISTER
BIT #NED, RMCS2(R0) ;IS "NED" SET??
BEQ 70$ ;NO!!
MOV #CLR, RMCS2(R0) ;CLEAR THE MASSBUS
MOVB (R1), RMCS2(R0) ;SELECT UNIT

MOV RMCS2(R0), $BDDAT ;STORE RMCS2 AT $BDDAT
BIT #NED, $BDDAT
BEQ 30$
MOVB (R1), $GDDAT
BIC #ICNTMSK, $GDDAT
BIS #IR, $GDDAT
MOV R0, $BDAOR
ADD #RMCS2, $BDAOR
ERROR 1
BR 60$

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

```



```

5316 006612 012713 000000      MOV      #0,(R3)          ;WRITE REGISTER
5317 006616 032760 010000 000010 BIT      #NED, RMCS2     (R0)  ;IS "NED" SET??
5318 006624 001432              BEQ      70$            ;NO!!
5319
5320 006626                      40$:
5321 ;COULD NOT READ OR WRITE THE REGISTER WITHOUT SETTING "NED" ERROR -
5322 ;ADVANCE THE REGISTER INDEX AND REPEAT THE TEST FOR THE NEXT
5323 ;AVAILABLE DEVICE REGISTER
5324 006626 062702 000002      ADD      #2,R2          ;ADVANCE TO NEXT REGISTER
5325 006632 022702 000002      CMP      #RMWC,R2      ;IS THIS RMWC??
5326 006636 001773              BEQ      40$            ;YES - TRY NEXT REGISTER
5327 006640 022702 000004      CMP      #RMA,R2      ;IS THIS RMA??
5328 006644 001770              BEQ      40$            ;YES - TRY NEXT REGISTER
5329 006646 022702 000010      CMP      #RMCS2,R2    ;IS THIS RMCS??
5330 006652 001765              BEQ      40$            ;YES - TRY ANOTHER REGISTER
5331 006654 022702 000016      CMP      #RMA5,R2     ;IS THIS RMA5??
5332 006660 001762              BEQ      40$            ;YES - TRY ANOTHER REGISTER
5333 006662 022702 000022      CMP      #RMOB,R2     ;IS THIS RMOB??
5334 006666 001757              BEQ      40$            ;YES - TRY ANOTHER REGISTER
5335 006670 022702 000046      CMP      #RMEC2,R2    ;IS THIS A LEGAL REGISTER
5336 006674 103251              BHS      10$            ;YES - TRY THIS REGISTER
5337
5338 006676                      50$:
5339
5340 ;GOT 'NONEXISTENT DEVICE' ERROR FOR EVERY REMOTE REGISTER ADDRESS
5341 006676 013737 001276 001136  MOV      $BASE,$BDAOR  ;STORE BASE ADDRESS
5342 006704 104002              ERROR    2             ;DEMAND OR TRANSFER FAILED
5343 006706 000137 057176      60$:  JMP      $EOSP         ;GO SELECT NEXT DEVICE
5344
5345 006712                      70$:
5346
5347 ;*****
5348 ;*TEST 2          CTOD TEST
5349
5350 ;*****
5351 006712 000004      TST2:  SCOPE
5352 006714 012737 000002 001226  MOV      #2,$TESTN    ;;SET TEST NUMBER IN APT MAIL BOX
5353 006722 000240      NOP
5354 006724 012737 000024 0J1120  MOV      #20,$ICNT    ;20 ITERATIONS
5355 006732 112737 000001 001131  MOVB    #1,$ERMAX     ;ONE ERROR ALLOWED
5356 006740 012737 006754 001122  MOV      #T2,$LPADR   ;LOAD LOOP ON TEST ADDRESS
5357 006746 012737 006754 001124  MOV      #T2,$LPERR   ;LOAD LOOP ON ERROR ADDRESS
5358 006754
5359 006754 012706 001100      T2:  MOV      #STACK,SP    ;LOAD THE STACK POINTER
5360 006760 013700 001276      MOV      $BASE,R0     ;R0 = UNIBUS ADDRESS OF UUT
5361 006764 013701 001456      MOV      TSTACK,R1    ;R1 = POINTER TO DEVICE
5362 006770 012760 000040 000010  MOV      #CLR,RMCS2(R0) ;CLEAR THE MASSBUS
5363 006776 111160 000010      MOVB    (R1),RMCS2(R0) ;SELECT UNIT
5364 ;WRITE ONES IN REMOTE REGISTERS
5365
5366 007002 012760 000076 000000      MOV      #ILF76,RMCS1(R0) ;LOAD RMCS1
5367
5368 007010 012760 177777 000006      MOV      #-1,RMDA(R0)  ;LOAD RMDA
5369
5370 007016 012760 001777 000034      MOV      #CYLSK,RMDC(R0) ;LOAD RMDC
5371

```

```

5372 007024 012760 016200 000032      MOV      #FMT16!ECI!HCI!OFD,RMOF(RO)      ;LOAD RMOF
5373                                     ;READ REMOTE REGISTERS TWICE
5374 007032 012702 000001      MOV      #1,R2
5375 007036                                     10$:
5376                                     MOV      RMCS1(RO),RMCS1I      ;STORE RMCS1 IN INPUT BUFFER
5377 007036 016037 000000 001326      MOV      RMDA(RO),RMDAI      ;STORE RMDA IN INPUT BUFFER
5378                                     MOV      RMDC(RO),RMDCI      ;STORE RMDC IN INPUT BUFFER
5379 007044 016037 000006 001334      MOV      RMOF(RO),RMOFI      ;STORE RMOF IN INPUT BUFFER
5380                                     DEC      R2
5381 007052 016037 000034 001362      BPL      10$
5382                                     ;SEE IF ANY ONE BITS CAME BACK
5383 007060 016037 000032 001360      BIC      #1CILF76,RMCS1I      ;IS RMCS1 0??
5384 007066 005302                                     BNE      20$                    ;NO!!
5385 007070 100362                                     TST     RMDAI                    ;IS RMDA 0??
5386                                     BNE      20$                    ;NO!!
5387 007072 042737 177701 001326      BIC      #XNUDC,RMDCI        ;IS RMDC 0??
5388 007100 001014                                     BNE      20$                    ;NO!!
5389 007102 005737 001334      BIC      #XNUOF,RMOFI        ;IS RMOF 0 ??
5390 007106 001011                                     BNE      20$                    ;NO!!
5391 007110 042737 176000 001362      BIC      #XNUOF,RMOFI        ;IS RMOF 0 ??
5392 007116 001005                                     BNE      20$                    ;NO!!
5393 007120 042737 161577 001360      BIC      #XNUOF,RMOFI        ;IS RMOF 0 ??
5394 007126 001001                                     BNE      20$                    ;NO!!
5395                                     ;CANNOT READ ANY ONE BITS FROM REMOTE REGISTERS
5396 007130 104003      ERROR 3      ;CTOD MUST BE STUCK
5397 007132      20$:
5398
5399      ;*****
5400      ;*TEST 3      MASSBUS INITIALIZE TEST
5401
5402      ;*****
5403 007132 000004      ST3:  SCOPE
5404 007134 012737 000003 001226      MOV      #3,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5405
5406      NOP
5407 007142 000240      MOV      #20,$ICNT      ;20 ITERATIONS
5408 007144 012737 000024 001120      MOVB    #1,$ERMAX      ;ONE ERROR ALLOWED
5409 007152 112737 000001 001131      MOV      #T3,$LPADR      ;LOAD LOOP ON TEST ADDRESS
5410 007160 012737 007174 001122      MOV      #T3,$LPERR      ;LOAD LOOP ON ERROR ADDRESS
5411 007166 012737 007174 001124      T3:
5412 007174 012706 001100      MOV      #STACK,SP      ;LOAD THE STACK POINTER
5413 007200 013700 001276      MOV      $BASE,RO      ;RO = UNIBUS ADDRESS OF UUT
5414 007204 013701 001456      MOV      TSTQUE,R1      ;R1 = POINTER TO DEVICE
5415 007210 012760 000040 000010      MOV      #CLR,RMCS2(RO)    ;CLEAR THE MASSBUS
5416 007216 111160 000010      MOVB    (R1),RMCS2(RO)    ;SELECT UNIT
5417      ;WRITE ONES IN SELECTED REGISTERS
5418
5419 007222 012760 000076 000000      MOV      #ILF76,RMCS1(RO)  ;LOAD RMCS1
5420
5421 007230 012760 177777 000014      MOV      #-1,RMER1(RO)    ;LOAD RMER1
5422
5423 007236 012760 177777 000042      MOV      #-1,RMER2(RO)    ;LOAD RMER2
5424      ;USING CONTROLLER CLEAR IE BIT 5 OF RMCS2, INITIALIZE THE MASSBUS
5425 007244 012760 000040 000010      MOV      #CLR,RMCS2(RO)    ;CLEAR THE MASSBUS
5426 007252 111160 000010      MOVB    (R1),RMCS2(RO)    ;SELECT UNIT
5427      ;READ THE REGISTERS THAT WERE WRITTEN

```

```

5428
5429 007256 016037 000000 001326      MOV      RMCS1(RO),RMCS1I      ;STORE RMCS1 IN INPUT BUFFER
5430
5431 007264 016037 000014 001342      MOV      RMER1(RO),RMER1I     ;STORE RMER1 IN INPUT BUFFER
5432
5433 007272 016037 000042 001370      MOV      RMER2(RO),RMER2I     ;STORE RMER2 IN INPUT BUFFER
5434 ;SEE IF ANY REGISTER BITS WERE CLEARED
5435 007300 052737 177701 001326      BIS      #1,CILF76,RMCS1I     ;SET ANY BIT NOT WRITTEN
5436 007306 052737 001567 001370      BIS      #XNUE2,RMER2I
5437 007314 022737 177777 001326      CMP      #-1,RMCS1I          ;ANY ZEROS IN RMCS1??
5438 007322 001011 177777 001326      BNE      10$                 ;YES!!
5439 007324 022737 177777 001342      CMP      #-1,RMER1I          ;ANY ZEROS IN RMER1??
5440 007332 001005 177777 001342      BNE      10$                 ;YES!!
5441 007334 022737 177777 001370      CMP      #-1,RMER2I          ;ANY ZEROS IN RMER2??
5442 007342 001001 177777 001370      BNE      10$
5443 ;NONE OF THE BITS WERE CLEARED
5444 007344 104004 177777 001370      ERROR   4                    ;MASSBUS INIT FAILED
5445 007346
5446
5447
5448 ;*****
5449 ;*TEST 4 CLEAR STUCK ACTIVE TEST
5450 ;*****
5451 007346 000004 000004 001226      ST4:   SCOPE
5452 007350 012737 000004 001226      MOV      #4,$TESTN           ;;SET TEST NUMBER IN APT MAIL BOX
5453
5454 007356 000240 000024 001120      NOP
5455 007360 012737 000001 001131      MOV      #20,$ICNT           ;20 ITERATIONS
5456 007366 112737 000001 001131      MOV      #1,$ERMAX           ;ONE ERROR ALLOWED
5457 007374 012737 007410 001122      MOV      #T4,$LPADR          ;LOAD LOOP ON TEST ADDRESS
5458 007402 012737 007410 001124      MOV      #T4,$LPERR          ;LOAD LOOP ON ERROR ADDRESS
5459 007410
5460 007410 012706 001100 000010      T4:   MOV      #STACK,SP        ;LOAD THE STACK POINTER
5461 007414 013700 001276 000010      MOV      $BASE,RO            ;RO = UNIBUS ADDRESS OF UUT
5462 007420 013701 001456 000010      MOV      TSTQUE,R1           ;R1 = POINTER TO DEVICE
5463 007424 012760 000040 000010      MOV      #CCR,RMCS2(RO)      ;CLEAR THE MASSBUS
5464 007432 111160 000010 000010      MOV      (R1),RMCS2(RO)      ;SELECT UNIT
5465 ;WRITE ONES IN TEST REGISTERS
5466
5467 007436 012760 177777 000014      MOV      #-1,RMER1(RO)       ;LOAD RMER1
5468
5469 007444 012760 177777 000042      MOV      #-1,RMER2(RO)       ;LOAD RMER2
5470
5471 007452 012760 000001 000024      MOV      #DMD,RMMR1(RO)      ;LOAD RMMR1
5472 ;READ TEST REGISTERS AND SEE IF ANY BITS ARE ON
5473
5474 007460 016037 000014 001342      MOV      RMER1(RO),RMER1I     ;STORE RMER1 IN INPUT BUFFER
5475
5476 007466 016037 000042 001370      MOV      RMER2(RO),RMER2I     ;STORE RMER2 IN INPUT BUFFER
5477
5478 007474 016037 000024 001352      MOV      RMMR1(RO),RMMR1I     ;STORE RMMR1 IN INPUT BUFFER
5479 007502 042737 040000 001342      BIC      #UNS,RMER1I          ;DONT ACCEPT UNSAFE
5480 007510 001011 040000 001342      BNE      10$                 ;BRANCH IF ANY OTHER BITS ON
5481 007512 042737 040200 001370      BIC      #SKI!DVC,RMER2I      ;DONT ACCEPT SKI OR DVC
5482 007520 001005 040200 001370      BNE      10$                 ;BRANCH IF ANY OTHER BITS ON
5483 007522 032737 000001 001352      BIT      #DMD,RMMR1I          ;BRANCH IF DMD IS ON

```

```

5484 007530 001001      BNE      105
5485 007532 104026      ERROR   26      ;MBA CLR IS STUCK ACTIVE
5486 007534
5487
5488
5489
5490
5491
5492
5493 007534 000004
5494 007536 012737 000005 001226
5495 007544 000240
5496 007546 012737 000024 001120
5497 007554 112737 000001 001131
5498 007562 012737 007576 001122
5499 007570 012737 007576 001124
5500 007576
5501 007576 012706 001100
5502 007602 013700 001276
5503 007606 013701 001456
5504 007612 005002
5505 007614 012760 000040 000010
5506 007622 111160 000010
5507
5508
5509 007626 012760 000076 000000
5510
5511 007634 012760 177777 000006
5512
5513 007642 012760 177777 000014
5514
5515 007650 012760 177777 000032
5516
5517 007656 012760 177777 000042
5518
5519
5520 007664 012760 000000 000000
5521
5522 007672 012760 000000 000006
5523
5524 007700 012760 000000 000014
5525
5526 007706 012760 000000 000032
5527
5528 007714 012760 000000 000034
5529
5530 007722 012760 000000 000042
5531
5532
5533 007730 016037 000000 001326
5534
5535 007736 016037 000006 001334
5536
5537 007744 016037 000014 001342
5538
5539 007752 016037 000032 001360

```

```

105:
;*****
;TEST 5      TRISTATE TRANSFER TEST
;*****
↑STS:  SCOPE
      MOV      #5,$STESTN      ;;SET TEST NUMBER IN APT MAIL BOX
      NOP
      MOV      #20,$ICNT      ;20 ITERATIONS
      MOV      #1,$ERMAX      ;ONE ERROR ALLOWED
      MOV      #TS,$LPAOR      ;LOAD LOOP ON TEST ADDRESS
      MOV      #TS,$LPERR      ;LOAD LOOP ON ERROR ADDRESS
15:
      MOV      #STACK_SP      ;LOAD THE STACK POINTER
      MOV      $BASE,R0      ;R0 = UNIBUS ADDRESS OF UUT
      MOV      TSTQUE,R1      ;R1 = POINTER TO DEVICE
      CLR      R2              ;CLEAR ERROR FLAGS
      MOV      #CLR, RMCS2(R0) ;CLEAR THE MASSBUS
      MOV      (R1), RMCS2(R0) ;SELECT UNIT
;WRITE ONES IN SELECTED REGISTERS
      MOV      #ILF76, RMCS1(R0) ;LOAD RMCS1
      MOV      #-1, RMDA(R0)      ;LOAD RMDA
      MOV      #-1, RMER1(R0)     ;LOAD RMER1
      MOV      #-1, RMOF(R0)     ;LOAD RMOF
      MOV      #-1, RMER2(R0)    ;LOAD RMER2
;WRITE ZEROS IN SELECTED REGISTERS
      MOV      #0, RMCS1(R0)     ;LOAD RMCS1
      MOV      #0, RMDA(R0)      ;LOAD RMDA
      MOV      #0, RMER1(R0)     ;LOAD RMER1
      MOV      #0, RMOF(R0)     ;LOAD RMOF
      MOV      #0, RMDC(R0)     ;LOAD RMDC
      MOV      #0, RMER2(R0)    ;LOAD RMER2
;READ BACK ALL REGISTERS
      MOV      RMCS1(R0), RMCS1I ;STORE RMCS1 IN INPUT BUFFER
      MOV      RMDA(R0), RMDAI  ;STORE RMDA IN INPUT BUFFER
      MOV      RMER1(R0), RMER1I ;STORE RMER1 IN INPUT BUFFER
      MOV      RMOF(R0), RMOFI  ;STORE RMOF IN INPUT BUFFER

```

G10

CZRMJBO RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07MACY11 30(1046) 23-NOV-77 12:14 PAGE 123
T5 TRISTATE TRANSFER TEST

SEQ 0123

```

5540
5541 007760 016037 000034 001362      MOV      RMDC(RO),RMDCI  ;STORE RMDC IN INPUT BUFFER
5542
5543 007766 016037 000042 001370      MOV      RMER2(RO),RMER2I ;STORE RMER2 IN INPUT BUFFER
5544 ;CHECK EACH REGISTER CONTENT FOR ZERO BITS WRITTEN & READ
5545 007774 012702 177777      MOV      #-1,R2          ;ACCUMULATE ZEROS IN R2
5546 010000 052737 177701 001326  BIS      #↑CILF76,RMCS1I ;SET ALL BITS NOT WRITTEN
5547 010006 052737 161577 001360  BIS      #XNUOF,RMOFI
5548 010014 052737 176000 001362  BIS      #XNUDC,RMDCI
5549 010022 052737 001567 001370  BIS      #XNUER2,RMER2I
5550 010030 005137 001326      COM      RMCS1I          ;COMPLEMENT REGISTER CONTENTS
5551 010034 005137 001334      COM      RMDAI
5552 010040 005137 001342      COM      RMER1I
5553 010044 005137 001360      COM      RMOFI
5554 010050 005137 001362      COM      RMDCI
5555 010054 005137 001370      COM      RMER2I
5556 010060 043702 001326      BIC      RMCS1I,R2        ;ACCUMULATE ALL ZERO BITS
5557 010064 043702 001334      BIC      RMDAI,R2
5558 010070 043702 001342      BIC      RMER1I,R2
5559 010074 043702 001360      BIC      RMOFI,R2
5560 010100 043702 001362      BIC      RMDCI,R2
5561 010104 043702 001370      BIC      RMER2I,R2
5562 010110 001407      BEQ      10$             ;BRANCH IF EACH BIT IS ZERO
5563 ;ONE OR MORE BIT POSITIONS ARE NOT ZERO
5564 010112 010237 001142      MOV      R2,$BDDAT      ;SAVE RESULT FOR TYPE
5565 010116 005037 001140      CLR      $GDDAT        ;LOAD EXPECTED RESULT
5566 010122 104005      ERROR   5              ;TRISTATE BUS IS STUCK AT ONE
5567 010124 052702 000001      BIS      #BIT0,R2      ;SET ERROR FLAG
5568 010130
5569 10$:
5570 010130 012760 000040 000010      MOV      #CLR,RMCS2(RO) ;CLEAR THE MASSBUS
5571 010136 111160 000010      MOV      (R1),RMCS2(RO) ;SELECT UNIT
5572 ;PRESET SELECTED REGISTERS TO ZEROS
5573 ;(ASSUME RMCS1, RMER1, RMER2 WERE CLEARED BY INIT)
5574
5575 010142 012760 000000 000006      MOV      #0,RMDA(RO)    ;LOAD RMDA
5576
5577 010150 012760 000000 000032      MOV      #0,RMOF(RO)    ;LOAD RMOF
5578
5579 010156 012760 000000 000034      MOV      #0,RMDC(RO)    ;LOAD RMDC
5580 ;WRITE ONES IN SELECTED REGISTERS
5581
5582 010164 012760 000076 000000      MOV      #ILF76,RMCS1(RO) ;LOAD RMCS1
5583
5584 010172 012760 177777 000006      MOV      #-1,RMDA(RO)    ;LOAD RMDA
5585
5586 010200 012760 016200 000032      MOV      #↑CXNUOF,RMOF(RO) ;LOAD RMOF
5587
5588 010206 012760 001777 000034      MOV      #↑CXNUDC,RMDC(RO) ;LOAD RMDC
5589
5590 010214 012760 177777 000014      MOV      #-1,RMER1(RO)   ;LOAD RMER1
5591
5592 010222 012760 176210 000042      MOV      #↑CXNUER2,RMER2(RO) ;LOAD RMER2
5593 ;READ ALL REGISTERS
5594
5595 010230 016037 000000 001326      MOV      RMCS1(RO),RMCS1I ;STORE RMCS1 IN INPUT BUFFER

```

```

5596
5597 010236 016037 000006 001334      MOV      RMDA(RO),RMDAI  ;STORE RMDA IN INPUT BUFFER
5598
5599 010244 016037 000032 001360      MOV      RMOF(RO),RMOFI  ;STORE RMOF IN INPUT BUFFER
5600
5601 010252 016037 000034 001362      MOV      RMDC(RO),RMDCI  ;STORE RMDC IN INPUT BUFFER
5602
5603 010260 016037 000014 001342      MOV      RMER1(RO),RMER1I  ;STORE RMER1 IN INPUT BUFFER
5604
5605 010266 016037 000042 001370      MOV      RMER2(RO),RMER2I  ;STORE RMER2 IN INPUT BUFFER
5606      ;CHECK EACH REGISTER CONTENT FOR ONE BITS WRITTEN & READ
5607 010274 042737 177701 001326      BIC      #1,CILF76,RMCS1I  ;CLEAR ALL BITS NOT WRITTEN
5608 010302 042737 161577 001360      BIC      #XNUOF,RMOFI
5609 010310 042737 176000 001362      BIC      #XNUDC,RMDCI
5610 010316 042737 001567 001370      BIC      #XNUER2,RMER2I
5611 010324 005002      CLR      R2  ;ACCUMULATE ONES IN R2
5612 010326 053702 001326      BIS      RMCS1I,R2  ;ACCUMULATE ALL ONE BITS
5613 010332 053702 001334      BIS      RMDAI,R2
5614 010336 053702 001360      BIS      RMOFI,R2
5615 010342 053702 001362      BIS      RMDCI,R2
5616 010346 053702 001342      BIS      RMER1I,R2
5617 010352 053702 001370      BIS      RMER2I,R2
5618 010356 022702 177777      CMP      #-1,R2  ;SEE IF EACH BIT POSITION WAS ONE
5619 010362 001410      BEQ      20$  ;BRANCH IF NONE STUCK
5620
5621 010364 010237 001142      ;ONE OR MORE BIT POSITIONS ARE NOT ONE
5622 010370 012737 177777 001140      MOV      R2,$BDDAT  ;SAVE RESULT FOR TYPE
5623 010376 104006      MOV      #-1,$GDDAT  ;EXPECTED RESULT
5624 010400 052702 000002      ERROR   6
5625 010404      BIS      #BIT1,R2  ;SET ERROR FLAG
5626 20$:
5627 010404 005702      TST      R2  ;ANY ERRORS DETECTED ??
5628 010406 001131      BNE      30$  ;YES - DONT DO BIT TEST
5629 010410 012702 000001      MOV      #1,R2  ;R2=BIT POSITION
5630 010414      25$:
5631 010414 012760 000040 000010      MOV      #CLR,RMCS2(RO) ;CLEAR THE MASSBUS
5632 010422 111160 000010      MOV      (R1),RMCS2(RO) ;SELECT UNIT
5633      ;WRITE THE BIT PATTERN IN SELECTED DEVICE REGISTERS
5634
5635 010426 010260 000006      MOV      R2,RMDA(RO)  ;LOAD RMDA
5636
5637 010432 010260 000032      MOV      R2,RMOF(RO)  ;LOAD RMOF
5638
5639 010436 010260 000034      MOV      R2,RMDC(RO)  ;LOAD RMDC
5640
5641 010442 010260 000014      MOV      R2,RMER1(RO) ;LOAD RMER1
5642
5643 010446 010260 000042      MOV      R2,RMER2(RO) ;LOAD RMER2
5644      ;READ BACK THE REGISTERS
5645
5646 010452 016037 000006 001334      MOV      RMDA(RO),RMDAI  ;STORE RMDA IN INPUT BUFFER
5647
5648 010460 016037 000032 001360      MOV      RMOF(RO),RMOFI  ;STORE RMOF IN INPUT BUFFER
5649
5650 010466 016037 000034 001362      MOV      RMDC(RO),RMDCI  ;STORE RMDC IN INPUT BUFFER
5651

```

```

5652 010474 016037 000014 001342      MOV      RMER1(R0),RMER1I      ;STORE RMER1 IN INPUT BUFFER
5653
5654 010502 016037 000042 001370      MOV      RMER2(R0),RMER2I      ;STORE RMER2 IN INPUT BUFFER
5655      ;CHECK REGISTER CONTENTS FOR CORRECT PATTERN
5656 010510 005003      CLR      R3                    ;R3=ACCUMULATED ONE BIT
5657 010512 012704 177777      MOV      #-1,R4                ;R4=ACCUMULATED ZERO BITS
5658 010516 013705 001334      MOV      RMDAI,R5              ;GET ANY GOOD BITS FROM RMDA
5659 010522 050503      BIS      R5,R3
5660 010524 005105      COM      R5
5661 010526 040504      BIC      R5,R4
5662 010530 013705 001360      MOV      RMOFI,R5              ;GET GOOD BITS FROM RMOF
5663 010534 042705 161577      BIC      #XNUOF,R5
5664 010540 050503      BIS      R5,R3
5665 010542 005105      COM      R5
5666 010544 042705 161577      BIC      #XNUOF,R5
5667 010550 040504      BIC      R5,R4
5668 010552 013705 001362      MOV      RMDCI,R5              ;GET GOOD BITS FROM RMDC
5669 010556 042705 176000      BIC      #XNUDC,R5
5670 010562 050503      BIS      R5,R3
5671 010564 005105      COM      R5
5672 010566 042705 176000      BIC      #XNUDC,R5
5673 010572 040504      BIC      R5,R4
5674 010574 013705 001342      MOV      RMER1I,R5              ;GET GOOD BITS FROM RMER1
5675 010600 050503      BIS      R5,R3
5676 010602 005105      COM      R5
5677 010604 040504      BIC      R5,R4
5678 010606 013705 001370      MOV      RMER2I,R5              ;GET GOOD BITS FROM RMER2
5679 010612 042705 001567      BIC      #XNUER2,R5
5680 010616 050503      BIS      R5,R3
5681 010620 005105      COM      R5
5682 010622 042705 001567      BIC      #XNUER2,R5
5683 010626 040504      BIC      R5,R4
5684 010630 010205      MOV      R2,R5                  ;RESET ALL ONES IN R3 EXCEPT
5685 010632 005105      COM      R5                      ;FOR THE TEST BIT
5686 010634 040503      BIC      R5,R3
5687 010636 040204      BIC      R2,R4                  ;RESET TEST BIT IN R4
5688 010640 050403      BIS      R4,R3                  ;COMBINE ACCUMULATED 1'S + 0'S
5689 010642 020302      CMP      R3,R2                  ;IS PATTERN OK??
5690 010644 001406      BEQ      26$                    ;YES!!
5691 010646 010237 001140      MOV      R2,$GDDAT              ;SAVE TEST PATTERN
5692 010652 010337 001142      MOV      R3,$BDDAT              ;SAVE RESULT
5693 010656 104007      ERROR   7                      ;BIT INTERFERENCE IN TRISTATE BUS
5694 010660 000404      BR       30$                    ;SKIP TO NEXT
5695 010662
5696      26$:
5697      ;ADVANCE R2 TO THE NEXT PATTERN AND REPEAT TEST
5698      ASL      R2
5699      BEQ      30$                    ;SHIFT THE BIT
5700      ;EXIT IF DONE
5701      JMP      25$
5702
5703      30$:
5704      ;*****
5705      ;*TEST 6 REGISTER SELECT TEST
5706      ;*****
5707      †$T6: SCOPE
5708      MOV      #6,$TESTN          ;:SET TEST NUMBER IN APT MAIL BOX

```

```

5708
5709 010702 000240          NOP
5710 010704 012737 000024 001120      MOV    #20, $ICNT      ;20 ITERATIONS
5711 010712 112737 000001 001131      MOV    #1, $ERMAX     ;ONE ERROR ALLOWED
5712 010720 012737 010734 001122      MOV    #T6, $LPADR   ;LOAD LOOP ON TEST ADDRESS
5713 010726 012737 010734 001124      MOV    #T6, $LPERR   ;LOAD LOOP ON ERROR ADDRESS
5714 010734
5715 010734 012706 001100      MOV    #STACK, SP    ;LOAD THE STACK POINTER
5716 010740 013700 001276      MOV    $BASE, R0     ;R0 = UNIBUS ADDRESS OF UUT
5717 010744 013701 001456      MOV    TSTQUE, R1    ;R1 = POINTER TO DEVICE
5718
5719 :THE FOLLOWING TABLE GIVES MASSBUS REGISTER SELECT VALUES FOR
5720 :EACH DEVICE REGISTER

```

REGISTER NAME	REG SEL (16,8,4,2,1)
RMCS1	00000
RMDS	00001
RMER1	00010
RMMR1	00011
RMA5	00100
RMDA	00101
RMDT	00110
RMLA	00111
RMSN	01000
RMOF	01001
RMDC	01010
RMHR	01011
RMMR2	01100
RMER2	01101
RMEC1	01110
RMEC2	01111

```

: 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-O, 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-O,
: THE ALL ONES WORD WILL BE WRITTEN IN RMER1, AND RMER1
: WILL NOT BE 0 WHEN READ BACK.

```

```

5750 010750 005002          CLR    R2              ;R2= ZEROS SOURCE
5751 010752 012703 177777      MOV    #1, R3         ;R3= ONES SOURCE
5752
5753 010756 012760 000040 000010 : TEST REG SEL 1 FOR S-A-O
5754 010764 111160 000010      MOV    #CLR, RMCS2(R0) ;CLEAR THE MASSBUS
5755
5756 010770 010260 000014      MOV    (R1), RMCS2(R0) ;SELECT UNIT
5757
5758 010774 010260 000034      MOV    R2, RMER1(R0)  ;LOAD RMER1
5759
5760 011000 010360 000024      MOV    R2, RMDC(R0)   ;LOAD RMDC
5761
5762 011004 010360 000036      MOV    R3, RMMR1(R0) ;LOAD RMMR1
5763
5764
5765
5766
5767
5768
5769
5770
5771
5772
5773
5774
5775
5776
5777
5778
5779
5780
5781
5782
5783
5784
5785
5786
5787
5788
5789
5790
5791
5792
5793
5794
5795
5796
5797
5798
5799

```


K10

CZRMJBO RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 127
T6 REGISTER SELECT TEST

SEQ 0127

```

5764 011010 016037 000014 001342      MOV      RMER1(RO),RMER1I      ;STORE RMER1 IN INPUT BUFFER
5765
5766 011016 016037 000034 001362      MOV      RMDC(RO),RMDCI      ;STORE RMDC IN INPUT BUFFER
5767 011024 020337 001342      CMP      R3,RMER1I
5768 011030 001007      BNE      10$
5769 011032 052737 176000 001362      BIS      #XNUDC,RMDCI
5770 011040 020337 001362      CMP      R3,RMDCI
5771 011044 001001      BNE      10$
5772 011046 104010      ERROR    10      ;REG SEL 1 IS S-A-0
5773 011050
5774
5775      10$:
5776 011050 012760 000040 000010      ;TEST REG SEL 1 FOR S-A-1
5777 011056 111160 000010      MOV      #CLR,RMCS2(RO)      ;CLEAR THE MASSBUS
5778      MOVB   (R1),RMCS2(RO)      ;SELECT UNIT
5779 011062 010260 000006      MOV      R2,RMDA(RO)      ;LOAD RMDA
5780
5781 011066 010260 000032      MOV      R2,RMOF(RO)      ;LOAD RMOF
5782
5783 011072 010260 000042      MOV      R2,RMER2(RO)      ;LOAD RMER2
5784
5785 011076 010360 000016      MOV      R3,RMAS(RO)      ;LOAD RMAS
5786
5787 011102 010360 000030      MOV      R3,RMSN(RO)      ;LOAD RMSN
5788
5789 011106 010360 000040      MOV      R3,RMMR2(RO)      ;LOAD RMMR2
5790
5791 011112 016037 000006 001334      MOV      RMDA(RO),RMDAI      ;STORE RMDA IN INPUT BUFFER
5792
5793 011120 016037 000032 001360      MOV      RMOF(RO),RMOFI      ;STORE RMOF IN INPUT BUFFER
5794
5795 011126 016037 000042 001370      MOV      RMER2(RO),RMER2I      ;STORE RMER2 IN INPUT BUFFER
5796 011134 020337 001334      CMP      R3,RMDAI
5797 011140 001015      BNE      20$
5798 011142 052737 161577 001360      BIS      #XNUOF,RMOFI
5799 011150 020337 001360      CMP      R3,RMOFI
5800 011154 001007      BNE      20$
5801 011156 052737 001567 001370      BIS      #XNUER2,RMER2I
5802 011164 020337 001370      CMP      R3,RMER2I
5803 011170 001001      BNE      20$
5804 011172 104011      ERROR    11      ;REG SEL 1 IS S-A-1
5805 011174
5806      20$:
5807      ;TEST REG SEL 2 FOR S-A-0
5808 011174 012760 000040 000010      MOV      #CLR,RMCS2(RO)      ;CLEAR THE MASSBUS
5809 011202 111160 000010      MOVB   (R1),RMCS2(RO)      ;SELECT UNIT
5810
5811 011206 010260 000006      MOV      R2,RMDA(RO)      ;LOAD RMDA
5812
5813 011212 010260 000032      MOV      R2,RMOF(RO)      ;LOAD RMOF
5814
5815 011216 010260 000042      MOV      R2,RMER2(RO)      ;LOAD RMER2
5816
5817 011222 010360 000020      MOV      R3,RMLA(RO)      ;LOAD RMLA
5818
5819 011226 010360 000036      MOV      R3,RMHR(RO)      ;LOAD RMHR

```

```

5820
5821 011232 010360 000046      MOV      R3,RMEC2(RO)      ;LOAD RMEC2
5822
5823 011236 016037 000006 001334      MOV      RMDA(RO),RMDAI    ;STORE RMDA IN INPUT BUFFER
5824
5825 011244 016037 000032 001360      MOV      RI:OF(RO),RMOFI    ;STORE RMOF IN INPUT BUFFER
5826
5827 011252 016037 000042 001370      MOV      RMER2(RO),RMER2I    ;STORE RMER2 IN INPUT BUFFER
5828 011260 020337 001334      CMP      R3,RMDAI
5829 011264 001015      BNE     30$
5830 011266 052737 161577 001360      BIS     #XNUOF,RMOFI
5831 011274 020337 001360      CMP      R3,RMOFI
5832 011300 001007      BNE     30$
5833 011302 052737 001567 001370      BIS     #XNUER2,RMER2I
5834 011310 020337 001370      CMP      R3,RMER2I
5835 011314 001001      BNE     30$
5836 011316 104012      ERROR   12                ;REG SEL 2 IS S-A-0
5837 011320      30$:
5838
5839      ;TEST REG SEL 2 FOR S-A-1
5840 011320 012760 000040 000010      MOV      #CLR,RMCS2(RO)    ;CLEAR THE MASSBUS
5841 011326 111160 000010      MOV      (R1),RMCS2(RO)    ;SELECT UNIT
5842
5843 011332 010260 000014      MOV      R2,RMER1(RO)      ;LOAD RMER1
5844
5845 011336 010260 000034      MOV      R2,RMDC(RO)       ;LOAD RMDC
5846
5847 011342 012760 000076 000000      MOV      #ILF76,RMCS1(RO)    ;LOAD RMCS1
5848
5849 011350 010360 000030      MOV      R3,RMSN(RO)       ;LOAD RMSN
5850
5851 011354 016037 000014 001342      MOV      RMER1(RO),RMER1I    ;STORE RMER1 IN INPUT BUFFER
5852
5853 011362 016037 000034 001362      MOV      RMDC(RO),RMDCI    ;STORE RMDC IN INPUT BUFFER
5854 011370 052737 177701 001342      BIS     #CILF76,RMER1I
5855 011376 020337 001342      CMP      R3,RMER1I
5856 011402 001007      BNE     40$
5857 011404 052737 176000 001362      BIS     #XNUDC,RMDCI
5858 011412 020337 001362      CMP      R3,RMDCI
5859 011416 001001      BNE     40$
5860 011420 104013      ERROR   13                ;REG SEL 2 IS S-A-1
5861 011422      40$:
5862
5863      ;TEST REG SEL 4 FOR S-A-0
5864 011422 012760 000040 000010      MOV      #CLR,RMCS2(RO)    ;CLEAR THE MASSBUS
5865 011430 111160 000010      MOV      (R1),RMCS2(RO)    ;SELECT UNIT
5866
5867 011434 010260 000014      MOV      R2,RMER1(RO)      ;LOAD RMER1
5868
5869 011440 010260 000032      MOV      R2,RMOF(RO)       ;LOAD RMOF
5870
5871 011444 010260 000034      MOV      R2,RMDC(RO)       ;LOAD RMDC
5872
5873 011450 010360 000026      MOV      R3,RMDT(RO)       ;LOAD RMDT
5874
5875 011454 010360 000042      MOV      R3,RMER2(RO)      ;LOAD RMER2

```

```

5876
5877 011460 010360 000044      MOV      R3,RMEC1(RO)      ;LOAD RMEC1
5878
5879 011464 016037 000014 001342  MOV      RMER1(RO),RMER1I      ;STORE RMER1 IN INPUT BUFFER
5880
5881 011472 016037 000032 001360  MOV      RMOF(RO),RMOFI      ;STORE RMOF IN INPUT BUFFER
5882
5883 011500 016037 000034 001362  MOV      RMDC(RO),RMDCI      ;STORE RMDC IN INPUT BUFFER
5884 011506 020337 001342      CMP      R3,RMER1I
5885 011512 001015      BNE     S0$
5886 011514 052737 161577 001360  BIS     #XNUOF,RMOFI
5887 011522 020337 001360      CMP      R3,RMOFI
5888 011526 001007      BNE     S1$
5889 011530 052737 176000 001362  BIS     #XNUDC,RMDCI
5890 011536 020337 001362      CMP      R3,RMDCI
5891 011542 001001      BNE     S0$
5892 011544 104014      ERROR   14      ;REG SEL 4 IS S-A-0
5893 011546
5894
5895
5896 011546 012760 000040 000010 ;TEST REG SEL 4 FOR S-A-1
5897 011554 111160 000010      MOV      #CLR,RMCS2(RO)      ;CLEAR THE MASSBUS
5898
5899 011560 010260 000006      MOVVB   (R1),RMCS2(RO)      ;SELECT UNIT
5900
5901 011564 010260 000042      MOV      R2,RMDA(RO)      ;LOAD RMDA
5902
5903 011570 010360 000012      MOV      R2,RMER2(RO)      ;LOAD RMER2
5904
5905 011574 010360 000032      MOV      R3,RMDS(RO)      ;LOAD RMDS
5906
5907 011600 016037 000006 001334  MOV      R3,RMOF(RO)      ;LOAD RMOF
5908
5909 011600 016037 000006 001334  MOV      RMDA(RO),RMDAI      ;STORE RMDA IN INPUT BUFFER
5910 011606 016337 000042 001370  MOV      RMER2(RO),RMER2I      ;STORE RMER2 IN INPUT BUFFER
5911 011614 020337 001334      CMP      R3,RMDAI
5912 011620 001007      BNE     60$
5913 011622 052737 001567 001370  BIS     #XNUER2,RMER2I
5914 011630 020337 001370      CMP      R3,RMER2I
5915 011634 001001      BNE     60$
5916 011636 104015      ERROR   15      ;REG SEL 4 IS S-A-1
5917 011640
5918
5919 011640 012760 000040 000010 ;TEST REG SEL 8 FOR S-A-0
5920 011646 111160 000010      MOV      #CLR,RMCS2(RO)      ;CLEAR THE MASSBUS
5921
5922 011652 010260 000014      MOVVB   (R1),RMCS2(RO)      ;SELECT UNIT
5923
5924 011652 010260 000014      MOV      R2,RMER1(RO)      ;LOAD RMER1
5925
5926 011656 010260 000006      MOV      R2,RMDA(RO)      ;LOAD RMDA
5927
5928 011662 010360 000034      MOV      R3,RMDC(RO)      ;LOAD RMDC
5929
5930 011666 010360 000042      MOV      R3,RMER2(RO)      ;LOAD RMER2
5931 011672 016037 000014 001342  MOV      RMER1(RO),RMER1I      ;STORE RMER1 IN INPUT BUFFER

```

```

5932 011700 016037 000006 001334      MOV      RMDA(R0),RMDAI ;STORE RMDA IN INPUT BUFFER
5933 011706 020337 001342      CMP      R3,RMER1I
5934 011712 001004      BNE     70$
5935 011714 020337 001334      CMP      R3,RMDAI
5936 011720 001001      BNE     70$
5937 011722 104016      ERROR   16 ;REG SEL 8 IS S-A-0
5938 011724
5939
5940 ;TEST REG SEL 8 FOR S-A-1
5941 011724 012760 000040 000010      MOV      #CLR,RMCS2(R0) ;CLEAR THE MASSBUS
5942 011732 111160 000010      MOVVB   (R1),RMCS2(R0) ;SELECT UNIT
5943
5944 011736 010260 000032      MOV      R2,RMOF(R0) ;LOAD RMOF
5945
5946 011742 010260 000034      MOV      R2,RMDC(R0) ;LOAD RMDC
5947
5948 011746 010260 000042      MOV      R2,RMER2(R0) ;LOAD RMER2
5949
5950 011752 010360 000012      MOV      R3,RMDS(R0) ;LOAD RMDS
5951
5952 011756 010360 000014      MOV      R3,RMER1(R0) ;LOAD RMER1
5953
5954 011762 010360 000006      MOV      R3,RMDA(R0) ;LOAD RMDA
5955
5956 011766 016037 000032 001360      MOV      RMOF(R0),RMOFI ;STORE RMOF IN INPUT BUFFER
5957
5958 011774 016037 000034 001362      MOV      RMDC(R0),RMDCI ;STORE RMDC IN INPUT BUFFER
5959
5960 012002 016037 000042 001370      MOV      RMER2(R0),RMER2I ;STORE RMER2 IN INPUT BUFFER
5961 012010 052737 161577 001360      BIS     #XNUOF,RMOFI
5962 012016 001015      BNE     80$
5963 012020 022737 176000 001362      CMP     #XNUDC,RMDCI
5964 012026 020337 001362      CMP     R3,RMDCI
5965 012032 001007      BNE     80$
5966 012034 052737 001567 001370      BIS     #XNUER2,RMER2I
5967 012042 020337 001370      CMP     R3,RMER2I
5968 012046 001001      BNE     80$
5969 012050 104017      ERROR   17 ;REG SEL 8 IS S-A-1
5970 012052
5971
5972 ;REGISTER SELECT 16 IS TESTED BY THE ILR TEST
5973
5974 ;*****
5975 ;TEST 7 DRIVE TYPE TEST
5976 ;*****
5977
5978 012052 000004      TST7:  SCOPE
5979 012054 012737 000007 001226      MOV     #7,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
5980
5981 012062 000240      NOP
5982 012064 012737 000024 001120      MOV     #20,$ICNT ;20 ITERATIONS
5983 012072 112737 000001 001131      MOVVB  #1,$ERMAX ;ONE ERROR ALLOWED
5984 012100 012737 012114 001122      MOV     #T7,$LPADR ;LOAD LOOP ON TEST ADDRESS
5985 012106 012737 012114 001124      MOV     #T7,$LPERR ;LOAD LOOP ON ERROR ADDRESS
5986 012114
5987 012114 012706 001100      T7:    MOV     #STACK,SP ;LOAD THE STACK POINTER

```

```

5988 012120 013700 001276      MOV      $BASE,RO      ;RO = UNIBUS ADDRESS OF UUT
5989 012124 013701 001456      MOV      TSTQUE,R1     ;R1 = POINTER TO DEVICE
5990
5991 012130 016037 000026 001142  MOV      RMDT(RO), $BDDAT ; STORE RMDT AT $BDDAT
5992 012136 022737 020024 001142  CMP      #SNGPRT, $BDDAT ; SINGLE PORT RMO3??
5993 012144 001432                BEQ      10$           ; YES!!
5994 012146 022737 024024 001142  CMP      #DULPRT, $BDDAT ; DUAL PORT RMO3??
5995 012154 001426                BEQ      10$           ; YES!!
5996 012156 022737 020025 001142  CMP      #SNGPRT!BITO, $BDDAT ; SINGLE PORT RMO2 ?
5997 012164 001422                BEQ      10$           ; BRANCH IF SO
5998 012166 022737 024025 001142  CMP      #DULPRT!BITO, $BDDAT ; DUAL PORT RMO2 ?
5999 012174 001416                BEQ      10$           ; BRANCH IF SO
6000 012176 012737 020024 001174  MOV      #SNGPRT, $TMP0 ; LOAD ACCEPTABLE VALUES
6001 012204 012737 024024 001176  MOV      #DULPRT, $TMP1
6002 012212 010037 001136      MOV      R0, $BADDR ; LOAD BAD ADDRESS
6003 012216 062737 000026 001136  ADD      #RMDT, $BADDR
6004 012224 104057      ERROR   57           ; DEVICE NOT AN RMO3
6005 012226 000137 057176      JMP      $EOSP        ; GO TO NEXT DEVICE
6006 012232
6007
6008
6009
6010
6011
6012 012232 000004      *TEST 10  DEVICE AVAILABLE TEST
6013 012234 012737 000010 001226  *ST10:  SCOPE
6014
6015 012242 000240      MOV      #10, $TESTN ; ; SET TEST NUMBER IN APT MAIL BOX
6016 012244 012737 000024 001120  NOP
6017 012252 112737 000001 001131  MOV      #20, $ICNT ; 20 ITERATIONS
6018 012260 012737 012274 001122  MOV      #1, $ERMAX ; ONE ERROR ALLOWED
6019 012266 012737 012274 001124  MOV      #T10, $LPADR ; LOAD LOOP ON TEST ADDRESS
6020 012274      MOV      #T10, $LPERR ; LOAD LOOP ON ERROR ADDRESS
6021 012274 012706 001100  T10:
6022 012300 013700 001276      MOV      #STACK, SP ; LOAD THE STACK POINTER
6023 012304 013701 001456      MOV      $BASE, RO ; RO = UNIBUS ADDRESS OF UUT
6024 012310 012760 000040 000010  MOV      TSTQUE, R1 ; R1 = POINTER TO DEVICE
6025 012316 111160 000010      MOV      #CLR, RMCS2(RO) ; CLEAR THE MASSBUS
6026
6027 012322 016037 000000 001142  MOV      (R1), RMCS2(RO) ; SELECT UNIT
6028 012330 042737 173777 001142  MOV      RMCS1(RO), $BDDAT ; STORE RMCS1 AT $BDDAT
6029 012336 001006      BIC      #1CDVA, $BDDAT ; CLEAR ALL BUT DVA
6030 012340 012737 004000 001140  BNE     10$           ; BRANCH IF DVA SET
6031 012346 010037 001136      MOV      #DVA, $GDDAT ; SETUP EXPECTED
6032 012352 104060      MOV      RO, $BADDR ; SETUP REG ADDRESS
6033 012354      ERROR   60           ; DEVICE NOT AVAILABLE
6034
6035
6036
6037
6038
6039 012354 000004      *TEST 11  HOLDING REGISTER TRANSFER TEST
6040 012356 012737 000011 001226  *ST11:  SCOPE
6041
6042 012364 000240      MOV      #11, $TESTN ; ; SET TEST NUMBER IN APT MAIL BOX
6043 012366 012737 000024 001120  NOP
6044
6045
6046
6047
6048
6049
6050
6051
6052
6053
6054
6055
6056
6057
6058
6059
6060
6061
6062
6063
6064
6065
6066
6067
6068
6069
6070
6071
6072
6073
6074
6075
6076
6077
6078
6079
6080
6081
6082
6083
6084
6085
6086
6087
6088
6089
6090
6091
6092
6093
6094
6095
6096
6097
6098
6099
6100
6101
6102
6103
6104
6105
6106
6107
6108
6109
6110
6111
6112
6113
6114
6115
6116
6117
6118
6119
6120
6121
6122
6123
6124
6125
6126
6127
6128
6129
6130
6131
6132
6133
6134
6135
6136
6137
6138
6139
6140
6141
6142
6143
6144
6145
6146
6147
6148
6149
6150
6151
6152
6153
6154
6155
6156
6157
6158
6159
6160
6161
6162
6163
6164
6165
6166
6167
6168
6169
6170
6171
6172
6173
6174
6175
6176
6177
6178
6179
6180
6181
6182
6183
6184
6185
6186
6187
6188
6189
6190
6191
6192
6193
6194
6195
6196
6197
6198
6199
6200
6201
6202
6203
6204
6205
6206
6207
6208
6209
6210
6211
6212
6213
6214
6215
6216
6217
6218
6219
6220
6221
6222
6223
6224
6225
6226
6227
6228
6229
6230
6231
6232
6233
6234
6235
6236
6237
6238
6239
6240
6241
6242
6243
6244
6245
6246
6247
6248
6249
6250
6251
6252
6253
6254
6255
6256
6257
6258
6259
6260
6261
6262
6263
6264
6265
6266
6267
6268
6269
6270
6271
6272
6273
6274
6275
6276
6277
6278
6279
6280
6281
6282
6283
6284
6285
6286
6287
6288
6289
6290
6291
6292
6293
6294
6295
6296
6297
6298
6299
6300
6301
6302
6303
6304
6305
6306
6307
6308
6309
6310
6311
6312
6313
6314
6315
6316
6317
6318
6319
6320
6321
6322
6323
6324
6325
6326
6327
6328
6329
6330
6331
6332
6333
6334
6335
6336
6337
6338
6339
6340
6341
6342
6343
6344
6345
6346
6347
6348
6349
6350
6351
6352
6353
6354
6355
6356
6357
6358
6359
6360
6361
6362
6363
6364
6365
6366
6367
6368
6369
6370
6371
6372
6373
6374
6375
6376
6377
6378
6379
6380
6381
6382
6383
6384
6385
6386
6387
6388
6389
6390
6391
6392
6393
6394
6395
6396
6397
6398
6399
6400
6401
6402
6403
6404
6405
6406
6407
6408
6409
6410
6411
6412
6413
6414
6415
6416
6417
6418
6419
6420
6421
6422
6423
6424
6425
6426
6427
6428
6429
6430
6431
6432
6433
6434
6435
6436
6437
6438
6439
6440
6441
6442
6443
6444
6445
6446
6447
6448
6449
6450
6451
6452
6453
6454
6455
6456
6457
6458
6459
6460
6461
6462
6463
6464
6465
6466
6467
6468
6469
6470
6471
6472
6473
6474
6475
6476
6477
6478
6479
6480
6481
6482
6483
6484
6485
6486
6487
6488
6489
6490
6491
6492
6493
6494
6495
6496
6497
6498
6499
6500

```

C11

CZRMJBO RM03/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 132
T11 HOLDING REGISTER TRANSFER TEST

SEQ 0132

6044	012374	112737	000001	001131	MOV	#1, \$ERMAX	; ONE ERROR ALLOWED
6045	012402	012737	012416	001122	MOV	#T11, \$LPAOR	; LOAD LOOP ON TEST ADDRESS
6046	012410	012737	012416	001124	MOV	#T11, \$LPEPR	; LOAD LOOP ON ERROR ADDRESS
6047	012416				T11:		
6048	012416	012706	001100		MOV	#STACK, SP	; LOAD THE STACK POINTER
6049	012422	013700	001276		MOV	\$BASE, R0	; R0 = UNIBUS ADDRESS OF UUT
6050	012426	013701	001456		MOV	TSTQUE, R1	; R1 = POINTER TO DEVICE
6051	012432	012760	000040	000010	MOV	#CLR, RMCS2(R0)	; CLEAR THE MASSBUS
6052	012440	111160	000010		MOV	(R1), RMCS2(R0)	; SELECT UNIT
6053	012444	005003			CLR	R3	; CLEAR ERROR FLAGS
6054	012446	010037	001136		MOV	R0, \$B0ADR	; SETUP REGISTER ADDRESS
6055	012452	062737	000036	001136	ADD	#RMHR, \$B0ADR	
6056					; WRITE ONES THEN ZEROS IN RMHR AND CHECK FOR S-A-1 BITS.		
6057					; NOTE THAT IT IS NECESSARY TO WRITE SOME OTHER REGISTER IN		
6058					; ORDER TO WRITE THE HOLDING REGISTER, AND RMDA IS USED FOR THIS		
6059					; PURPOSE.		
6060							
6061	012460	012760	177777	000006	MOV	#-1, RMDA(R0)	; LOAD RMDA
6062							
6063	012466	012760	000000	000006	MOV	#0, RMDA(R0)	; LOAD RMDA
6064							
6065	012474	016037	000036	001142	MOV	RMHR(R0), \$B0DAT	; STORE RMHR AT \$B0DAT
6066	012502	005137	001142		COM	\$B0DAT	; ANY ERROR??
6067	012506	001405			BEQ	10\$; NO!!
6068	012510	005037	001140		CLR	\$G0DAT	; LOAD EXPECTED
6069	012514	104061			ERROR	61	
6070	012516	052703	000001		BIS	#BIT0, R3	; SET ERROR FLAGS
6071	012522						
6072					10\$:		
6073					; *****		
6074					; WRITE ZEROS THEN ONES IN RMHR AND CHECK FOR S-A-0 BITS.		
6075	012522	012760	000000	000006	MOV	#0, RMDA(R0)	; LOAD RMDA
6076							
6077	012530	012760	177777	000006	MOV	#-1, RMDA(R0)	; LOAD RMDA
6078							
6079	012536	016037	000036	001142	MOV	RMHR(R0), \$B0DAT	; STORE RMHR AT \$B0DAT
6080	012544	005137	001142		COM	\$B0DAT	; RMHR IS COMPLEMENTED WHEN READ
6081	012550	012737	177777	001140	MOV	#-1, \$G0DAT	; SETUP EXPECTED
6082	012556	023737	001140	001142	CMP	\$G0DAT, \$B0DAT	; ANY ERROR??
6083	012564	001403			BEQ	20\$; NO!!
6084	012566	104062			ERROR	62	
6085	012570	052703	000002		BIS	#BIT1, R3	; SET ERROR FLAG
6086	012574						
6087					20\$:		
6088					; *****		
6089	012574	005703			; IF NO PREVIOUS ERRORS, WRITE AND READ SHIFTING ONE BIT PATTERN.		
6090	012576	001025			TST	R3	; ANY FLAGS SET??
6091	012600	012702	000001		BNE	50\$; YES!!
6092					MOV	#1, R2	; R2=DATA PATTERN
6093	012604						
6094					30\$:		
6095	012604	012760	000000	000006	MOV	#0, RMDA(R0)	; LOAD RMDA
6096							
6097	012612	010260	000006		MOV	R2, RMDA(R0)	; LOAD RMDA
6098							
6099	012616	016037	000036	001142	MOV	RMHR(R0), \$B0DAT	; STORE RMHR AT \$B0DAT

```

6100 012624 005137 001142          COM      $BDDAT          ;RMHR IS COMPLEMENTED
6101 012630 023702 001142          CMP      $BDDAT,R2      ;ANY ERROR??
6102 012634 001404                    BEQ      40$             ;NO!!
6103 012636 010237 001140          MOV      R2,$GDDAT      ;SETUP EXPECTED
6104 012642 104063                    ERROR   63
6105 012644 000402                    BR       50$             ;DO NOT COLLECT ALL ERRORS
6106
6107 012646 006302          40$:   ASL      R2          ;SHIFT TO NEXT PATTERN
6108 012650 001355                    BNE     30$             ;CONTINUE IF NOT DONE
6109
6110 012652          50$:
6111
6112          ;*****
6113          ;*TEST 12          CONTROL STATUS #1 TRANSFER TEST
6114          ;*****
6115          ;*****
6116 012652 000004          †ST12:  SCOPE
6117 012654 012737 000012 001226          MOV      #12,$TESTN    ;;SET TEST NUMBER IN APT MAIL BOX
6118
6119 012662 000240                    NOP
6120 012664 012737 000024 001120          MOV      #20,$ICNT     ;20 ITERATIONS
6121 012672 112737 000001 001131          MOV      #1,$ERMAX    ;ONE ERROR ALLOWED
6122 012700 012737 012714 001122          MOV      #T12,$LPADR  ;LOAD LOOP ON TEST ADDRESS
6123 012706 012737 012714 001124          MOV      #T12,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
6124 012714
6125 012714 012706 001100          T12:   MOV      #STACK,SP     ;LOAD THE STACK POINTER
6126 012720 013700 001276          MOV      $BASE,R0     ;R0 = UNIBUS ADDRESS OF UUT
6127 012724 013701 001456          MOV      TSTQUE,R1    ;R1 = POINTER TO DEVICE
6128 012730 005003          CLR      R3           ;R3 = ERROR INDICATOR
6129 012732 012760 000040 000010          MOV      #CLR, RMCS2(R0) ;CLEAR THE MASSBUS
6130 012740 111160 000010          MOV      (R1),RMCS2(R0) ;SELECT UNIT
6131          ;WRITE ONES IN RMCS1, BITS 01-05, THEN CLEAR. READ AND
6132          ;CHECK FOR S-A-1 BITS.
6133
6134 012744 012760 000076 000000          MOV      #ILF76, RMCS1(R0) ;LOAD RMCS1
6135 012752 012760 000040 000010          MOV      #CLR, RMCS2(R0) ;CLEAR THE MASSBUS
6136 012760 111160 000010          MOV      (R1),RMCS2(R0) ;SELECT UNIT
6137
6138 012764 016037 000000 001142          MOV      RMCS1(R0), $BDDAT ;STORE RMCS1 AT $BDDAT
6139 012772 042737 177701 001142          BIC     #†CILF76, $BDDAT
6140 013000 001410                    BEQ      5$
6141 013002 005037 001140                    CLR     $GDDAT
6142 013006 010037 001136                    MOV     R0, $BDAOR
6143 013012 062737 000000 001136          ADD     #RMCS1, $BDAOR
6144 013020 104043                    ERROR   43             ;CANT CLEAR RMCS1
6145 013022
6146          5$:
6147          ;WRITE ONES IN RMCS1, BITS 01-05, THEN WRITE ZEROS. READ AND CHECK FOR
6148          ;S-A-1 BITS.
6149 013022 012760 000076 000000          MOV      #ILF76, RMCS1(R0) ;LOAD RMCS1
6150
6151 013030 012760 000000 000000          MOV      #0, RMCS1(R0)  ;LOAD RMCS1
6152
6153 013036 016037 000000 001142          MOV      RMCS1(R0), $BDDAT ;STORE RMCS1 AT $BDDAT
6154 013044 042737 177701 001142          BIC     #†CILF76, $BDDAT
6155 013052 001412                    BEQ     10$

```

E11

CZRMJBO RM03/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 134
T12 CONTROL STATUS #1 TRANSFER TEST

SEQ 0134

```

6156 013054 005037 001140 CLR $GDDAT
6157 013060 010037 001136 MOV RO,$BDADR
6158 013064 062737 000000 001136 ADD #RMCS1,$BDADR
6159 013072 104023 ERROR 23 ;CANT WRITE 0'S
6160 013074 052703 000001 BIS #BIT0,R3 ;SET ERROR FLAG
6161 013100
6162 10$: ;WRITE ZEROS IN RMCS1, THEN ONES, READ AND CHECK S-A-O BITS.
6163
6164 013100 012760 000000 000000 MOV #0,RMCS1(RO) ;LOAD RMCS1
6165
6166 013106 012760 000076 000000 MOV #ILF76,RMCS1(RO) ;LOAD RMCS1
6167
6168 013114 016037 000000 001142 MOV RMCS1(RO),$BDDAT ;STORE RMCS1 AT $BDDAT
6169 013122 042737 177701 001142 BIC #CILF76,$BDDAT
6170 013130 012737 000076 001140 MOV #ILF76,$GDDAT
6171 013136 023737 001140 001142 CMP $GDDAT,$BDDAT
6172 013144 001410 BEQ 20$
6173 013146 010037 001136 MOV RO,$BDADR
6174 013152 062737 000000 001136 ADD #RMCS1,$BDADR
6175 013160 104024 ERROR 24 ;CANT WRITE ONES
6176 013162 052703 000002 BIS #BIT1,R3 ;SET ERROR FLAG
6177 013166
6178 20$: ;WRITE A SHIFTING ONE BIT PATTERN IN RMCS1, READ AND CHECK FOR STUCK BITS.
6179 013166 005703 TST R3 ;OMIT IF ANY ERRORS
6180 013170 001035 BNE 50$
6181 013172 012702 000002 MOV #2,R2 ;R2 = TEST PATTERN
6182 013176
6183 013176 010203 30$: MOV R2,R3 ;R3 = EXPECTED RESULT, BITS 1-5
6184 013200 042703 177701 BIC #CILF76,R3
6185
6186 013204 012760 000000 000000 MOV #0,RMCS1(RO) ;LOAD RMCS1
6187
6188 013212 010260 000000 MOV R2,RMCS1(RO) ;LOAD RMCS1
6189
6190 013216 016037 000000 001142 MOV RMCS1(RO),$BDDAT ;STORE RMCS1 AT $BDDAT
6191 013224 042737 177701 001142 BIC #CILF76,$BDDAT
6192 013232 020337 001142 CMP R3,$BDDAT
6193 013236 001410 BEQ 40$
6194 013240 010337 001140 MOV R3,$GDDAT
6195 013244 010037 001136 MOV RO,$BDADR
6196 013250 062737 000000 001136 ADD #RMCS1,$BDADR
6197 013256 104025 ERROR 25 ;CANT WRITE SHIFTING ONES
6198 013260 006302 40$: ASL R2 ;SHIFT TO NEXT BIT
6199 013262 001345 BNE 30$ ;CONTINUE IF R2 NOT ZERO
6200 013264
6201
6202
6203 ;*****
6204 ;*TEST 13 ERROR REGISTER 1 TRANSFER TEST
6205 ;*****
6206 013264 000004 6206: ST13: SCOPE
6207 013266 012737 000013 001226 MOV #13,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
6208
6209 013274 000240 NOP
6210 013276 012737 000024 001120 MOV #20,$ICNT ;20 ITERATIONS
6211 013304 112737 000001 001131 MOVB #1,$ERMAX ;ONE ERROR ALLOWED

```


F11

CZRMJ80 RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 135
T13 ERROR REGISTER 1 TRANSFER TEST

SEQ 0135

6212	013312	012737	013326	001122		MOV	#T13,\$LPADR	;LOAD LOOP ON TEST ADDRESS
6213	013320	012737	013326	001124		MOV	#T13,\$LPERR	;LOAD LOOP ON ERROR ADDRESS
6214	013326				T13:			
6215	013326	012706	001100			MOV	#STACK,SP	;LOAD THE STACK POINTER
6216	013332	013700	001276			MOV	\$BASE,R0	;R0 = UNIBUS ADDRESS OF UUT
6217	013336	013701	001456			MOV	TSTQUE,R1	;R1 = POINTER TO DEVICE
6218	013342	005003				CLR	R3	;CLEAR ERROR FLAG
6219								;WRITE ONES IN RMER1, CLEAR AND CHECK FOR S-A-I BITS
6220								
6221	013344	012760	177777	000014		MOV	#-1,RMER1(R0)	;LOAD RMER1
6222	013352	012760	000040	000010		MOV	#CLR,RMCS2(R0)	;CLEAR THE MASSBUS
6223	013360	111160	000010			MOV	(R1),RMCS2(R0)	;SELECT UNIT
6224								
6225	013364	016037	000014	001342		MOV	RMER1(R0),RMER1I	;STORE RMER1 IN INPUT BUFFER
6226	013372	013737	001342	001142		MOV	RMER1I,\$BDDAT	
6227	013400	042737	177760	001142		BIC	#1C<PAR!RMR!ILF!ILR>,\$BDDAT	
6228	013406	001410				BEQ	10\$	
6229	013410	005037	001140			CLR	\$GDDAT	
6230	013414	010037	001136			MOV	R0,\$BDADR	
6231	013420	062737	000014	001136		ADD	#RMER1,\$BDADR	
6232	013426	104027				ERROR	27	;CANT CLEAR RMER1
6233	013430				10\$:			
6234	013430	013737	001342	001142		MOV	RMER1I,\$BDDAT	
6235	013436	042737	074017	001142		BIC	#1C<DCK!IAE!AOE!HCRC!HCE!ECH!WCF!FER>,\$BDDAT	
6236	013444	001410				BEQ	20\$	
6237	013446	005037	001140			CLR	\$GDDAT	
6238	013452	010037	001136			MOV	R0,\$BDADR	
6239	013456	062737	000014	001136		ADD	#RMER1,\$BDADR	
6240	013464	104030				ERROR	30	
6241	013466				20\$:			
6242	013466	013737	001342	001142		MOV	RMER1I,\$BDDAT	
6243	013474	042737	147777	001142		BIC	#1C<OP1!DTE>,\$BDDAT	
6244	013502	001410				BEQ	30\$	
6245	013504	005037	001140			CLR	\$GDDAT	
6246	013510	010037	001136			MOV	R0,\$BDADR	
6247	013514	062737	000014	001136		ADD	#RMER1,\$BDADR	
6248	013522	104031				ERROR	31	
6249	013524				30\$:			
6250								;WRITE ONES THEN ZEROS IN RMER1, READ AND CHECK FOR S-A-I BITS
6251								
6252	013524	012760	177777	000014		MOV	#-1,RMER1(R0)	;LOAD RMER1
6253								
6254	013532	012760	000000	000014		MOV	#0,RMER1(R0)	;LOAD RMER1
6255								
6256	013540	016037	000014	001342		MOV	RMER1(R0),RMER1I	;STORE RMER1 IN INPUT BUFFER
6257	013546	013737	001342	001142		MOV	RMER1I,\$BDDAT	
6258	013554	042737	177770	001142		BIC	#1C<RMR!ILF!ILR>,\$BDDAT	
6259	013562	001412				BEQ	40\$	
6260	013564	005037	001140			CLR	\$GDDAT	
6261	013570	010037	001136			MOV	R0,\$BDADR	
6262	013574	062737	000014	001136		ADD	#RMER1,\$BDADR	
6263	013602	104032				ERROR	32	
6264	013604	052703	000001			BIS	#BIT0,R3	;SET ERROR FLAG
6265	013610	013737	001342	001142	40\$:	MOV	RMER1I,\$BDDAT	
6266	013616	042737	074017	001142		BIC	#1C<DCK!IAE!AOE!HCRC!HCE!ECH!WCF!FER>,\$BDDAT	
6267	013624	001412				BEQ	50\$	

G11

CZRMJ80 RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 136
T13 ERROR REGISTER 1 TRANSFER TEST

6268	013626	005037	001140			CLR	\$GDDAT	
6269	013632	010037	001136			MOV	RO, \$BDADR	
6270	013636	062737	000014	001136		ADD	#RMER1, \$BDADR	
6271	013644	104033				ERROR	33	
6272	013646	052703	000001			BIS	#BIT0, R3	; SET ERROR FLAG
6273	013652				50\$:			
6274	013652	013737	001342	001142		MOV	RMER1, \$BDDAT	
6275	013660	042737	147777	001142		BIC	#C(OP!DTE), \$BDDAT	
6276	013666	001412				BEQ	60\$	
6277	013670	005037	001140			CLR	\$GDDAT	
6278	013674	010037	001136			MOV	RO, \$BDADR	
6279	013700	062737	000014	001136		ADD	#RMER1, \$BDADR	
6280	013706	104034				ERROR	34	
6281	013710	052703	030000			BIS	#BIT, R3	
6282	013714				60\$:			
6283						;WRITE ZEROS THEN ONES IN RMER1, READ AND CHECK FOR S-A-O BITS		
6284								
6285	013714	012760	000000	000014		MOV	#0, RMER1(RO)	; LOAD RMER1
6286								
6287	013722	012760	177777	000014		MOV	#-1, RMER1(RO)	; LOAD RMER1
6288								
6289	013730	016037	000014	001142		MOV	RMER1(RO), \$BDDAT	; STORE RMER1 AT \$BDDAT
6290	013736	012737	177777	001140		MOV	#-1, \$GDDAT	
6291	013744	023737	001140	001142		CMP	\$GDDAT, \$BDDAT	
6292	013752	001410				BEQ	70\$	
6293	013754	010037	001136			MOV	RO, \$BDADR	
6294	013760	062737	000014	001136		ADD	#RMER1, \$BDADR	
6295	013766	104035				ERROR	35	
6296	013770	052703	000002			BIS	#BIT1, R3	
6297	013774				70\$:			
6298						;WRITE A SHIFTING 1 BIT IN RMER1 AND CHECK FOR STUCK BITS		
6299						;NOTE: DONT TEST UNSAFE OR PARITY		
6300	013774	005703				TST	R3	; SKIP THIS PART IF ANY ERRORS
6301	013776	001045				BNE	120\$	
6302	014000	012702	000001			MOV	#1, R2	; R2 = TEST PATTERN
6303	014004				80\$:			
6304	014004	012760	000040	000010		MOV	#CLR, RMCS2(RO)	; CLEAR THE MASSBUS
6305	014012	111160	000010			MOVB	(R1), RMCS2(RO)	; SELECT UNIT
6306								
6307	014016	010260	000014			MOV	R2, RMER1(RO)	; LOAD RMER1
6308								
6309	014022	016037	000014	001142		MOV	RMER1(RO), \$BDDAT	; STORE RMER1 AT \$BDDAT
6310	014030	032702	000010			BIT	#PAR, R2	; DONT TEST PAR = 0
6311	014034	001003				BNE	90\$	
6312	014036	042737	000010	001142		BIC	#PAR, \$BDDAT	
6313	014044	032702	040000		90\$:	BIT	#UNS, R2	; DONT TEST UNS = 0
6314	014050	001003				BNE	100\$	
6315	014052	042737	040000	001142		BIC	#UNS, \$BDDAT	
6316	014060	020237	001142		100\$:	CMP	R2, \$BDDAT	
6317	014064	001410				BEQ	110\$	
6318	014066	010237	001140			MOV	R2, \$GDDAT	
6319	014072	010037	001136			MOV	RO, \$BDADR	
6320	014076	062737	000014	001136		ADD	#RMER1, \$BDADR	
6321	014104	104036				ERROR	36	
6322								
6323	014106	006302			110\$:	ASL	R2	; SHIFT TO NEXT BIT

```

6324 014110 001335
6325 014112
6326
6327
6328
6329
6330
6331
6332 014112 000004
6333 014114 012737 000014 001226
6334
6335 014122 000240
6336 014124 012737 000024 001120
6337 014132 112737 000001 001131
6338 014140 012737 014154 001122
6339 014146 012737 014154 001124
6340 014154
6341 014154 012706 001100
6342 014160 013700 001276
6343 014164 013701 001456
6344 014170 005003
6345 014172 010037 001136
6346 014176 062737 000042 001136
6347 014204 005037 001140
6348
6349
6350
6351 014210 012760 177777 000042
6352 014216 012760 000040 000010
6353 014224 111160 000010
6354
6355 014230 016037 000042 001370
6356
6357
6358 014236 013737 001370 001142
6359 014244 042737 176210 001142
6360 014252 001403
6361 014254 104044
6362 014256 052703 000001
6363 014262
6364
6365 014262 013737 001370 001142
6366 014270 042737 143777 001142
6367 014276 001403
6368 014300 104045
6369 014302 052703 000001
6370 014306
6371
6372 014306 013737 001370 001142
6373 014314 042737 175767 001142
6374 014322 001403
6375 014324 104046
6376 014326 052703 000001
6377 014332
6378
6379

```

```

;*****
;TEST 14 ERROR REGISTER 2 TRANSFER TEST
;*****
T14:
ST14: SCOPE
MOV #14,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
NOP
MOV #20,$ICNT ;20 ITERATIONS
MOVB #1,$EMAX ;ONE ERROR ALLOWED
MOV #T14,$LPADR ;LOAD LOOP ON TEST ADDRESS
MOV #T14,$LPERR ;LOAD LOOP ON ERROR ADDRESS
MOV #STACK,$SP ;LOAD THE STACK POINTER
MOV $BASE,$R0 ;R0 = UNIBUS ADDRESS OF UUT
MOV TSTQUE,$R1 ;R1 = POINTER TO DEVICE
CLR R3 ;RESET ERROR FLAGS
MOV R0,$BADDR ;SETUP BAD ADDRESS
ADD #RMR2,$BADDR
CLR $GDDAT ;SETUP EXPECTED DATA
;*****
;WRITE ONES IN RMR2, CLEAR AND CHECK FOR S-A-1 BITS
MOV #-1,RMR2($R0) ;LOAD RMR2
MOV #CLR,RMCS2($R0) ;CLEAR THE MASSBUS
MOVB (R1),RMCS2($R0) ;SELECT UNIT
MOV RMR2($R0),RMR2I ;STORE RMR2 IN INPUT BUFFER
;
TEST UNUSED BITS FOR ZERO-FAILURE ON IF
MOV RMR2I,$BDDAT
BIC #1<CXNUER2>,$BDDAT
BEQ 10$ ;BRANCH IF NO ERROR
ERROR 44 ;UNUSED BITS NOT ZERO
BIS #BIT0,R3 ;SET ERROR FLAG
10$:
;
TEST "OPE" "IVC" "LSC" FOR ZERO-FAILURE ON CS, IF
MOV RMR2I,$BDDAT
BIC #1<OPE!IVC!LSC>,$BDDAT
BEQ 20$ ;BRANCH IF NO ERROR
ERROR 45 ;CANT CLEAR OPE, IVC, LSC
BIS #BIT0,R3 ;SET ERROR FLAG
20$:
;
TEST "LBC" "DPE" FOR ZERO-FAILURE ON DS, IF
MOV RMR2I,$BDDAT
BIC #1<LBC!DPE>,$BDDAT
BEQ 30$ ;BRANCH IF NO ERROR
ERROR 46 ;CANT CLEAR LBC, DPE
BIS #BIT0,R3 ;SET ERROR FLAG
30$:
;*****

```

```

6380 ;WRITE ONES IN RMER2 THEN WRITE ZEROS AND CHECK FOR S-A-I BITS.
6381
6382 014332 012760 177777 000042      MOV    #-1,RMER2(RO)  ;LOAD RMER2
6383
6384 014340 012760 000000 000042      MOV    #0,RMER2(RO)  ;LOAD RMER2
6385
6386 014346 016037 000042 001370      MOV    RMER2(RO),RMER2I ;STORE RMER2 IN INPUT BUFFER
6387
6388 ;
6389 014354 013737 001370 001142      TEST  "OPE" "IVC" "LSC" FOR ZERO-FAILURE ON CS, IF
6390 014362 042737 143777 001142      MOV    RMER2I,$BDDAT
        BIC  #1C(OPE!IVC!LSC),$BDDAT

```

CZRMJBO RM03/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

6391 014370 001403

MACY11 30(1046)
T14

23-NOV-77 12:14 PAGE 139
ERROR REGISTER 2 TRANSFER TEST

J11

BEQ 405

;BRANCH IF NO ERROR

SEQ 0139

K11

CZRMJ80 RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046)
T14

23-NOV-77 12:14 PAGE 140
ERROR REGISTER 2 TRANSFER TEST

SEQ 0140

6392 014372 104047
6393 014374 052703 000001
6394 014400
6395
6396 014400 013737 001370 001142

405:
:

ERROR 47 ;CANT WRITE ZEROS
BIS #BIT0,R3 ;SET ERROR FLAG
TEST "LBC", "DPE" FOR ZERO-FAILURE ON DS, IF
MOV RMÉR2I, \$BDDAT

```

6397 014406 042737 175767 001142      BIC    #1C(LBC:DPE), $BDDAT
6398 014414 001403                      BEQ    50$      ;BRANCH IF NO ERROR
6399 014416 104050                      ERROR  50      ;CANT WRITE ZEROS
6400 014420 052703 000001                BIS    #BIT0,R3 ;SET ERROR FLAG
6401 014424
6402
6403
6404      ;*****
6405      ;WRITE ZEROS IN RMER2 THEN WRITE ONES AND CHECK FOR S-A-O BITS.
6406 014424 012760 000000 000042      MOV    #0,RMER2(R0) ;LOAD RMER2
6407
6408 014432 012760 177777 000042      MOV    #-1,RMER2(R0) ;LOAD RMER2
6409
6410 014440 016037 000042 001370      MOV    RMER2(R0),RMER2I ;STORE RMER2 IN INPUT BUFFER
6411      ;
6412 014446 013737 001370 001142      TEST  UNUSED BITS FOR ZERO-FAILURE ON IF
6413 014454 042737 176210 001142      MOV    RMER2I,$BDDAT
6414 014462 001403                      BIC    #1CXNUE2,$BDDAT
6415 014464 104044                      BEQ    60$      ;BRANCH IF NO ERROR
6416 014466 052703 000001                ERROR  44      ;UNUSED BITS NOT ZERO
6417 014472                      BIS    #BIT0,R3 ;SET ERROR FLAG
6418
6419      ;
6420 014472 013737 001370 001142      TEST  USED BITS FOR ONE-FAILURE ON IF
6421 014500 042737 001567 001142      MOV    RMER2I,$BDDAT
6422 014506 012737 176210 001140      BIC    #XNUE2,$BDDAT
6423 014514 023737 001140 001142      MOV    #1CXNUE2,$GDDAT
6424 014522 001403                      CMP    $GDDAT,$BDDAT
6425 014524 104051                      BEQ    70$      ;BRANCH IF NO ERROR
6426 014526 052703 000002                ERROR  51      ;CANT WRITE ONES
6427 014532                      BIS    #BIT1,R3 ;SET ERROR FLAG
6428
6429      ;*****
6430      ;IF NO PREVIOUS ERROR, TEST BIT INTERFERENCE WITH SHIFTING ONE BIT.
6431 014532 005703                      TST    R3      ;ANY ERRORS?
6432 014534 001044                      BNE    120$    ;YES!!
6433 014536 012702 000001                MOV    #1,R2   ;R2=DATA PATTERN
6434 014542
6435 014542 012760 000040 000010      80$:   MOV    #CLR, RMCS2(R0) ;CLEAR THE MASSBUS
6436 014550 111160 000010                MOV    (R1),RMCS2(R0) ;SELECT UNIT
6437 014554 010260 000042      MOV    R2,RMER2(R0) ;LOAD RMER2
6438
6439 014560 016037 000042 001142      MOV    RMER2(R0),$BDDAT ;STORE RMER2 AT $BDDAT
6440 014566 032702 040000                      BIT    #SKI,R2   ;IS SKI BEING SET?
6441 014572 001003                      BNE    90$      ;YES!!
6442 014574 042737 040000 001142      BIC    #SKI,$BDDAT ;DONT TEST SKI FOR ZERO
6443 014602 032702 000200                BIT    #DVC,R2  ;IS DVC BEING SET??
6444 014606 001003                      BNE    100$    ;YES!!
6445 014610 042737 000200 001142      BIC    #DVC,$BDDAT ;DONT TEST DVC FOR ZERO
6446 014616 010237 001140 001142      100$:  MOV    R2,$GDDAT
6447 014622 042737 001567 001140      BIC    #XNUE2,$GDDAT ;UNUSED BITS SHOULD BE ZERO
6448 014630 023737 001140 001142      CMP    $GDDAT,$BDDAT ;ANY ERRORS??
6449 014636 001401                      BEQ    110$    ;NO!!
6450 014640 104052                      ERROR  52      ;CANT WRITE SHIFTING ONES
6451
6452 014642 006302                110$:  ASL    R2      ;SHIFT TO NEXT DATA BIT

```

```

6453 014644 001336          BNE      B05          ;CONTINUE IF NOT DONE
6454
6455 014646          1205:
6456
6457
6458 ;*****
6459 ;*TEST 15          CLEAR OFFSET STUCK ACTIVE TEST
6460 ;*****
6461 014646 000004          †ST15: SCOPE
6462 014650 012737 000015 001226      MOV      #15,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6463
6464 014656 000240          NOP
6465 014660 012737 000024 001120      MOV      #20,$ICNT      ;20 ITERATIONS
6466 014666 112737 000001 001131      MOV      #1,$ERMAX      ;ONE ERROR ALLOWED
6467 014674 012737 014710 001122      MOV      #T15,$LPADR    ;LOAD LOOP ON TEST ADDRESS
6468 014702 012737 014710 001124      MOV      #T15,$LPERR    ;LOAD LOOP ON ERROR ADDRESS
6469 014710
6470 014710 012706 001100          T15:  MOV      #STACK,SP      ;LOAD THE STACK POINTER
6471 014714 013700 001276          MOV      $BASE,R0      ;R0 = UNIBUS ADDRESS OF UUT
6472 014720 013701 001456          MOV      TSTQUE,R1     ;R1 = POINTER TO DEVICE
6473 014724 012760 000040 000010      MOV      #CLR,RMCS2(R0) ;CLEAR THE MASSBUS
6474 014732 111160 000010          MOV      (R1),RMCS2(R0);SELECT UNIT
6475
6476 014736 012760 177777 000032      MOV      #-1,RMOF(R0)  ;LOAD RMOF
6477
6478 014744 016037 000032 001142      MOV      RMOF(R0),$BDDAT;STORE RMOF AT $BDDAT
6479 014752 042737 177577 001142      BIC      #COFD,$BDDAT
6480 014760 001011          BNE      105          ;BRANCH IF OFD IS A ONE
6481 014762 012737 000200 001140      MOV      #OFD,$GDDAT   ;SETUP ERROR MESSAGE
6482 014770 010037 001136          MOV      R0,$B0ADR
6483 014774 062737 000032 001136      ADD      #RMOF,$B0ADR
6484 015002 104172          ERROR    172
6485 015004          105:
6486
6487
6488 ;*****
6489 ;*TEST 16          OFFSET REGISTER TRANSFER TEST
6490 ;*****
6491
6492 015004 000004          †ST16: SCOPE
6493 015006 012737 000016 001226      MOV      #16,$TESTN    ;;SET TEST NUMBER IN APT MAIL BOX
6494
6495 015014 000240          NOP
6496 015016 012737 000024 001120      MOV      #20,$ICNT     ;20 ITERATIONS
6497 015024 112737 000001 001131      MOV      #1,$ERMAX     ;ONE ERROR ALLOWED
6498 015032 012737 015046 001122      MOV      #T16,$LPADR   ;LOAD LOOP ON TEST ADDRESS
6499 015040 012737 015046 001124      MOV      #T16,$LPERR   ;LOAD LOOP ON ERROR ADDRESS
6500 015046
6501 015046 012706 001100          T16:  MOV      #STACK,SP      ;LOAD THE STACK POINTER
6502 015052 013700 001276          MC.     $BASE,R0      ;R0 = UNIBUS ADDRESS OF UUT
6503 015056 013701 001456          MOV      TSTQUE,R1     ;R1 = POINTER TO DEVICE
6504 015062 005003          CLR      R3           ;RESET ERROR FLAGS
6505 015064 010037 001136          MOV      R0,$B0ADR    ;SETUP BAD ADDRESS
6506 015070 062737 000032 001136      ADD      #RMOF,$B0ADR
6507 015076 005037 001140          CLR      $GDDAT       ;SETUP EXPECTED DATA
6508 ;*****

```



```

6509 ;WRITE ONES THEN ZEROS IN RMOF AND CHECK FOR S-A-1 BITS.
6510
6511 015102 012760 177777 000032      MOV      #-1,RMOF(R0)      ;LOAD RMOF
6512
6513 015110 012760 000000 000032      MOV      #0,RMOF(R0)      ;LOAD RMOF
6514
6515 015116 016037 000032 001360      MOV      RMOF(R0),RMOFI    ;STORE RMOF IN INPUT BUFFER
6516 ; CHECK UNUSED BITS OF RMOF FOR ZERO
6517 015124 013737 001360 001142      MOV      RMOFI,$BDDAT     ;GET UNUSED BITS
6518 015132 042737 016200 001142      BIC      #1CXNUOF,$BDDAT
6519 015140 001403                      BEQ      10$              ;BRANCH IF NO ERROR
6520 015142 104053                      ERROR    53              ;UNUSED BITS NOT ZERO RMOF
6521 015144 052703 000001      BIS      #BIT0,R3        ;SET ERROR FLAG
6522 015150
6523 10$:
6524 ; CHECK USED BITS OF RMOF FOR ZERO
6525 015150 013737 001360 001142      MOV      RMOFI,$BDDAT     ;GET USED BITS
6526 015156 042737 161577 001142      BIC      #XNUOF,$BDDAT
6527 015164 001403                      BEQ      20$              ;BRANCH IF NO ERROR
6528 015166 104054                      ERROR    54              ;USED BITS RMOF S-A-1
6529 015170 052703 000001      BIS      #BIT0,R3        ;SET ERROR FLAG
6530 015174
6531 ;*****
6532 ;WRITE ZEROS THEN ONES ON RMOF AND CHECK FOR S-A-0 BITS.
6533
6534 015174 012760 000000 000032      MCV      #0,RMOF(R0)      ;LOAD RMOF
6535
6536 015202 012760 177777 000032      MOV      #-1,RMOF(R0)      ;LOAD RMOF
6537
6538 015210 016037 000032 001360      MOV      RMOF(R0),RMOFI    ;STORE RMOF IN INPUT BUFFER
6539 ; CHECK UNUSED BITS OF RMOF FOR ZERO
6540 015216 013737 001360 001142      MOV      RMOFI,$BDDAT     ;GET UNUSED BITS
6541 015224 042737 016200 001142      BIC      #1CXNUOF,$BDDAT
6542 015232 001403                      BEQ      30$              ;BRANCH IF NO ERROR
6543 015234 104053                      ERROR    53              ;UNUSED BITS OF RMOF NOT ZERO
6544 015236 052703 000001      BIS      #BIT0,R3
6545 015242
6546 30$:
6547 ; CHECK USED BITS OF RMOF FOR ONE
6548 015242 013737 001360 001142      MOV      RMOFI,$BDDAT     ;GET USED BITS
6549 015250 042737 161577 001142      BIC      #XNUOF,$BDDAT
6550 015256 012737 016200 001140      MOV      #FMT16!ECI!HCI!OFD,$GDDAT
6551 015264 023737 001140 001142      CMP      $GDDAT,$BDDAT
6552 015272 001403                      BEQ      40$              ;BRANCH IF NO ERROR
6553 015274 104055                      ERROR    55              ;USED BITS RMOF S-A-0
6554 015276 052703 000002      BIS      #BIT1,R3
6555 015302
6556 ;*****
6557 ;IF NO PREVIOUS ERRORS, TEST RMOF WITH SHIFTING ONE BIT PATTERN.
6558 015302 005703                      TST      R3              ;ANY ERROR??
6559 015304 001025                      BNE      70$              ;YES!!
6560 015306 012702 000001      MOV      #1,R2            ;STARTING DATA PATTERN
6561 015312
6562 50$:
6563 015312 012760 000000 000032      MOV      #0,RMOF(R0)      ;LOAD RMOF
6564

```

```

6565 015320 010260 000032      MOV      R2,RMOF(R0)      ;LOAD RMOF
6566
6567 015324 016037 000032 001142      MOV      RMOF(R0),%BDDAT ;STORE RMOF AT %BDDAT
6568 015332 010203      MOV      R2,R3           ;SETUP EXPECTED RESULT
6569 015334 042703 161577      BIC      %XNLOF,R3       ;CLEAR UNUSED BITS
6570 015340 020337 001142      CMP      R3,%BDDAT       ;COMPARE EXPECTED & RECEIVED
6571 015344 001403      BEQ      60$             ;BRANCH IF NO ERROR
6572 015346 010337 001140      MOV      R3,%GDDAT       ;LOAD EXPECTED
6573 015352 104056      ERROR    56             ;CANT WRITE SHIFTING 1 BIT
6574 015354 006302 60$:      HSL      R2             ;SHIFT TO NEXT BIT
6575 015356 001355      BNE      50$             ;CONTINUE IF NOT DONE
6576
6577 015360      70$:
6578
6579 ;*****
6580 ;*TEST 17 SERIAL NUMBER TEST
6581 ;*****
6582
6583 015360 000004      *ST17: SCOPE
6584 015362 012737 000017 001226      MOV      #17,%TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
6585
6586 015370 000240      NOP
6587 015372 012737 000024 001120      MOV      #20,%ICNT      ;20 ITERATIONS
6588 015400 112737 000001 001131      MOV      #1,%ERRMAX     ;ONE ERROR ALLOWED
6589 015406 012737 015422 001122      MOV      #T17,%LPADR    ;LOAD LOOP ON TEST ADDRESS
6590 015414 012737 015422 001124      MOV      #T17,%LPERR    ;LOAD LOOP ON ERROR ADDRESS
6591
6592 T17:
6593 015422 012706 001100      MOV      #STACK,SP      ;LOAD THE STACK POINTER
6594 015426 013700 001276      MOV      %BASE,R0       ;R0 = UNIBUS ADDRESS OF UUT
6595 015430 013701 001456      MOV      TSTQUE,R1      ;R1 = POINTER TO DEVICE
6596 015436 012760 000040 000010      MOV      #CLR,RMCS2(R0) ;CLEAR THE MASSBUS
6597 015444 111160 000010      MOV      (R1),RMCS2(R0) ;SELECT UNIT
6598 015450 010037 001136      MOV      R0,%BDAOR      ;SETUP REGISTER ADDRESS FOR TYPEOUT
6599 015454 062737 000030 001136      ADD      #RMSN,%BDAOR
6600 015462 012702 000031      MOV      #25,R2         ;READ RMSN 25 TIMES
6601 ;READ RMSN AND USE THE RESULT AS EXPECTED VALUE
6602 015466 016037 000030 001140      MOV      RMSN(R0),%GDDAT ;STORE RMSN AT %GDDAT
6603 015474
6604 10$:
6605 ;READ RMSN AND COMPARE WITH INITIAL VALUE
6606 015474 016037 000030 001142      MOV      RMSN(R0),%BDDAT ;STORE RMSN AT %BDDAT
6607 015502 023737 001140 001142      CMP      %GDDAT,%BDDAT
6608 015510 001401      BEQ      20$             ;BRANCH IF SERIAL NUMBER CONSISTENT
6609 015512 104136      ERROR    136            ;SERIAL NUMBER INCONSISTENT
6610 015514
6611 20$:
6612 ;DECREMENT COUNT AND CONTINUE IF NOT DONE
6613 015516 100366      DEC      R2
6614 015520      BPL      10$
6615
6616 30$:
6617 ;END OF TEST
6618 ;*****
6619 ;*TEST 20 CONTROL BUS PARITY DETECTION TEST
6620 ;*****
6620 015520 000004      *ST20: SCOPE

```

```

6621 015522 012737 000020 001226      MOV      #20,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6622
6623 015530 000240                NOP
6624 015532 012737 000024 001120      MOV      #20,$ICNT      ;20 ITERATIONS
6625 015540 112737 000001 001131      MOV      #1,$ERMAX      ;ONE ERROR ALLOWED
6626 015546 012737 015562 001122      MOV      #T20,$LPAOR    ;LOAD LOOP ON TEST ADDRESS
6627 015554 012737 015562 001124      MOV      #T20,$LPERR    ;LOAD LOOP ON ERROR ADDRESS
6628 015562
6629 015562 012706 001100      T20:     MOV      #STACK,SP      ;LOAD THE STACK POINTER
6630 015566 013700 001276      MOV      $BASE,R0      ;R0 = UNIBUS ADDRESS OF UUT
6631 015572 013701 001456      MOV      TSTQUE,R1     ;R1 = POINTER TO DEVICE
6632
6633 015576 005037 001140      ;SETUP FOR FIRST LOOP (NO ERROR)
6634 015602 111137 001412      CLR      $GDDAT        ;"PAR" SHOULD BE ZERO
6635 015606 042737 177770 001412      MOV      (R1),RMCS20   ;SETUP RMCS2 VALUE
6636 015614 012737 000001 001440      BIC      #1,CUNTMASK,RMCS20
6637 015622 000402                MOV      #1,RMHRO      ;INITIALIZE DATA PATTERN
6638 015624 006337 001440      BR       65            ;SKIP INCREMENT FIRST TIME
6639 015630
6640
6641 015630 012760 000040 000010      55:     ASL      RMHRO      ;SHIFT TO NEXT PATTERN
6642 015636 111160 000010      65:     ;CLEAR AND VERIFY THAT "PAR" IS RESET
6643
6644 015642 01603. 000014 001142      MOV      RMER1(R0),$BDDAT ;STORE RMER1 AT $BDDAT
6645
6646 015650 016037 000042 001370      MOV      RMER2(R0),RMER2I ;STORE RMER2 IN INPUT BUFFER
6647 015656 042737 177767 001142      BIC      #1,CPAR,$BDDAT ;DID "PAR" RESET?
6648 015664 001415                BEQ      20$           ;YES!!
6649 015666 010037 001136                MOV      R0,$BDDADR    ;SETUP REGISTER ADDRESS
6650 015672 062737 000014 001136      ADD      #RMER1,$BDDADR
6651 015700 032737 000010 001370      BIT      #DPE,RMER2I   ;IS "DPE" SET??
6652 015706 001002                BNE      10$          ;YES!!
6653 015710 104067                ERROR    67           ;CANT CLEAR "PAR"
6654 015712 000453                BR       50$          ;CANT CLEAR "PAR", "DPE"
6655 015714 104070                10$:    ERROR    70
6656 015716 000451                BR       50$
6657 015720                20$:
6658
6659
6660
6661 015720 013760 001412 000010      ;WRITE TEST PATTERN AND VERIFY "PAR" STATUS
6662
6663 015726 013760 001440 000036      MOV      RMCS20,RMCS2(R0) ;LOAD RMCS2
6664
6665 015734 016037 000014 001142      MOV      RMHRO,RMHR(R0) ;LOAD RMHR
6666 015742 042737 177767 001142      MOV      RMER1(R0),$BDDAT ;STORE RMER1 AT $BDDAT
6667 015750 023737 001140 001142      BIC      #1,CPAR,$BDDAT
6668 015756 001410                CMP      $GDDAT,$BDDAT ;IS "PAR" CORRECT??
6669 015760 032737 000020 001412      BEQ      40$           ;YES!!
6670 015766 001002                BIT      #PAT,RMCS20   ;SHOULD "PAR" BE SET?
6671 015770 104071                BNE      30$          ;YES!!
6672 015772 000423                ERROR    71           ;"PAR" SHOULD NOT BE SET
6673 015774 104072                BR       50$
6674 015776 000421                30$:    ERROR    72           ;"PAR" SHOULD BE SET
6675
6676 016000 005737 001440      ;GO TO NEXT PATTERN
40$:    TST      RMHRO      ;IS DATA PATTERN COMPLETE??

```

```

6677 016004 001307          BNE      5$          ;NO!!
6678 016006 032737 000020 001412 BIT      #PAT, RMCS20 ;IS TEST COMPLETE??
6679 016014 001012          BNE      50$         ;YES!!
6680          ;SETUP FOR SECOND TEST LOOP (ERROR)
6681 016016 052737 000020 001412 BIS      #PAT, RMCS20 ;TURN ON BAD PARITY
6682 016024 012737 000010 001140 MOV      #PAR, $GDDAT ;EXPECT ERROR
6683 016032 012737 000001 001440 MOV      #1, RMHRO    ;INITIALIZE DATA PATTERN
6684 016040 000673          BR       6$          ;START LOOP
6685
6686 016042          50$:          ;END OF TEST
6687
6688          ;*****
6689          ;*TEST 21 CONTROL BUS PARITY GENERATION TEST
6690          ;*****
6691          ;*****
6692 016042 000004          †ST21: SCOPE
6693 016044 012737 000021 001226 MOV      #21, $TESTN ;;SET TEST NUMBER IN APT MAIL BOX
6694
6695 016052 000240          NOP
6696 016054 012737 000024 001120 MOV      #20, $ICNT  ;20 ITERATIONS
6697 016062 112737 000001 001131 MOVB    #1, $ERMAX  ;ONE ERROR ALLOWED
6698 016070 012737 016104 001122 MOV      #T21, $LPADR ;LOAD LOOP ON TEST ADDRESS
6699 016076 012737 016104 001124 MOV      #T21, $LPERR ;LOAD LOOP ON ERROR ADDRESS
6700 016104
6701 016104 012706 001100          T21: MOV      #STACK, SP ;LOAD THE STACK POINTER
6702 016110 013700 001276          MOV      $BASE, R0 ;R0 = UNIBUS ADDRESS OF UUT
6703 016114 013701 001456          MOV      †STQUE, R1 ;R1 = POINTER TO DEVICE
6704
6705          ;SETUP FOR TEST (NO ERROR)
6706 016120 012737 000001 001440 MOV      #1, RMHRO  ;INITIALIZE DATA PATTERN
6707 016126 005037 001140          CLR      $GDDAT    ;MCPE SHOULD BE ZERO
6708 016132 000402          BR       20$       ;DONT SHIFT FIRST TIME
6709 016134
6710          10$:
6711 016134 006337 001440          ;SHIFT DATA PATTERN
6712 016140          ASL      RMHRO
6713          20$:
6714 016140 012760 000040 000010 ;CLEAR, THEN TRANSFER DATA TO RMO3
6715 016146 111160 000010          MOV      #CLR, RMCS2(R0) ;CLEAR THE MASSBUS
6716          MOVB    (R1), RMCS2(R0) ;SELECT UNIT
6717 016152 013760 001440 000036          MOV      RMHRO, RMHR(R0) ;LOAD RMHR
6718          ;TRANSFER DATA TO RH, VERIFY NO "MCPE" ERROR
6719          MOV      RMHR(R0), RMHRI ;STORE RMHR IN INPUT BUFFER
6720 016160 016037 000036 001364
6721          MOV      RMCS1(R0), $BDDAT ;STORE RMCS1 AT $BDDAT
6722 016166 016037 000000 001142          BIC      #†CMCPE, $BDDAT ;WAS BAD PARITY DETECTED??
6723 016174 042737 157777 001142          BEQ     30$         ;NO!!
6724 016202 001402          ERROR  73         ;BAD PARITY DETECTED BY RH
6725 016204 104073          BR       40$
6726 016206 000403
6727
6728 016210          30$:
6729          ;GO TO NEXT PATTERN
6730 016210 005737 001440          TST     RMHRO
6731 016214 001347          BNE     10$         ;DONE ALL PATTERNS??
6732          ;NO!!

```

```

6733 016216
6734
6735
6736
6737
6738
6739 016216 000004
6740 016220 012737 000022 001226
6741
6742 016226 000240
6743 016230 012737 000024 001120
6744 016236 112737 000001 001131
6745 016244 012737 016260 001122
6746 016252 012737 016260 001124
6747 016260
6748 016260 012706 001100
6749 016264 013700 001276
6750 016270 013701 001456
6751 016274 012760 000040 000010
6752 016302 111160 000010
6753
6754
6755
6756 016306 012760 000000 000006
6757
6758 016314 012760 000000 000034
6759
6760 016322 012760 177777 000006
6761
6762 016330 012760 177777 000034
6763
6764 016336 016037 000006 001334
6765
6766 016344 016037 000034 001362
6767 016352 022737 177777 001334
6768 016360 001410
6769 016362 052737 176000 001362
6770 016370 022737 177777 001362
6771 016376 001401
6772 016400 104042
6773
6774 016402
6775
6776
6777
6778
6779
6780
6781
6782 016402 000004
6783 016404 012737 000023 001226
6784 016412 000240
6785 016414 012737 000024 001120
6786 016422 112737 000001 001131
6787 016430 012737 016444 001122
6788 016436 012737 016444 001124

```

```

405: ;END OF TEST
:*****
;*TEST 22 RMDA,RMDC FAULT TEST
:*****
†ST22: SCOPE
MOV #22,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
NOP
MOV #20,$ICNT ;20 ITERATIONS
MOVB #1,$ERMAX ;ONE ERROR ALLOWED
MOV #T22,$LPADR ;LOAD LOOP ON TEST ADDRESS
MOV #T22,$LPERR ;LOAD LOOP ON ERROR ADDRESS
T22:
MOV #STACK,SP ;LOAD THE STACK POINTER
MOV $BASE,R0 ;R0 = UNIBUS ADDRESS OF UUT
MOV TSTQUE,R1 ;R1 = POINTER TO DEVICE
MOV #CLR,RMCS2(R0) ;CLEAR THE MASSBUS
MOVB (R1),RMCS2(R0) ;SELECT UNIT
;WRITE ZEROS, THEN ONES IN RMDA,RMDC-READ AND TEST FOR S-A-0
MOV #0,RMDA(R0) ;LOAD RMDA
MOV #0,RMDC(R0) ;LOAD RMDC
MOV #-1,RMDA(R0) ;LOAD RMDA
MOV #-1,RMDC(R0) ;LOAD RMDC
MOV RMDA(R0),RMDAI ;STORE RMDA IN INPUT BUFFER
MOV RMDC(R0),RMDCI ;STORE RMDC IN INPUT BUFFER
CMP #-1,RMDAI ;IS ANY REGISTER ALL ONES??
BEQ 105 ;YES!!
BIS #XNUDC,RMDCI ;SET UNUSED BITS
CMP #-1,RMDCI
BEQ 105 ;YES!!
ERROR 42 ;RMDC AND RMDA S-A-0
105:
:*****
;*TEST 23 DISK ADDRESS TRANSFER TEST
:*****
†ST23: SCOPE
MOV #23,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
NOP
MOV #20,$ICNT ;20 ITERATIONS
MOVB #1,$ERMAX ;ONE ERROR ALLOWED
MOV #T23,$LPADR ;LOAD LOOP ON TEST ADDRESS
MOV #T23,$LPERR ;LOAD LOOP ON ERROR ADDRESS

```

6789	016444				T23:			
6790	016444	012706	001100		MOV	#STACK, SP		;LOAD THE STACK POINTER
6791	016450	013700	001276		MOV	\$BASE, R0		;R0 = UNIBUS ADDRESS OF UUT
6792	016454	013701	001456		MOV	TSTQUE, R1		;R1 = POINTER TO DEVICE
6793	016460	005003			CLR	R3		;R3 = ERROR INDICATOR
6794	016462	012760	000040	000010	MOV	#CLR, RMCS2(R0)		;CLEAR THE MASSBUS
6795	016470	111160	000010		MOV	(R1), RMCS2(R0)		;SELECT UNIT
6796								
6797								
6798	016474	012760	177777	000006				
6799					MOV	#-1, RMDA(R0)		;LOAD RMDA
6800	016502	012760	000000	000006	MOV	#0, RMDA(R0)		;LOAD RMDA
6801								
6802	C16510	016037	000006	001142	MOV	RMDA(R0), \$BDDAT		;STORE RMDA AT \$BDDAT
6803	016516	005737	001142		TST	\$BDDAT		
6804	016522	001412			BEQ	10\$		
6805	016524	005037	001140		CLR	\$GDDAT		
6806	016530	010037	001136		MOV	R0, \$BDADR		
6807	016534	062737	000006	001136	ADD	#RMDA, \$BDADR		
6808	016542	104020			ERROR	20		
6809	016544	052703	000001		BIS	#BIT0, R3		;SET ERROR FLAG
6810								
6811	016550							
6812								
6813	016550	012760	000000	000006				
6814					MOV	#0, RMDA(R0)		;LOAD RMDA
6815	016556	012760	177777	000006	MOV	#-1, RMDA(R0)		;LOAD RMDA
6816								
6817	016564	016037	000006	001142	MOV	RMDA(R0), \$BDDAT		;STORE RMDA AT \$BDDAT
6818	016572	023727	001142	177777	CMP	\$BDDAT, #-1		
6819	016600	001413			BEQ	20\$		
6820	016602	012737	177777	001140	MOV	#-1, \$GDDAT		
6821	016610	010037	001136		MOV	R0, \$BDADR		
6822	016614	062737	000006	001136	ADD	#RMDA, \$BDADR		
6823	016622	104021			ERROR	21		
6824	016624	052703	000002		BIS	#BIT1, R3		;SET ERROR FLAG
6825								
6826	016630	005703						
6827	016632	001027						
6828	016634	012702	000001					
6829	016640							
6830								
6831	016640	012760	000000	000006				
6832					MOV	#0, RMDA(R0)		;LOAD RMDA
6833	016646	010260	000006		MOV	R2, RMDA(R0)		;LOAD RMDA
6834								
6835	016652	016037	000006	001142	MOV	RMDA(R0), \$BDDAT		;STORE RMDA AT \$BDDAT
6836	016660	023702	001142		CMP	\$BDDAT, R2		
6837	016664	001410			BEQ	40\$		
6838	016666	010237	001140		MOV	R2, \$GDDAT		
6839	016672	010037	001136		MOV	R0, \$BDADR		
6840	016676	062737	000006	001136	ADD	#RMDA, \$BDADR		
6841	016704	104022			ERROR	22		
6842	016706	006302			40\$:	ASL	R2	;SHIFT TO NEXT BIT
6843	016710	001353			BNE	30\$;CONTINUE IF R2 NOT ZERO
6844	016712				50\$:			

```

6845
6846
6847
6848
6849
6850 016712 000004
6851 016714 012737 000024 001226
6852
6853 016722 000240
6854 016724 012737 000024 001120
6855 016732 112737 000001 001131
6856 016740 012737 016754 001122
6857 016746 012737 016754 001124
6858 016754
6859 016754 012706 001100
6860 016760 013700 001276
6861 016764 013701 001456
6862 016770 005003
6863 016772 012760 000040 000010
6864 017000 111160 000010
6865 017004 005037 001140
6866 017010 010037 001136
6867 017014 062737 000034 001136
6868
6869
6870
6871 017022 012760 177777 000034
6872
6873 017030 016037 000034 001142
6874 017036 042737 001777 001142
6875 017044 001403
6876 017046 104134
6877 017050 052703 000001
6878 017054
6879
6880
6881 017054 012760 177777 000034
6882
6883 017062 012760 000000 000034
6884
6885 017070 016037 000034 001142
6886 017076 042737 176000 001142
6887 017104 001403
6888 017106 104037
6889 017110 052703 000001
6890 017114
6891
6892
6893 017114 012760 000000 000034
6894
6895 017122 012760 177777 000034
6896
6897 017130 016037 000034 001142
6898 017136 052737 176000 001142
6899 017144 012737 177777 001140
6900 017152 023737 001140 001142

```

```

;*****
;TEST 24 DESIRED CYLINDER TRANSFER TEST
;*****
T24: SCOPE
MOV #24,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
NOP
MOV #20,$ICNT ;20 ITERATIONS
MOVB #1,$ERMAX ;ONE ERROR ALLOWED
MOV #T24,$LPADR ;LOAD LOOP ON TEST ADDRESS
MOV #T24,$LPERR ;LOAD LOOP ON ERROR ADDRESS
T24:
MOV #STACK,SP ;LOAD THE STACK POINTER
MOV $BASE,R0 ;R0 = UNIBUS ADDRESS OF UUT
MOV TSTQUE,R1 ;R1 = POINTER TO DEVICE
CLR R3 ;RESET ERROR FLAGS
MOV #CLR,RMCS2(R0) ;CLEAR THE MASSBUS
MOVB (R1),RMCS2(R0) ;SELECT UNIT
CLR $GDDAT ;LOAD EXPECTED
MOV R0,$BDADR ;LOAD REG ADDRESS
ADD #RMDC,$BDADR
;WRITE ONES IN RMDC AND VERIFY THAT UNUSED BITS ARE ZERO
MOV #-1,RMDC(R0) ;LOAD RMDC
MOV RMDC(R0),$BDDAT ;STORE RMDC AT $BDDAT
BIC #XNUDC,$BDDAT ;CLEAR ALL USED BITS
BEQ 55 ;BRANCH IF NO ERROR
ERROR 134 ;UNUSED BITS OF RMDC NOT ZERO
BIS #BIT0,R3 ;SET ERROR FLAG
55:
;WRITE ONES IN RMDC, THEN WRITE ZEROS, READ AND CHECK FOR S-A-1 BITS
MOV #-1,RMDC(R0) ;LOAD RMDC
MOV #0,RMDC(R0) ;LOAD RMDC
MOV RMDC(R0),$BDDAT ;STORE RMDC AT $BDDAT
BIC #XNUDC,$BDDAT ;CLEAR UNUSED BITS
BEQ 105 ;BRANCH IF NO ERROR
ERROR 37 ;CANT WRITE ONES RMDC
BIS #BIT0,R3 ;SET ERROR FLAG
105:
;WRITE ZEROS, THEN ONES IN RMDC, READ AND CHECK FOR S-A-0 BITS
MOV #0,RMDC(R0) ;LOAD RMDC
MOV #-1,RMDC(R0) ;LOAD RMDC
MOV RMDC(R0),$BDDAT ;STORE RMDC AT $BDDAT
BIS #XNUDC,$BDDAT ;SET UNUSED BITS
MOV #-1,$GDDAT ;LOAD EXPECTED RESULT
CMP $GDDAT,$BDDAT ;IS RMDC ALL ONES ??

```

```

6901 017160 001403          BEQ      20$          ;YES !!
6902 017162 104040          ERROR   40
6903 017164 052703 000002  BIS      #BIT1,R3      ;SET ERROR FLAG
6904 017170
6905
6906 017170 005703          20$:
6907 017172 001026          ;OMIT BIT TEST IF ANY ERRORS
6908 017174 012702 000001  TST      R3
6909 017200          BNE     60$
6910
6911
6912 017200 012760 000000 000034  MOV      #0,RMDC(RO)    ;LOAD RMDC
6913
6914 017206 010260 000034  MOV      R2,RMDC(RO)    ;LOAD RMDC
6915 017212 010203          MOV      R2,R3          ;R3 = EXPECTED RESULT
6916 017214 042703 176000  BIC     #XNUDC,R3      ;CLEAR ANY UNUSED BITS
6917
6918 017220 016037 000034 001142  MOV      RMDC(RO), $BDDAT ;STORE RMDC AT $BDDAT
6919 017226 023703 001142  CMP     $BDDAT,R3
6920 017232 001404          BEQ     50$
6921 017234 010260 001140  MOV      R3,$GDDAT
6922 017240 104041          ERROR   41
6923 017242 000402          BR      60$          ;SKIP TO NEXT
6924
6925 017244 006302          50$:
6926 017246 001354          ASL     R2          ;SHIFY TO NEXT BIT
6927
6928 017250          BNE     30$          ;CONTINUE IF R2 NOT ZERO
6929
6930
6931
6932
6933
6934 017250 000004          60$:
6935 017252 012737 000025 001226  ;*****
6936 017260 000240          ;TEST 25 ILLEGAL REGISTER TEST
6937 017262 012737 000024 001120  ;*****
6938 017270 112737 000001 001131  ;ST25: SCOPE
6939 017276 012737 017312 001122  MOV      #25,$TESTN    ;;SET TEST NUMBER IN APT MAIL BCX
6940 017304 012737 017312 001124  NOP
6941 017312
6942 017312 012706 001100  MOV      #20,$ICNT     ;20 ITERATIONS
6943 017316 013700 001276  MOV      #1,$ERMAX     ;ONE ERROR ALLOWED
6944 017322 013701 001456  MOV      #T25,$LPADR   ;LOAD LOOP ON TEST ADDRESS
6945 017326 005037 001140  MOV      #T25,$LPERR   ;LOAD LOOP ON ERROR ADDRESS
6946 017332 012702 000000  T25:
6947 017336          MOV      #STACK,SP     ;LOAD THE STACK POINTER
6948
6949
6950 017336 012760 000040 000010  MOV      $BASE,R0      ;R0 = UNIBUS ADDRESS OF UUT
6951 017344 111160 000010          MOV      TSTQUF,R1     ;R1 = POINTER TO DEVICE
6952
6953 017350 016037 000014 001142  CLR     $GDDAT        ;"ILR" SHOULD BE ZERO
6954 017356 042737 177775 001142  MOV      #0,R2        ;R2=REGISTER INDEX
6955 017364 001411
6956 017366 005037 001140          10$:
6957
6958
6959
6960
6961
6962
6963
6964
6965
6966
6967
6968
6969
6970
6971
6972
6973
6974
6975
6976
6977
6978
6979
6980
6981
6982
6983
6984
6985
6986
6987
6988
6989
6990
6991
6992
6993
6994
6995
6996
6997
6998
6999

```



```

6957 017372 010037 001136      MOV      R0,$B0ADR
6958 017376 062737 000014 001136  ADD      #RMER1,$B0ADR
6959 017404 104064      ERROR    64      ;CANT CLEAR ILR
6960 017406 000550      BR       140$
6961 017410
6962
6963
6964 017410 010003      20$:
6965 017412 060203      ;*****
6966 017414 013746 000004      ;WRITE THE REGISTER (INDEX=R2) AND TEST ILR STATUS
6967 017420 013746 000006      MOV      R0,R3      ;R3=REG ADDRESS
6968 017424 012737 017466 000004      ADD      R2,R3
6969 017432 012737 000300 000006      MOV      ERRVEC, -(SP) ;: PUSH ERRVEC ON STACK
6970 017440 005004      MOV      ERRVEC+2, -(SP) ;: PUSH ERRVEC+2 ON STACK
6971 017442 022702 000010      MOV      #40$,ERRVEC ;: SETUP FOR BUS TIMEOUT
6972 017446 001001      MOV      #R6,ERRVEC+2
6973 017450 111104      CLR      R4      ;R4=REGISTER VALUE
6974 017452 010413      CMP      #RMCS2,R2
6975 017454 012637 000006      BNE     30$
6976 017460 012637 000004      MOV     B (R1),R4 ;: SELECT DRIVE IF RMCS2
6977 017464 000416      MOV     R4,(R3) ;: WRITE TEST REGISTER
6978 017466 012716 017474      MOV     (SP)+,ERRVEC+2 ;: POP STACK INTO ERRVEC+2
6979 017472 000002      MOV     (SP)+,ERRVEC ;: POP STACK INTO ERRVEC
6980 017474      BR     60$ ;: POP STACK INTO ERRVEC+2
6981 017474 012637 000006      MOV     R2,#RMEC2 ;: WERE ALL RM03 REGISTERS READ??
6982 017500 012637 000004      CMP     R2,#RMEC2 ;: YES!!
6983 017504 020227 000046      BHI     50$
6984 017510 101003      MOV     R3,$B0ADR
6985 017512 010337 001136      MOV     R3,$B0ADR
6986 017516 104074      ERROR   74      ;UNEXPECTED BUS TIMEOUT
6987 017520 000503      BR     140$
6988
6989 017522      50$:
6990
6991 017522 016037 000014 001142      MOV     RMER1(R0),$B0DAT ;: STORE RMER1 AT $B0DAT
6992 017530 042737 177775 001142      BIC     #1CILR,$B0DAT
6993 017536 023737 001140 001142      CMP     $GDDAT,$B0DAT ;: IS "ILR" OK??
6994 017544 001411      BEQ     80$ ;: YES!!
6995 017546 010337 001174      MOV     R3,$TMPD ;: SAVE ADDRESS
6996 017552 032737 000002 001140      BIT     #ILR,$GDDAT ;: SHOULD "ILR" BE SET??
6997 017560 001002      BNE     70$ ;: YES!!
6998 017562 104065      ERROR   65      ;: "ILR" SHOULD NOT BE SET
6999 017564 000401      BR     80$
7000 017566 104066      70$:
7001      ERROR   66      ;: "ILR" SHOULD BE SET
7002 017570
7003      80$:
7004 017570 062702 000002      ;ADVANCE TO THE NEXT REGISTER ADDRESS
7005 017574 022702 000050      ADD     #2,R2 ;: INCREMENT INDEX
7006 017600 101256      CMP     #50,R2 ;: TIME TO TRY RM70 ?
7007 017602 103437      BHI     10$ ;: BRANCH IF NOT
7008      BLO     110$ ;: BRANCH IF ALREADY CHECKED
7009      CMP     #RMBAE,R2 ;: IS THIS RMBAE??
7010      BNE     100$ ;: NO!!
7011 017604 013746 000004      MOV     ERRVEC, -(SP) ;: PUSH ERRVEC ON STACK
7012 017610 013746 000006      MOV     ERRVEC+2, -(SP) ;: PUSH ERRVEC+2 ON STACK
7013 017614 012737 017712 000004      MOV     #130$,ERRVEC ;: SETUP FOR TIMEOUT

```

```

7013 017622 012737 000300 000006      MOV      #PR6,ERRVEC+2
7014                                     PUTCS1  #A17!A16
7015                                     GETBAE
7016                                     BIC      #1<BIT1!BIT0>,RMBAEI
7017                                     BNE     90$      ;BRANCH IF RH70 CONTROLLER
7018                                     BIS     #ILR,$GDDAT ;"ILR" SHOULD BE SET
7019                                     POP     ERRVEC+2
7020                                     POP     ERRVEC
7021                                     100$:  CMP     #RMCS3+2,R2      ;DONE ALL LEGAL REGISTERS
7022                                     BNE     110$
7023                                     BIS     #ILR,$GDDAT ;"ILR" SHOULD BE SET
7024 017630 052737 000002 001140      BIS     #ILR,$GDDAT ;SET ILR
7025 017636 012702 000054              MOV     #54,R2      ;START AT INDEX 54 IF RH70/22REG
7026 017642 012760 001400 00000C      MOV     #A17!A16,RMCS1(R0) ;SET EXTEND BITS
7027 017650 016037 000050 001376      MOV     50(R0),RMBAEI ;READ THE EXTENDED BITS
7028 017656 042737 177774 001376      BIC     #177774,RMBAEI ;CHOP OFF
7029 017664 001002              BNE     90$      ;BRANCH IF RH70/22-REG
7030 017666 012702 000050              MOV     #50,R2      ;OTHERWISE NOT A RH70 OR RH70 WITH32REG
7031 017672 012637 000006      90$:  MOV     (SP)+,ERRVEC+2
7032 017676 012637 000004              MOV     (SP)+,ERRVEC
7033 017702 022702 000074      110$:  CMP     #74,R2
7034 017706 101410              BLOS   140$
7035 017710 000612      120$:  BR     10$
7036
7037 017712 012716 017720      130$:  MOV     #135$,(SP) ;DUMMY RTI ADDRESS
7038 017716 000002              RTI
7039 017720      135$:
7040 017720 012637 000004              MOV     (SP)+,ERRVEC ;;POP STACK INTO ERRVEC
7041 017724 012637 000006              MOV     (SP)+,ERRVEC+2 ;;POP STACK INTO ERRVEC+2
7042 017730      140$:
7043
7044 ;*****
7045 ;*TEST 26 RESET GO BY INIT TEST
7046 ;*****
7047
7048 017730 000004      †ST26: SCOPE
7049 017732 012737 000026 001226      MOV     #26,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
7050
7051 017740 000240              NOP
7052 017742 012737 000024 001120      MOV     #20,$ICNT ;20 ITERATIONS
7053 017750 112737 000001 001131      MOV     #1,$ERMAX ;ONE ERROR ALLOWED
7054 017756 012737 017772 001122      MOV     #T26,$LPADR ;LOAD LOOP ON TEST ADDRESS
7055 017764 012737 017772 001124      MOV     #T26,$LPERR ;LOAD LOOP ON ERROR ADDRESS
7056 017772      T26:
7057 017772 012706 001100              MOV     #STACK,SP ;LOAD THE STACK POINTER
7058 017776 013700 001276              MOV     $BASE,R0 ;R0 = UNIBUS ADDRESS OF UUT
7059 020002 013701 001456              MOV     TSTQUE,R1 ;R1 = POINTER TO DEVICE
7060 020006 012760 000040 000010      MOV     #CLR,RMCS2(R0) ;CLEAR THE MASSBUS
7061 020014 111160 000010              MOV     (R1),RMCS2(R0) ;SELECT UNIT
7062 020020 010037 001136              MOV     R0,$BOADR ;SETUP REGISTER ADDRESS FOR MSG
7063 ;SET GO, INITIALIZE AND VERIFY THAT GO IS RESET
7064
7065 020024 012760 000001 000000      MOV     #GO,RMCS1(R0) ;LOAD RMCS1
7066 020032 012760 000040 000010      MOV     #CLR,RMCS2(R0) ;CLEAR THE MASSBUS
7067 020040 111160 000010              MOV     (R1),RMCS2(R0) ;SELECT UNIT
7068

```

```

7069 020044 016037 000000 001142      MOV      RMCS1(RO), $BDDAT      ;STORE RMCS1 AT $BDDAT
7070 020052 042737 177776 001142      BIC      #↑CGO, $BDDAT
7071 020060 001403                BEQ      10$                   ;BRANCH IF GO IS RESET
7072 020062 005037 001140      CLR      $GDDAT
7073 020066 104137                ERROR   137                   ;GO NOT CLEARED BY INIT
7074
7075 020070                10$:                           ;END OF TEST
7076
7077
7078 ;:*****
; *TEST 27      DIAGNOSTIC MODE TEST
7079
7080 ;:*****
7081 020070 000004      †ST27: SCOPE
7082 020072 012737 000027 001226      MOV      #27, $TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
7083
7084 020100 000240                NOP
7085 020102 012737 000024 001120      MOV      #20, $ICNT           ;20 ITERATIONS
7086 020110 112737 000001 001131      MOVB    #1, $ERMAX           ;ONE ERROR ALLOWED
7087 020116 012737 020132 001122      MOV      #T27, $LPADR        ;LOAD LOOP ON TEST ADDRESS
7088 020124 012737 020132 001124      MOV      #T27, $LPERR        ;LOAD LOOP ON ERROR ADDRESS
7089 020132
7090 020132 012706 001100      MOV      #STACK, SP          ;LOAD THE STACK POINTER
7091 020136 013700 001276      MOV      $BASE, RO           ;RO = UNIBUS ADDRESS OF UUT
7092 020142 013701 001456      MOV      TSTQUE, R1          ;R1 = POINTER TO DEVICE
7093 020146 010037 001136      MOV      RO, $BDDADR         ;SETUP REGISTER ADDRESS
7094 020152 062737 000024 001136      ADD     #RMMR1, $BDDADR
7095 020160 005003                CLR     R3                    ;INITIALIZE ERROR FLAGS
7096 ; INITIALIZE AND VERIFY THAT "DMD" IS RESET
7097 020162 012760 000040 000010      MOV      #CLR, RMCS2(RO)     ;CLEAR THE MASSBUS
7098 020170 111160 000010      MOVB    (R1), RMCS2(RO)     ;SELECT UNIT
7099
7100 020174 016037 000024 001142      MOV      RMMR1(RO), $BDDAT    ;STORE RMMR1 AT $BDDAT
7101 020202 042737 177776 001142      BIC      #↑CMD, $BDDAT
7102 020210 001403                BEQ      10$                   ;BRANCH IF "DMD" IS ZERO
7103 020212 005037 001140      CLR      $GDDAT
7104 020216 104075                ERROR   75                   ;CANT CLEAR "DMD"
7105 020220
7106 ;SET AND RESET "DMD" USING REGISTER TRANSFER-VERIFY "DMD" NOT S-A-1
7107
7108 020220 012760 000001 000024      MOV      #DMD, RMMR1(RO)     ;LOAD RMMR1
7109
7110 020226 012760 000000 000024      MOV      #0, RMMR1(RO)       ;LOAD RMMR1
7111
7112 020234 016037 000024 001142      MOV      RMMR1(RO), $BDDAT    ;STORE RMMR1 AT $BDDAT
7113 020242 042737 177776 001142      BIC      #↑CMD, $BDDAT
7114 020250 001403                BEQ      20$                   ;BRANCH IF DMD NOT S-A-1
7115 020252 005037 001140      CLR      $GDDAT
7116 020256 104076                ERROR   76                   ;CANT WRITE DMD=0
7117 020260 052703 000001      BIS     #BIT0, R3            ;SET ERROR FLAG
7118 020264
7119 ;RESET AND SET "DMD" USING REGISTER TRANSFER-VERIFY "DMD" NOT S-A-0
7120
7121 020264 012760 000000 000024      MOV      #0, RMMR1(RO)       ;LOAD RMMR1
7122
7123 020272 012760 000001 000024      MOV      #DMD, RMMR1(RO)     ;LOAD RMMR1
7124

```

```

7125 020300 016037 000024 001142      MOV    RMMR1(RO), $BDDAT      ;STORE RMMR1 AT $BDDAT
7126 020306 042737 177776 001142      BIC    #1CDMD, $BDDAT
7127 020314 001006             BNE    30$                   ;BRANCH IF DMD NOT S-A-O
7128 020316 012737 000001 001140      MOV    #DMD, $GDDAT
7129 020324 104077             ERROR  77                     ;CAN'T WRITE DMD=1
7130 020326 052703 000002             BIS    #BIT1, R3              ;SET ERROR FLAG
7131 020332
7132      30$:
; IF NO PREVIOUS ERROR, TEST FOR BIT INTERFERENCE WITH SHIFTING
; ONE BIT PATTERN
7133      TST    R3                ;ANY ERRORS DETECTED??
7134 020332 005703             BNE    60$                   ;YES!!
7135 020334 001027             MOV    #BIT1, R2              ;INITIALIZE DATA PATTERN
7136 020336 012702 000002             40$:
7137 020342
7138
7139 020342 012760 000000 000024      MOV    #0, RMMR1(RO)         ;LOAD RMMR1
7140
7141 020350 010260 000024             MOV    R2, RMMR1(RO)        ;LOAD RMMR1
7142
7143 020354 016037 000024 001142      MOV    RMMR1(RO), $BDDAT     ;STORE RMMR1 AT $BDDAT
7144 020362 042737 177776 001142      BIC    #1CDMD, $BDDAT
7145 020370 001407             BEQ    50$                   ;BRANCH IF DMD NOT SET
7146 020372 010237 001174             MOV    R2, $TMPD             ;SAVE DATA
7147 020376 010237 001174             MOV    R2, $TMPD             ;SAVE DATA
7148 020402 005037 001140             CLR    $GDDAT                ;DMD SHOULD BE ZERO
7149 020406 104100             ERROR  100                   ;DMD SET BY WRONG BIT
7150 020410
7151      50$:
; SHIFT TO NEXT DATA BIT AND CONTINUE TEST IF NOT DONE
7152 020410 006302             ASL    R2
7153 020412 001353             BNE    40$
7154 020414
7155      60$:
; *****
; *TEST 30      MOL TEST
7156
7157
7158      ; *****
7159 020414 000004             †ST30: SCOPE
7160 020416 012737 000030 001226      MOV    #30, $TESTN           ;;SET TEST NUMBER IN APT MAIL BOX
7161
7162 020424 000240             NOP
7163 020426 012737 000024 001120      MOV    #20, $ICNT           ;20 ITERATIONS
7164 020434 112737 000001 001131      MOV    #1, $ERMAX           ;ONE ERROR ALLOWED
7165 020442 012737 020456 001122      MOV    #T30, $LPADR         ;LOAD LOOP ON TEST ADDRESS
7166 020450 012737 020456 001124      MOV    #T30, $LPERR         ;LOAD LOOP ON ERROR ADDRESS
7167 020456
7168 020456 012706 001100             T30: MOV    #STACK_SP           ;LOAD THE STACK POINTER
7169 020462 013700 001276             MOV    $BASE, R0            ;R0 = UNIBUS ADDRESS OF UUT
7170 020466 013701 001456             MOV    TSTACK, R1          ;R1 = POINTER TO DEVICE
7171 020472 012760 000040 000010      MOV    #CLR, RMCS2(RO)      ;CLEAR THE MASSBUS
7172 020500 111160 000010             MOV    (R1), RMCS2(RO)     ;SELECT UNIT
7173 020504 010037 001136             MOV    R0, $DADR           ;R0=REGISTER ADDRESS
7174 020510 062737 000012 001136      ADD    #RMS, $DADR
7175 020516 005003             CLR    R3                   ;R3=ERROR FLAG
7176      ;SET DIAGNOSTIC MODE AND VERIFY THAT "MOL" IS ZERO
7177
7178 020520 012760 000001 000024      MOV    #DMD, RMMR1(RO)     ;LOAD RMMR1
7179
7180 020526 016037 000012 001142      MOV    RMS(RO), $BDDAT     ;STORE RMS AT $BDDAT

```

7181	020534	042737	167777	001142	BIC	#1CMOL,\$BDDAT	
7182	020542	001405			BEQ	10\$	
7183	020544	005037	001140		CLR	\$GDDAT	
7184	020550	104101			ERROR	101	;"MOL" NOT ZERO
7185	020552	052703	000001		BIS	#BIT0,R3	;"SET ERROR FLAG
7186	020556				10\$:		
7187					;"SET MAINTENANCE UNIT READY AND VERIFY THAT "MOL" IS ONE		
7188							
7189	020556	012760	000001	000024	MOV	#DMD,RMMR1(RO)	;"LOAD RMMR1
7190							
7191	020564	012760	001001	000024	MOV	#DMD!MUR,RMMR1(RO)	;"LOAD RMMR1
7192							
7193	020572	016037	000012	001142	MOV	RMDS(RO), \$BDDAT	;"STORE RMDS AT \$BDDAT
7194	020600	042737	167777	001142	BIC	#1CMOL,\$BDDAT	
7195	020606	001006			BNE	20\$	
7196	020610	012737	010000	001140	MOV	#MOL,\$GDDAT	
7197	020616	104102			ERROR	102	;"MOL" NOT ONE
7198	020620	052703	000002		BIS	#BIT1,R3	
7199	020624				20\$:		
7200					;"IF NO PREVIOUS ERROR, VERIFY VOLUME VALID IS RESET AND		
7201					;"TEST FOR BIT INTERFERENCE		
7202	020624	005703			TST	R3	;"ANY ERROR DETECTED?"
7203	020626	001057			BNE	70\$;"YES!!
7204							
7205	020630	016037	000012	001142	MOV	RMDS(RO), \$BDDAT	;"STORE RMDS AT \$BDDAT
7206	020636	042737	177677	001142	BIC	#1CVV,\$BDDAT	
7207	020644	001403			BEQ	25\$;"BRANCH IF VV RESET
7208	020646	005037	001140		CLR	\$GDDAT	
7209	020652	104135			ERROR	135	;"VV NOT RESET
7210	020654				25\$:		
7211	020654	012702	000001		MOV	#1,R2	;"INITIALIZE DATA PATTERN
7212	020660				30\$:		
7213							
7214	020660	012760	000001	000024	MOV	#DMD,RMMR1(RO)	;"LOAD RMMR1
7215							
7216	020666	010260	000024		MOV	R2,RMMR1(RO)	;"LOAD RMMR1
7217							
7218	020672	016037	000012	001142	MOV	RMDS(RO), \$BDDAT	;"STORE RMDS AT \$BDDAT
7219	020700	042737	167777	001142	BIC	#1CMOL,\$BDDAT	
7220	020706	005003			CLR	R3	;"SETUP EXPECTED "MOL"
7221	020710	032702	001000		BIT	#MUR,R2	
7222	020714	001402			BEQ	40\$	
7223	020716	052703	010000		BIS	#MOL,R3	;"MOL" SHOULD BE ONE
7224	020722	020337	001142		40\$:		;"IS MOL OK?"
7225	020726	001405			CMP	R3,\$BDDAT	;"YES!!
7226	020730	010237	001174		BEQ	50\$;"SAVE TEST PATTERN
7227	020734	010337	001140		MOV	R2,\$TMP0	
7228	020740	104103			MOV	R3,\$GDDAT	
7229	020742				ERROR	103	;"MUR SET BY WRONG BIT
7230					50\$:		
7231	020742	042702	000001		;"SHIFT TO NEXT PATTERN		
7232	020746	001002			BIC	#DMD,R2	;"DONT SHIFT DMD
7233	020750	012702	000001		BNE	60\$	
7234	020754	006302			MOV	#DMD,R2	;"DONT TRUNCATE TEST
7235	020756	001403			60\$:		;"SHIFT TO NEXT BIT
7236	020760	052702	000001		ASL	R2	;"BRANCH IF DONE
					BEQ	70\$;"KEEP DMD ON
					BIS	#DMD,R2	

```

7237 020764 000735          BR      30$          ;CONTINUE
7238
7239 020766          70$:          ;END OF TEST
7240
7241          ;:*****
7242          ;*TEST 31      WRITE LOCK TEST
7243          ;:*****
7244          ;TST31:  SCOPE
7245 020766 000004          MOV      #31,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7246 020770 012737 000031 001226
7247
7248 020776 000240          NOP
7249 021000 012737 000024 001120          MOV      #20,$ICNT      ;20 ITERATIONS
7250 021006 112737 000001 001131          MOVB    #1,$ERMAX      ;ONE ERROR ALLOWED
7251 021014 012737 021030 001122          MOV      #T31,$LPADR   ;LOAD LOOP ON TEST ADDRESS
7252 021022 012737 021030 001124          MOV      #T31,$LPERR   ;LOAD LOOP ON ERROR ADDRESS
7253 021030
7254 021030 012706 001100          T31:    MOV      #STACK,SP      ;LOAD THE STACK POINTER
7255 021034 013700 001276          MOV      $BASE,R0      ;R0 = UNIBUS ADDRESS OF UUT
7256 021040 013701 001456          MOV      TSTQUE,R1     ;R1 = POINTER TO DEVICE
7257 021044 005003          CLR      R3            ;R3=ERROR FLAG
7258 021046 010037 001136          MOV      R0,$BADR      ;SETUP REGISTER ADDRESS
7259 021052 062737 000012 001136          ADD     #RMD5,$BADR
7260 021060 005037 001140          CLR     $GDDAT        ;WRL SHOULD BE ZERO
7261 021064 012760 000040 000010          MOV     #CLR,RMCS2(R0) ;CLEAR THE MASSBUS
7262 021072 111160 000010          MOVB   (R1),RMCS2(R0) ;SELECT UNIT
7263          ;SET DIAGNOSTIC MODE AND VERIFY "WRL" IS ZERO
7264
7265 021076 012760 000001 000024          MOV     #DMD,RMMR1(R0) ;LOAD RMMR1
7266
7267 021104 016037 000012 001142          MOV     RMD5(R0),$BDDAT ;STORE RMD5 AT $BDDAT
7268 021112 042737 173777 001142          BIC    #↑CWRL,$BDDAT
7269 021120 001403          BEQ    10$            ;BRANCH IF WRL IS ZERO
7270 021122 104104          ERROR  104           ;WRL NOT ZERO
7271 021124 052703 000001          BIS    #BIT0,R3      ;SET ERROR FLAG
7272 021130
7273          10$:
7274          ;SET MAINTENANCE WRITE PROTECT AND VERIFY "WRL" IS ONE
7275 021130 012760 000001 000024          MOV     #DMD,RMMR1(R0) ;LOAD RMMR1
7276
7277 021136 012760 000011 000024          MOV     #DMD!MWP,RMMR1(R0) ;LOAD RMMR1
7278
7279 021144 016037 000012 001142          MOV     RMD5(R0),$BDDAT ;STORE RMD5 AT $BDDAT
7280 021152 042737 173777 001142          BIC    #↑CWRL,$BDDAT
7281 021160 001006          BNE    20$            ;BRANCH IF WRL IS NOE
7282 021162 012737 004000 001140          MOV     #WRL,$GDDAT   ;WRL SHOULD BE SET
7283 021170 104105          ERROR  105           ;CANT SET WRL
7284 021172 052703 000002          BIS    #BIT1,R3      ;SET ERROR FLAG
7285 021176
7286          20$:
7287          ;IF NO PREVIOUS ERROR, TEST FOR BIT INTERFERENCE ON "MWP"
7288          TST     R3      ;ANY OTHER ERROR??
7289          BNE    70$      ;YES!!
7290          MOV     #1,R2   ;INITIALIZE DATA PATTERN
7291 021206
7292          30$:
          ; TRANSFER DATA TO RMMR1, READ RMD5 AND VERIFY WRL

```

```

7293
7294 021206 012760 000001 000024      MOV      #DMD,RMMR1(RO) ;LOAD RMMR1
7295
7296 021214 010260 000024              MOV      R2,RMMR1(RO) ;LOAD RMMR1
7297
7298 021220 016037 000012 001142      MOV      RMS(RO), $BDDAT ;STORE RMS AT $BDDAT
7299 021226 042737 173777 001142      BIC      #1CWAL,$BDDAT ;CLEARUP RECEIVED "WRL"
7300 021234 005003              CLR      R3 ;GENERATE EXPECTED "WRL"
7301 021236 032702 000010      BIT      #MWP,R2
7302 021242 001402              BEQ      40$
7303 021244 052703 004000      BIS      #WAL,R3 ;WAL SHOULD BE SET
7304 021250 020337 001142      CMP      R3,$BDDAT ;IS WAL OK??
7305 021254 001405              BEQ      50$ ;YES!!
7306 021256 010337 001140      MOV      R3,$GDDAT ;SAVE EXPECTED
7307 021262 010237 001174      MOV      R2,$TMP0 ;SAVE DATA PATTERN
7308 021266 104106              ERROR   106 ;BIT INTERFERENCE WITH MWP
7309 021270
7310 ;
7311 021270 042702 000001      ; ADVANCE TO NEXT DATA BIT
7312 021274 001002              BIC      #DMD,R2 ;DONT SHIFT DMD
7313 021276 012702 000001      BNE      60$
7314 021302 006302              MOV      #DMD,R2 ;DONT TRUNCATE TEST
7315 021304 001403              ASL      R2 ;SHIFT DATA BIT
7316 021306 052702 000001      BEQ      70$ ;EXIT IF DONE
7317 021312 000735              BIS      #DMD,R2 ;KEEP DIAGNOSTIC MODE ON
7318
7319 021314              BR       30$ ;CONTINUE TEST
7320
7321 ;*****
7322 ;#TEST 32 DRIVE FAULT TEST
7323 ;*****
7324
7325 021314 000004      TST32: SCOPE
7326 021316 012737 000032 001226      MOV      #32,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
7327
7328
7329 021324 000240              NOP
7330 021326 012737 000024 001120      MOV      #20,$ICNT ;20 ITERATIONS
7331 021334 112737 000001 001131      MOV      #1,$ERMAX ;ONE ERROR ALLOWED
7332 021342 012737 021356 001122      MOV      #T32,$LPADR ;LOAD LOOP ON TEST ADDRESS
7333 021350 012737 021356 001124      MOV      #T32,$LPERR ;LOAD LOOP ON ERROR ADDRESS
7334 021356
7335 021356 012706 001100      T32: MOV      #STACK,SP ;LOAD THE STACK POINTER
7336 021362 013700 001276      MOV      $BASE,R0 ;R0 = UNIBUS ADDRESS OF UUT
7337 021366 013701 001456      MOV      TSTQUE,R1 ;R1 = POINTER TO DEVICE
7338 021372 012760 000040 000010      MOV      #CLR,RMCS2(RO) ;CLEAR THE MASSBUS
7339 021400 111160 000010      MOV      (R1),RMCS2(RO) ;SELECT UNIT
7340 021404 005003              CLR      R3 ;INITIALIZE ERROR FLAGS
7341 021406 010037 001136      MOV      R0,$BDAOR ;SETUP REGISTER ADDRESS
7342 021412 062737 000042 001136      ADD      #RMR2,$BDAOR
7343 021420 005037 001140      CLR      $GDDAT ;"DVC" AND "UNS" SHOULD BE ZERO
7344 ;SET AND RESET MAINTENANCE DRIVE FAULT, VERIFY THAT "DVC" IS NOT
7345 ;STUCK-AT-ONE.
7346 021424 012760 000001 000024      MOV      #DMD,RMMR1(RO) ;LOAD RMMR1
7347
7348 021432 012760 000000 000042      MOV      #0,RMR2(RO) ;LOAD RMR2

```

```

7349
7350 021440 012760 000000 000014      MOV      #0,RMER1(RO)      ;LOAD RMER1
7351
7352 021446 016037 000042 001142      MOV      RMER2(RO), $BDDAT ;STORE RMER2 AT $BDDAT
7353 021454 042737 177577 001142      BIC      #↑CDVC, $BDDAT    ;IS "DVC" RESET??
7354 021462 001406          BEQ      10$              ;YES!!
7355 021464 104107          ERROR   107              ;CANT RESET DVC
7356 021466 052703 000001          BIS      #BIT0, R3        ;SET ERROR FLAG
7357 021472 012737 040000 001140      MOV      #UNS, $GDDAT
7358 021500
7359
7360
7361
7362
7363
7364
7365
7366
7367
7368
7369
7370
7371
7372
7373
7374
7375
7376
7377
7378
7379
7380
7381
7382
7383
7384
7385
7386
7387
7388
7389
7390
7391
7392
7393
7394
7395
7396
7397
7398
7399
7400
7401
7402
7403
7404

```

10\$:
;VERIFY THAT "UNS" IS SAME AS "DVC"

```

MOV      RMER1(RO), $BDDAT ;STORE RMER1 AT $BDDAT
BIC      #↑CUNS, $BDDAT
CMP      $GDDAT, $BDDAT    ;IS "UNS" OK??
BEQ      30$              ;YES!!
MOV      RO, $BDAOR        ;SETUP REGISTER ADDRESS
ADD      #RMER1, $BDAOR
BIT      #UNS, $GDDAT      ;SHOULD "UNS" BE ON??
BNE      20$              ;YES!!
ERROR   110              ;UNS IS SET, DVS IS RESET
BR       30$

```

20\$:
ERROR 111 ;UNS IS RESET, DVC IS SET

30\$:
;RESET AND SET 'MDF', VERIFY THAT "DVC" IS NOT S-A-O.
MOV #DVC, \$GDDAT ;DVC SHOULD BE ON

```

MOV      #DMD, RMMR1(RO) ;LOAD RMMR1
MOV      #DMD!MDF, RMMR1(RO) ;LOAD RMMR1
MOV      RMER2(RO), $BDDAT ;STORE RMER2 AT $BDDAT
BIC      #↑CDVC, $BDDAT
BNE      40$              ;BRANCH IF DVC IS SET
MOV      RO, $BDAOR        ;SETUP REGISTER ADDRESS
ADD      #RMER2, $BDAOR
ERROR   112              ;CANT SET DVC
BIS      #BIT1, R3        ;SET ERROR FLAG
CLR      $GDDAT          ;UNS SHOULD BE OFF

```

40\$:
;VERIFY THAT "UNS" IS SAME AS "DVC"

```

TST      $GDDAT ;CHANGE DVC TO UNS
BEQ      50$
MOV      #UNS, $GDDAT

```

50\$:
MOV RMER1(RO), \$BDDAT ;STORE RMER1 AT \$BDDAT
BIC #↑CUNS, \$BDDAT
CMP \$GDDAT, \$BDDAT
BEQ 70\$;BRANCH IF UNS IS OK
MOV RO, \$BDAOR ;SETUP REGISTER ADDRESS
ADD #RMER1, \$BDAOR


```

7405 021712 032737 040000 001140 BIT #UNS,$GDDAT ;SHOULD UNS BE ON??
7406 021720 001002 BNE 60$ ;YES!!
7407 021722 104113 ERROR 113 ;UNS IS SET, DVC IS RESET
7408 021724 000401 BR 70$
7409 021726 60$: ERROR 114 ;UNS IS RESET, DVC IS SET
7410 021726 104114
7411 021730 70$:
7412
7413 ;IF THERE WERE NO PREVIOUS ERRORS, TEST FOR BIT INTERFERENCE ON
7414 ;"MDF"
7415 021730 005703 TST R3
7416 021732 001056 BNE 120$ ;BRANCH IF ANY OTHER ERRORS
7417 021734 012702 000001 MOV #1,R2 ;INITIALIZE TEST PATTERN
7418 021740 80$:
7419
7420 021740 012760 000001 000024 MOV #DMD,RMMR1(RO) ;LOAD RMMR1
7421
7422 021746 012760 000000 000042 MOV #0,RMER2(RO) ;LOAD RMER2
7423
7424 021754 010260 000024 MOV R2,RMMR1(RO) ;LOAD RMMR1
7425
7426 021760 016037 000042 001142 MOV RMER2(RO),$BDDAT ;STORE RMER2 AT $BDDAT
7427 021766 042737 177577 001142 BIC #1CDVC,$BDDAT ;GET RESULTS
7428 021774 005037 001140 CLR $GDDAT ;SETUP EXPECTED
7429 022000 032702 000100 BIT #MDF,R2 ;WAS MDF SET??
7430 022004 001403 BEQ 90$ ;NO!!
7431 022006 052737 000200 001140 BIS #DVC,$GDDAT ;YES-DVC SHOULD BE ON
7432 022014 023737 001140 001142 90$: CMP $GDDAT,$BDDAT
7433 022022 001410 BEQ 100$ ;BRANCH IF DVC IS OK
7434 022024 010037 001136 001136 MOV RO,$BDADR ;SETUP REGISTER ADDRESS
7435 022030 062737 000042 ADD #RMER2,$BDADR
7436 022036 010237 001174 MOV R2,$TMP0 ;SAVE TEST PATTERN
7437 022042 104115 ERROR 115 ;MDF SET BY WRONG BIT
7438 022044 100$:
7439 ;SHIFT TO NEXT BIT POSITION
7440 022044 042702 000001 BIC #DMD,R2 ;DONT SHIFT DMD
7441 022050 001002 BNE 110$
7442 022052 012702 000001 MOV #DMD,R2 ;DONT TRUNCATE TEST
7443 022056 006302 110$: ASL R2 ;SHIFT
7444 022060 001403 BEQ 120$ ;EXIT IF DONE
7445 022062 052702 000001 BIS #DMD,R2 ;KEEP DMD ON
7446 022066 000724 BR 80$ ;CONTINUE
7447
7448 022070 120$: ;END OF TEST
7449
7450 ;*****
7451 ;TEST 33 SEEK ERROR TEST
7452 ;*****
7453 ;*****
7454 022070 000004 TST33: SCOPE
7455 022072 012737 000033 001226 MOV #33,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
7456
7457 022100 000240 NOP
7458 022102 012737 000024 001120 MOV #20,$ICNT ;20 ITERATIONS
7459 022110 112737 000001 001131 MOVB #1,$ERMAX ;ONE ERROR ALLOWED
7460 022116 012737 022132 001122 MOV #T33,$LPADR ;LOAD LOOP ON TEST ADDRESS

```

```

7461 022124 012737 022132 001124      MOV      #T33,$LPERR ;LOAD LOOP ON ERROR ADDRESS
7462 022132      T33:
7463 022132      MOV      #STACK,SP ;LOAD THE STACK POINTER
7464 022136 012706 001100      MOV      $BASE,R0 ;R0 = UNIBUS ADDRESS OF UUT
7465 022142 013700 001276      MOV      TSTQUE,R1 ;R1 = POINTER TO DEVICE
7466 022146 012760 000040 000010      MOV      #CLR,RMCS2(R0) ;CLEAR THE MASSBUS
7467 022154 111160 000010      MOV      (R1),RMCS2(R0) ;SELECT UNIT
7468 022160 005003      CLR      R3 ;CLEAR ERROR FLAGS
7469 022162 010037 001136      MOV      R0,$BDADR ;SETUP REGISTER ADDRESS
7470 022166 062737 000042 001136      ADD      #RMR2,$BDADR
7471
7472      ;SET DIAGNOSTIC MODE AND VERIFY THAT "SKI" CAN BE RESET
7473
7474 022174 012760 000001 000024      MOV      #DMD,RMMR1(R0) ;LOAD RMMR1
7475
7476 022202 012760 000000 000042      MOV      #0,RMER2(R0) ;LOAD RMER2
7477
7478 022210 016037 000042 001142      MOV      RMER2(R0),$BDDAT ;STORE RMER2 AT $BDDAT
7479 022216 042737 137777 001142      BIC      #↑CSKI,$BDDAT
7480 022224 001405      BEQ      10$ ;BRANCH IF SKI IS RESET
7481 022226 005037 001140      CLR      $GDDAT ;SKI SHOULD BE ZERO
7482 022232 104116      ERROR   116 ;CANT RESET "SKI"
7483 022234 052703 000001      BIS      #BIT0,R3 ;SET ERROR FLAG
7484 022240      10$:
7485
7486      ;SET MAINTENANCE SEEK ERROR AND VERIFY THAT "SKI" CAN BE SET
7487
7488 022240 012760 000001 000024      MOV      #DMD,RMMR1(R0) ;LOAD RMMR1
7489
7490 022246 012760 000000 000042      MOV      #0,RMER2(R0) ;LOAD RMER2
7491
7492 022254 012760 000201 000024      MOV      #DMD!MSER,RMMR1(R0) ;LOAD RMMR1
7493
7494 022262 016037 000042 001142      MOV      RMER2(R0),$BDDAT ;STORE RMER2 AT $BDDAT
7495 022270 042737 137777 001142      BIC      #↑CSKI,$BDDAT
7496 022276 001005      BNE     20$ ;BRANCH IF SKI IS SET
7497 022300 012737 040000 001140      MOV      #SI↑,$GDDAT ;CANT SET SKI
7498 022306 052703 000002      BIS      #BIT1,R3 ;SET ERROR FLAG
7499 022312      20$:
7500
7501      ; IF NO PREVIOUS ERROR, CHECK FOR BIT INTERFERENCE SETTING MAINTENANCE
7502      ; SEEK ERROR.
7503 022312 005703      TST      R3
7504 022314 001051      BNE     70$ ;BRANCH IF ANY OTHER ERRORS
7505 022316 012702 000001      MOV      #1,R2 ;INITIALIZE TEST PATTERN
7506 022322      30$:
7507
7508 022322 012760 000001 000024      MOV      #DMD,RMMR1(R0) ;LOAD RMMR1
7509
7510 022330 012760 000000 000042      MOV      #0,RMER2(R0) ;LOAD RMER2
7511
7512 022336 010260 000024      MOV      R2,RMMR1(R0) ;LOAD RMMR1
7513
7514 022342 016037 000042 001142      MOV      RMER2(R0),$BDDAT ;STORE RMER2 AT $BDDAT
7515 022350 042737 137777 001142      BIC      #↑CSKI,$BDDAT ;GET SKI STATUS
7516 022356 005037 001140      CLR      $GDDAT ;SETUP EXPECTED RESULT

```

```

7517 022362 032702 000200          BIT      #MSER,R2
7518 022366 001403          BEQ      40$
7519 022370 052737 040000 001140  BIS     $SKI,$GDDAT      ;SKI SHOULD BE ON
7520 022376 023737 001140 001142 40$:   CMP     $GDDAT,$BDDAT
7521 022404 001403          BEQ      50$              ;BRANCH IF SKI IS OK
7522 022406 010237 001174          MOV     R2,$TMPD        ;SAVE TEST PATTERN
7523 022412 104120          ERROR   120             ;SKI SET BY WRONG BIT
7524 022414          50$:
7525          :
7526 022414 042702 000001          ADVANCE TEST PATTERN IN R2
7527 022420 001002          BIC     #DMD,R2          ;DONT SHIFT DMD BIT
7528 022422 012702 000001          BNE     60$
7529 022426 006302          MOV     #DMD,R2          ;DONT TRUNCATE TEST
7530 022430 001403          ASL     R2               ;SHIFT TO NEXT BIT
7531 022432 052702 000001          BEQ     70$              ;EXIT IF DONE
7532 022436 000731          BIS     #DMD,R2          ;KEEP DMD ON
7533          BR      30$          ;CONTINUE TEST
7534 022440          70$:
7535          ;END OF TEST
7536          ;*****
7537          ;*TEST 34      PIP TEST
7538          ;*****
7539          ;*****
7540 022440 000004          T34:   SCOPE
7541 022442 012737 000034 001226          MOV     #34,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7542          :
7543 022450 000240          NOP
7544 022452 012737 000024 001120          MOV     #20,$ICNT       ;20 ITERATIONS
7545 022460 112737 000001 001131          MOV     #1,$ERMAX       ;ONE ERROR ALLOWED
7546 022466 012737 022502 001122          MOV     #T34,$LPADR    ;LOAD LOOP ON TEST ADDRESS
7547 022474 012737 022502 001124          MOV     #T34,$LPERR    ;LOAD LOOP ON ERROR ADDRESS
7548 022502          T34:
7549 022502 012706 001100          MOV     #STACK,SP       ;LOAD THE STACK POINTER
7550 022506 013700 001276          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
7551 022512 013701 001456          MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
7552 022516 012760 000040 000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
7553 022524 111160 000010          MOV     (R1),RMCS2(R0)  ;SELECT UNIT
7554 022530 005003          CLP     R3              ;RESET ERROR FLAGS
7555 022532 010037 001136          MOV     R0,$BDADR      ;SETUP REGISTER ADDRESS
7556 022536 062737 000012 001136          ADD     #RMS,$BDADR
7557          :
7558          ;SET MAINTENANCE ON CYLINDER "MOC" AND VERIFY THAT "PIP" CAN BE RESET.
7559          :
7560 022544 012760 000001 000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
7561          :
7562 022552 012760 000401 000024          MOV     #DMD!MOC,RMMR1(R0) ;LOAD RMMR1
7563          :
7564 022560 016037 000012 001142          MOV     RMS(R0),$BDDAT  ;STORE RMS AT $BDDAT
7565 022566 042737 157777 001142          BIC     #CPIP,$BDDAT
7566 022574 001405          BEQ     10$              ;BRANCH IF PIP IS RESET
7567 022576 005037 001140          CLR     $GDDAT
7568 022602 104121          ERROR   121             ;CANT RESET PIP
7569 022604 052703 000001          BIS     #BIT0,R3        ;SET ERROR FLAG
7570 022610          10$:
7571          ;RESET MAINTENANCE ON CYLINDER AND VERIFY THAT "PIP" CAN BE SET
7572          :

```

```

7573 022610 012760 000001 000024      MOV      #DMD,RMMR1(RO) ;LOAD RMMR1
7574                                     ;
7575 022616 016037 000012 001142      MOV      RMDS(RO),SBDDAT ;STORE RMDS AT SBDDAT
7576 022624 042737 157777 001142      BIC      #↑CPIP,SBDDAT  ;GET PIP STATUS
7577 022632 001006                                     ;
7578 022634 012737 020000 001140      BNE      20$             ;BRANCH IF PIP IS SET
7579 022642 104122                                     ;
7580 022644 052703 000002                                     ;
7581 022650                                     ;
7582                                     ;
7583 022650 005703                                     ;
7584 022652 001046                                     ;
7585 022654 012702 000001                                     ;
7586 022660                                     ;
7587                                     ;
7588                                     ;
7589 022660 012760 000001 000024      MOV      #DMD,RMMR1(RO) ;LOAD RMMR1
7590                                     ;
7591 022666 010260 000024                                     ;
7592                                     ;
7593 022672 016037 000012 001142      MOV      RMDS(RO),SBDDAT ;STORE RMDS AT SBDDAT
7594 022700 042737 157777 001142      BIC      #↑CPIP,SBDDAT  ;GET PIP STATUS
7595 022706 005037 001140                                     ;
7596 022712 032702 000400                                     ;
7597 022716 001003                                     ;
7598 022720 052737 020000 001140      CLR      $GDDAT         ;SETUP EXPECTED RESULT
7599 022726 023737 001140 001142      BIT      #MOC,R2
7600 022734 001403                                     ;
7601 022736 010237 001174                                     ;
7602 022742 104123                                     ;
7603 022744                                     ;
7604                                     ;
7605 022744 042702 000001                                     ;
7606 022750 001002                                     ;
7607 022752 012702 000001                                     ;
7608 022756 006302                                     ;
7609 022760 001403                                     ;
7610 022762 052702 000001                                     ;
7611 022766 000734                                     ;
7612                                     ;
7613 022770                                     ;
7614                                     ;
7615                                     ;
7616                                     ;
7617                                     ;
7618                                     ;
7619 022770 000004                                     ;
7620 022772 012737 000035 001226      BIC      #DMD,R2       ;DONT SHIFT DMD
7621                                     ;
7622 023000 000240                                     ;
7623 023002 012737 000024 001120      MOV      #DMD,R2       ;DONT TRUNCATE TEST
7624 023010 112737 000001 001131      BNE      60$           ;SHIFT BIT
7625 023016 012737 023032 001122      MOV      #DMD,R2       ;KEEP DMD ON
7626 023024 012737 023032 001124      BEQ      70$           ;EXIT IF DONE
7627 023032                                     ;
7628 023032 012706 001100      BIS      #DMD,R2       ;KEEP DMD ON
                                     ;CONTINUE TEST
                                     ;END OF TEST
                                     ;*****
                                     ;*TEST 35      EBL TEST
                                     ;*****
7629 022770 000004                                     ;
7630 022772 012737 000035 001226      ;ST35: SCOPE
7631                                     ;
7632                                     ;
7633                                     ;
7634                                     ;
7635                                     ;
7636                                     ;
7637                                     ;
7638                                     ;
7639                                     ;
7640                                     ;
7641                                     ;
7642                                     ;
7643                                     ;
7644                                     ;
7645                                     ;
7646                                     ;
7647                                     ;
7648                                     ;
7649                                     ;
7650                                     ;
7651                                     ;
7652                                     ;
7653                                     ;
7654                                     ;
7655                                     ;
7656                                     ;
7657                                     ;
7658                                     ;
7659                                     ;
7660                                     ;
7661                                     ;
7662                                     ;
7663                                     ;
7664                                     ;
7665                                     ;
7666                                     ;
7667                                     ;
7668                                     ;
7669                                     ;
7670                                     ;
7671                                     ;
7672                                     ;
7673                                     ;
7674                                     ;
7675                                     ;
7676                                     ;
7677                                     ;
7678                                     ;
7679                                     ;
7680                                     ;
7681                                     ;
7682                                     ;
7683                                     ;
7684                                     ;
7685                                     ;
7686                                     ;
7687                                     ;
7688                                     ;
7689                                     ;
7690                                     ;
7691                                     ;
7692                                     ;
7693                                     ;
7694                                     ;
7695                                     ;
7696                                     ;
7697                                     ;
7698                                     ;
7699                                     ;
7700                                     ;
7701                                     ;
7702                                     ;
7703                                     ;
7704                                     ;
7705                                     ;
7706                                     ;
7707                                     ;
7708                                     ;
7709                                     ;
7710                                     ;
7711                                     ;
7712                                     ;
7713                                     ;
7714                                     ;
7715                                     ;
7716                                     ;
7717                                     ;
7718                                     ;
7719                                     ;
7720                                     ;
7721                                     ;
7722                                     ;
7723                                     ;
7724                                     ;
7725                                     ;
7726                                     ;
7727                                     ;
7728                                     ;
7729                                     ;
7730                                     ;
7731                                     ;
7732                                     ;
7733                                     ;
7734                                     ;
7735                                     ;
7736                                     ;
7737                                     ;
7738                                     ;
7739                                     ;
7740                                     ;
7741                                     ;
7742                                     ;
7743                                     ;
7744                                     ;
7745                                     ;
7746                                     ;
7747                                     ;
7748                                     ;
7749                                     ;
7750                                     ;
7751                                     ;
7752                                     ;
7753                                     ;
7754                                     ;
7755                                     ;
7756                                     ;
7757                                     ;
7758                                     ;
7759                                     ;
7760                                     ;
7761                                     ;
7762                                     ;
7763                                     ;
7764                                     ;
7765                                     ;
7766                                     ;
7767                                     ;
7768                                     ;
7769                                     ;
7770                                     ;
7771                                     ;
7772                                     ;
7773                                     ;
7774                                     ;
7775                                     ;
7776                                     ;
7777                                     ;
7778                                     ;
7779                                     ;
7780                                     ;
7781                                     ;
7782                                     ;
7783                                     ;
7784                                     ;
7785                                     ;
7786                                     ;
7787                                     ;
7788                                     ;
7789                                     ;
7790                                     ;
7791                                     ;
7792                                     ;
7793                                     ;
7794                                     ;
7795                                     ;
7796                                     ;
7797                                     ;
7798                                     ;
7799                                     ;
7800                                     ;
7801                                     ;
7802                                     ;
7803                                     ;
7804                                     ;
7805                                     ;
7806                                     ;
7807                                     ;
7808                                     ;
7809                                     ;
7810                                     ;
7811                                     ;
7812                                     ;
7813                                     ;
7814                                     ;
7815                                     ;
7816                                     ;
7817                                     ;
7818                                     ;
7819                                     ;
7820                                     ;
7821                                     ;
7822                                     ;
7823                                     ;
7824                                     ;
7825                                     ;
7826                                     ;
7827                                     ;
7828                                     ;
7829                                     ;
7830                                     ;
7831                                     ;
7832                                     ;
7833                                     ;
7834                                     ;
7835                                     ;
7836                                     ;
7837                                     ;
7838                                     ;
7839                                     ;
7840                                     ;
7841                                     ;
7842                                     ;
7843                                     ;
7844                                     ;
7845                                     ;
7846                                     ;
7847                                     ;
7848                                     ;
7849                                     ;
7850                                     ;
7851                                     ;
7852                                     ;
7853                                     ;
7854                                     ;
7855                                     ;
7856                                     ;
7857                                     ;
7858                                     ;
7859                                     ;
7860                                     ;
7861                                     ;
7862                                     ;
7863                                     ;
7864                                     ;
7865                                     ;
7866                                     ;
7867                                     ;
7868                                     ;
7869                                     ;
7870                                     ;
7871                                     ;
7872                                     ;
7873                                     ;
7874                                     ;
7875                                     ;
7876                                     ;
7877                                     ;
7878                                     ;
7879                                     ;
7880                                     ;
7881                                     ;
7882                                     ;
7883                                     ;
7884                                     ;
7885                                     ;
7886                                     ;
7887                                     ;
7888                                     ;
7889                                     ;
7890                                     ;
7891                                     ;
7892                                     ;
7893                                     ;
7894                                     ;
7895                                     ;
7896                                     ;
7897                                     ;
7898                                     ;
7899                                     ;
7900                                     ;
7901                                     ;
7902                                     ;
7903                                     ;
7904                                     ;
7905                                     ;
7906                                     ;
7907                                     ;
7908                                     ;
7909                                     ;
7910                                     ;
7911                                     ;
7912                                     ;
7913                                     ;
7914                                     ;
7915                                     ;
7916                                     ;
7917                                     ;
7918                                     ;
7919                                     ;
7920                                     ;
7921                                     ;
7922                                     ;
7923                                     ;
7924                                     ;
7925                                     ;
7926                                     ;
7927                                     ;
7928                                     ;
7929                                     ;
7930                                     ;
7931                                     ;
7932                                     ;
7933                                     ;
7934                                     ;
7935                                     ;
7936                                     ;
7937                                     ;
7938                                     ;
7939                                     ;
7940                                     ;
7941                                     ;
7942                                     ;
7943                                     ;
7944                                     ;
7945                                     ;
7946                                     ;
7947                                     ;
7948                                     ;
7949                                     ;
7950                                     ;
7951                                     ;
7952                                     ;
7953                                     ;
7954                                     ;
7955                                     ;
7956                                     ;
7957                                     ;
7958                                     ;
7959                                     ;
7960                                     ;
7961                                     ;
7962                                     ;
7963                                     ;
7964                                     ;
7965                                     ;
7966                                     ;
7967                                     ;
7968                                     ;
7969                                     ;
7970                                     ;
7971                                     ;
7972                                     ;
7973                                     ;
7974                                     ;
7975                                     ;
7976                                     ;
7977                                     ;
7978                                     ;
7979                                     ;
7980                                     ;
7981                                     ;
7982                                     ;
7983                                     ;
7984                                     ;
7985                                     ;
7986                                     ;
7987                                     ;
7988                                     ;
7989                                     ;
7990                                     ;
7991                                     ;
7992                                     ;
7993                                     ;
7994                                     ;
7995                                     ;
7996                                     ;
7997                                     ;
7998                                     ;
7999                                     ;
8000                                     ;

```

7629	023036	013700	001276		MOV	\$BASE,R0	;R0 = UNIBUS ADDRESS OF UUT
7630	023042	013701	001456		MOV	TSTQUE,R1	;R1 = POINTER TO DEVICE
7631	023046	005003			CLR	R3	;RESET ERROR FLAGS
7632	023050	010037	001136		MOV	R0,\$BDAOR	;SETUP REGISTER ADDRESS
7633	023054	062737	000024	001136	ADD	#RMMR1,\$BDAOR	
7634	023062	005037	001140		CLR	\$GDDAT	;SETUP EXPECTED RESULT
7635					;CLEAR AND VERIFY THAT END OF BLOCK IS RESET		
7636	023066	012760	000040	000010	MOV	#CLR, RMCS2(R0)	;CLEAR THE MASSBUS
7637	023074	111160	000010		MOVB	(R1),RMCS2(R0)	;SELECT UNIT
7638							
7639	023100	016037	000024	001142	MOV	RMMR1(R0), \$BDDAT	;STORE RMMR1 AT \$BDDAT
7640	023106	042737	157777	001142	BIC	#ICEBL, \$BDDAT	
7641	023114	001403			BEQ	10\$;BRANCH IF EBL IS RESET
7642	023116	104125			ERROR	124	;CANT CLEAR EBL
7643	023120	052703	000001		BIS	#BIT0,R3	;SET ERROR FLAG
7644	023124				10\$:		
7645					;SET AND RESET DIAGNOSTIC END OF BLOCK, CHECK FOR EBL S-A-1.		
7646							
7647	023124	012760	000001	000024	MOV	#DMD,RMMR1(R0)	;LOAD RMMR1
7648							
7649	023132	012760	020001	000024	MOV	#DMD!DEBL,RMMR1(R0)	;LOAD RMMR1
7650							
7651	023140	012760	000001	000024	MOV	#DMD,RMMR1(R0)	;LOAD RMMR1
7652							
7653	023146	016037	000024	001142	MOV	RMMR1(R0), \$BDDAT	;STORE RMMR1 AT \$BDDAT
7654	023154	042737	157777	001142	BIC	#ICEBL, \$BDDAT	
7655	023162	001403			BEQ	20\$;BRANCH IF EBL IS RESET
7656	023164	104125			ERROR	125	;DEBL APPEARS S-A-1
7657	023166	052703	000001		BIS	#BIT0,R3	;SET ERROR FLAG
7658	023172				20\$:		
7659					;RESET AND SET DIAGNOSTIC END OF BLOCK, CHECK FOR EBL S-A-0		
7660							
7661	023172	012760	000001	000024	MOV	#DMD,RMMR1(R0)	;LOAD RMMR1
7662							
7663	023200	012760	020001	000024	MOV	#DMD!DEBL,RMMR1(R0)	;LOAD RMMR1
7664							
7665	023206	016037	000024	001142	MOV	RMMR1(R0), \$BDDAT	;STORE RMMR1 AT \$BDDAT
7666	023214	042737	157777	001142	BIC	#ICEBL, \$BDDAT	
7667	023222	001006			BNE	30\$;BRANCH IF EBL IS SET
7668	023224	012737	020000	001140	MOV	#EBL, \$GDDAT	
7669	023232	104126			ERROR	126	;CANT SET EBL
7670	023234	052703	000002		BIS	#BIT1,R3	;SET ERROR FLAG
7671	023240				30\$:		
7672					;IF NO PREVIOUS ERRORS, TEST FOR ADJACENT BIT INTERFERENCE ON "DEBL".		
7673	023240	005703			TST	R3	
7674	023242	001042			BNE	70\$;BRANCH IF ANY ERROR
7675	023244	012702	000001		MOV	#1,R2	;INITIALIZE TEST PATTERN
7676	023250				40\$:		
7677					;WRITE, READ AND VERIFY THE TEST PATTERN IN R2		
7678							
7679	023250	012760	000001	000024	MOV	#DMD,RMMR1(R0)	;LOAD RMMR1
7680							
7681	023256	010260	000024		MOV	R2,RMMR1(R0)	;LOAD RMMR1
7682							
7683	023262	016037	000024	001142	MOV	RMMR1(R0), \$BDDAT	;STORE RMMR1 AT \$BDDAT
7684	023270	042737	157777	001142	BIC	#ICEBL, \$BDDAT	

```

7685 023276 010203          MOV      R2,R3          ;GENERATE EXPECTED RESULT
7686 023300 042703 157777    BIC      #1,CEBL,R3
7687 023304 020337 001142    CMP      R3,$BDDAT
7688 023310 001405          BEQ      50$           ;BRANCH IF EBL IS OK
7689 023312 010337 001140    MOV      R3,$GDDAT    ;SAVE EXPECTED RESULT
7690 023316 010237 001174    MOV      R2,$TMPD     ;SAVE TEST PATTERN
7691 023322 104127          ERROR    127         ;DEBL SET(RESET) BY WRONG BIT
7692 023324          50$:
7693          ;
7694 023324 042702 000001    ;SHIFT TO NEXT BIT POSITION
7695 023330 001002          BIC      #DMD,R2      ;DONT SHIFT DMD
7696 023332 012702 000001    BNE      60$
7697 023336 006302          MOV      #DMD,R2      ;DONT TRUNCATE DMD
7698 023340 001403          ASL      R2           ;SHIFT TO NEXT BIT
7699 023342 052702 000001    BEQ      70$         ;EXIT IF DONE
7700 023346 000740          BIS      #DMD,R2      ;KEEP DMD ON
7701          BR       40$         ;CONTINUE TEST
7702 023350          70$:
7703          ;
7704          ;*****
7705          ;*TEST 36      LAST SECTOR, LAST TRACK TEST
7706          ;*****
7707          ;*****
7708 023350 000004          †T36:  SCOPE
7709 023352 012737 000036 001226    MOV      #36,$TESTN   ;;SET TEST NUMBER IN APT MAIL BOX
7710 023360 000240          NOP
7711 023362 012737 000024 001120    MOV      #20,$ICNT    ;20 ITERATIONS
7712 023370 112737 000001 001131    MOV      #1,$ERMAX    ;ONE ERROR ALLOWED
7713 023376 012737 023412 001122    MOV      #T36,$LPADR  ;LOAD LOOP ON TEST ADDRESS
7714 023404 012737 023412 001124    MOV      #T36,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
7715 023412          T36:
7716 023412 012706 001100          MOV      #STACK,SP    ;LOAD THE STACK POINTER
7717 023416 013700 001276          MOV      $BASE,R0     ;R0 = UNIBUS ADDRESS OF UUT
7718 023422 013701 001456          MOV      TSTQUE,R1    ;R1 = POINTER TO DEVICE
7719 023426 012760 000040 000010    MOV      #CLR,RMCS2(R0);CLEAR THE MASSBUS
7720 023434 111160 000010          MOV      (R1),RMCS2(R0);SELECT UNIT
7721 023440 005002          CLR      R2           ;INITIALIZE TEST PATTERN
7722 023442 010037 001136          MOV      R0,$BDADR    ;SETUP REGISTER ADDRESS
7723 023446 062737 000024 001136    ADD      #RMMR1,$BDADR
7724 023454 005037 001434          CLR      RMOFO        ;START IN 18 BIT MODE
7725 023460          10$:
7726          ;TRANSFER TEST PATTERN TO RMDA THEN VERIFY LAST SECTOR "LS" AND LAST
7727          ;SECTOR/TRACK "LST" FOR EACH TRANSFER. THE TABLE BELOW LISTS THE VALUE
7728          ;OF RMDA FOR WHICH LS AND LST ARE SET.
7729          ;
7730          ;
7731          ;LS=      18 BIT MODE      16 BIT MODE
7732          ;          XXX035          XXX037
7733          ;
7734          ;LST=     002035          002037
7735          ;
7736 023460 013760 001434 000032    MOV      RMOFO,RMOF(R0);LOAD RMOF
7737          ;
7738 023466 010260 000006          MOV      R2,RMDA(R0) ;LOAD RMDA
7739          ;
7740 023472 016037 000024 001352    MOV      RMMR1(R0),RMMR1I ;STORE RMMR1 IN INPUT BUFFER

```

```

7741 023500 013737 001352 001142 MOV RMMR1I,$BDDAT ;VERIFY "LS"
7742 023506 042737 177773 001142 BIC #1CL5,$BDDAT
7743 023514 005037 001140 CLR $GDDAT ;GENERATE EXPECTED "LS"
7744 023520 032737 010000 001434 BIT #FMT16,RMOFO ;16 BIT MODE??
7745 023526 001007 BNE 20$ ;YES!!
7746 023530 122702 000035 CMPB #035,R2
7747 023534 001012 BNE 30$
7748 023536 052737 000004 001140 BIS #LS,$GDDAT ;LS SHOULD BE ON-18 BIT MODE
7749 023544 000406 BR 30$
7750 023546 122702 000037 20$: CMPB #037,R2
7751 023552 001003 BNE 30$
7752 023554 052737 000004 001140 BIS #LS,$GDDAT ;LS SHOULD BE ON-16 BIT MODE
7753 023562 023737 001140 001142 30$: CMP $GDDAT,$BDDAT
7754 023570 001407 BEQ 40$ ;BRANCH IF LS IS CORRECT
7755 023572 010237 001174 MOV R2,$TMP0 ;SAVE TEST PATTERN
7756 023576 013737 001434 001176 MOV RMOFO,$TMP1 ;SAVE OFFSET REGISTER FOR ERROR
7757 023604 104130 ERROR 130 ;LS NOT CORRECT FOR RMDA
7758 023606 000453 BR 80$ ;SKIP TO NEXT
7759 023610 013737 001352 001142 40$: MOV RMMR1I,$BDDAT ;VERIFY "LST"
7760 023616 042737 177775 001142 BIC #1CLST,$BDDAT
7761 023624 005037 001140 CLR $GDDAT ;GENERATE EXPECTED "LST"
7762 023630 032737 010000 001434 BIT #FMT16,RMOFO ;16 BIT MODE??
7763 023636 001007 BNE 50$ ;YES!!
7764 023640 022702 002035 CMP #002035,R2
7765 023644 001012 BNE 60$
7766 023646 052737 000002 001140 BIS #LST,$GDDAT ;LST SHOULD BE ON-18 BIT MODE
7767 023654 000406 BR 60$
7768 023656 022702 002037 50$: CMP #002037,R2
7769 023662 001003 BNE 60$
7770 023664 052737 000002 001140 BIS #LST,$GDDAT ;LST SHOULD BE ON-16 BIT MODE
7771 023672 023737 001140 001142 60$: CMP $GDDAT,$BDDAT
7772 023700 001404 BEQ 70$
7773 023702 010237 001174 MOV R2,$TMP0 ;SAVE TEST PATTERN
7774 023706 104131 ERROR 131 ;LST NOT CORRECT FOR RMDA
7775 023710 000412 BR 80$ ;SKIP TO NEXT
7776
7777 023712 70$:
7778 ; ADVANCE TO NEXT TEST PATTERN, CHANGE TO 16 BIT MODE IF ALL
7779 ; 18 BIT TESTS DONE.
7780 023712 005202 INC R2 ;INCREMENT PATTERN
7781 023714 001261 BNE 10$ ;CONTINUE IF NOT DONE
7782 023716 032737 010000 001434 BIT #FMT16,RMOFO ;DONE 16 BIT TOO??
7783 023724 001004 BNE 80$ ;YES!!
7784 023726 012737 010000 001434 MOV #FMT16,RMOFO ;DO 16 BIT FORMAT TEST
7785 023734 000651 BR 10$
7786
7787 023736 80$: ;END OF TEST
7788
7789 ;*****
7790 ;*TEST 37 RMDA COUNT TEST
7791
7792 ;*****
7793 023736 000004 †ST37: SCOPE
7794 023740 012737 000037 001226 MOV #37,$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
7795
7796 023746 000240 NOP

```

7797	023750	012737	000024	001120	MOV	#20, \$ICNT	:20 ITERATIONS
7798	023756	112737	000001	001131	MOVB	#1, \$ERMAX	:ONE ERROR ALLOWED
7799	023764	012737	024000	001122	MOV	#T37, \$LPADR	:LOAD LOOP ON TEST ADDRESS
7800	023772	012737	024000	001124	MOV	#T37, \$LPERR	:LOAD LOOP ON ERROR ADDRESS
7801	024000				T37:		
7802	024000	012706	001100		MOV	#STACK, SP	:LOAD THE STACK POINTER
7803	024004	013700	001276		MOV	\$BASE, RO	:RO = UNIBUS ADDRESS OF UUT
7804	024010	013701	001456		MOV	TSTQUE, R1	:R1 = POINTER TO DEVICE
7805	024014	010037	001136		MOV	RO, \$BDADR	:SETUP REGISTER ADDRESS
7806	024020	062737	000006	001136	ADD	#RMDA, \$BDADR	
7807	024026	005037	001434		CLR	RMOFO	:START WITH 18 BIT FORMAT
7808	024032	012737	000035	024454	MOV	#035, \$110\$:SETUP LAST SECTOR FOR 18 BIT
7809	024040	012737	000001	001140	MOV	#1, \$GDDAT	:SETUP FIRST COUNT
7810					::	*****	*****

L13

CZRMJBO RMC3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 167
T37 RMDA COUNT TEST

SEQ 0167

7811
7812

; INCREMENT SECTOR COUNT USING DIAGNOSTIC END OF BLOCK STARTING AT
; SECTOR 0 AND CONTINUING UNTIL TRACK ADDRESS INCREMENTS

7813	024046				105:	
7814					:	.CLEAR THE MASSBUS
7815					:	.SET FORMAT
7816					:	.LOAD SECTOR AND TRACK ADDRESS
7817					:	.ENABLE DEBUG CLOCK
7818					:	.SET GO BIT
7819	024046	012760	000040	000010	MOV	#CLR, RMCS2(RO) ;CLEAR THE MASSBUS
7820	024054	111160	000010		MOV6	(R1), RMCS2(RO) ;SELECT UNIT
7821						
7822	024060	012760	000001	000024	MOV	#DMD, RMMR1(RO) ;LOAD RMMR1
7823						
7824	024066	012760	000000	000014	MOV	#0, RMER1(RO) ;LOAD RMER1
7825						
7826	024074	012760	000000	000042	MOV	#0, RMER2(RO) ;LOAD RMER2
7827						
7828	024102	012760	000000	000006	MOV	#0, RMDA(RO) ;LOAD RMDA
7829						
7830	024110	013760	001434	000032	MOV	RMOFO, RMOF(RO) ;LOAD RMOF
7831	024116	013737	001434	001174	MOV	RMOFO, STMPD ;SAVE OFFSET FOR ERROR TYPE
7832						
7833	024124	012760	040001	000024	MOV	#DMD!DBEN, RMMR1(RO) ;LOAD RMMR1
7834						
7835	024132	012760	000001	000000	MOV	#GO, RMCS1(RO) ;LOAD RMCS1

```

7836 024140      25$:
7837
7838           ; SET AND RESET EBL TO INCREMENT RMDA THEN VERIFY RMDA.
7839
7840 024140 012760 060001 000024      MOV      #DMD!DBEN!DEBL,RMMR1(RO)      ;LOAD RMMR1
7841
7842 024146 012760 040001 000024      MOV      #DMD!DBEN,RMMR1(RO)      ;LOAD RMMR1
7843
7844 024154 016037 000006 001142      MOV      RMDA(RO), $BDDAT ;STORE RMDA AT $BDDAT
7845 024162 023737 001142 001140      CMP      $BDDAT,$GDDAT
7846 024170 001402                      BEQ      30$ ;BRANCH IF RMDA OK
7847 024172 104132                      ERROR   132 ;SECTOR COUNT NOT INCREMENTING
7848 024174 000416                      BR      50$ ;OUT OF SYNC-SKIP TO NEXT
7849 024176
7850           30$: ADVANCE EXPECTED SECTOR COUNT AND CONTINUE IF ONE CYCLE NOT
7851           ; COMPLETE
7852 024176 005237 001140      INC      $GDDAT ;INCREMENT EXPECTED SECTOR
7853 024202 123737 001142 024454      CMPB    $BDDAT,110$ ;WAS THE LAST SECTOR JUST COUNTED??
7854 024210 001004                      BNE     40$ ;NO!!
7855 024212 105037 001140      CLRB    $GDDAT ;YES-NEXT SECTOR SHOULD BE ZERO
7856 024216 105237 001141      INCB    $GDDAT+1 ;INCREMENT TRACK ADDRESS
7857 024222 105737 001142      40$: TSTB    $BDDAT ;HAS A FULL CYCLE BEEN COUNTED??
7858 024226 001401                      BEQ     50$ ;YES-DO NEXT
7859 024230 000743                      BR      25$ ;CONTINUE SECTOR TEST
7860
7861 024232           50$:
7862           ;*****
7863           ; INCREMENT TRACK COUNT USING DIAGNOSTIC END OF BLOCK. START AT TRACK 0.
7864           ; LAST SECTOR AND COUNT ONE COMPLETE TRACK CYCLE.
7865 024232 013737 024454 001410      MOV      110$,RMDAO ;STARTING TRACK ADDRESS
7866 024240 012737 000400 001140      MOV      #000400,$GDDAT ;FIRST VALUE AFTER INCREMENT
7867
7868 024246           60$:
7869           ;
7870           ; .CLEAR THE MASSBUS
7871           ; .SET FORMAT
7872           ; .LOAD LAST SECTOR ADDRESS AND TEST TRACK ADDRESS
7873           ; .ENABLE DEBUG CLOCK
7874           ; .SET GO BIT
7875
7876 024246 012760 000040 000010      MOV      #CLR,RMCS2(RO) ;CLEAR THE MASSBUS
7877 024254 111160 000010                      MOVB    (R1),RMCS2(RO) ;SELECT UNIT
7878
7879 024260 013760 001434 000032      MOV      RMOFO,RMOF(RO) ;LOAD RMOF
7880
7881 024266 013760 001410 000006      MOV      RMDAO,RMDA(RO) ;LOAD RMDA
7882
7883 024274 012760 000001 000024      MOV      #DMD,RMMR1(RO) ;LOAD RMMR1
7884
7885 024302 012760 040001 000024      MOV      #DMD!DBEN,RMMR1(RO) ;LOAD RMMR1
7886
7887 024310 012760 000001 000000      MOV      #GO,RMCS1(RO) ;LOAD RMCS1
7888           ;CLOCK RMDA USING DIAGNOSTIC END OF BLOCK
7889
7890 024316 012760 060001 000024      MOV      #DMD!DBEN!DEBL,RMMR1(RO) ;LOAD RMMR1
7891

```



```
7925  
7926  
7927  
7928  
7929  
7930  
7931 024456 000004  
7932 024460 012737 000040 001226  
7933  
7934 024466 000240  
7935 024470 012737 000024 001120  
7936 024476 112737 000001 001131  
7937 024504 012737 024520 001122  
7938 024512 012737 024520 001124  
7939 024520  
7940 024520 012706 001100  
7941 024524 013700 001276  
7942 024530 013701 001456  
7943 024534 012760 000040 000010  
7944 024542 111160 000010  
7945  
7946 024546 012737 001136  
7947 024552 062737 000034 001136  
7948 024560 005037 001434  
7949 024564 012737 002035 001410  
7950 024572 012737 000001 001140  
7951  
7952 024600 012760 000000 000034  
7953  
7954 024606 013760 001434 000032  
7955 024614 013737 001434 001174  
7956 024622  
7957  
7958  
7959  
7960  
7961  
7962  
7963  
7964 024622 012760 000040 000010  
7965 024630 111160 000010  
7966  
7967 024634 013760 001434 000032  
7968  
7969 024642 013760 001410 000006  
7970  
7971 024650 012760 000001 000024  
7972  
7973 024656 012760 040001 000024  
7974  
7975 024664 012760 000001 000000  
7976  
7977  
7978 024672 012760 060001 000024  
7979  
7980 024700 012760 040001 000024
```

```
*****  
; *TEST 40 RMDA COUNT TEST  
*****  
T40: SCOPE  
MOV #40,$TESTN ; ; SET TEST NUMBER IN APT MAIL BOX  
  
NOP  
MOV #20,$ICNT ; ; 20 ITERATIONS  
MOVB #1,$ERMAX ; ; ONE ERROR ALLOWED  
MOV #T40,$LPADR ; ; LOAD LOOP ON TEST ADDRESS  
MOV #T40,$LPERR ; ; LOAD LOOP ON ERROR ADDRESS  
  
T40: MOV #STACK,SP ; ; LOAD THE STACK POINTER  
MOV $BASE,R0 ; ; R0 = UNIBUS ADDRESS OF UUT  
MOV $TQUEUE,R1 ; ; R1 = POINTER TO DEVICE  
MOV #CLR,RMCS2(R0) ; ; CLEAR THE MASSBUS  
MOVB (R1),RMCS2(R0) ; ; SELECT UNIT  
; CLEAR RMDA, SET FORMAT AND SETUP PROGRAM PARAMETERS  
MOV R0,$BDAOR ; ; SETUP REGISTER ADDRESS  
ADD #RMDA,$BDAOR  
CLR RMOFO ; ; START WITH 18 BIT FORMAT  
MOV #002035,RMDA0 ; ; LOAD LAST SECTOR/TRACK VALUE  
105: MOV #1,$GDDAT ; ; LOAD FIRST INCREMENTAL VALUE  
  
MOV #0,RMDC(R0) ; ; LOAD RMDC  
  
MOV RMOFO,RMOF(R0) ; ; LOAD RMOF  
MOV RMOFO,$TMPD ; ; SAVE FOR ERROR MSG  
  
205: .CLEAR THE MASSBUS  
; .SET OFFSET  
; .LOAD LAST SECTOR AND TRACK ADDRESS  
; .ENABLE DEBUG CLOCK  
; .SET GO BIT  
  
MOV #CLR,RMCS2(R0) ; ; CLEAR THE MASSBUS  
MOVB (R1),RMCS2(R0) ; ; SELECT UNIT  
  
MOV RMOFO,RMOF(R0) ; ; LOAD RMOF  
  
MOV RMDA0,RMDA(R0) ; ; LOAD RMDA  
  
MOV #DMD,RMMR1(R0) ; ; LOAD RMMR1  
  
MOV #DMD!DBEN,RMMR1(R0) ; ; LOAD RMMR1  
  
MOV #GO,RMCS1(R0) ; ; LOAD RMCS1  
; CLOCK THE CYLINDER ADDRESS USING DEBL  
  
MOV #DMD!DBEN!DEBL,RMMR1(R0) ; ; LOAD RMMR1  
  
MOV #DMD!DBEN,RMMR1(R0) ; ; LOAD RMMR1
```

```

7981
7982 024706 016037 000034 001142      MOV      RMDC(RO), $BDDAT ;STORE RMDC AT $BDDAT
7983 024714 023737 001140 001142      CMP      $GDDAT, $BDDAT
7984 024722 001402                      BEQ      30$              ;BRANCH IF RMDC=RMDC+1
7985 024724 104140                      ERROR   140              ;CANT INCREMENT RMDC
7986 024726 000427                      BR      60$              ;OUT OF SYNC-SKIP TO END
7987 024730
7988
7989 024730 005237 001140      30$:      ;ADVANCE EXPECTED RESULT FOR NEXT INCREMENT
7990 024734 022737 002000 001140      INC      $GDDAT          ;ADVANCE NEXT RESULT
7991 024742 001002                      CMP      #1777+1, $GDDAT ;SHOULD NEXT VALUE BE ZERO??
7992 024744 005037 001140      BNE      40$              ;NO!!
7993 024750 005737 001142      CLR      $GDDAT          ;YES-RMDC SHOULD OVERFLOW
7994 024754 001401      40$:      TST      $BDDAT          ;IS ONE CYCLE COMPLETE??
7995 024756 000721                      BEQ      50$              ;YES!!
7996 024760                      BR      20$              ;CONTINUE
7997
7998 024760 032737 010000 001434      50$:      ;REPEAT TEST IN 16 BIT MODE-ELSE DONE
7999 024766 001007                      BIT      #FMT16, RMOFO   ;DONE BOTH MODES??
8000 024770 012737 010000 001434      BNE      60$              ;YES!!
8001 024776 012737 002037 001410      MOV      #FMT16, RMOFO   ;SET 16 BIT FORMAT
8002 025004 000672                      MOV      #002037, RMDA0  ;LOAD LAST SECTOR/TRACK VALUE
8003
8004 025006                      BR      10$              ;REPEAT TEST
8005
8006
8007
8008
8009
8010 025006 000004      60$:      ;END OF TEST
8011 025010 012737 000041 001226      ;*****
8012
8013
8014
8015
8016
8017
8018
8019
8020
8021
8022
8023
8024
8025
8026
8027
8028
8029
8030
8031
8032
8033
8034
8035
8036

```

```

8037 025124 012760 001466 000034      MOV      #822.,RMDC(RO) ;LOAD RMDC
8038
8039 025132 016037 000012 001142      MOV      RMDS(RO), $BDDAT ;STORE RMDS AT $BDDAT
8040 025140 042737 175777 001142      BIC      #↑CLBT, $BDDAT
8041 025146 001403          BEQ      10$ ;BRANCH IF LBT IS RESET
8042 025150 005037 001140      CLR      $GDDAT ;LBT SHOULD BE ZERO
8043 025154 104141          ERROR   141 ;CANNOT RESET "LBT"
8044
8045 025156          10$:
8046          :      ENABLE DEBUG CLOCK
8047          :      SET GO
8048          :      FORCE EBL
8049          :
8050          :      VERIFY THAT LBT IS SET
8051          :
8052 025156 012760 000001 000024      MOV      #DMD, RMMR1(RO) ;LOAD RMMR1
8053
8054 025164 012760 000000 000014      MOV      #0, RMER1(RO) ;LOAD RMER1
8055
8056 025172 012760 000000 000042      MOV      #0, RMER2(RO) ;LOAD RMER2
8057
8058 025200 012760 000001 000000      MOV      #GO, RMCS1(RO) ;LOAD RMCS1
8059
8060 025206 012760 060001 000024      MOV      #DMD!DBEN!DEBL, RMMR1(RO) ;LOAD RMMR1
8061
8062 025214 012760 040001 000024      MOV      #DMD!DBEN, RMMR1(RO) ;LOAD RMMR1
8063
8064 025222 016037 000012 001142      MOV      RMDS(RO), $BDDAT ;STORE RMDS AT $BDDAT
8065 025230 042737 175777 001142      BIC      #↑CLBT, $BDDAT
8066 025236 001004          BNE      20$ ;BRANCH IF LBT IS SET
8067 025240 012737 002000 001140      MOV      #LBT, $GDDAT
8068 025246 104142          ERROR   142 ;CANT SET LBT
8069
8070 025250          20$: ;END OF TEST
8071
8072          : *****
8073          : *TEST 42 COMPOSITE ERROR TEST
8074          : *****
8075          : *****
8076 025250 000004      †ST42: SCOPE
8077 025252 012737 000042 001226      MOV      #42, $TESTN ;;SET TEST NUMBER IN APT MAIL BOX
8078
8079 025260 000240      NOP
8080 025262 012737 000024 001120      MOV      #20, $ICNT ;20 ITERATIONS
8081 025270 112737 000001 001131      MOV      #1, $ERMAX ;ONE ERROR ALLOWED
8082 025276 012737 025312 001122      MOV      #T42, $LPADR ;LOAD LOOP ON TEST ADDRESS
8083 025304 012737 025312 001124      MOV      #T42, $LPERR ;LOAD LOOP ON ERROR ADDRESS
8084 025312
8085 025312 012706 001100      MOV      #STACK, SP ;LOAD THE STACK POINTER
8086 025316 013700 001276      MOV      $BASE, RO ;RO = UNIBUS ADDRESS OF UUT
8087 025322 013701 001456      MOV      TSTQUE, R1 ;R1 = POINTER TO DEVICE
8088 025326 012760 000040 000010      MOV      #CLR, RMCS2(RO) ;CLEAR THE MASSBUS
8089 025334 111160 000010      MOV      (R1), RMCS2(RO) ;SELECT UNIT
8090 025340 010037 001136      MOV      RO, $BDADR ;SETUP REGISTER ADDRESS
8091 025344 062737 000012 001136      ADD      #RMDS, $BDADR
8092          :USING DIAGNOSTIC MODE, CLEAR ALL ERRORS AND VERIFY THAT COMPOSITE

```

```

8093 ;ERROR IS RESET.
8094
8095 025352 012760 000001 000024      MOV      #DMD,RMMR1(RO) ;LOAD RMMR1
8096
8097 025360 012760 000000 000014      MOV      #0,RMER1(RO)  ;LOAD RMER1
8098
8099 025366 012760 000000 000042      MOV      #0,RMER2(RO)  ;LOAD RMER2
8100
8101 025374 016037 000012 001142      MOV      RMDS(RO), $BDDAT ;STORE RMDS AT $BDDAT
8102 025402 042737 137777 001142      BIC      #↑CERR,$BDDAT
8103 025410 001403                BEQ      10$ ;BRANCH IF ERR IS RESET
8104 025412 005037 001140      CLR      $GDDAT
8105 025416 104143                ERROR   143 ;CANT READ ZERO FROM ERR
8106 025420 012737 040000 001140 10$: MOV      #ERR,$GDDAT
8107 ;SET BOTH ERROR REGISTERS AND VERIFY THAT COMPOSITE ERROR IS SET
8108
8109 025426 012760 177777 000014      MOV      #-1,RMER1(RO) ;LOAD RMER1
8110
8111 025434 012760 177777 000042      MOV      #-1,RMER2(RO) ;LOAD RMER2
8112
8113 025442 016037 000012 001142      MOV      RMDS(RO), $BDDAT ;STORE RMDS AT $BDDAT
8114 025450 042737 137777 001142      BIC      #↑CERR,$BDDAT
8115 025456 001001                BNE     20$ ;BRANCH IF ERR IS SET
8116 025460 104144                ERROR   144 ;CANT SET ERR
8117 025462
8118 20$:
8119 ;VERIFY THAT COMPOSITE ERROR SETS FOR EACH BIT OF RMER1
8120 025462 012702 000001      MOV      #1,R2 ;INITIALIZE TEST PATTERN
8121 025466
8122 ;
8123 025466 012760 000040 000010      MOV      #CLR, RMCS2(RO) ;CLEAR THE MASSBUS
8124 025474 111160 000010      MOV      (R1), RMCS2(RO) ;SELECT UNIT
8125
8126 025500 012760 000001 000024      MOV      #DMD,RMMR1(RO) ;LOAD RMMR1
8127 025506 012760 000000 000014      MOV      #0,RMER1(RO)  ;LOAD RMER1
8128
8129 025514 012760 000000 000042      MOV      #0,RMER2(RO)  ;LOAD RMER2
8130
8131 025522 010260 000014      MOV      R2,RMER1(RO)  ;LOAD RMER1
8132
8133 025526 016037 000012 001142      MOV      RMDS(RO), $BDDAT ;STORE RMDS AT $BDDAT
8134 025534 042737 137777 001142      BIC      #↑CERR,$BDDAT
8135 025542 001005                BNE     40$ ;BRANCH IF COMPOSITE ERROR SET
8136 025544 010237 001174      MOV      R2,$TMP0 ;SAVE RMER1 TEST PATTERN
8137 025550 005037 001176      CLR      $TMP1 ;SAVE RMER2 TEST PATTERN
8138 025554 104145                ERROR   145 ;ERR NOT SET BY RMER1 ERROR
8139 025556
8140 40$:
8141 ; ADVANCE THE TEST PATTERN FOR RMER1
8142 025556 006302      ASL     R2
8143 025562 001342      BNE     30$ ;CONTINUE IF TEST NOT DONE
8144
8145 50$:
8146 ;VERIFY THAT COMPOSITE ERROR SETS FOR EACH BIT OF RMER2
8147 025562 012702 000001      MOV      #1,R2 ;INITIALIZE TEST PATTERN
8148 025566
8149 60$:
8150 ; WRITE THE TEST PATTERN AND VERIFY THAT ERR IS SET
8151 025566 012760 000040 000010      MOV      #CLR, RMCS2(RO) ;CLEAR THE MASSBUS

```



```

8149 025574 111160 000010      MOV      (R1),RMCS2(RO) ;SELECT UNIT
8150
8151 025600 012760 000001 000024      MOV      #DMD,RMMR1(RO) ;LOAD RMMR1
8152
8153 025606 012760 000000 000014      MOV      #0,RMER1(RO) ;LOAD RMER1
8154
8155 025614 012760 000000 000042      MOV      #0,RMER2(RO) ;LOAD RMER2
8156
8157 025622 010260 000042      MOV      R2,RMER2(RO) ;LOAD RMER2
8158
8159 025626 016037 000012 001142      MOV      RMD5(RO), $BDDAT ;STORE RMD5 AT $BDDAT
8160 025634 042737 137777 001142      BIC      #CERR,$BDDAT
8161 025642 012737 040000 001140      MOV      #ERR,$GDDAT ;SETUP EXPECTED VALUE FOR COMP ERROR
8162 025650 032702 001567      BIT      #XNUMR2,R2
8163 025654 001402      BEQ      65$ ;BRANCH IF TEST BIT IS A USED BIT
8164 025656 005037 001140      CLR      $GDDAT ;TEST BIT IS NOT USED - ERR SHOULD BE 0
8165 025662 023737 001140 001142 65$:      CMP      $GDDAT,$BDDAT
8166 025670 001405      BEQ      70$ ;BRANCH IF COMP ERROR IS OK
8167 025672 005037 001174      CLR      $TMP0 ;SAVE RMER1 TEST PATTERN
8168 025676 010237 001176      MOV      R2,$TMP1 ;SAVE RMER2 TEST PATTERN
8169 025702 104145      ERROR    14$ ;ERR NOT SET BY RMER2 ERROR
8170 025704      70$:
8171      ;      ADVANCE THE TEST PATTERN FOR RMER2
8172 025704 006302      ASL      R2
8173 025706 001327      BNE      60$ ;CONTINUE IF TEST NOT DONE
8174
8175 025710      80$: ;END OF TEST
8176
8177      ;*****
8178      ;TEST 43 WRITE GO TEST
8179
8180      ;*****
8181 025710 000004      TST43:  SCOPE
8182 025712 012737 000043 001226      MOV      #43,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
8183
8184 025720 000240      NOP
8185 025722 012737 000024 001120      MOV      #20,$ICNT ;20 ITERATIONS
8186 025730 112737 000001 001131      MOV      #1,$ERMAX ;ONE ERROR ALLOWED
8187 025736 012737 025752 001122      MOV      #T43,$LPADR ;LOAD LOOP ON TEST ADDRESS
8188 025744 012737 025752 001124      MOV      #T43,$LPERR ;LOAD LOOP ON ERROR ADDRESS
8189 025752      T43:
8190 025752 012706 001100      MOV      #STACK,SP ;LOAD THE STACK POINTER
8191 025756 013700 001276      MOV      $BASE,R0 ;R0 = UNIBUS ADDRESS OF UUT
8192 025762 013701 001456      MOV      TSTQUE,R1 ;R1 = POINTER TO DEVICE
8193 025766 010037 001136      MOV      R0,$B0ADR ;COPY RMCS1 ADDRESS
8194 025772 005002      CLR      R2 ;INITIALIZE FUNCTION CODE
8195 025774      10$:
8196
8197      ;CLEAR THE MASSBUS, SET DIAGNOSTIC MODE AND ENABLE DEBUG CLOCK
8198 025774 012760 000040 000010      MOV      #CLR,RMCS2(RO) ;CLEAR THE MASSBUS
8199 026002 111160 000010      MOV      (R1),RMCS2(RO) ;SELECT UNIT
8200
8201 026006 012760 000001 000024      MOV      #DMD,RMMR1(RO) ;LOAD RMMR1
8202
8203 026014 012760 041001 000024      MOV      #DMD!DBEN!MUR,RMMR1(RO) ;LOAD RMMR1
8204

```

```

8205 026022 012760 000000 000014      MOV      #0,RMER1(RO)      ;LOAD RMER1
8206 026030 012760 000000 000042      MOV      #0,RMER2(RO)      ;LOAD RMER2
8207 026030 012760 000000 000042      ;TRANSFER THE FUNCTION CODE AND GO BIT TO RMCS1, VERIFY GO IS SET
8208 026036 010203 000001 000001      MOV      R2,R3              ;SETUP FUNCTION CODE
8209 026040 052703 000001 000001      BIS      #GO,R3
8210 026044 010360 000000 000000      MOV      R3,RMCS1(RO)      ;LOAD RMCS1
8211 026050 016037 000000 001142      MOV      RMCS1(RO),SBDDAT    ;STORE RMCS1 AT SBDDAT
8212 026056 032737 000001 001142      BIT      #GO,SBDDAT
8213 026064 001007 000000 001142      BNE      20$                ;BRANCH IF GO IS SET
8214 026066 042737 177700 001142      ;REPORT THE ERROR-CANT SET GO WITH THIS FUNCTION CODE
8215 026074 010337 001140 001142      BIC      #CFNCMSK,SBDDAT
8216 026100 104146 000405 000405      MOV      R3,SGDDAT          ;SAVE FUNCTION CODE
8217 026102 000405 000405 000405      ERROR   146                 ;CANT SET GO
8218 026104 000405 000405 000405      BR      30$
8219 026104 062702 000002 000002      20$:
8220 026110 022702 000076 000076      ;ADVANCE R2 TO THE NEXT FUNCTION CODE
8221 026114 10332 10332 10332      ADD      #2,R2
8222 026116 10332 10332 10332      CMP      #1LF76,R2
8223 026116 10332 10332 10332      BHIS    10$
8224 026116 10332 10332 10332
8225 026116 10332 10332 10332
8226 026116 10332 10332 10332
8227 026116 10332 10332 10332
8228 026116 10332 10332 10332
8229 026116 10332 10332 10332
8230 026116 10332 10332 10332
8231 026116 10332 10332 10332
8232 026116 10332 10332 10332
8233 026116 10332 10332 10332
8234 026116 000004 000004 001226      30$:
8235 026120 012737 000044 001226      ;*****
8236 026126 000240 000240 000240      ;TEST 44 BRANCH MULTIPLEXOR TEST
8237 026130 012737 000024 001120      ;*****
8238 026136 112737 000001 001131      †ST44. SCOPE
8239 026144 012737 026160 001122      MOV      #44,$TESTN        ;,SET TEST NUMBER IN APT MAIL BOX
8240 026152 012737 026160 001124      NOP
8241 026160 012706 001100 001100      MOV      #20,$ICNT          ;20 ITERATIONS
8242 026164 013700 001276 001276      MOV      #1,$ERMAX          ;ONE ERROR ALLOWED
8243 026170 013701 001456 001456      MOV      #T44,$LPADR        ;LOAD LOOP ON TEST ADDRESS
8244 026174 010037 001136 001136      MOV      #T44,$LPERR        ;LOAD LOOP ON ERROR ADDRESS
8245 026200 062737 000040 001136      T44:
8246 026206 012702 026440 026440      MOV      #STACK,SP          ;LOAD THE STACK POINTER
8247 026212 012760 000040 000010      MOV      $BASE,RO           ;RO = UNIBUS ADDRESS OF UUT
8248 026220 111160 000010 000010      MOV      TSTQUE,R1          ;R1 = POINTER TO DEVICE
8249 026224 012760 000001 000024      MOV      RO,$BADDR          ;COPY REGISTER ADDRESS
8250 026232 012760 041001 000024      ADD      #RMMR2,$BADDR
8251 026240 012760 000000 000014      MOV      #100$,R2           ;INITIALIZE TABLE POINTER
8252 026246 012760 000000 000042      ;CLEAR THE MASSBUS AND SET DEBUG CLOCK ENABLE
8253 026212 012760 000040 000010      10$:
8254 026220 111160 000010 000010      MOV      #CLR,RMCS2(RO)     ;CLEAR THE MASSBUS
8255 026224 012760 000001 000024      MOV      (R1),RMCS2(RO)     ;SELECT UNIT
8256 026232 012760 041001 000024      MOV      #DMD,RMMR1(RO)     ;LOAD RMMR1
8257 026240 012760 000000 000014      MOV      #DMD!DBEN!MUR,RMMR1(RO) ;LOAD RMMR1
8258 026246 012760 000000 000042      MOV      #0,RMER1(RO)       ;LOAD RMER1
8259 026246 012760 000000 000042      MOV      #0,RMER2(RO)       ;LOAD RMER2

```

; THE TEST BIT SHOULD BE ONE BECAUSE THE ADDRESS IS ALL ONES WHEN
; THE COMMAND SEQUENCER IS INITIALIZED.

8261									
8262									
8263									
8264	026254	016037	000040	001142	MOV	RMMR2(RO), \$BDDAT		; STORE RMMR2 AT \$BDDAT	
8265	026262	032737	010000	001142	BIT	#TST, \$BDDAT			
8266	026270	001010			BNE	15\$; BRANCH IF TEST BIT IS ON	
8267	026272	042737	167777	001142	BIC	#+CTST, \$BDDAT		; SETUP FOR ERROR TYPE	
8268	026300	012737	010000	001140	MOV	#TST, \$GDDAT			
8269	026306	104147			ERROR	147		; TEST BIT NOT SET	
8270	026310	000452			BR	40\$; SKIP REST OF TEST	

15\$:
; GET THE FUNCTION CODE FROM THE TABLE AND TRANSFER IT TO THE DEVICE,
; THEN STEP THE COMMAND SEQUENCER ACCORDING TO THE TABLE.

8271	026312								
8272									
8273									
8274									
8275	026312	111203			MOVB	(R2), R3			
8276	026314	052703	000001		BIS	#GO, R3			
8277	026320	042703	177700		BIC	#+CFNCMSK, R3		; R3=FUNCTION CODE, GO BIT	
8278									
8279	026324	010360	000000		MOV	R3, RMCS1(RO)		; LOAD RMCS1	
8280	026330	010337	001174		MOV	R3, \$TMP0		; SAVE R3 FOR ERROR MSG	
8281									
8282	026334	116203	000001		MOVB	1(R2), R3		; GET CLOCK COUNT IN R3	
8283	026340	042703	177400		BIC	#+C377, R3			

20\$:

8284	026344								
8285									
8286	026344	012760	141001	000024	MOV	#DMD!DBEN!MUR!DBCK, RMMR1(RO)		; LOAD RMMR1	
8287									
8288	026352	012760	041001	000024	MOV	#DMD!DBEN!MUR, RMMR1(RO)		; LOAD RMMR1	
8289	026360	005303			DEC	R3		; DECREMENT CLOCK COUNT	
8290	026362	001370			BNE	20\$; ISSUE CLOCKS TILL ZERO	

; GET THE TEST BIT AND COMPARE IT WITH THE TABLE ENTRY

8291									
8292									
8293									
8294	026364	016037	000040	001142	MOV	RMMR2(RO), \$BDDAT		; STORE RMMR2 AT \$BDDAT	
8295	026372	042737	167777	001142	BIC	#+CTST, \$BDDAT			
8296	026400	016237	000002	001140	MOV	2(R2), \$GDDAT			
8297	026406	023737	001140	001142	CMP	\$GDDAT, \$BDDAT			
8298	026414	001402			BEG	30\$; BRANCH IF TEST BIT OK	
8299	026416	104150			ERROR	150		; TEST BIT IS INCORRECT	
8300	026420	000406			BR	40\$; SKIP REST OF TEST	

30\$:
; MOVE THE TABLE POINTER AND CONTINUE IF NEXT ENTRY POSITIVE

8301	026422								
8302									
8303	026422	062702	000004		ADD	#4, R2			
8304	026426	105762	000001		TSTB	1(R2)			
8305	026432	100401			BMI	40\$; BRANCH IF DONE TEST	
8306	026434	000666			BR	10\$; REPEAT TEST	
8307	026436	000436			BR	200\$; JUMP OVER TABLE	

40\$:
; JUMP OVER TABLE

8308									
8309	026440								
8310									
8311									
8312									
8313	026440	000			.BYTE	NOP		; MUX ADDRESS=DATA COMMAND	
8314	026441	001			.BYTE	1			
8315	026442	000000			.WORD	0		; TEST BIT=0	
8316									

; TABLE OF FUNCTION CODES, CLOCK COUNTS, AND TEST BITS

8317	026444	000	.BYTE	NOP	; MUX ADDRESS=UNIT READY
8318	026445	002	.BYTE	2	
8319	026446	000000	.WORD	0	; TEST BIT=0
8320					
8321	026450	010	.BYTE	DRVCLR	; MUX ADDRESS=F4
8322	026451	001	.BYTE	1	
8323	026452	010000	.WORD	TST	; TEST BIT=1
8324					
8325	026454	050	.BYTE	WCD	; MUX ADDRESS=F4
8326	026455	001	.BYTE	1	
8327	026456	000000	.WORD	0	; TEST BIT=0
8328					
8329	026460	012	.BYTE	RELEASE	; MUX ADDRESS=F4
8330	026461	001	.BYTE	1	
8331	026462	010000	.WORD	TST	; TEST BIT=1
8332					
8333	026464	052	.BYTE	WCH	; MUX ADDRESS=F4
8334	026465	001	.BYTE	1	
8335	026466	000000	.WORD	0	; TEST BIT=0
8336					
8337	026470	020	.BYTE	RIP	; MUX ADDRESS=F4
8338	026471	001	.BYTE	1	
8339	026472	010000	.WORD	TST	; TEST BIT=1
8340					
8341	026474	060	.BYTE	WD	; MUX ADDRESS=F4
8342	026475	001	.BYTE	1	
8343	026476	000000	.WORD	0	; TEST BIT=0
8344					
8345	026500	022	.BYTE	PAKACK	; MUX ADDRESS=F4
8346	026501	001	.BYTE	1	
8347	026502	010000	.WORD	TST	; TEST BIT=1
8348					
8349	026504	062	.BYTE	WH	; MUX ADDRESS=F4
8350	026505	001	.BYTE	1	
8351	026506	000000	.WORD	0	; TEST BIT=0
8352					
8353	026510	030	.BYTE	SEARCH	; MUX ADDRESS=F4
8354	026511	001	.BYTE	1	
8355	026512	010000	.WORD	TST	; TEST BIT=1
8356					
8357	026514	070	.BYTE	RD	; MUX ADDRESS=F4
8358	026515	001	.BYTE	1	
8359	026516	000000	.WORD	0	; TEST BIT=0
8360					
8361	026520	032	.BYTE	ILF32	; MUX ADDRESS=F4
8362	026521	001	.BYTE	1	
8363	026522	010000	.WORD	TST	; TEST BIT=1
8364					
8365	026524	072	.BYTE	RH	; MUX ADDRESS=F4
8366	026525	001	.BYTE	1	
8367	026526	000000	.WORD	0	; TEST BIT=0
8368					
8369	026530	000	.BYTE		; END OF TABLE
8370	026531	377	.BYTE	#-1	
8371	026532	000000	.WORD		
8372	026534				; END OF TEST

200\$:

```

8373
8374
8375
8376
8377
8378 026534 000004
8379 026536 012737 000045 001226
8380
8381 026544 000240
8382 026546 012737 000024 001120
8383 026554 112737 000001 001131
8384 026562 012737 026576 001122
8385 026570 012737 026576 001124
8386 026576
8387 026576 012706 001100
8388 026602 013700 001276
8389 026606 013701 001456
8390 026612 012702 027234
8391
8392
8393 026616
8394 026616 012760 000040 000010
8395 026624 111160 000010
8396
8397 026630 012760 000001 000024
8398
8399 026636 012760 041001 000024
8400
8401 026644 012760 000000 000014
8402
8403 026652 012760 000000 000042
8404
8405 026660 111203
8406 026662 042703 177701
8407 026666 052703 000001
8408
8409 026672 010360 000000
8410
8411 026676 016037 000000 001142
8412 026704 032737 000001 001142
8413 026712 001011
8414 026714 042737 177700 001142
8415 026722 010337 001140
8416 026726 010037 001136
8417 026732 104151
8418 026734 000536
8419
8420 026736
8421
8422 026736 005037 001140
8423 026742 032737 000001 001142
8424 026750 001003
8425 026752 012737 000200 001140
8426 026760
8427
8428 026760 016037 000012 001142

```

```

;*****
;#TEST 45      SET/RESET GO TEST
;*****
T44: SCOPE
MOV #45,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
NOP
MOV #20,$ICNT      ;20 ITERATIONS
MOVB #1,$ERMAX     ;ONE ERROR ALLOWED
MOV #T45,$LPAOR   ;LOAD LOOP ON TEST ADDRESS
MOV #T45,$LPERR   ;LOAD LOOP ON ERROR ADDRESS
T45:
MOV #STACK,SP     ;LOAD THE STACK POINTER
MOV $BASE,R0      ;R0 = UNIBUS ADDRESS OF UUT
MOV TSTQUE,R1     ;R1 = POINTER TO DEVICE
MOV #200,$R2      ;INITIALIZE FUNCTION CODE POINTER
;CLEAR, THEN SET DIAGNOSTIC MODE, CLEAR COMPOSITE ERROR, SET MEDIUM
;ON LINE AND ENABLE DEBUG CLOCK
10$:
MOV #CLR,RMCS2(R0) ;CLEAR THE MASSBUS
MOVB (R1),RMCS2(R0) ;SELECT UNIT
MOV #DMD,RMMR1(R0) ;LOAD RMMR1
MOV #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
MOV #0,RMER1(R0)   ;LOAD RMER1
MOV #0,RMER2(R0)   ;LOAD RMER2
;TRANSFER THE FUNCTION CODE AND GO BIT TO RMCS1 AND VERIFY GO IS SET
MOVB (R2),R3      ;GET FUNCTION CODE
BIC #C1,F76,R3   ;CLEAR UNUSED BITS
BIS #GO,R3       ;SET GO
MOV R3,RMCS1(R0) ;LOAD RMCS1
MOV RMCS1(R0),$BDDAT ;STORE RMCS1 AT $BDDAT
BIT #GO,$BDDAT
BNE 20$          ;BRANCH IF GO IS SET
BIC #CFNCMSK,$BDDAT
MOV R3,$GDDAT   ;SAVE EXPECTED RESULT
MOV R0,$BDADR   ;COPY REGISTER ADDRESS
ERROR 151      ;CANT SET GO
BR 100$
20$:
;GET READY STATUS AND VERIFY THAT IT IS THE COMPLEMENT OF GO
CLR $GDDAT      ;EXPECT DRY TO BE OFF
BIT #GO,$BDDAT ;WAS GO SET??
BNE 30$         ;YES!!
MOV #DRY,$GDDAT ;GO WAS NOT SET, DRY SHOULD BE
30$:
MOV RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT

```

```

8429 026766 042737 177577 001142      BIC      #1CDRY,$BDDAT
8430 026774 023737 001140 001142      CMP      $GDDAT,$BDDAT
8431 027002 001406                BEQ      40$      ;BRANCH IF DRY IS OK
8432 027004 010037 001136                MOV      R0,$B0ADR ;COPY REGISTER ADDRESS
8433 027010 062737 000012 001136      ADD      #RMS,$B0ADR
8434 027016 104152                ERROR    152      ;DRY NOT COMPLEMENT OF GO
8435 027020                40$:
8436                ;STEP THE DEBUG CLOCK AND VERIFY THAT GO REMAINS SET
8437 027020 116204 000001      MOV      1(R2),R4 ;GET NUMBER OF CLOCK CYCLES
8438 027024 042704 177400      BIC      #1C377,R4
8439 027030                50$:
8440
8441 027030 012760 141001 000024      MOV      #DMD!MUR!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1
8442
8443 027036 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
8444
8445 027044 016037 000000 001142      MOV      RMCS1(RO),$BDDAT ;STORE RMCS1 AT $BDDAT
8446 027052 042737 177700 001142      BIC      #1CFNCMSK,$BDDAT ;CLEAR UNUSED BITS
8447 027060 010037 001136                MOV      R0,$B0ADR ;SETUP REGISTER ADDRESS
8448 027064 010337 001140                MOV      R3,$GDDAT ;SAVE EXPECTED RESULT
8449                ;DECREMENT CLOCK COUNT AND EXIT LOOP IF ZERO
8450 027070 005304                DEC      R4
8451 027072 001406                BEQ      60$
8452 027074 032737 000001 001142      BIT      #GO,$BDDAT ;IS GO STILL SET??
8453 027102 001352                BNE     50$      ;YES!!
8454 027104 104153                ERROR    153      ;GO RESET EARLY
8455 027106 000451                BR      100$     ;OUT OF SYNC=SKIP TO NEXT
8456 027110                60$:
8457                ;GO SHOULD NOW BE RESET AND DRY SHOULD BE SET
8458 027110 032737 000001 001142      BIT      #GO,$BDDAT ;IS GO RESET??
8459 027116 001405                BEQ      70$      ;YES!!
8460 027120 042737 000001 001140      BIC      #GO,$GDDAT ;SETUP EXPECTED RESULT
8461 027126 104154                ERROR    154      ;GO DID NOT RESET
8462 027130 000440                BR      100$
8463 027132                70$:
8464 027132 012737 000200 001140      MOV      #DRY,$GDDAT ;EXPECT DRIVE READY TO BE SET
8465 027140 032737 000001 001142      BIT      #GO,$BDDAT ;DID GO RESET??
8466 027146 001402                BEQ      80$      ;YES!!
8467 027150 005037 001140                CLR      $GDDAT   ;GO IS SET-
8468 027154                80$:
8469
8470 027154 016037 000012 001142      MOV      RMS(RO),$BDDAT ;STORE RMS AT $BDDAT
8471 027162 042737 177577 001142      BIC      #1CDRY,$BDDAT
8472 027170 010037 001136                MOV      R0,$B0ADR ;COPY REGISTER ADDRESS
8473 027174 062737 000012 001136      ADD      #RMS,$B0ADR
8474 027202 023737 001140 001142      CMP      $GDDAT,$BDDAT ;IS DRIVE READY OK??
8475 027210 001401                BEQ      90$      ;YES!!
8476 027212 104152                ERROR    152      ;DRY NOT COMPLEMENT OF GO
8477 027214                90$:
8478                ;ADVANCE TO THE NEXT FUNCTION CODE TO BE TESTED-EXIT IF DONE
8479 027214 062702 000002      ADD      #2,R2    ;MOVE TABLE POINTER
8480 027220 105762 000001      TSTB    1(R2)    ;END OF TABLE??
8481 027224 100402                BMI     100$     ;YES!!
8482 027226 000137 026616      JMP     10$      ;TEST THIS FUNCTION CODE
8483 027232 000423                100$:
8484                BR      300$ ;GO TO NEXT TEST

```

Address	Hex	Dec	Function	Description
8485	027234			
8486				
8487				
8488	027234	002	.BYTE ILF02	; ILLEGAL FUNCTION CODE #2
8489	027235	001	.BYTE 1	
8490				
8491	027236	004	.BYTE SEEK	; SEEK COMMAND
8492	027237	001	.BYTE 1	
8493				
8494	027240	006	.BYTE RECAL	; RECALIBRATE COMMAND
8495	027241	001	.BYTE 1	
8496				
8497	027242	014	.BYTE OFFSET	; OFFSET COMMAND
8498	027243	001	.BYTE 1	
8499				
8500	027244	016	.BYTE RTC	; RETURN TO CENTER LINE COMMAND
8501	027245	001	.BYTE 1	
8502				
8503	027246	024	.BYTE ILF24	; ILLEGAL FUNCTION CODE #24
8504	027247	001	.BYTE 1	
8505				
8506	027250	026	.BYTE ILF26	; ILLEGAL FUNCTION CODE #26
8507	027251	001	.BYTE 1	
8508				
8509	027252	034	.BYTE ILF34	; ILLEGAL FUNCTION CODE #34
8510	027253	001	.BYTE 1	
8511				
8512	027254	036	.BYTE ILF36	; ILLEGAL FUNCTION CODE #36
8513	027255	001	.BYTE 1	
8514				
8515	027256	042	.BYTE ILF42	; ILLEGAL FUNCTION CODE #42
8516	027257	001	.BYTE 1	
8517				
8518	027260	044	.BYTE ILF44	; ILLEGAL FUNCTION CODE #44
8519	027261	001	.BYTE 1	
8520				
8521	027262	046	.BYTE ILF46	; ILLEGAL FUNCTION CODE #46
8522	027263	001	.BYTE 1	
8523				
8524	027264	054	.BYTE ILF54	; ILLEGAL FUNCTION CODE #54
8525	027265	001	.BYTE 1	
8526				
8527	027266	056	.BYTE ILF56	; ILLEGAL FUNCTION CODE #56
8528	027267	001	.BYTE 1	
8529				
8530	027270	064	.BYTE ILF64	; ILLEGAL FUNCTION CODE #64
8531	027271	001	.BYTE 1	
8532				
8533	027272	066	.BYTE ILF66	; ILLEGAL FUNCTION CODE #66
8534	027273	001	.BYTE 1	
8535				
8536	027274	074	.BYTE ILF74	; ILLEGAL FUNCTION CODE #74
8537	027275	001	.BYTE 1	
8538				
8539	027276	076	.BYTE ILF76	; ILLEGAL FUNCTION CODE #76
8540	027277	001	.BYTE 1	

2005:
 ;*****
 ;TABLE OF FUNCTION CODES AND CLOCK COUNTS USED DURING TEST
 ;

```

8541
8542 027300 000 .BYTE ;END OF TABLE
8543 027301 377 .BYTE #-1
8544
8545 027302 300$: ;END OF TEST
8546
8547 ;*****
8548 ;*TEST 46 END 1 RESET GO TEST
8549
8550 ;*****
8551 027302 000004 †ST46: SCOPE
8552 027304 012737 000046 001226 MOV #46,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
8553
8554 027312 000240 NOP
8555 027314 012737 000024 001120 MOV #20,$ICNT ;20 ITERATIONS
8556 027322 112737 000001 001131 MOV#B #1,$ERMAX ;ONE ERROR ALLOWED
8557 027330 012737 027344 001122 MOV #T46,$LPADR ;LOAD LOOP ON TEST ADDRESS
8558 027336 012737 027344 001124 MOV #T46,$LPERR ;LOAD LOOP ON ERROR ADDRESS
8559 027344 T46:
8560 027344 012706 001100 MOV #STACK,SP ;LOAD THE STACK POINTER
8561 027350 013700 001276 MOV $BASE,R0 ;R0 = UNIBUS ADDRESS OF UUT
8562 027354 013701 001456 MOV TSTQUE,R1 ;R1 = POINTER TO DEVICE
8563 027360 012702 027624 MOV #100$,R2 ;INITIALIZE TABLE POINTER
8564 027364 010037 001136 MOV R0,$BDDADR ;COPY RMCS1 ADDRESS
8565
8566 027370 10$:
8567 ;CLEAR MASSBUS, THEN SET MEDIUM ON LINE AND ENABLE DEBUG CLOCK
8568 027370 012760 000040 000010 MOV #CLR, RMCS2(R0) ;CLEAR THE MASSBUS
8569 027376 111160 000010 MOV#B (R1),RMCS2(R0) ;SELECT UNIT
8570
8571 027402 012760 000001 000024 MOV #DMD,RMMR1(R0) ;LOAD RMMR1
8572
8573 027410 012760 041001 000024 MOV #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
8574
8575 027416 012760 000000 000014 MOV #0,RMER1(R0) ;LOAD RMER1
8576
8577 027424 012760 000000 000042 MOV #0,RMER2(R0) ;LOAD RMER2
8578 ;TRANSFER THE FUNCTION CODE AND GO BIT TO RMCS1, VERIFY GO IS SET
8579 027432 111203 MOV#B (R2),R3 ;GET FUNCTION CODE FROM
8580 027434 042703 177701 BIC #1C1F76,R3 ;TABLE AND SET GO
8581 027440 052703 000001 BIS #GO,R3
8582 027444 010337 001140 MOV R3,$GDDAT ;SAVE FUNCTION CODE FOR MSG
8583
8584 027450 010360 000000 MOV R3,RMCS1(R0) ;LOAD RMCS1
8585
8586 027454 016037 000000 001142 MOV RMCS1(R0),$BDDAT ;STORE RMCS1 AT $BDDAT
8587 027462 032737 000001 001142 BIT #GO,$BDDAT
8588 027470 001005 BNE 20$ ;BRANCH IF GO IS SET
8589 027472 042737 177700 001142 BIC #1CFNCMSK,$BDDAT
8590 027500 104151 ERROR 151 ;GO DID NOT SET
8591 027502 000447 BR 60$ ;OUT OF SYNC-SKIP
8592 027504
8593 20$:
8594 027504 116204 000001 ;GET THE NUMBER OF CLOCK CYCLES FROM THE TABLE, SAVE EXPECTED STATUS
8595 027510 042704 177400 MOV#B 1(R2),R4 ;R4=CLOCK COUNT
8596 027514 BIC #1C377,R4
30$:

```


;STEP THE DEBUG CLOCK AND VERIFY GO STATUS ON UNTIL CLOCK COUNT EXPIRES.

```

8597
8598
8599 027514 012760 141001 000024      MOV      #DMD!MUR!DBEN!DBCK,RMMR1(RC      ;LOAD RMMR1
8600
8601 027522 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RD) ;LOAD RMMR1
8602
8603 027530 016037 000000 001142      MOV      RMCS1(RD), $BDDAT          ;STORE RMCS1 AT $BDDAT
8604 027536 042737 177700 001142      BIC      #1CFNCMSK,$BDDAT
8605 027544 005304          DEC      R4
8606 027546 001406          BEQ      40$                       ;BRANCH IF GO SHOULD BE OFF
8607 027550 032737 000001 001142      BIT      #GO,$BDDAT
8608 027556 001356          BNE      30$                       ;CONTINUE IF GO IS ON
8609 027560 104153          ERROR   153                       ;GO RESET EARLY
8610 027562 000417          BR      60$                       ;OUT OF SYNC-SKIP
8611 027564
8612
8613 027564 032737 000001 001142      40$:
;VERIFY THAT GO RESET AT END1
8614 027572 001405          BIT      #GO,$BDDAT              ;DID GO RESET??
8615 027574 042737 000001 001140          BEQ      50$                       ;YES!!
8616 027602 104154          BIC      #G,$BDDAT
8617 027604 000406          ERROR   154                       ;GO NOT RESET AT END1
8618 027606          BR      60$
8619
8620 027606 062702 000002          50$:
;GET THE NEXT FUNCTION CODE FROM THE TABLE
8621 027612 105762 000001          ADD      #2,R2
8622 027616 100401          TSTB    1(R2)
8623 027620 000663          BMI     60$                       ;BRANCH IF END OF TABLE
8624 027622 000404          BR      10$                       ;TEST THIS FUNCTION CODE
8625
8626 027624          60$:
;JUMP OVER TABLE
8627
8628
8629 027624 012          100$:
;TABLE OF FUNCTION CODES AND CLOCK COUNTS USED DURING TEST
8630 027625 002          .BYTE   RELEASE                   ;RELEASE COMMAND
8631
8632 027626 030          .BYTE   SEARCH                    ;SEARCH COMMAND
8633 027627 002          .BYTE   2
8634
8635 027630 032          .BYTE   ILF32                     ;ILLEGAL FUNCTION #32
8636 027631 002          .BYTE   2
8637
8638 027632 000          .BYTE   ;END OF TABLE
8639 027633 377          .BYTE   #-1
8640 027634
8641
8642
8643
8644
8645
8646 027634 000004          200$:
;*****
;TEST 47 SET PULSE TEST
8647 027636 012737 000047 001226      ;*****
;ST47: SCOPE
MOV      #47,$TESTN                ;;SET *ST NUMBER IN APT MAIL BOX
8648
8649 027644 000240          NOP
8650 027646 012737 000024 001120      MOV      #20,$ICNT                ;20 ITERATIONS
8651 027654 112737 000001 001131      MOV      #1,$ERMAX                 ;ONE ERROR ALLOWED
8652 027662 012737 027676 001122      MOV      #T47,$LPADR              ;LOAD LOOP ON TEST ADDRESS

```

```

8653 027670 012737 027676 001124      MOV      #T47,$LPERR ;LOAD LOOP ON ERROR ADDRESS
8654 027676                                T47:
8655 027676 012706 001100      MOV      #STACK,SP ;LOAD THE STACK POINTER
8656 027702 013700 001276      MOV      $BASE,R0 ;R0 = UNIBUS ADDRESS OF UUT
8657 027706 013701 001456      MOV      TSTQUE,R1 ;R1 = POINTER TO DEVICE
8658 027712 010037 001136      MOV      R0,$BDADR ;COPY REG ADDRESS FOR MSG
8659 027716 062737 000024 001136      ADD     #RMMR1,$BDADR
8660 027724 012702 030170      MOV     #100$,R2 ;INITIALIZE TABLE POINTER
8661 027730                                10$:
8662                                ;CLEAR THE MASS BUS, ENABLE DEBUG CLOCK, AND RESET ERROR REGISTERS
8663 027730 012760 000040 000010      MOV     #CLR,RMCS2(R0) ;CLEAR THE MASSBUS
8664 027736 111160 000010      MOV     (R1),RMCS2(R0) ;SELECT UNIT
8665
8666 027742 012760 000001 000024      MOV     #DMD,RMMR1(R0) ;LOAD RMMR1
8667
8668 027750 012760 041001 000024      MOV     #DMD!DBEN!MUR,RMMR1(R0) ;LOAD RMMR1
8669
8670 027756 012760 000000 000014      MOV     #0,RMER1(R0) ;LOAD RMER1
8671
8672 027764 012760 000000 000042      MOV     #0,RMER2(R0) ;LOAD RMER2
8673                                ;VERIFY THAT CONTINUE, "CONT" IS RESET AFTER CLEAR
8674
8675 027772 016037 000024 001142      MOV     RMMR1(R0),$BDAT ;STORE RMMR1 AT $BDAT
8676 030000 042737 177677 001142      BIC     #1CCONT,$BDAT
8677 030006 001404                                BEQ     20$ ;BRANCH IF CONT WAS CLEARED
8678 030010 005037 001140      CLR     $GDDAT ;FOR ERROR MSG
8679 030014 104155                                ERROR  15$ ;CANT CLEAR CONTINUE
8680 030016 000463                                BR      70$
8681 030020                                20$:
8682                                ;GET THE FUNCTION CODE FROM THE TABLE AND TRANSFER IT TO RMCS1
8683 030020 111203                                MOV     (R2),R3
8684 030022 052703 000001      BIS     #GO,R3
8685 030026 042703 177700      BIC     #1CFNCMSK,R3 ;R3=FUNCTION CODE AND GO
8686
8687 030032 010360 000000      MOV     R3,RMCS1(R0) ;LOAD RMCS1
8688 030036 010337 001174      MOV     R3,$TMP0 ;SAVE FUNCTION CODE FOR MSG
8689                                ;GET THE CLOCK COUNT FROM THE TABLE
8690 030042 116203 000001      MOV     1(R2),R3
8691 030046 042703 177400      BIC     #1C377,R3
8692                                ;GET THE BIT STREAM FOR CONTINUE FROM THE TABLE
8693 030052 016204 000002      MOV     2(R2),R4
8694 030056                                30$:
8695                                ;STEP THE COMMAND SEQUENCER AND VERIFY CONTINUE STATUS
8696
8697 030056 012760 141001 000024      MOV     #DMD!DBEN!MUR!DBCK,RMMR1(R0) ;LOAD RMMR1
8698
8699 030064 012760 041001 000024      MOV     #DMD!DBEN!MUR,RMMR1(R0) ;LOAD RMMR1
8700
8701 030072 016037 000024 001142      MOV     RMMR1(R0),$BDAT ;STORE RMMR1 AT $BDAT
8702 030100 042737 177677 001142      BIC     #1CCONT,$BDAT
8703 030106 005037 001140      CLR     $GDDAT ;GENERATE EXPECTED CONTINUE
8704 030112 032704 000001      BIT     #BIT0,R4
8705 030116 001403                                BEQ     40$
8706 030120 012737 000100 001140      MOV     #CONT,$GDDAT
8707 030126 023737 001140 001142 40$:      CMP     $GDDAT,$BDAT
8708 030134 001402                                BEQ     50$ ;BRANCH IF CONTINUE IS OK

```

8709	030136	104156		ERROR	156		;CONTINUE IS INCORRECT
8710	030140	000412		BR	70\$;SKIP
8711	030142			50\$:			
8712				;DECREMENT CLOCK COUNT AND SHIFT BIT STREAM			
8713	030142	005303		DEC	R3		
8714	030144	001402		BEG	60\$;BRANCH IF CLOCK COUNT EXPIRED
8715	030144	006204		ASR	R4		;SHIFT TO NEXT CONTINUE BIT
8716	030150	000742		BR	30\$;TEST NEXT CLOCK CYCLE
8717	030152			60\$:			
8718				;ADVANCE TABLE POINTER-EXIT IF DONE			
8719	030152	062702	000004	ACC	#4,R2		
8720	030156	105762	000001	TSTB	1(R2)		
8721	030162	100401		BMI	70\$;EXIT IF CLOCK COUNT NEGATIVE
8722	030164	000661		BR	10\$;CONTINUE TEST
8723	030166	000442		70\$:	BR	200\$;JUMP OVER TABLE
8724				100\$:			
8725	030170			;TABLE OF FUNCTION CODES, CLOCK COUNTS AND CONTINUE BITS FOR TEST			
8726							
8727							
8728	030170	000		.BYTE	NOP		;NOP COMMAND
8729	030171	004		.BYTE	4		;4 CLOCKS
8730	030172	000000		.WORD	↑80000		;CONTINUE=0000
8731							
8732	030174	002		.BYTE	ILF02		;ILLEGAL FUNCTION 2
8733	030175	002		.BYTE	2		
8734	030176	000000		.WORD	↑800		
8735							
8736	030200	004		.BYTE	SEEK		;SEEK COMMAND
8737	030201	002		.BYTE	2		
8738	030202	000000		.WORD	↑800		
8739							
8740	030204	006		.BYTE	RECAL		;RECALIBRATE COMMAND
8741	030205	002		.BYTE	2		
8742	030206	000000		.WORD	↑800		
8743							
8744	030210	010		.BYTE	DRVCLR		;DRIVE CLEAR COMMAND
8745	030211	002		.BYTE	2		
8746	030212	000001		.WORD	↑801		
8747							
8748	030214	012		.BYTE	RELEASE		;RELEASE COMMAND
8749	030215	003		.BYTE	3		
8750	030216	000000		.WORD	↑8000		
8751							
8752	030220	014		.BYTE	OFFSET		;OFFSET COMMAND
8753	030221	002		.BYTE	2		
8754	030222	000000		.WORD	↑800		
8755							
8756	030224	016		.BYTE	RTC		;RETURN TO CENTER COMMAND
8757	030225	002		.BYTE	2		
8758	030226	000000		.WORD	↑800		
8759							
8760	030230	020		.BYTE	RIP		;READ IN PRESET COMMAND
8761	030231	004		.BYTE	4		
8762	030232	000016		.WORD	↑21110		
8763							
8764	030234	022		.BYTE	PAKACK		;PACK ACKNOWLEDGE

8765	030235	004				.BYTE	4		
8766	030236	000016				.WORD	↑B1110		
8767									
8768	030240	024				.BYTE	ILF24		; ILLEGAL FUNCTION 24
8769	030241	002				.BYTE	2		
8770	030242	000000				.WORD	↑B00		
8771									
8772	030244	026				.BYTE	ILF26		; ILLEGAL FUNCTION 26
8773	030245	002				.BYTE	2		
8774	030246	000000				.WORD	↑B00		
8775									
8776	030250	030				.BYTE	SEARCH		; SEARCH COMMAND
8777	030251	003				.BYTE	3		
8778	030252	000000				.WORD	↑B000		
8779									
8780	030254	032				.BYTE	ILF32		; ILLEGAL FUNCTION 32
8781	030255	003				.BYTE	3		
8782	030256	000000				.WORD	↑B000		
8783									
8784	030260	034				.BYTE	ILF34		; ILLEGAL FUNCTION 34
8785	030261	002				.BYTE	2		
8786	030262	000000				.WORD	↑B00		
8787									
8788	030264	036				.BYTE	ILF36		; ILLEGAL FUNCTION 36
8789	030265	002				.BYTE	2		
8790	030266	000000				.WORD	↑B00		
8791									
8792	030270	000				.BYTE			; END OF TABLE
8793	030271	377				.BYTE	-1		
8794	030272	000000				.WORD			
8795									
8796	030274				200\$:				; END OF TEST
8797									
8798									
8799									
8800									
8801									
8802	030274	000004				↑ST50:	SCOPE		
8803	030276	012737	000050	001226		MOV	#50,\$TESTN		; ; SET TEST NUMBER IN APT MAIL BOX
8804									
8805	030 4	000240				NOP			
8806	030306	012737	000024	001120		MOV	#20,\$ICNT		; 20 ITERATIONS
8807	030314	112737	000001	001131		MOV	#1,\$ERMAX		; ONE ERROR ALLOWED
8808	030322	012737	030336	001122		MOV	#T50,\$LPADR		; LOAD LOOP ON TEST ADDRESS
8809	030330	012737	030336	001124		MOV	#T50,\$LPERR		; LOAD LOOP ON ERROR ADDRESS
8810	030336				T50:				
8811	030336	012706	001100			MOV	#STACK,SP		; LOAD THE STACK POINTER
8812	030342	013700	001276			MOV	\$BASE,R0		; R0 = UNIBUS ADDRESS OF JUT
8813	030346	013701	001456			MOV	TSTQUE,R1		; R1 = POINTER TO DEVICE
8814	030352	010037	001136			MOV	R0,\$BDAOR		; SETUP REG ADDRESS
8815	030356	062737	000042	001136		ADD	#RMR2,\$BDAOR		
8816	030364	005002				CLR	R2		; R2=FUNCTION CODE
8817	030366								
8818					10\$:				
8819	030366	012760	000040	000010		MOV	#CLR,AMCS2(R0)		; INITIALIZE AND VERIFY THAT IVC STATUS IS ZERO.
8820	030374	111160	000010			MOV	(R1),AMCS2(R0)		; CLEAR THE MASSBUS
						MOV	(R1),AMCS2(R0)		; SELECT UNIT

```

8821
8822 030400 012760 000001 000024      MOV      #DMD,RMMR1(RO) ;LOAD RMMR1
8823
8824 030406 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
8825
8826 030414 012760 000000 000014      MOV      #0,RMER1(RO) ;LOAD RMER1
8827
8828 030422 012760 000000 000042      MOV      #0,RMER2(RO) ;LOAD RMER2
8829
8830 030430 016037 000042 001142      MOV      RMER2(RO), $BDDAT ;STORE RMER2 AT $BDDAT
8831 030436 042737 167777 001142      BIC      #1CIVC,$BDDAT
8832 030444 001404                BEQ      20$ ;BRANCH IF IVC IS ZERO
8833 030446 005037 001140                CLR      $GDDAT
8834 030452 104157                ERROR   157 ;CANT CLEAR IVC
8835 030454 000444                BR      40$ ;SKIP REST OF TEST
8836 030456
20$:
8837 ;LOAD THE FUNCTION CODE WITH GO BIT, STEP THE COMMAND SEQUENCER OFF
8838 ;ADDRESS 0 AND VERIFY IVC STATUS.
8839 030456 010203                MOV      R2,R3 ;SETUP FUNCTION CODE
8840 030460 052703 000001                BIS      #GO,R3
8841
8842 030464 010360 000000                MOV      R3,RMCS1(RO) ;LOAD RMCS1
8843
8844 030470 012760 141001 000024      MOV      #DMD!MUR!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1
8845
8846 030476 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
8847
8848 030504 016037 000042 001142      MOV      RMER2(RO), $BDDAT ;STORE RMER2 AT $BDDAT
8849 030512 042737 167777 001142      BIC      #1CIVC,$BDDAT ;SET ACTUAL STATUS
8850 030520 016237 066142 001140      MOV      FNCDTB(R2), $GDDAT ;SETUP EXPECTED STATUS FROM
8851 030526 042737 167777 001140      BIC      #1CIVC,$GDDAT ;FUNCTION CODE TABLE
8852 030534 023737 001140 001142      CMP      $GDDAT,$BDDAT
8853 030542 001403                BEQ      30$ ;BRANCH IF IVC IS OK
8854 030544 010237 001174                MOV      R2,$TMPO ;SAVE FUNCTION CODE FOR MSG
8855 030550 104160                ERROR   160 ;IVC IS INCORRECT
8856 030552
30$:
8857 ;ADVANCE FUNCTION CODE AND REPEAT TEST IF NOT DONE
8858 030552 062702 000002                ADD      #2,R2
8859 030556 022702 000076                CMP      #1,F76,R2
8860 030562 103401                BLO     40$ ;BRANCH IF DONE TEST
8861 030564 000700                BR      10$
8862
8863 030566
40$:
8864 ;END OF TEST
8865
8866 ;*****
8867 ;*TEST S1 SET LSC TEST
8868
8869 ;*****
8870 †ST51: SCOPE
8871 MOV      #51,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
8872
8873 NOP
8874 MOV      #20,$ICNT ;20 ITERATIONS
8875 MOV      #1,$ERMAX ;ONE ERROR ALLOWED
8876 MOV      #TS1,$LPADR ;LOAD LOOP ON TEST ADDRESS
8877 MOV      #TS1,$LPERR ;LOAD LOOP ON ERROR ADDRESS

```

```

8877 030630
8878 030630 012706 001100
8879 030634 013700 001276
8880 030640 013701 001456
8881 030644 010037 001136
8882 030650 062737 000042 001136
8883
8884 ;INITIALIZE AND VERIFY THAT LOSS OF SYSTEM CLOCK, "LSC", IS RESET
8885 030656 012760 000040 000010
8886 030664 111.60 000010
8887
8888 030670 012760 000001 000024
8889
8890 030676 012760 040001 000024
8891
8892 030704 012760 000000 000014
8893
8894 030712 012760 000000 000042
8895
8896 030720 016037 000042 001142
8897 030726 042737 173777 001142
8898 030734 001403
8899 030736 005037 001140
8900 030742 104161
8901 ;WITH DEBUG CLOCK ENABLED, SET GO AND WAIT FOR ONE SHOT TO SET
8902 030744 LSC:
8903
8904 030744 012760 000001 000000
8905 030752 012737 000001 001524
8906 030760 004777 150542
8907 030764 005737 001524
8908 030770 001375
8909 030772 004777 150532
8910
8911
8912 030776 012760 000001 000024
8913
8914 031004 016037 000042 001142
8915 031012 042737 173777 001142
8916 031020 001004
8917 031022 012737 004000 001140
8918 031030 104162
8919
8920 031032
8921
8922 ;*****
8923 ;*TEST 52 DECODE TEST
8924 ;*****
8925
8926 031032 000004
8927 031034 012737 000052 001226
8928
8929 031042 000240
8930 031044 012737 000024 001120
8931 031052 112737 000001 001131
8932 031060 012737 031074 001122

```

TS1:
MOV #STACK, SP ;LOAD THE STACK POINTER
MOV \$BASE, RO ;RO = UNIBUS ADDRESS OF UUT
MOV TSTQUE, R1 ;R1 = POINTER TO DEVICE
MOV RO, \$BDAOR
ADD #RMR2, \$BDAOR

;INITIALIZE AND VERIFY THAT LOSS OF SYSTEM CLOCK, "LSC", IS RESET
MOV #CLR, RMCS2(RO) ;CLEAR THE MASSBUS
MOVB (R1), RMCS2(RO) ;SELECT UNIT
MOV #DMD, RMMR1(RO) ;LOAD RMMR1
MOV #DMD!DBEN, RMMR1(RO) ;LOAD RMMR1
MOV #0, RMR1(RO) ;LOAD RMR1
MOV #0, RMR2(RO) ;LOAD RMR2
MOV RMR2(RO), \$BDDAT ;STORE RMR2 AT \$BDDAT
BIC #CLSC, \$BDDAT
BEQ 10\$;BRANCH IF LSC IS ZERO
CLR \$GDDAT
ERROR 161 ;CANT CLEAR LSC
;WITH DEBUG CLOCK ENABLED, SET GO AND WAIT FOR ONE SHOT TO SET LSC:
MOV #GO, RMCS1(RO) ;LOAD RMCS1
MOV #1, WATCH ;SET WATCHDOG TIMER VALUE
JSR PC, \$CLOCK ;START THE CLOCK
TST WATCH
BNE 20\$;WAIT FOR WATCH ZERO
JSR PC, \$STOP ;STOP THE CLOCK
;ONE SHOT SHOULD BE SET-DISABLE DIAGNOSTIC CLOCK AND LSC SHOULD SET.
MOV #DMD, RMMR1(RO) ;LOAD RMMR1
MOV RMR2(RO), \$BDDAT ;STORE RMR2 AT \$BDDAT
BIC #CLSC, \$BDDAT
BNE 30\$;BRANCH IF LSC SET
MOV #LSC, \$GDDAT
ERROR 162 ;CANT SET LSC
;END OF TEST
30\$:
;*****
;*TEST 52 DECODE TEST
;*****
†ST52: SCOPE
MOV #52, \$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
NOP
MOV #20, \$ICNT ;20 ITERATIONS
MOVB #1, \$ERMAX ;ONE ERROR ALLOWED
MOV #TS2, \$LPADR ;LOAD LOOP ON TEST ADDRESS

```

8933 031066 012737 031074 001124      MOV      #T52,$LPERR ;LOAD LOOP ON ERROR ADDRESS
8934 031074                                T52:
8935 031074 012706 001100      MOV      #STACK,SP ;LOAD THE STACK POINTER
8936 031100 013700 001276      MOV      $BASE,R0 ;R0 = UNIBUS ADDRESS OF UUT
8937 031104 013701 001456      MOV      TSTQUE,R1 ;R1 = POINTER TO DEVICE
8938 031110 005037 001416      CLR      RMER10 ;NO ERROR FIRST TEST
8939 031114                                5$:
8940 031114 004737 031474      JSR      PC,100$ ;INITIALIZE
8941
8942 ;EXECUTE A PACK ACKNOWLEDGE AND CHECK VOLUME VALID
8943
8944 031120 013760 001416 000014      MOV      RMER10,RMER1(R0) ;LOAD RMER1
8945
8946 031126 012760 000023 000000      MOV      #PACACK!GO,RMCS1(R0) ;LOAD RMCS1
8947 031134 012703 000003      MOV      #3,R3
8948 031140                                10$:
8949
8950 031140 012760 141001 000024      MOV      #DMD!MUR!DBEN!DBCK,RMMR1(R0) ;LOAD RMMR1
8951
8952 031146 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
8953 031154 005303      DEC      R3
8954 031156 001370      BNE      10$ ;ISSUE NEXT CLOCK IF COUNT NOT 0
8955
8956 031160 016037 000012 001142      MOV      RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
8957 031166 042737 177677 001142      BIC      #!CVV,$BDDAT
8958 031174 001414      BEQ      20$ ;BRANCH IF VV IS ZERO
8959 031176 005737 001416      TST      RMER10
8960 031202 001527      BEQ      70$ ;BRANCH IF VV SHOULD BE SET
8961 031204 005037 001140      CLR      $GDDAT ;SETUP ERROR MESSAGE
8962 031210 010037 001136      MOV      R0,$BDDCR
8963 031214 062737 000012 001136      ADD      #RMDS,$BDDCR
8964 031222 104163      ERROR   163 ;DECODE SET WITH COMP ERROR ACTIVE
8965 031224 000522      BR      80$ ;SKIP
8966 031226                                20$:
8967 031226 004737 031474      JSR      PC,100$ ;INITIALIZE AND SET DIAGNOSTIC MODE
8968
8969 ;EXECUTE A READ IN PRESET AND CHECK VOLUME VALID
8970
8971 031232 013760 001416 000014      MOV      RMER10,RMER1(R0) ;LOAD RMER1
8972
8973 031240 012760 000021 000000      MOV      #RIP!GO,RMCS1(R0) ;LOAD RMCS1
8974 031246 012703 000003      MOV      #3,R3 ;R3=CLOCK COUNT
8975 031252                                30$:
8976
8977 031252 012760 141001 000024      MOV      #DMD!MUR!DBEN!DBCK,RMMR1(R0) ;LOAD RMMR1
8978
8979 031260 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
8980 031266 005303      DEC      R3
8981 031270 001370      BNE      30$ ;ISSUE NEXT CLOCK IF COUNT NOT ZERO
8982
8983 031272 016037 000012 001142      MOV      RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
8984 031300 042737 177677 001142      BIC      #!CVV,$BDDAT
8985 031306 001414      BEQ      40$ ;BRANCH IF VOLUME VALID NOT SET
8986 031310 005737 001416      TST      RMER10
8987 031314 001466      BEQ      70$ ;BRANCH IF VOLUME VALID SHOULD BE SET
8988 031316 005037 001140      CLR      $GDDAT ;SETUP ERROR MESSAGE

```

```

8989 031322 010037 001136      MOV      RO, $BDADR
8990 031326 062737 000012 001136      ADD      #RMS, $BDADR
8991 031334 104163      ERROR    163      ;DECODE SET WITH COMP ERROR ACTIVE
8992 031336 000455      BR       80$      ;SKIP
8993 031340      40$:
8994 031340 004737 031474      JSR      PC, 100$      ;INITIALIZE AND SET DIAGNOSTIC MODE
8995
8996      ;EXECUTE A WRITE CHECK DATA AND CHECK OCCUPIED
8997
8998 031344 013760 001416 000014      MOV      RMER10, RMER1(RO)      ;LOAD RMER1
8999
9000 031352 012760 000051 000000      MOV      #WCD!GO, RMCS1(RO)      ;LOAD RMCS1
9001 031360 012703 000002      MOV      #2, R3      ;R3=CLOCK COUNT
9002 031364      50$:
9003
9004 031364 012760 141001 000024      MOV      #DMD!MUR!DBEN!DBCK, RMMR1(RO)      ;LOAD RMMR1
9005
9006 031372 012760 041001 000024      MOV      #DMD!MUR!DBEN, RMMR1(RO) ;LOAD RMMR1
9007 031400 005303      DEC      R3
9008 031402 001370      BNE     50$      ;ISSUE NEXT CLOCK IF COUNT NOT ZERO
9009
9010 031404 016037 000024 001142      MOV      RMMR1(RO), $BDDAT      ;STORE RMMR1 AT $BDDAT
9011 031412 042737 077777 001142      BIC     #!COCC, $BDDAT
9012 031420 001414      BEQ     60$      ;BRANCH IF OCCUPIED IS RESET
9013 031422 005737 001416      TST     RMER10
9014 031426 001415      BEQ     70$      ;BRANCH IF OCCUPIED SHOULD BE SET
9015 031430 005737 001140      CLR     $GDDAT      ;SETUP ERROR MESSAGE
9016 031434 010037 001136      MOV      RO, $BDADR
9017 031440 062737 000024 001136      ADD      #RMMR1, $BDADR
9018 031446 104164      ERROR    164      ;DECODE SET WITH COMP ERROR ACTIVE
9019 031450 000410      BR       80$
9020 031452      60$:
9021      ;VOLUME VALID AND OCCUPIED DID NOT SET-SEE IF COMP ERROR WAS ACTIVE
9022 031452 005737 001416      TST     RMER10
9023 031456 001005      BNE     80$      ;BRANCH IF COMP ERROR WAS SET
9024      ;COULD NOT SET VV OR OCCUPIED-SUSPECT DECODE FLOP NOT SETTING
9025 031460 104165      ERROR    165      ;DECODE DOES NOT SET
9026
9027 031462      70$:
9028
9029      ;REPEAT TEST WITH COMPOSITE ERROR ACTIVE-VERIFY THAT DECODE FLOP
9030      ;DOES NOT SET, AS INDICATED BY VOLUME VALID AND OCCUPIED.
9031 031462 012737 040000 001416      MOV      #UNS, RMER10      ;USE UNSAFE TO SET COMP ERROR
9032 031470 000611      BR       5$
9033
9034 031472 000513      80$:      BR       200$      ;END OF TEST
9035
9036 031474      100$:
9037
9038      ;SUBROUTINE USED DURING TEST
9039
9040      ;USING DIAGNOSTIC MODE, RESET VOLUME VALID AND COMPOSITE ERROR.
9041      ;VERIFY THAT VV, ERR, AND OCC ARE ZERO.
9042 031474 012760 000040 000010      MOV      #CLR, RMCS2(RO)      ;CLEAR THE MASSBUS
9043 031502 111160 000010      MOVB    (R1), RMCS2(RO)      ;SELECT UNIT
9044

```



```

9045 031506 012760 000001 000024      MOV      #DMD,RMMR1(RO) ;LOAD RMMR1
9046
9047 031514 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
9048
9049 031522 012760 000000 000014      MOV      #0,RMER1(RO) ;LOAD RMER1
9050
9051 031530 012760 000000 000042      MOV      #0,RMER2(RO) ;LOAD RMER2
9052 031536 005037 001140      CLR      $GDDAT ;SETUP FOR ERROR MSG
9053 031542 010037 001136      MOV      RO,$BDADR
9054 031546 062737 000012 001136      ADD      #RMS,$BDADR
9055
9056 031554 016037 000012 001142      MOV      RMS(RO),$BDAT ;STORE RMS AT $BDAT
9057 031562 042737 137777 001142      BIC      #1CERR,$BDAT
9058 031570 001402      BEQ      110$ ;BRANCH IF COMP ERROR ZERO
9059 031572 104143      ERROR   143 ;CANT CLEAR COMP ERROR
9060 031574 000447      BR       140$ ;SKIP TEST
9061 031576
9062
9063 031576 016037 000012 001142      MOV      RMS(RO),$BDAT ;STORE RMS AT $BDAT
9064 031604 042737 177677 001142      BIC      #1CVV,$BDAT
9065 031612 001402      BEQ      120$ ;BRANCH IF VOLUME VALID ZERO
9066 031614 104135      ERROR   135 ;CANT RESET VOLUME VALID
9067 031616 000436      BR       140$ ;SKIP TEST
9068 031620
9069
9070 031620 016037 000024 001142      MOV      RMMR1(RO),$BDAT ;STORE RMMR1 AT $BDAT
9071 031626 042737 077777 001142      BIC      #1COCC,$BDAT
9072 031634 001407      BEQ      130$ ;BRANCH IF OCCUPIED ZERO
9073 031636 010037 001136      MOV      RO,$BDADR ;SETUP ERROR MESSAGE
9074 031642 062737 000024 001136      ADD      #RMMR1,$BDADR
9075 031650 104166      ERROR   166 ;CANT CLEAR OCCUPIED
9076 031652 000420      BR       140$ ;SKIP TEST
9077 031654
9078
9079
9080
9081 031654 012760 000024 000000      MOV      #ILF24, RMCS1(RO) ;LOAD RMCS1
9082
9083 031662 016037 000014 001142      MOV      RMER1(RO),$BDAT ;STORE RMER1 AT $BDAT
9084 031670 042737 177776 001142      BIC      #1CILF,$BDAT
9085 031676 001410      BEQ      150$ ;BRANCH IF ILF IS ZERO
9086 031700 010037 001136      MOV      RO,$BDADR ;SETUP ERROR MESSAGE
9087 031704 062737 000014 001136      ADD      #RMER1,$BDADR
9088 031712 104167      ERROR   167 ;DECODE FLOP APPEARS ON
9089 031714 012716 031722 140$: MOV      #200$,(SP) ;DONT GO BACK TO TEST
9090
9091 031720 000207 150$: RTS      PC ;RETURN TO TEST OR EXIT TEST
9092
9093 031722 200$:
9094
9095
9096
9097
9098
9099 031722 000004
9100 031724 012737 000053 001226
;*****
; *TEST 53 SET/RESET VOLUME VALID TEST
;*****
↑ST53: SCOPE
MOV      #53,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX

```

```

9101
9102 031732 000240
9103 031734 012737 000024 001120
9104 031742 112737 000001 001131
9105 031750 012737 031764 001122
9106 031756 012737 031764 001124
9107 031764
9108 031764 012706 001100
9109 031770 013700 001276
9110 031774 013701 001456
9111 032000 010037 001136
9112 032004 062737 000012 001136
9113 032012 012702 032224
9114 032016
9115
9116 032016 012760 000040 000010
9117 032024 111160 000010
9118
9119 032030 012760 000001 000024
9120
9121 032036 012760 041001 000024
9122
9123 032044 012760 000000 000014
9124
9125 032052 012760 000000 000042
9126
9127 032060 016037 000012 001142
9128 032066 042737 177677 001142
9129 032074 001403
9130 032076 005037 001140
9131 032102 104135
9132 032104
9133
9134 032104 111203
9135 032106 042703 177701
9136 032112 052703 000001
9137
9138 032116 010360 000000
9139 032122 116204 000001
9140 032126 042704 177400
9141 032132
9142
9143 032132 012760 141001 000024
9144
9145 032140 012760 041001 000024
9146 032146 005304
9147 032150 001370
9148
9149 032152 016037 000012 001142
9150 032160 042737 177677 001142
9151 032166 001007
9152 032170 010337 001174
9153 032174 012737 000100 001140
9154 032202 104170
9155 032204 000406
9156 032206

```

```

NOP
MOV #20,$ICNT ;20 ITERATIONS
MOVB #1,$ERMAX ;ONE ERROR ALLOWED
MOV #T53,$LPADR ;LOAD LOOP ON TEST ADDRESS
MOV #T53,$LPERR ;LOAD LOOP ON ERROR ADDRESS
T53:
MOV #STACK,SP ;LOAD THE STACK POINTER
MOV $BASE,R0 ;R0 = UNIBUS ADDRESS OF UUT
MOV TSTQUE,R1 ;R1 = POINTER TO DEVICE
MOV R0,$BDADR ;SETUP REGISTER ADDRESS
ADD #RMD5,$BDADR
MOV #100$,R2 ;R2=TABLE POINTER
10$:
;INITIALIZE AND USE DIAGNOSTIC MODE TO RESET VOLUME VALID
MOV #CLR,RMCS2(R0) ;CLEAR THE MASSBUS
MOVB (R1),RMCS2(R0) ;SELECT UNIT
MOV #DMD,RMMR1(R0) ;LOAD RMMR1
MOV #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
MOV #0,RMER1(R0) ;LOAD RMER1
MOV #0,RMER2(R0) ;LOAD RMER2
MOV RMD5(R0),$BDDAT ;STORE RMD5 AT $BDDAT
BIC #↑CVV,$BDDAT
BEQ 20$ ;BRANCH IF VOLUME VALID ZERO
CLR $GDDAT
ERROR 135 ;CANT RESET VV
20$:
;EXECUTE THE FUNCTION CODE IN THE TABLE
MOVB (R2),R3 ;GET FUNCTION CODE
BIC #↑CILF76,R3
BIS #GO,R3
MOV R3,RMCS1(R0) ;LOAD RMCS1
MOVB 1(R2),R4 ;GET CLOCK COUNT
BIC #↑C377,R4
30$:
MOV #DMD!DBEN!MUR!DBCK,RMMR1(R0) ;LOAD RMMR1
MOV #DMD!DBEN!MUR,RMMR1(R0) ;LOAD RMMR1
DEC R4
BNE 30$ ;ISSUE COCKS TIL R4 ZERO
MOV RMD5(R0),$BDDAT ;STORE RMD5 AT $BDDAT
BIC #↑CVV,$BDDAT
BNE 40$ ;BRANCH IF VOLUME VALID SET
MOV R3,$TMPD ;SAVE FUNCTION CODE FOR MSG
MOV #VV,$GDDAT
ERROR 170 ;CANT SET VOLUME VALID
BP 50$
40$:

```

```

9157
9158 032206 062702 000002
9159 032212 105762 000001
9160 032216 100401
9161 032220 000676
9162 032222 000403
9163
9164 032224
9165
9166 032224 020
9167 032225 003
9168
9169 032226 022
9170 032227 003
9171
9172 032230 000
9173 032231 377
9174
9175 032232
9176
9177
9178
9179
9180
9181 032232 000004
9182 032234 012737 000054 001226
9183
9184 032242 000240
9185 032244 012737 000024 001120
9186 032252 112737 000001 001131
9187 032260 012737 032274 001122
9188 032266 012737 032274 001124
9189 032274
9190 032274 012706 001100
9191 032300 013700 01276
9192 032304 013701 001456
9193 032310 005002
9194 032312
9195
9196 032312 004737 060654
9197 032316 000402
9198 032320 104000
9199 032322 000460
9200 032324 012704 000002
9201
9202
9203 032330 012760 041001 000024
9204 032336 010203
9205 032340 052703 000001
9206
9207 032344 010360 000000
9208 032350
9209
9210 032350 012760 141001 000024
9211
9212 032356 012760 041001 000024

```

```

;ADVANCE THE TABLE POINTER, EXIT IF DONE
ADD #2,R2
TSTB 1(R2)
BMI 50$ ;EXIT IF COUNT IS NEGATIVE
BR 10$
50$: BR 200$ ;JUMP OVER TABLE

100$:
;TABLE OF FUNCTION CODES AND CLOCK COUNTS
.BYTE RIP ;READ IN PRESET COMMAND
.BYTE 3
.BYTE PAKACK ;PACK ACKNOWLEDGE COMMAND
.BYTE 3
.BYTE ;END OF TABLE
.BYTE -1

200$: ;END OF TEST

;*****
;*TEST 54 ILLEGAL FUNCTION TEST
;*****
†ST54: SCOPE
MOV #54,$TESTN ;;SET TEST NUMBER IN AP* MAIL BOX

NOP
MOV #20,$ICNT ;20 ITERATIONS
MOVB #1,$ERRMAX ;ONE ERROR ALLOWED
MOV #T54,$PADR ;LOAD LOOP ON TEST ADDRESS
MOV #T54,$LPERR ;LOAD LOOP ON ERROR ADDRESS

T54:
MOV #STACK,SP ;LOAD THE STACK POINTER
MOV $BASE,R0 ;R0 = UNIBUS ADDRESS OF UUT
MOV TSTQUE,R1 ;R1 = POINTER TO DEVICE
CLR R2 ;INITIALIZE FUNCTION CODE VALUE

10$:
;SET VOLUME VALID USING SUBROUTINE
JSR PC,SETVV ;GO SET VOLUME VALID
BR 20$ ;BRANCH TO 20$ IF NO ERROR
ERROR ;RETURN HERE IF ERROR
BR 50$ ;SKIP TEST IF ERROR

20$: MOV #2,R4 ;R4=CLOCK COUNT
;EXECUTE THE TEST FUNCTION CODE AND VERIFY ILF

MOV #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
MOV R2,R3 ;SETUP FUNCTION CODE IN R3
BIS #G0,R3

MOV R3,RMCS1(R0) ;LOAD RMCS1

30$:
MOV #DMD!MUR!DBEN!DBCK,RMMR1(R0) ;LOAD RMMR1
MOV #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1

```

```

9213 032364 005304          DEC      R4
9214 032366 001370          BNE     30$
9215
9216 032370 016037 000014 001142      MOV     RMER1(R0), $BDDAT      ; STORE RMER1 AT $BDDAT
9217 032376 042737 177776 001142      BIC     #1,CILF,$BDDAT      ; SETUP ACTUAL ILF STATUS
9218 032404 016237 066142 001140      MOV     FNCDTB(R2), $GDDAT    ; GET EXPECTED ILF STATUS
9219 032412 042737 177776 001140      BIC     #1,CILF,$GDDAT
9220 032420 023737 001140 001142      CMP     $GDDAT,$BDDAT
9221 032426 001410          BEQ     40$                  ; BRANCH IF ILF IS OK
9222 032430 010037 001136          MOV     R0,$BDDADR          ; SETUP FOR ERROR MSG
9223 032434 062737 000014 001136      ADD     #RMER1,$BDDADR
9224 032442 010237 001174          MOV     R2,$TMP0
9225 032446 104171          ERROR  171                  ; ILF IS NOT CORRECT
9226 032450
9227
9228 032450 062702 000002          40$:
;ADVANCE TO THE NEXT FUNCTION CODE AND REPEAT TEST
9229 032454 022702 000076          ADD     #2,R2
9230 032460 103401          CMP     #ILF76,R2
9231 032462 000713          BLO    50$
9232 032464
9233
9234
9235
9236
9237
9238 032464 000004          50$:
;END OF TEST
9239 032466 012737 000055 001226      ;*****
;TEST 55 OCCUPIED TEST
9240
9241
9242
9243
9244
9245
9246
9247
9248
9249
9250
9251
9252 032544
9253
9254
9255 032544 004737 060654          ;*****
;TST55: SCOPE
9256 032550 000402          MOV     #55,$TESTN          ;; SET TEST NUMBER IN APT MAIL BOX
9257 032552 104000          NOP
9258 032554 000464          MOV     #20,$ICNT          ; 20 ITERATIONS
9259
9260
9261
9262
9263
9264
9265
9266 032572 010360 000000          MOV     #1,$ERRMAX         ; ONE ERROR ALLOWED
9267 032576 012704 000002          MOV     #T55,$LPADR        ; LOAD LOOP ON TEST ADDRESS
9268
;STEP THE DEBUG CLOCK UNTIL SET PULSE IS ACTIVE
;*****
;T55:
MOV     #STACK_SP          ; LOAD THE STACK POINTER
MOV     $BASE,R0           ; R0 = UNIBUS ADDRESS OF UUT
MOV     TSTQUE,R1          ; R1 = POINTER TO DEVICE
CLR     R2                  ; INITIALIZE FUNCTION CODE

10$:
;GET THE DEVICE READY
;SET VOLUME VALID USING SUBROUTINE
JSR     PC,SETVV           ; GO SET VOLUME VALID
BR      20$                ; BRANCH TO 20$ IF NO ERROR
ERROR  20$                  ; RETURN HERE IF ERROR
BR      50$

20$:
;ENABLE DEBUG CLOCK AND LOAD THE FUNCTION CODE
MOV     #DMD!MUR!DBEN,RMMR1(R0) ; LOAD RMMR1
MOV     R2,R3              ; ASSEMBLE FUNCTION CODE AND
BIS     #GO,R3             ; GO BIT IN R3

MOV     R3,RMCS1(R0)       ; LOAD RMCS1
MOV     #2,R4              ; R4=CLOCK COUNT

```

CZRMJ80 RM03/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 195
T55 OCCUPIED TEST

SEQ 0195

```

9269 032502          30$:
9270
9271 032602 012760 141001 000024      MOV      #DMD!MUR!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1
9272
9273 032610 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR!
9274 032616 005304      DEC      R4
9275 032620 001370      BNE     30$ ;ISSUE NEXT CLOCK TIL R4 ZERO
9276          ;VERIFY OCCUPIED STATUS
9277
9278 032622 016037 000024 001142      MOV      RMMR1(RO), $BDDAT ;STORE RMMR1 AT $BDDAT
9279 032630 042737 077777 001142      BIC     #1COCC, $BDDAT
9280 032636 005037 001140      CLR     $GDDAT ;GENERATE OCC FROM AOE
9281 032642 032762 001000 066142      BIT     #AGE, FNCDTB(R2)
9282 032650 001403      BEQ     35$

```

```

9283 032652 012737 100000 001140      MOV      #OCC,$GDDAT
9284 032660 023737 001140 001142 35$:    CMP      $GDDAT,$BDDAT
9285 032666 001411                BEQ      40$                ;BRANCH IF OCC IS OK
9286 032670 010237 001174                MOV      R2,$TMP0          ;SAVE FUNCTION CODE
9287 032674 010037 001136                MOV      R0,$BDAOR         ;SETUP REGISTER ADDRESS
9288 032700 062737 000024 001136                ADD      #RMMR1,$BDAOR
9289 032706 104173                ERROR   173                ;OCCUPIED IS INCORRECT
9290 032710 000406                BR       50$
9291 032712
9292
9293 032712 062702 000002                40$:    ;ADVANCE TO NEXT FUNCTIONCODE, EXIT IF DONE
9294 032716 022702 000076                ADD      #2,R2
9295 032722 103401                CMP      #ILF76,R2
9296 032724 000707                BLO     50$                ;EXIT IF DONE
9297
9298 032726                BR       10$
9299
9300
9301
9302
9303
9304 032726 000004                50$:    ;END OF TEST
9305 032730 012737 000056 001226                ;*****
; *TEST 56      READ IN PRESET TEST
;*****
9306
9307 032736 000240                †T56:  SCOPE
9308 032740 012737 000024 001120                MOV      #56,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
9309 032746 112737 000001 001131                NOP
9310 032754 012737 032770 001122                MOV      #20,$ICNT      ;20 ITERATIONS
9311 032762 012737 032770 001124                MOV      #1,$ERMAX      ;ONE ERROR ALLOWED
9312 032770                MOV      #T56,$LPAOR    ;LOAD LOOP ON TEST ADDRESS
9313 032770 012706 001100                MOV      #T56,$LPERR    ;LOAD LOOP ON ERROR ADDRESS
9314 032774 013700 001276                T56:    MOV      #STACK,SP      ;LOAD THE STACK POINTER
9315 033000 013701 001456                MOV      $BASE,R0      ;R0 = UNIBUS ADDRESS OF UUT
9316                MOV      TSTQUE,R1     ;R1 = POINTER TO DEVICE
9317 033004 012760 000040 000010                ;CLEAR AND ENABLE DEBUG CLOCK - LEAVE VOLUME VALID RESET
9318 033012 111160 000010                MOV      #CLR, RMCS2(R0) ;CLEAR THE MASSBUS
9319                MOV      (R1),RMCS2(R0) ;SELECT UNIT
9320 033016 012760 000001 000024                MOV      #DMD,RMMR1(R0) ;LOAD RMMR1
9321
9322 033024 012760 041001 000024                MOV      #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
9323
9324 033032 012760 000000 000014                MOV      #0,RMER1(R0)   ;LOAD RMER1
9325
9326 033040 012760 000000 000042                MOV      #0,RMER2(R0)   ;LOAD RMER2
9327                ;LOAD ALL ONES IN RMDA, RMDC AND RMOF
9328
9329 033046 012760 177777 000006                MOV      #-1,RMDA(R0)  ;LOAD RMDA
9330
9331 033054 012760 177777 000034                MOV      #-1,RMDC(R0)  ;LOAD RMDC
9332
9333 033062 012760 177777 000032                MOV      #-1,RMOF(R0)  ;LOAD RMOF
9334                ;LOAD READ IN PRESET COMMAND AND STEP THE CLOCK TILL SET PULSE
9335
9336 033070 012760 000021 000050                MOV      #RIP!GO, RMCS1(R0) ;LOAD RMCS1
9337 033076 012702 000003                MOV      #3,R2         ;R2=CLOCK COUNT
9338 033102

```

```

9339
9340 033102 012760 141001 000024      MOV      #DMD!MUR!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1
9341
9342 033110 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
9343 033116 005302
9344 033120 001370      DEC      R2
9345      BNE     10$ ;ISSUE 3 CLOCKS
;SEE IF RMDA OR RMDC OR RMOF IS ZERO
9346
9347 033122 016002 000006      MOV      RMDA(RO),R2 ;STORE RMDA AT R2
9348 033126 005702      TST     R2
9349 033130 001413      BEQ     20$ ;BRANCH IF RMDA IS ZERO
9350
9351 033132 016002 000034      MOV      RMDC(RO),R2 ;STORE RMDC AT R2
9352 033136 042702 176000      BIC     #XNUDC,R2 ;CLEAR UNUSED BITS
9353 033142 001406      BEQ     20$ ;BRANCH IF RMDC IS ZERO
9354
9355 033144 016002 000032      MOV      RMOF(RO),R2 ;STORE RMOF AT R2
9356 033150 042702 161577      BIC     #XNUOF,R2 ;CLEAR UNUSED BITS
9357 033154 001401      BEQ     20$ ;BRANCH IF RMOF IS ZERO
;READ IN PRESET COMMAND DIDNT CLEAR ANY OF THE 3 REGISTERS
9358      ;READ IN PRESET FAILED
9359 033156 104174      ERROR   174
9360
9361 033160      20$: ;END OF TEST
9362
9363      ;*****
9364      ;*TEST 57 RIP/RMOF TEST
9365
9366      ;*****
9367 033160 000004      †T57: SCOPE
9368 033162 012737 000057 001226      MOV      #57,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
9369
9370 033170 000240      NOP
9371 033172 012737 000024 001120      MOV      #20,$ICNT ;20 ITERATIONS
9372 033200 112737 000001 001131      MOV     #1,$ERMAX ;ONE ERROR ALLOWED
9373 033206 012737 033222 001122      MOV      #T57,$LPADR ;LOAD LOOP ON TEST ADDRESS
9374 033214 012737 033222 001124      MOV      #T57,$LPERR ;LOAD LOOP ON ERROR ADDRESS
9375 033222
9376 033222 012706 001100      T57: MOV      #STACK_SP ;LOAD THE STACK POINTER
9377 033226 013700 001276      MOV      $BASE,R0 ;R0 = UNIBUS ADDRESS OF UUT
9378 033232 013701 001456      MOV      TSTQUE,R1 ;R1 = POINTER TO DEVICE
9379 033236 010037 001136      MOV      R0,$B0ADR ;SETUP REGISTER ADDRESS AND
9380 033242 062737 000032 001136      ADD      #RMOF,$B0ADR
9381 033250 005037 001140      CLR      $GDATA ;EXPECTED RMOF
9382
9383      ;INITIALIZE AND SET BITS IN RMOF
9384 033254 012760 000040 000010      MOV      #CLR,RMCS2(RO) ;CLEAR THE MASSBUS
9385 033262 111160 000010      MOV     (R1),RMCS2(RO) ;SELECT UNIT
9386
9387 033266 012760 000001 000024      MOV      #DMD,RMMR1(RO) ;LOAD RMMR1
9388
9389 033274 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
9390
9391 033302 012760 000000 000014      MOV      #0,RMER1(RO) ;LOAD RMER1
9392
9393 033310 012760 000000 000042      MOV      #0,RMER2(RO) ;LOAD RMER2
9394

```

```

9395 033316 012760 177777 000032      MOV      #-1,RMOF(RO)      ;LOAD RMOF
9396                                     :EXECUTE A READ IN PRESET IN DIAGNOSTIC MODE TILL SET PULSE
9397
9398 033324 012760 000021 000000      MOV      #RIP!GO,RMCS1(RO) ;LOAD RMCS1
9399 033332 012702 000003      MOV      #3,R2              ;R2=CLOCK COUNT
9400 033336
9401
9402 033336 012760 141001 000024      MOV      #DMD!MUR!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1
9403
9404 033344 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
9405 033352 005302
9406 033354 001370      DEC      R2
9407                                     BNE      10$                ;ISSUE 3 CLOCKS
9408                                     :VERIFY THAT RMOF IS ZERO
9409 033356 016037 000032 001142      MOV      RMOF(RO), $BDDAT ;STORE RMOF AT $BDDAT
9410 033364 042737 161577 001142      BIC      #XNUOF,$BDDAT
9411 033372 001401
9412 033374 104175      BEQ      20$                ;BRANCH IF RMOF IS ZERO
9413                                     ERROR    17$                ;CANT CLEAR RMOF WITH RIP
9414 033376
9415
9416
9417
9418
9419
9420 033376 00C004
9421 033400 012737 000060 001226      *ST60:  SCOPE
9422                                     MOV      #60,$TESTN        ;;SET TEST NUMBER IN APT MAIL BOX
9423 033406 000240
9424 033410 012737 000024 001120      NOP
9425 033416 112737 000001 001131      MOV      #20,$ICNT        ;20 ITERATIONS
9426 033424 012737 033440 001122      MOV      #1,$ERMAX        ;ONE ERROR ALLOWED
9427 033432 012737 023440 001124      MOV      #T60,$LPADR      ;LOAD LOOP ON TEST ADDRESS
9428 033440      MOV      #T60,$LPERR      ;LOAD LOOP ON ERROR ADDRESS
9429 033440 012706 001100      T60:    MOV      #STACK,SP    ;LOAD THE STACK POINTER
9430 033444 013700 001276      MOV      $BASE,R0         ;R0 = UNIBUS ADDRESS OF UUT
9431 033450 013700 001456      MOV      TSTQUE,R1        ;R1 = PCINTER TO DEVICE
9432 033454 005037 001140      CLR      $GDDAT
9433
9434
9435 033460 012760 000040 000010      ;CLEAR, ENABLE DEBUG CLOCK, THEN PRESET RMDA AND RMDC
9436 033466 111160 000010      MOV      #CLR,RMCS2(RO)   ;CLEAR THE MASSBUS
9437                                     MOV      (R1),RMCS2(RO)   ;SELECT UNIT
9438 033472 012760 000001 000024      MOV      #DMD,RMMR1(RO)  ;LOAD RMMR1
9439
9440 033500 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
9441
9442 033506 012760 000000 000014      MOV      #0,RMER1(RO)    ;LOAD RMER1
9443
9444 033514 012760 000000 000014      MOV      #0,RMER1(RO)    ;LOAD RMER1
9445
9446 033522 012760 177777 000006      MOV      #-1,RMDA(RO)    ;LOAD RMDA
9447
9448 033530 012760 177777 000034      MOV      #-1,RMDC(RO)    ;LOAD RMDC
9449                                     :EXECLTE READ IN PRESET TILL SET PULSE
9450

```


E16

CZRMJ80 RM03/2 DSALS DIAG
CZRMJ8.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 199
T60 RMDA/RMDC/RIP TEST

SEQ 0199

```

9451 033536 012760 000021 000000      MOV      #R.P!GO,RMCS1(RO)      ;LOAD RMCS1
9452 033544 012702 000003                MOV      #3,R2
9453 033550                10$:
9454
9455 033550 012760 141001 000024      MOV      #DMD!MUR!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1
9456
9457 033556 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
9458 033564 005302                DEC      R2
9459 033566 001370                BNE     10$
9460                ;VERIFY RMDA IS ZERO
9461
9462 033570 016037 000006 001142      MOV      RMDA(RO),SBDDAT ;STORE RMDA AT SBDDAT
9463 033576 005737 001142      TST     SBDDAT
9464 033602 001406                BEQ     20$ ;BRANCH IF RMDA RESET
9465 033604 010037 001136      MOV      RO,SBADR
9466 033610 062737 000006 001136      ADD     #RMDA,SBADR
9467 033616 104176                ERROR   176 ;RMDA NOT RESET BY RIP
9468 033620                20$:
9469                ;VERIFY RMDC IS ZERO
9470
9471 033620 016037 000034 001142      MOV      RMDC(RO),SBDDAT ;STORE RMDC AT SBDDAT
9472 033626 042737 176000 001142      BIC     #XNUDC,SBDDAT
9473 033634 001406                BEQ     30$ ;BRANCH IF RMDC RESET
9474 033636 010037 001136      MOV      RO,SBADR
9475 033642 062737 000034 001136      ADD     #RMDC,SBADR
9476 033650 104177                ERROR   177 ;RMDC NOT RESET BY RIP
9477
9478 033652                30$:
9479                ;END OF TEST
9480
9481                ;*****
9482                ;*TEST 61      OFFSET COMMAND TEST
9483                ;*****
9484 033652 000004      TST61: SCOPE
9485 033654 012737 000061 001226      MOV     #61,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
9486
9487 033662 000240      NOP
9488 033664 012737 000024 001120      MOV     #20,$ICNT ;20 ITERATIONS
9489 033672 112737 000001 001131      MOV     #1,$EMAX ;ONE ERROR ALLOWED
9490 033700 012737 033714 001122      MOV     #T61,$LPADR ;LOAD LOOP ON TEST ADDRESS
9491 033706 012737 033714 001124      MOV     #T61,$LPERK ;LOAD LOOP ON ERROR ADDRESS
9492 033714                T61:
9493 033714 012706 001100      MOV     #STACK,SP ;LOAD THE STACK POINTER
9494 033720 013700 001276      MOV     $BASE,RO ;RO = UNIBUS ADDRESS OF UUT
9495 033724 013701 001456      MOV     TSTQUE,R1 ;R1 = POINTER TO DEVICE
9496 033730 010037 001136      MOV     RO,SBADR
9497 033734 062737 000012 001136      ADD     #RMDA,SBADR
9498 033742 012737 000001 001140      MOV     #OM,SBDDAT
9499
9500                ;SET VOLUME VALID USING SUBROUTINE
9501 033750 004737 060654      JSR     PC,SETVV ;GO SET VOLUME VALID
9502 033754 000402                BR     10$ ;BRANCH TO 10$ IF NO ERROR
9503 033756 104000                ERPOR
9504 033760 000433                BR     40$ ;RETURN HERE IF ERROR
9505
9506                10$:

```

```

9507 033762 012760 000000 000034      MOV      #0,RMDC(RO)      ;LOAD RMDC
9508      ;ENABLE DEBUG CLOCK AND EXECUTE OFFSET COMMAND
9509
9510 033770 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
9511
9512 033776 012760 000015 000000      MOV      #OFFSET!GO,RMCS1(RO)    ;LOAD RMCS1
9513 034004 012702 000002                MOV      #2,R2                ;R2=CLOCK COUNT
9514 034010      20$:
9515
9516 034010 012760 141001 000024      MOV      #DMD!MUR!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1
9517
9518 034016 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
9519 034024 005302                DEC      R2
9520 034026 001370                BNE     20$                    ;ISSUE 2 CLOCKS
9521      ;VERIFY THAT OFFSET MODE IS SET
9522
9523 034030 016037 000012 001142      MOV      RMDS(RO), $BDDAT ;STORE RMDS AT $BDDAT
9524 034036 042737 177776 001142      BIC     #1COM,$BDDAT
9525 034044 001001                BNE     40$                    ;BRANCH IF OM IS SET
9526 034046 104200                ERROR   200                    ;CANT SET ON BY COMMAND
9527 034050      40$:                          ;END OF TEST
9528
9529      ;*****
9530      ;*TEST 62      RETURN TO CENTER TEST
9531
9532      ;*****
9533 034050 000004      †ST62:  SCOPE
9534 034052 012737 000062 001226      MOV      #62,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
9535
9536      NOP
9537 034062 012737 000024 001120      MOV      #20,$ICNT          ;20 ITERATIONS
9538 034070 112737 000001 001131      MOV     #1,$ERMAX          ;ONE ERROR ALLOWED
9539 034076 012737 034112 001122      MOV     #T62,$LPADR        ;LOAD LOOP ON TEST ADDRESS
9540 034104 012737 034112 001124      MOV     #T62,$LPERR        ;LOAD LOOP ON ERROR ADDRESS
9541      T62:
9542 034112 012706 001100      MOV     #STACK,SP          ;LOAD THE STACK POINTER
9543 034116 013700 001276      MOV     $BASE,R0           ;R0 = UNIBUS ADDRESS OF JUT
9544 034122 013701 001456      MOV     TSTQUE,R1          ;R1 = POINTER TO DEVICE
9545 034126 010037 001136      MOV     R0,$BDADR
9546 034132 062737 000012 001136      ADD     #RMDS,$BDADR
9547      ;SET OFFSET DIRECTION AND OFFSET MODE
9548      ;SET VOLUME VALID USING SUBROUTINE
9549 034140 004737 060654      JSR     PC,SETVV           ;GO SET VOLUME VALID
9550 034144 000402      BR     10$                 ;BRANCH TO 10$ IF NO ERROR
9551 034146 104000      ERROR   ;RETURN HERE IF ERROR
9552 034150 000465      BR     60$
9553 034152      10$:
9554
9555 034152 012760 000200 000032      MOV     #OFD,RMOF(RO)      ;LOAD RMOF
9556      ;SET OFFSET MODE USING SUBROUTINE
9557 034160 004737 061004      JSR     PC,SETOM          ;GO SET OFFSET MODE
9558 034164 000401      BR     20$                 ;BRANCH TO 20$ IF NO ERROR
9559 034166 104000      ERROR   ;RETURN HERE IF ERROR
9560 034170      20$:
9561      ;ENABLE DEBUG CLOCK AND EXECUTE RETURN TOCENTER COMMAND
9562      ;SET VOLUME VALID USING SUBROUTINE

```

```

9563 034170 004737 060654 JSR PC,SETVV ;GO SET VOLUME VALID
9564 034174 000402 BR 30$ ;BRANCH TO 30$ IF NO ERROR
9565 034176 104000 ERROR ;RETURN HERE IF ERROR
9566 034200 000451 BR 60$
9567 C34202 30$:
9568
9569 034202 012760 041001 000024 MOV #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
9570
9571 034210 012760 000017 000000 MOV #RTC!GO,RMCS1(RO) ;LOAD RMCS1
9572 034216 012702 000002 MOV #2,R2
9573 034222 40$:
9574
9575 034222 012760 141001 000024 MOV #DMD!MUR!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1
9576
9577 034230 012760 041001 000024 MOV #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
9578 034236 005302 DEC R2
9579 034240 001370 BNE 40$ ;ISSUE 2 CLOCKS
9580 ;VERIFY THAT OFFSET MODE IS RESET
9581
9582 034242 016037 000012 001142 MOV RMDS(RO), $BDDAT ;STORE RMDS AT $BDDAT
9583 034250 042737 177776 001142 BIC #1COM,$BDDAT
9584 034256 001403 BEQ 50$ ;BRANCH IF OFFSET MODE RESET
9585 034260 005037 001140 CLR $GDDAT
9586 034264 104201 ERROR 201 ;CANT RESET OFFSETMODE BY RTC
9587 034266 50$:
9588 ;VERIFY THAT OFFSET DIRECTION IS RESET
9589
9590 034266 016037 000032 001142 MOV RMOF(RO), $BDDAT ;STORE RMOF AT $BDDAT
9591 034274 042737 177577 001142 BIC #1COFD,$BDDAT
9592 034302 001410 BEQ 60$ ;BRANCH IF OFD IS RESET
9593 034304 005037 001140 CLR $GDDAT
9594 034310 010037 001136 MOV RO,$BDADR
9595 034314 062737 000032 001136 ADD #RMOF,$BDADR
9596 034322 104202 ERROR 202 ;CANT RESET OFD BY RTC
9597 034324 60$: ;END OF TEST
9598
9599
9600 ;*****
9601 ;*TEST 63 RMDC CLEAR OFFSET TEST
9602 ;*****
9603 034324 000004 TEST63: SCOPE
9604 034326 012737 000063 001226 MOV #63,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
9605
9606 034334 000240 NOP
9607 034336 012737 000024 001120 MOV #20,$ICNT ;20 ITERATIONS
9608 034344 112737 000001 001131 MOVB #1,$EMAX ;ONE ERROR ALLOWED
9609 034352 012737 034366 001122 MOV #T63,$LPADR ;LOAD LOOP ON TEST ADDRESS
9610 034360 012737 034366 001124 MOV #T63,$LPERR ;LOAD LOOP ON ERROR ADDRESS
9611 034366 T63:
9612 034366 012706 001100 MOV #STACK,SP ;LOAD THE STACK POINTER
9613 034372 013700 001276 MOV $BASE,R0 ;R0 = UNIBUS ADDRESS OF UUT
9614 034376 013701 001456 MOV TSTQUE,R1 ;R1 = POINTER TO DEVICE
9615 034402 010037 001136 MOV RC,$BDADR
9616 034406 062737 000012 001136 ADD #RMDS,$BDADR
9617 ;SET VOLUME VALID USING SUBROUTINE
9618 034414 004737 060654 JSR PC,SETVV ;GO SET VOLUME VALID

```

H16

CZRMJBD RM03/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 202
T63 RMDC CLEAR OFFSET TEST

SEQ 0202

```

9619 034420 000402          BR      10$          ;BRANCH TO 10$ IF NO ERROR
9620 034422 104000          ERROR          ;RETURN HERE IF ERROR
9621 034424 000421          BR      40$          ;SKIP REST OF TEST
9622 034426
9623
9624 034426 004737 061004      10$:
;SET OFFSET MODE USING SUBROUTINE
JSR      PC,SETOM          ;GO SET OFFSET MODE
9625 034432 000401          BR      20$          ;BRANCH TO 20$ IF NO ERROR
9626 034434 104000          ERROR          ;RETURN HERE IF ERROR
9627 034436
9628
9629
9630 034436 012760 000000 000034      20$:
;WRITE THE DESIRED CYLINDER REGISTER AND VERIFY THAT OFFSET IS ZERO
MOV      #0,RMDC(RO)      ;LOAD RMDC
9631
9632 034444 016037 000012 001142      MOV      RMDS(RO), $BDDAT ;STORE RMDS AT $BDDAT
9633 034452 042737 177776 001142      BIC      #1COM, $BDDAT
9634 034460 001403          BEQ      40$
9635 034462 005037 001140          CLR      $GDDAT
9636 034466 104203          ERROR      20$          ;CANT RESET OFFSET BY RMDC
9637
9638 034470          40$:
;END OF TEST
9639
9640
9641
9642
9643
9644 034470 000004          ;*****
9645 034472 012737 000064 001226      ;TEST 64          EBL CLEAR OFFSET TEST
9646
9647
9648 034500 000240          ;*****
9649 034502 012737 000024 001120      †ST64: SCOPE
MOV      #64, $TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
9650 034510 112737 000001 001131      NOP
9651 034516 012737 034532 001122      MOV      #20, $ICNT      ;20 ITERATIONS
9652 034524 012737 034532 001124      MOVVB   #1, $ERMAX      ;ONE ERROR ALLOWED
9653 034532 012706 001100          MOV      #T64, $LPADR   ;LOAD LOOP ON TEST ADDRESS
9654 034536 013700 001276          MOV      #T64, $LPERR  ;LOAD LOOP ON ERROR ADDRESS
9655 034542 013701 001456          T64:
MOV      #STACK, SP      ;LOAD THE STACK POINTER
MOV      $BASE, RO      ;RO = UNIBUS ADDRESS OF LUT
MOV      †STQUE, R1     ;R1 = POINTER TO DEVICE
9656 034546 010037 001136          RO, $BDADR      ;SETUP REGISTER FOR ERROR MSG
9657 034552 062737 000012 001136      ADD      #RMDS, $BDADR
9658
9659 034560 004737 060654      ;SET VOLUME VALID USING SUBROUTINE
JSR      PC,SETVV          ;GO SET VOLUME VALID
9660 034564 000402          BR      10$          ;BRANCH TO 10$ IF NO ERROR
9661 034566 104000          ERROR          ;RETURN HERE IF ERROR
9662 034570 000440          BR      30$          ;SKIP REST OF TEST IF ERROR
9663 034572
9664
9665 034572 004737 061004      10$:
;SET OFFSET MODE USING SUBROUTINE
JSR      PC,SETOM          ;GO SET OFFSET MODE
9666 034576 000401          BR      20$          ;BRANCH TO 20$ IF NO ERROR
9667 034600 104000          ERROR          ;RETURN HERE IF ERROR
9668 034602
9669
9670
9671 034602 012760 010000 000032      20$:
;SET 16 BIT FORMAT AND LOAD LAST SECTOR/TRACK ADDRESS
MOV      #FMT16, RMOF(RO) ;LOAD RMOF
9672
9673 034610 012760 002037 000006      MOV      #002037, RMDA(RO) ;LOAD RMDA
9674
;FORCE END OF BLOCK AND VERIFY THAT OFFSET MODE IS CLEARED

```

```

9675
9676 034616 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
9677
9678 034624 012760 000001 000000      MOV      #GO, RMCS1(RO) ;LOAD RMCS1
9679
9680 034632 012760 061001 000024      MOV      #DMD!MUR!DBEN!DEBL,RMMR1(RO) ;LOAD RMMR1
9681
9682 034640 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
9683
9684 034646 016037 000012 001142      MOV      RMD5(RO), $BDDAT ;STORE RMD5 AT $BDDAT
9685 034654 042737 177776 001142      BIC      #1COM, $BDDAT
9686 034662 001403                      BEQ      30$ ;BRANCH IF OFFSET IS ZERO
9687 034664 005037 001140      CLR      $GDDAT
9688 034670 104204                      ERROR   204 ;OFFSET NO CLEARED BY EBL
9689 034672                      30$: ;END OF TEST
9690
9691
9692 ;*****
9693 ;*TEST 65 RUN AND GO TEST
9694 ;*****
9695 034672 000004      †$T65: SCOPE
9696 034674 012737 000065 001226      MOV      #65, $TESTN ;;SET TEST NUMBER IN APT MAIL BOX
9697 034702 000240      NOP
9698 034704 012737 000024 001120      MOV      #20, $ICNT ;20 ITERATIONS
9699 034712 112737 000001 001131      MOV      #1, $ERMAX ;ONE ERROR ALLOWED
9700 034720 012737 034734 001122      MOV      #T65, $LPADR ;LOAD LOOP ON TEST ADDRESS
9701 034726 012737 034734 001124      MOV      #T65, $LPERR ;LOAD LOOP ON ERROR ADDRESS
9702 034734
9703 034734 012706 001100      MOV      #STACK, SP ;LOAD THE STACK POINTER
9704 034740 013700 001276      MOV      $BASE, RO ;RO = UNIBUS ADDRESS OF UUT
9705 034744 013701 001456      MOV      TSTQUE, R1 ;R1 = POINTER TO DEVICE
9706 034750 005037 001140      CLR      $GDDAT ;INITIALIZE EXPECTED RESULT
9707 034754 005002                      CLR      R2 ;INITIALIZE FUNCTION CODE
9708 034756 010037 001136      MOV      RO, $BDADR
9709 034762 062737 000024 001136      ADD      #RMMR1, $BDADR
9710 034770
9711
9712 034770 012760 000040 000010      10$: ;CLEAR THE MASSBUS AND ENABLE DEBUG CLOCK
9713 034776 111160 000010      MOV      #CLR, RMCS2(RO) ;CLEAR THE MASSBUS
9714
9715 035002 012760 000001 000024      MOV      #DMD, RMMR1(RO) ;LOAD RMMR1
9716
9717 035010 012760 040001 000024      MOV      #DMD!DBEN, RMMR1(RO) ;LOAD RMMR1
9718
9719 035016 012760 000000 000014      MOV      #0, RMER1(RO) ;LOAD RMER1
9720
9721 035024 012760 000000 000042      MOV      #0, RMER2(RO) ;LOAD RMER2
9722 ;LOAD THE FUNCTION CODE AND VERIFY RUN AND GO FLOP
9723
9724 035032 010203                      MOV      R2, R3 ;ASSEMBLE FUNCTION CODE AND GO
9725 035034 052703 000001                      BIS      #GO, R3
9726 035040 012737 000200 001524      MOV      #200, WATCH ;SET WATCHDOG TIMER VALUE
9727 035046 004777 144454                      JSR      PC, @CLOCK ;START THE CLOCK
9728
9729 035052 010360 000000      15$: MOV      R3, RMCS1(RO) ;LOAD RMCS1
9730 035056

```

```

9731
9732 035056 016037 000024 001142      MOV      RMMR1(RO),SBDDAT      ;STORE RMMR1 AT SBDDAT
9733 035064 042737 137777 001142      BIC      #1CRG,SBDDAT
9734 035072 023737 001140 001142      CMP      $GDDAT,SBDDAT
9735 035100 001411                BEQ      20$                    ;BRANCH IF RUN AND GO FLOP OK
9736 035102 005737 001524                TST      WATCH                  ;TAKE ANOTHER SAMPLE IF CLOCK NOT ZERO
9737 035106 001363                BNE      15$
9738 035110 004777 144414                JSR      PC,2STOP                ;STOP THE CLOCK
9739 035114 010237 001174                MOV      R2,STMPD                ;SAVE FUNCTION CODE FOR MSG
9740 035120 104205                ERROR   20$                    ;RUN AND GO INCORRECT
9741 035122 000416                BR       40$                    ;SKIP REST OF
9742 035124
9743 035124 004777 144400                JSR      PC,2STOP                ;STOP THE CLOCK
9744                ;ADVANCE TO NEXT FUNCTION CODE - EXIT IF DONE
9745 035130 062702 000002                ADD      #2,R2
9746 035134 022702 000076                CMP      #1LF76,R2
9747 035140 103407                BLO      40$                    ;EXIT IF DONE
9748 035142 020227 000050                CMP      R2,#WCD                ;CHANGE EXPECTED RESJLTI IF
9749 035146 103403                BLO      30$                    ;DATA COMMAND
9750 035150 012737 040000 001140                MOV      #RG,$GDDAT
9751 035156 000704                BR       10$                    ;REPEAT TEST
9752
9753 035160                40$:                               ;END OF TEST
9754                ;*****
9755                ;*TEST 66      SET IAE TEST
9756                ;*****
9757                ;*****
9758 035160 000004                †ST66: SCOPE
9759 035162 012737 000066 001226                MOV      #66,$TESTN            ;;SET TEST NUMBER IN APT MAIL BOX
9760
9761 035170 000240                NOP
9762 035172 012737 000024 001120                MOV      #20,$ICNT            ;20 ITERATIONS
9763 035200 112737 000001 001131                MOV      #1,$ERMAX            ;ONE ERROR ALLOWED
9764 035206 012737 035222 001122                MOV      #T66,$LPADR          ;LOAD LOOP ON TEST ADDRESS
9765 035214 012737 035222 001124                MOV      #T66,$LPERR          ;LOAD LOOP ON ERROR ADDRESS
9766 035222
9767 035222 012706 001100                MOV      #STACK,SP            ;LOAD THE STACK POINTER
9768 035226 013700 001276                MOV      $BASE,R0            ;R0 = UNIBUS ADDRESS OF UUT
9769 035232 013701 001456                MOV      TSTQUE,R1           ;R1 = POINTER TO DEVICE
9770 035236 012702 035404                MOV      #100$,R2           ;R2=TABLE POINTER
9771 035242
9772                10$:                               ;SET VOLUME VALID USING SUBROUTINE
9773 035242 004737 060654                JSR      PC,SETVV            ;GO SET VOLUME VALID
9774 035246 000402                BR       20$                    ;BRANCH TO 20$ IF NO ERROR
9775 035250 104000                ERROR   ;RETURN HERE IF ERROR
9776 035252 000453                BR       50$                    ;SKIP REST OF TEST
9777 035254
9778                20$:                               ;LOAD INVALID TRACK, SECTOR AND CYLINDER ADDRESS AND SET FORMAT TO 18
9779
9780 035254 012760 177777 000006                MOV      #-1,RMDA(RO)        ;LOAD RMDA
9781
9782 035262 012760 177777 000034                MOV      #-1,RMDC(RO)        ;LOAD RMDC
9783
9784 035270 012760 000000 000032                MOV      #0,RMOF(RO)         ;LOAD RMOF
9785                ;ENABLE DEBUG CLOCK AND LOAD FUNCTION CODE IN RMCSI WITH GO ON
9786

```

9787	035276	012760	041001	000024	MOV	#DMD!MUR!DBEN,RMMR1(RO)	;LOAD RMMR1
9788	035304	111203			MOV	(R2),R3	
9789	035306	042703	177701		BIC	#+CILF76,R3	
9790	035312	052703	0000C1		BIS	#GO,R3	
9791							
9792	035316	010360	000000		MOV	R3, RMCS1(RO)	;LOAD RMCS1
9793	035322	116204	000001		MOV	1(R2),R4	;GET CLOCK COUNT
9794	035326	042704	177400		BIC	#+C377,R4	
9795	035332				30\$:		
9796					;CLOCK THE COMMAND SEQUENCER		
9797							
9798	035332	012760	141001	000024	MOV	#DMD!MUR!DBEN!DBCK,RMMR1(RO)	;LOAD RMMR1
9799							
9800	035340	012760	041001	000024	MOV	#DMD!MUR!DBEN,RMMR1(RO)	;LOAD RMMR1
9801	035346	005304			DEC	R4	
9802	035350	001370			BNE	30\$	
9803					;SEE IF IAE HAS SET		
9804							
9805	035352	016004	000014		MOV	RMR1(RO),R4	;STORE RMR1 AT R4
9806	035356	042704	175777		BIC	#+CIAE,R4	
9807	035362	001007			BNE	50\$;BRANCH IF IAE SET
9808					;IAE DID NOT SET - TRY ANOTHER FUNCTION CODE		
9809	035364	062702	000002		ADD	#2,R2	
9810	035370	105762	000001		TSTB	1(R2)	
9811	035374	100401			BMI	40\$;BRANCH IF ALL CODES TRIED
9812	035376	000721			BR	10\$	
9813	035400				40\$:		
9814					;CANNOT SET IAE WITH ANY COMBINATION OF ADDRESS AND FUNCTION CODE		
9815	035400	104206			ERROR	20\$	
9816	035402				50\$:		
9817	035402	000411			BR	200\$;JUMP OVER TABLE
9818							
9819	035404				100\$:		
9820					;TABLE OF FUNCTION CODES AND CLOCK COUNTS FOR TEST		
9821							
9822	035404	C30			.BYTE	SEARCH	;SEARCH COMMAND
9823	035405	002			.BYTE	2	
9824							
9825	035406	004			.BYTE	SEEK	;SEEK COMMAND
9826	035407	002			.BYTE	2	
9827							
9828	035410	062			.BYTE	WH	;WRITE HEADER COMMAND
9829	035411	002			.BYTE	2	
9830							
9831	035412	052			.BYTE	WCH	;WRITE CHECK HEADER COMMAND
9832	035413	002			.BYTE	2	
9833							
9834	035414	072			.BYTE	RH	;READ HEADER COMMAND
9835	035415	002			.BYTE	2	
9836							
9837	035416	060			.BYTE	WD	;WRITE DATA COMMAND
9838	035417	002			.BYTE	2	
9839							
9840	035420	050			.BYTE	WCD	;WRITE CHECK DATA COMMAND
9841	035421	002			.BYTE	2	
9842							

```

9843 035422 070 .BYTE RD ;READ DATA COMMAND
9844 035423 002 .BYTE 2
9845
9846 035424 000 .BYTE ;END OF TABLE
9847 035425 377 .BYTE -1
9848
9849 035426 200$ ;END OF TEST
9850
9851 ;*****
9852 ;*TEST 67 SEARCH, SEEK, READ WRITE TEST
9853 ;*****
9854
9855 035426 000004 †ST67: SCOPE
9856 035430 012737 000067 001226 MOV #67,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
9857
9858 035436 070240 NOP
9859 035440 012737 000024 001120 MOV #20,$ICNT ;20 ITERATIONS
9860 035446 112737 000001 001131 MOVB #1,$ERMAX ;ONE ERROR ALLOWED
9861 035454 012737 035470 001122 MOV #T67,$LPADR ;LOAD LOOP ON TEST ADDRESS
9862 035462 012737 035470 001124 MOV #T67,$LPERR ;LOAD LOOP ON ERROR ADDRESS
9863 035470
9864 035470 012706 001100 T67: MCV #STACK,SP ;LOAD THE STACK POINTER
9865 035474 013700 001276 MOV $BASE,R0 ;R0 = UNIBUS ADDRESS OF JUT
9866 035500 013701 001456 MOV TSTQUE,R1 ;R1 = POINTER TO DEVICE
9867 035504 005002 CLR R2 ;INITIALIZE FUNCTION CODE
9868 035506
9869
9870 035506 004737 060654 10$: ;SET VOLUME VALID USING SUBROUTINE
9871 035512 000402 JSP PC,SETVV ;GO SET VOLUME VALID
9872 035514 104000 BR 20$ ;BRANCH TO 20$ IF NO ERROR
9873 035516 000472 ERROR ;RETURN HERE IF ERROR
9874 035520 BR 50$
9875
9876 20$: ;LOAD INVALID TRACK, SECTOR AND CYLINDER ADDRESS AND SET FORMAT
9877 ;TO 18 BIT MODE
9878 035520 012760 177777 000006 MOV #-1,RMDA(R0) ;LOAD RMDA
9879
9880 035526 012760 177777 000034 MOV #-1,RMDC(R0) ;LOAD RMDC
9881
9882 035534 012760 000000 000032 MOV #0,RMOF(R0) ;LOAD RMOF
9883 ;ENABLE DEBUG CLOCK AND LOAD FUNCTION CODE IN RMCS1 WITH GO ON
9884
9885 035542 012760 041001 000024 MOV #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
9886 035550 010203 MOV R2,R3 ;ASSEMBLE CODE AND GO
9887 035552 052703 000001 BIS #GO,R3
9888
9889 035556 010360 000000 MOV R3,RMCS1(R0) ;LOAD RMCS1
9890 ;CLOCK THE COMMAND SEQUENCER TO SET PULSE
9891 035562 012704 000002 MOV #2,R4
9892
9893 30$:
9894 035566 012760 141001 000024 MOV #DMD!MUR!DBEN!DBCK,RMMR1(R0) ;LOAD RMMR1
9895
9896 035574 012760 041001 000024 MOV #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
9897 035602 005304 DEC R4
9898 035604 001370 BNE 30$

```


M16

CZRMJBO RM03/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 207
T67 SEARCH, SEEK, READ WRITE TEST

SEQ 0207

```

9899          ;VERIFY IAE ACCORDING TO FUNCTION CODE TABLE
9900
9901 035606 016037 000014 001142      MOV      RMER1(R0), $BDDAT      ;STORE RMER1 AT $BDDAT
9902 035614 042737 175777 001142      BIC      #1, CIAE, $BDDAT
9903 035622 016237 066142 001140      MOV      FNCOTB(R2), $GDDAT    ;ASSEMBLE EXPECTED IAE
9904 035630 042737 175777 001140      BIC      #1, CIAE, $GDDAT
9905 035636 023737 001140 001142      CMP      $GDDAT, $BDDAT
9906 035644 001411                      BEQ      40$                    ;BRANCH IF IAE OK
9907 035646 010037 001136                      MOV      R0, $BDDADR          ;SET UP ERROR MSG
9908 035652 062737 000014 001136      ADD      #RMER1, $BDDADR
9909 035660 010237 001174                      MOV      R2, $TMPD
9910 035664 104207                      ERROR   20$                    ;IAE IS INCORRECT
9911 035666 000406                      BR       50$                    ;SKIP REST OF TEST
9912 035670
9913
9914 035670 062702 000002                      ;ADVANCE TO NEXT FUNCTIONCODE - EXIT IF DONE
9915 035674 023702 000076                      ADD      #2, R2
9916 035700 103401                      CMP      ILF7S, R2
9917 035702 000701                      BLO     50$
9918 035704                      BR       10$
9919
9920
9921          ;*****
9922          ;*TEST 70      INVALID SECTOR/TRACK TEST
9923          ;*****
9924 035704 000004      †ST70: SCOPE
9925 035706 012737 000070 001226      MOV      #70, $TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
9926
9927 035714 000240      NOP
9928 035716 012737 000024 001120      MOV      #20, $ICNT          ;20 ITERATIONS
9929 035724 112737 000001 001131      MOVB    #1, $ERRMAX          ;ONE ERROR ALLOWED
9930 035732 012737 035746 001122      MOV      #T70, $LPAOR        ;LOAD LOOP ON TEST ADDRESS
9931 035740 012737 035746 001124      MOV      #T70, $LPERR        ;LOAD LOOP ON ERROR ADDRESS
9932 035746
9933 035746 012706 001100      T70:   MOV      #STACK, SP          ;LOAD THE STACK POINTER
9934 035752 013700 001276      MOV      $BASE, R0           ;R0 = UNIBUS ADDRESS OF UUT
9935 035756 013701 001456      MOV      †STQUE, R1          ;R1 = POINTER TO DEVICE
9936 035762 012702 036142      MOV      #100$, R2           ;INITIALIZE TABLE POINTER
9937
9938 035766
9939
9940 035766 004737 060654      10$:   ;SET VOLUME VALID USING SUBROUTINE
9941 035772 000402      JSR      PC, SETVV          ;GO SET VOLUME VALID
9942 035774 104000      BR       20$                ;BRANCH TO 20$ IF NO ERROR
9943 035776 000460      ERROR   ;RETURN HERE IF ERROR
9944 036000      BR       50$                ;SKIP REST OF TEST
9945
9946          20$:   ;CLEAR DESIRED CYLINDER, LOAD SECTOR/TRACK ADDRESS FROM TABLE, AND SET
9947          ;18 BIT FORMAT
9948 036000 012760 000000 000034      MOV      #0, RMDC(R0)        ;LOAD RMDC
9949
9950 036006 012760 000000 000032      MOV      #0, RMOF(R0)        ;LOAD RMOF
9951
9952 036014 011260 000006      MOV      (R2), RMDA(R0)      ;LOAD RMDA
9953          ;EXECUTE A SEARCH COMMAND TO WHERE "SET PULSE" IS ACTIVE
9954

```

```

9955 036020 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
9956
9957 036026 012760 000031 000000      MOV      #SEARCH!G0,RMCSI(RO) ;LOAD RMCSI
9958 036034 012703 000002
9959 036040      30$:
9960
9961 036040 012760 141001 000024      MOV      #DMD!MUR!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1
9962
9963 036046 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
9964 036054 005303      DEC      R3
9965 036056 001370      BNE      30$ ;ISSUE 2 CLOCKS
9966 ;VERIFY IAE IS SET
9967
9968 036060 016037 000014 001142      MOV      RMER1(RO),SBD0AT ;STORE RMER! AT SBD0AT
9969 036066 042737 175777 001142      BIC      #1CIAE,SBD0AT
9970 036074 001014      BNE      40$ ;BRANCH IF IAE IS ON
9971 036076 012737 002000 001140      MOV      #IAE,SBD0AT ;SETUP ERROR MESSAGE
9972 036104 010037 001136      MOV      RO,SBDADR
9973 036110 062737 000014 001136      ADD      #RMER1,SBDADR
9974 036116 011237 001174      MOV      (R2),S1MPO
9975 036122 104210      ERROR   210 ;IAE NOT SET BY RMDR ADDRESS
9976 036124 000405      BR       50$
9977 036126
9978      40$:
9979 ;ADVANCE TO NEXT ENTRY IN TABLE - EXIT IF DONE
9980 036126 062702 000002      ADD      #2,R2
9981 036132 005712      TST     (R2)
9982 036134 001401      BEQ     50$ ;EXIT IF END OF TABLE
9983 036136 000713      BR      10$ ;REPEAT TEST
9984 036140 000414      BR      200$ ;JUMP OVER TABLE
9985 036142
9986
9987      50$:
9988 ;TABLE OF SECTOR AND TRACK ADDRESSES FOR TEST
9989 .BYTE 0 ;SECTOR ADDRESS = 0
9990 .BYTE 1800000101 ;TRACK ADDRESS = 5
9991 .BYTE 0 ;SECTOR = 0
9992 .BYTE 1800000110 ;TRACK = 6
9993 .BYTE 0 ;SECTOR = 0
9994 .BYTE 1800001000 ;TRACK = 8
9995 .BYTE 0 ;SECTOR = 0
9996 .BYTE 1800010000 ;TRACK = 16
9997 .BYTE 0 ;SECTOR = 0
9998 .BYTE 1800100000 ;TRACK = 32
9999 .BYTE 0 ;SECTOR = 0
10000 .BYTE 1801000000 ;TRACK = 64
10001 .BYTE 0 ;SECTOR = 0
10002 .BYTE 1801000000 ;TRACK = 128
10003 .BYTE 0 ;SECTOR = 0
10004 .BYTE 1801000000 ;TRACK = 128
10005 .BYTE 0 ;SECTOR = 0
10006 .BYTE 1800011110 ;SECTOR = 31
10007 .BYTE 0 ;TRACK = 0
10008
10009
10010

```

CO1

CZRMJ80 RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 209
170 INVALID SECTOR/TRACK TEST

SEQ 0209

10011											
10012	036162	040				.BYTE	#B00100000			;SECTOR = 32	
10013	036163	000				.BYTE	0			;TRACK = 0	
10014											
10015	036164	100				.BYTE	#B01000000			;SECTOR = 64	
10016	036165	000				.BYTE	0			;TRACK = 0	
10017											
10018	036166	200				.BYTE	#B10000000			;SECTOR = 128	
10019	036167	000				.BYTE	0			;TRACK = 0	
10020											
10021	036170	000				.BYTE	0			;END OF TABLE	
10022	036171	000				.BYTE	0				
10023											
10024	036172							200\$:		;END OF TEST	
10025											
10026											
10027											
10028											
10029											
10030	036172	000004									
10031	036174	012737	000071	001226		ST71:	SCOPE				
10032						MOV	#71,\$TESTN			;SET TEST NUMBER IN APT MAIL BOX	
10033	036202	000240				NOP					
10034	036204	012737	000024	001120		MOV	#20,\$ICNT			;20 ITERATIONS	
10035	036212	112737	000001	001131		MOV#	#1,\$ERMAX			;ONE ERROR ALLOWED	
10036	036220	012737	036234	001122		MOV	#T71,\$LPADR			;LOAD LOOP ON TEST ADDRESS	
10037	036226	012737	036234	001124		MOV	#T71,\$LPERR			;LOAD LOOP ON ERROR ADDRESS	
10038	036234							T71:			
10039	036234	012706	001100			MOV	#STACK,SP			;LOAD THE STACK POINTER	
10040	036240	013700	001276			MOV	\$BASE,R0			;R0 = UNIBUS ADDRESS OF UUT	
10041	036244	013701	001456			MOV	TSTQUE,R1			;R1 = POINTER TO DEVICE	
10042	036290	012702	036422			MOV	#100\$,R2			;INITIALIZE TABLE POINTER	
10043	036254										
10044											
10045	036254	004737	060654								
10046	036260	000402				JSR	PC,SETVV			;GO SET VOLUME VALID	
10047	036262	104000				BR	20\$;BRANCH TO 20\$ IF NO ERROR	
10048	036264	000455				ERROR				;RETURN HERE IF ERROR	
10049	036266					BR	50\$;SKIP IF ERROR	
10050											
10051											
10052	036266	012760	000000	000006		MOV	#0,RMDA(R0)			;LOAD RMDA	
10053											
10054	036274	011260	000034			MOV	(R2),RMDC(R0)			;LOAD RMDC	
10055											
10056											
10057	036300	012760	041001	000024		MOV	#DMD!MUR!DBEN,RMMR1(R0)			;LOAD RMMR1	
10058											
10059	036306	012760	000031	000000		MOV	#SEARCH!GO,RMCS1(R0)			;LOAD RMCS1	
10060	036314	012703	000002			MOV	#2,R3				
10061	036320										
10062											
10063	036320	012760	141001	000024		MOV	#DMD!MUR!DBEN!DBCK,RMMR1(R0)			;LOAD RMMR1	
10064											
10065	036326	012760	041001	000024		MOV	#DMD!MUR!DBEN,RMMR1(R0)			;LOAD RMMR1	
10066	036334	005303				DEC	R3				

```

10067 036336 001370          BNE      30$          ;ISSUE 2 CLOCKS
10068          ;VERIFY IAE IS SET
10069
10070 036340 016037 000014 001142      MOV     RMER1(RO), $BDDAT      ;STORE RMER1 AT $BDDAT
10071 036346 042737 175777 001142      BIC     #1,IAE,$BDDAT
10072 036354 001014          BNE     40$          ;BRANCH IF IAE IS SET
10073 036356 012737 002000 001140      MOV     #IAE,$GDDAT          ;SETUP ERROR MESSAGE
10074 036364 010037 001136          MOV     RO,$DADR
10075 036370 062737 000014 001136      ADD     #RMER1,$DADR
10076 036376 011237 001174          MOV     (R2),$TMPD
10077 036402 104211          ERROR   211          ;IAE NOT SET BY RMDC
10078 036404 000405          BR      50$
10079 036406
10080          40$:
10081 036406 062702 000002          ;ADVANCE TABBLE POINTER - EXIT IF DONE
10082 036412 005712          ADD     #2,R2
10083 036414 001401          TST     (R2)
10084 036416 000716          BEQ     50$
10085 036420 000405          BR      10$
10086
10087 036422          50$:
10088          BR      200$          ;JUMP OVER TABLE
10089
10090          100$:
10091 036422 001467          ;TABLE OF CYLINER ADDRESSES USED DURING TEST
10092          .WORD   #B1100110111          ;CYLINDER 823
10093 036424 001470          .WORD   #B1100111000          ;CYLINDER 824
10094
10095 036426 001500          .WORD   #B1101000000          ;CYLINDER 832
10096
10097 036430 001600          .WORD   #B1110000000          ;CYLINDER 896
10098
10099 036432 000000          .WORD   0                    ;END OF TABLE
10100
10101 036434          200$:
10102          ;END OF TEST
10103
10104          ;*****
10105          ;*TEST 72      SET AOE TEST
10106          ;*****
10107 036434 000004          #ST72: SCOPE
10108 036436 012737 000072 001226      MOV     #72,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
10109
10110 036444 000240          NOP
10111 036446 012737 000024 001120      MOV     #20,$ICNT          ;20 ITERATIONS
10112 036454 112737 000001 001131      MOV     #1,$ERMAX          ;ONE ERROR ALLOWED
10113 036462 012737 036476 001122      MOV     #T72,$LPADR        ;LOAD LOOP ON TEST ADDRESS
10114 036470 012737 036476 001124      MOV     #T72,$LPERR        ;LOAD LOOP ON ERROR ADDRESS
10115 036476
10116 036476 012706 001100          T72:  MOV     #STACK,SP          ;LOAD THE STACK POINTER
10117 036502 013700 001276          MOV     $BASE,RO          ;RO = UNIBUS ADDRESS OF UUT
10118 036506 013701 001456          MOV     TSTQUE,R1         ;R1 = POINTER TO DEVICE
10119
10120 036512 004737 060654          ;SET VOLUME VALID USING SUBROUTINE
10121 036516 000402          JSR     PC,SETVV          ;GO SET VOLUME VALID
10122 036520 104000          BR      10$          ;BRANCH TO 10$ IF NO ERROR
          ERROR          ;RETURN HERE IF ERROR

```

E01

CZRMJBO RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 211
T72 SET AOE TEST

SEQ 0211

```

10123 036522 000465          BR      30$          ;SKIP TEST IF ERROR
10124 036524          10$:
;ENABLE DEBUG CLOCK AND LOAD LAST SECTOR ADDRESS, MEMORY ADDRESS AND
10125          ;WORD COUNT, THEN LOAD WRITE DATA COMMAND WITH GO SET
10126
10127
10128 036524 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
10129
10130 036532 012760 000000 000032      MOV      #0,RMOF(RO) ;LOAD RMOF
10131
10132 036540 012760 002035 000006      MOV      #002035,RMDA(RO) ;LOAD RMDA
10133
10134 036546 012760 001466 000034      MOV      #001466,RMDC(RO) ;LOAD RMDC
10135
10136 036554 012760 107030 000004      MOV      #BUFFER,RMBA(RO) ;LOAD RMBA
10137
10138 036562 012760 177244 000002      MOV      #1C<2*256>+1,RMWC(RO) ;LOAD RMWC
10139
10140 036570 012760 000061 000000      MOV      #WD!GO,RMCS1(RO) ;LOAD RMCS1
10141 036576 012702 000014 000000      MOV      #14,R2 ;R2 = CLOCK COUNT
;CLOCK COUNT 2-05 3-04 14-60...
10142          ;CLOCK THE COMMAND SEQUENCER TO GENERATE SET PULSE
10143          20$:
10144 036602          MOV      #DMD!MUR!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1
10145
10146 036602 012760 141001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
10147
10148 036610 012760 041001 000024      DEC      R2
10149 036616 005302          BNE     20$ ;ISSUE 2 CLOCKS
10150 036620 001370          ;FORCE EBL TO GET TO OVERFLOW ADDRESS
10151
10152
10153 036622 012760 061001 000024      MOV      #DMD!MUR!DBEN!DEBL,RMMR1(RO) ;LOAD RMMR1
10154
10155 036630 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
10156          ;VERIFY THAT ADDRESS OVERFLOW ERROR IS SET
10157
10158 036636 016037 000014 001142      MOV      RMER1(RO), $BDDAT ;STORE RMER1 AT $BDDAT
10159 036644 042737 176777 001142      BIC     #1CAOE,$BDDAT
10160 036652 001011          BNE     30$ ;BRANCH IF AOE IS SET
10161 036654 010037 001136      MOV      RO,$BDAOR ;SETUP ERROR MESSAGE
10162 036660 062737 000014 001136      ADD     #RMER1,$BDAOR
10163 036666 012737 001000 001140      MOV     #AOE,$GDDAT
10164 036674 104212          ERROR  212 ;CANT SET AOE
10165 036676          30$: ;END OF TEST
10166
10167          ;*****
10168          ;*TEST 73 SET RMR TEST
10169
10170          ;*****
10171 036676 000004          †ST73: SCOPE
10172 036700 012737 000073 001226      MOV     #73,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
10173
10174 036706 000240          NOP
10175 036710 012737 000024 001120      MOV     #20,$ICNT ;20 ITERATIONS
10176 036716 112737 000001 001131      MOV     #1,$ERMAX ;ONE ERROR ALLOWED
10177 036724 012737 036740 001122      MOV     #T73,$LPADR ;LOAD LOOP ON TEST ADDRESS
10178 036732 012737 036740 001124      MOV     #T73,$LPERR ;LOAD LOOP ON ERROR ADDRESS

```

```

10179 036740
10180 036740 012706 001100
10181 036744 013700 001276
10182 036750 013701 001456
10183 036754 010037 001136
10184 036760 062737 000014 001136
10185 036766 012702 037142
10186 036772
10187
10188 036772 012760 000040 000010
10189 037000 111160 000010
10190
10191 037004 012760 000001 000024
10192
10193 037012 012760 041001 000024
10194
10195 037020 012760 000000 000014
10196
10197 037026 012760 000000 000042
10198
10199
10200 037034 012760 000001 000000
10201 037042 011203
10202 037044 060003
10203 037046 012713 041001
10204
10205
10206 037052 016037 000014 001142
10207 037060 042737 177773 001142
10208 037066 016237 000002 001140
10209 037074 023737 001140 001142
10210 037102 001411
10211 037104 011237 001174
10212 037110 032762 000004 000002
10213 037116 001002
10214 037120 104213
10215 037122 000401
10216 037124 104214
10217
10218 037126
10219
10220 037126 062702 000004
10221 037132 005712
10222 037134 100401
10223 037136 000715
10224 037140 000410
10225
10226 037142
10227
10228
10229
10230 037142 000016
10231 037144 000000
10232
10233 037146 000024
10234 037150 000000

```

```

T73:
MOV #STACK, SP ;LOAD THE STACK POINTER
MOV $BASE, R0 ;R0 = UNIBUS ADDRESS OF UUT
MOV TSTQUE, R1 ;R1 = POINTER TO DEVICE
MOV R0, $BDADR ;SETUP REGISTER ADDRESS FOR MSG
ADD #RMR1, $BDADR
MOV #100$, R2 ;INITIALIZE TABLE POINTER

10$:
;CLEAR THE DEVICE AND ENABLE DEBUG CLOCK
MOV #CLR, RMCS2(R0) ;CLEAR THE MASSBUS
MOV (R1), RMCS2(R0) ;SELECT UNIT

MOV #DMD, RMMR1(R0) ;LOAD RMMR1
MOV #DMD!MUR!DBEN, RMMR1(R0) ;LOAD RMMR1
MOV #0, RMR1(R0) ;LOAD RMR1
MOV #0, RMR2(R0) ;LOAD RMR2
;SET GO THEN WRITE THE REGISTER SPECIFIED BY THE TABLE
MOV #GO, RMCS1(R0) ;LOAD RMCS1
MOV (R2), R3 ;GENERATE REGISTER ADDRESS
ADD R0, R3
MOV #DMD!MUR!DBEN, (R3) ;WRITE THE REGISTER
;VERIFY RMR ACCORDING TO TABLE
MOV RMR1(R0), $BDADR ;STORE RMR1 AT $BDADR
BIC #CRMR, $BDADR
MOV 2(R2), $GDDAT ;GET EXPECTED RESULT FROM TABLE
CMP $GDDAT, $BDADR
BEQ 30$ ;BRANCH IF RMR IS OK
MOV (R2), $TMPD ;SAVE TEST REGISTER
BIT #RMR, 2(R2)
BNE 27$ ;BRANCH IF ERROR SHOULD BE ONE
ERROR 213 ;RMR SET WHEN WRITING REGISTER
BR 30$
20$: ERROR 214 ;RMR DID NOT SET

30$:
;ADVANCE TABLE POINTER, EXIT IF DONE
ADD #4, R2
TST (R2)
BMI 40$ ;EXIT IF ENTRY NEGATIVE
BR 10$
40$: BR 200$ ;JUMP OVER TABLE

100$:
;TABLE OF REGISTER ADDRESSES AND RMR VALUES
.word #RMAS ;ATTENTION SUMMARY REG
.word ;RMR = 0
.word #RMMR1 ;MAINTENANCE REG
.word ;RMR = 0

```

```

10235
10236 037152 000006 .WORD #RMDA ;DISK ADDRESS REG
10237 037154 000004 .WORD RMR ;RMR = 1
10238
10239 037156 177777 .WORD #-1 ;END OF TABLE
10240 037160 000000 .WORD
10241
10242 037162 200$: ;END OF TEST
10243
10244 ;*****
10245 ;*TEST 74 PGM STATUS CHECK
10246 ;*****
10247 ;*****
10248 037162 000004 †ST74: SCOPE
10249 037164 012737 000074 001226 MOV #74,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
10250
10251 037172 000240 NOP
10252 037174 012737 000024 001120 MOV #20,$ICNT ;20 ITERATIONS
10253 037202 112737 000001 001131 MOVB #1,$ERMAX ;ONE ERROR ALLOWED
10254 037210 012737 037224 001122 MOV #T74,$LPADR ;LOAD LOOP ON TEST ADDRESS
10255 037216 012737 037224 001124 MOV #T74,$LPERR ;LOAD LOOP ON ERROR ADDRESS
10256 037224
10257 037224 012706 001100 T74: MOV #STACK,SP ;LOAD THE STACK POINTER
10258 037230 013700 001276 MOV $BASE,RO ;RO = UNIBUS ADDRESS OF UUT
10259 037234 013701 001456 MOV TSTQUE,R1 ;R1 = POINTER TO DEVICE
10260 ;CLEAR AND READ DRIVE TYPE AND DRIVE STATUS
10261 037240 012760 000040 000010 MOV #CLR,RMCS2(RO) ;CLEAR THE MASSBUS
10262 037246 111160 000010 MOVB (R1),RMCS2(RO) ;SELECT UNIT
10263
10264 037252 016037 000026 001174 MOV RMDT(RO),$TMPD ;STORE RMDT AT $TMPD
10265
10266 037260 016037 000012 001142 MOV RMDS(RO),$BDDAT ;STORE RMDS AT $BDDAT
10267 ;OMIT TEST IF DRQ IS ON - ELSE VERIFY THAT PGM IS OFF
10268 037266 032737 004000 001174 BIT #DRQ,$TMPD
10269 037274 001014 BNE 10$ ;BRANCH IF DRQ IS ON
10270 037276 042737 176777 001142 BIC #ICPGM,$BDDAT
10271 037304 001410 BEQ 10$ ;BRANCH IF PGM IS OFF
10272 037306 005037 001140 CLR $GDDAT
10273 037312 010037 001136 MOV RO,$BDADR
10274 037316 062737 000012 001134 ADD #RMDS,$GDADR
10275 037324 104215 ERROR 21$ ;DRQ IS OFF BUT PGM IS ON
10276 037326 10$: ;END OF TEST
10277
10278 ;*****
10279 ;*TEST 75 DPR STATUS CHECK
10280 ;*****
10281 ;*****
10282 037326 000004 †ST75: SCOPE
10283 037330 012737 000075 001226 MOV #75,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
10284
10285 037336 000240 NOP
10286 037340 012737 000024 001120 MOV #20,$ICNT ;20 ITERATIONS
10287 037346 112737 000001 001131 MOVB #1,$ERMAX ;ONE ERROR ALLOWED
10288 037354 012737 037370 001122 MOV #T75,$LPADR ;LOAD LOOP ON TEST ADDRESS
10289 037362 012737 037370 001124 MOV #T75,$LPERR ;LOAD LOOP ON ERROR ADDRESS
10290 037370
T75:

```

HO1

CZPMJBO RM03/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 214
T75 DPR STATUS CHECK

SEQ 0214

```

10291 037370 012706 001100      MOV      #STACK,SP      ;LOAD THE STACK POINTER
10292 037374 013700 001276      MOV      $BASE,R0      ;R0 = UNIBUS ADDRESS OF UUT
10293 037400 013701 001456      MOV      TSTQUE,R1     ;R1 = POINTER TO DEVICE
10294                                     ;CLEAR AND VERIFY THAT DVA IS SET
10295 037404 012760 000040 000010  MOV      #CLR, RMCS2(R0) ;CLEAR THE MASSBUS
10296 037412 111160 000010      MOVVB   (R1),RMCS2(R0) ;SELECT UNIT
10297
10298 037416 011037 000000 001142  MOV      RMCS1(R0), $BDDAT ;STORE RMCS1 AT $BDDAT
10299 037424 042737 173777 001142  BIC     #1CDVA, $BDDAT
10300 037432 001006                                     ;BRANCH IF DVA IS ON
10301 037434 012737 004000 001140  BNE     10$
10302 037442 010037 001136      MOV      #DVA, $GDADR
10303 037446 104216      MOV      R0, $BDADR
10304 037450      ERROR   216          ;DVA NOT SET
10305
10306      10$:
10307 037450 016037 000012 001142  ;VERIFY THAT DPR IS SET
10308 037456 042737 173777 001142  MOV      RMDS(R0), $BDDAT ;STORE RMDS AT $BDDAT
10309 037464 001011      BIC     #1CDPR, $BDDAT
10310 037466 012737 000400 001140  BNE     20$
10311 037474 010037 001136      MOV      #DPR, $GDADR
10312 037500 062737 000012 001136  MOV      R0, $BDADR
10313 037506 104217      ADD     #RMDS, $BDADR
10314      ERROR   217          ;DPR NOT SET
10315 037510      20$:
10316                                     ;END OF TEST
10317
10318      ;*****
10319      ;*TEST 76      PORT REQUEST TEST, PART 1
10320      ;*****
10321 037510 000004      ;ST76: SCOPE
10322 037512 012737 000076 001226  MOV      #76, $TESTN    ;;SET TEST NUMBER IN APT MAIL BOX
10323
10324 037520 000240      NOP
10325 037522 012737 000024 001120  MOV      #20, $ICNT    ;20 ITERATIONS
10326 037530 112737 000001 001131  MOVVB   #1, $ERMAX     ;ONE ERROR ALLOWED
10327 037536 012737 037552 001122  MOV      #T76, $LPADR  ;LOAD LOOP ON TEST ADDRESS
10328 037544 012737 037552 001124  MOV      #T76, $LPERR  ;LOAD LOOP ON ERROR ADDRESS
10329 037552
10330 037552 012706 001100      T76:
10331 037556 013700 001276      MOV      #STACK,SP    ;LOAD THE STACK POINTER
10332 037562 013701 001456      MOV      $BASE,R0    ;R0 = UNIBUS ADDRESS OF UUT
10333                                     ;SET VOLUME VALID USING SUBROUTINE
10334 037566 004737 060654      JSR     PC, SETVV     ;GO SET VOLUME VALID
10335 037572 000402      BR     10$          ;BRANCH TO 10$ IF NO ERROR
10336 037574 104000      ERROR   10$          ;RETURN HERE IF ERROR
10337 037576 000434      BR     20$
10338 037600
10339      10$:
10340      ;EXECUTE A RELEASE TO RESET REQUEST FLOP
10341 037600 012760 000013 000000  MOV      #RELEASE!GO, RMCS1(R0) ;LOAD RMCS1
10342      ;READ RMMR2 AND SKIP TEST IF REQUEST FLOPS ARENT RESET
10343
10344 037606 016002 000040      MOV      RMMR2(R0), R2 ;STORE RMMR2 AT R2
10345 037612 042702 037777      BIC     #1C(RQA!RQB), R2
10346 037616 001024      BNE     20$

```


I01

CZRMJBO RMG3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 215
T76 PORT REQUEST TEST, PART 1

SEQ 0215

```

10347      ;READ RMCS1 TO SET REQUEST FLOP
10348
10349 037620 016002 000000      MOV      RMCS1(R0),R2      ;STORE RMCS1 AT R2
10350      ;VERIFY THAT REQUEST FLOP IS SET IF PGM IS SET
10351
10352 037624 016005 000012      MOV      RMDS(R0),R5      ;STORE RMDS AT R5
10353 037630 032705 001000      BIT      #PGM,R5          ;SEE IF PGM IS SET
10354 037634 001415                BEQ      20$              ;DONT TEST REQUEST IF PGM IS ZERO
10355
10356 037636 016037 000040 001142      MOV      RMMR2(R0),SBDDAT ;STORE RMMR2 AT SBDDAT
10357 037644 042737 037777 001142      BIC      #1C(RQA!RQB),SBDDAT
10358 037652 001005                BNE      20$              ;BRANCH IF REQUEST IS SET
10359 037654 010037 001136      MOV      R0,$BDADR
10360 037660 062737 000040 001136      ADD      #RMMR2,$BDADR
10361 037666 104220                ERROR    220              ;CANT SET PEQUEST BY RMCS1
10362 037670
10363
10364      ;*****
10365      ;*TEST 77      PORT REQUEST TEST, PART 2
10366
10367      ;*****
10368 037670 000004      TST77: SCOPE
10369 037672 012737 000077 001226      MOV      #77,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
10370
10371 037700 000240      NOP
10372 037702 012737 000024 001120      MOV      #20,$ICNT      ;20 ITERATIONS
10373 037710 112737 000001 001131      MOV      #1,$ERMAX      ;ONE ERROR ALLOWED
10374 037716 012737 037732 001122      MOV      #T77,$LPADR    ;LOAD LOOP ON TEST ADDRESS
10375 037724 012737 037732 001124      MOV      #T77,$LPERR    ;LOAD LOOP ON ERROR ADDRESS
10376 037732
10377 037732 012706 001100      T77:      MOV      #STACK,SP      ;LOAD THE STACK POINTER
10378 037736 013700 001276      MOV      $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
10379 037742 013701 001456      MOV      TSTQUE,R1       ;R1 = POINTER TO DEVICE
10380
10381 037746 004737 060654      ;SET VOLUME VALID USING SLROUTINE
10382 037752 000402      JSR      PC,SETVV        ;GO SET VOLUME VALID
10383 037754 104000      BR       10$             ;BRANCH TO 10$ IF NO ERROR
10384 037756 000435      ERROR    20$             ;RETURN HERE IF ERROR
10385 037760
10386
10387      10$:
10388      ;EXECUTE A RELEASE TO RESET REQUEST FLOP
10389
10390 037760 012760 000013 000000      MOV      #RELEASE!GO,RMCS1(R0) ;LOAD RMCS1
10391      ;READ RMMR2 AND SKIP TEST IF REQUEST FLOPS ARENT RESET
10392
10393 037766 016002 000040      MOV      RMMR2(R0),R2      ;STORE RMMR2 AT R2
10394 037772 042702 037777      BIC      #1C(RQA!RQB),R2
10395 037776 001025                BNE      20$
10396
10397      ;WRITE THE ATTENTION SUMMARY REGISTER TO SET REQUEST FLOP
10398
10399 040000 012760 177777 000016      MOV      #-1,RMCS1(R0)    ;LOAD RMCS1
10400      ;VERIFY THAT REQUEST FLOP IS SET IF PGM IS SET
10401
10402 040006 016005 000012      MOV      RMDS(R0),R5      ;STORE RMDS AT R5
10403 040012 032705 001000      BIT      #PGM,R5          ;SEE IF PGM IS SET
10404 040016 001415                BEQ      20$              ;DONT TEST REQUEST IF PGM IS ZERO

```

J01

CZRMJ80 RM03/2 DSKLS DIAG
CZRMJ6.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 216
T77 PORT REQUEST TEST, PART 2

SEQ 0216

```

10403 040020 016037 000040 001142      MOV      RMMR2(R0), $BDDAT      ;STORE RMMR2 AT $BDDAT
10404 040026 042737 037777 001142      BIC      #1C<RQA!RQB>, $BDDAT
10405 040034 001006                BNE      20$
10406 040036 010037 001136      MOV      R0, $BDADR
10407 040041 062737 000040 001136      ADD      #RMMR2, $BDADR
10408 040030 104221                ERROR    221                    ;CANT SEE REQUEST BY RMA$
10409 040052                20$:
10410
10411      ;*****
10412      ;*TEST 100      PORT REQUEST TEST, PART 3
10413
10414      ;*****
10415 040052 000001                T$T100: SCOPE
10416 040054 012737 000100 001226      MOV      #100, $TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
10417
10418 040062 000240                NOP
10419 040064 012737 000024 001120      MOV      #20, $ICNT           ;20 ITERATIONS
10420 040072 112737 000001 001131      MOV      #1, $ERMAX           ;ONE ERROR ALLOWED
10421 040100 012737 040114 001122      MOV      #T100, $SLPADR       ;LOAD LOOP ON TEST ADDRESS
10422 040106 012737 040114 001124      MOV      #T100, $SLPERA       ;LOAD LOOP ON ERROR ADDRESS
10423 040114
10424 040114 012706 001100      MOV      #STACK, SP           ;LOAD THE STACK POINTER
10425 040120 013700 001276      MOV      $BASE, R0            ;R0 = UNIBUS ADDRESS OF UUT
10426 040124 013701 001456      MOV      TSTQUE, R1           ;R1 = POINTER TO DEVICE
10427      ;SET VOLUME VALID USING SUBROUTINE
10428 040130 004737 060654      JSR      PC, SETVV            ;GO SET VOLUME VALID
10429 040134 000402                BR       10$                  ;BRANCH TO 10$ IF NO ERROR
10430 040136 104000                ERROR    10$                  ;RETURN HERE IF ERROR
10431 040140 000435                BR       20$
10432 040142
10433      10$:
10434      ;EXECUTE A RELEASE COMMAND TO RESET REQUEST FLOP
10435 040142 012760 000013 000000      MOV      #RELEASE!GO, RMCS1(R0) ;LOAD RMCS1
10436      ;READ RMMR2 AND SKIP TEST IF REQUEST FLOPS ARENT RESET
10437
10438 040150 016002 000040      MOV      RMMR2(R0), R2        ;STORE RMMR2 AT R2
10439 040154 042702 037777      BIC      #1C<RQA!RQB>, R2
10440 040160 001025                BNE      20$
10441      ;WRITE RMDA TO SET REQUEST FLOP
10442
10443 040162 012760 000000 000006      MOV      #0, RMDA(R0)         ;LOAD RMDA
10444      ;VERIFY THAT REQUEST FLOP IS SET IF PGM IS SET
10445
10446 040170 016005 000012      MOV      RMD5(R0), R5         ;STORE RMD5 AT R5
10447 040174 032705 001000      BIT      #PGM, R5             ;SEE IF PGM IS SET
10448 040200 001415                BEQ      20$                  ;DONT TEST REQUEST IF PGM IS ZERO
10449
10450 040202 016037 000040 001142      MOV      RMMR2(R0), $BDDAT      ;STORE RMMR2 AT $BDDAT
10451 040210 042737 037777 001142      BIC      #1C<RQA!RQB>, $BDDAT
10452 040216 001006                BNE      20$
10453 040220 010037 001136      MOV      R0, $BDADR
10454 040224 062737 000040 001136      ADD      #RMMR2, $BDADR
10455 040232 104222                ERROR    222                    ;CANT SET REQUEST BY RMDA
10456
10457 040234                20$:
10458

```

K01

CZRMJBO RM03/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 217
T101 RELEASE TEST

SEQ 0217

```

10459
10460
10461
10462
10463 040234 000004
10464 040236 012737 000101 001226
10465
10466 040244 000240
10467 040246 012737 000024 001120
10468 040254 112737 000001 001131
10469 040262 012737 040276 001122
10470 040270 012737 040276 001124
10471 040276
10472 040276 012706 001100
10473 040302 013700 001276
10474 040306 013701 001456
10475
10476
10477 040312 004737 060654
10478 040316 000402
10479 040320 104000
10480 040322 000454
10481 040324
10482
10483
10484 040324 01276C 041001 000024
10485
10486 040332 012760 000013 000000
10487 040340 012702 000002
10488 040344
10489
10490 040344 012760 141001 000024
10491
10492 040352 012760 041001 000024
10493 040360 005302
10494 040362 001370
10495
10496
10497 040364 015737 000026 001174
10498
10499 040372 016737 000040 001142
10500 040400 042737 037777 001142
10501 040406 001422
10502 040410 032737 004000 001174
10503 040416 001410
10504 040420 032737 100000 001142
10505 040426 001412
10506 040430 032737 040000 001142
10507 040436 001406
10508 040440
10509
10510
10511 040440 010037 001136
10512 040444 062737 000040 001136
10513 040452 104223
10514 040454

;*****
;TEST 101 RELEASE TEST
;*****
T101: SCOPE
MOV #101,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
NOP
MOV #20,$ICNT ;20 ITERATIONS
MOVB #1,$ERMAX ;ONE ERROR ALLOWED
MOV #T101,$LPADR ;LOAD LOOP ON TEST ADDRESS
MOV #T101,$LPERR ;LOAD LOOP ON ERROR ADDRESS
T101:
MOV #STACK,SP ;LOAD THE STACK POINTER
MOV $BASE,R0 ;R0 = UNIBUS ADDRESS OF UUT
MOV TSTGUE,R1 ;R1 = POINTER TO DEVICE
;REQUEST FLOP SHOULD SET WHEN WRITING RMCS1
;SET VOLUME VALID USING SUBROUTINE
JSR PC,SETVV ;GO SET VOLUME VALID
BR 10$ ;BRANCH TO 10$ IF NO ERROR
ERROR BR 40$ ;RETURN HERE IF ERROR
10$:
;EXECUTE A RELEASE COMMAND
MOV #DMD!DBEN!MUR,RMMR1(R0) ;LOAD RMMR1
MOV #RELEASE!GO,RMCS1(R0) ;LOAD RMCS1
MOV #2,R2
20$:
MOV #DMD!DBEN!MUR!DBCK,RMMR1(R0) ;LOAD RMMR1
MOV #DMD!DBEN!MUR,RMMR1(R0) ;LOAD RMMR1
DEC R2
BNE 20$ ;ISSUE 2 CLOCKS
;VERIFY REQUEST FLOPS ARE RESET
MOV RMDT(R0),$TMPD ;STORE RMDT AT $TMPD
MOV RMMR2(R0),$BDDAT ;STORE RMMR2 AT $BDDAT
BIC #C<RQA!RQB>,$BDDAT
BEQ 40$ ;BRANCH IF REQUESTS ARE RESET
BIT #DRQ,$TMPD
BEQ 30$ ;BRANCH IF SINGLE PORT DEVICE
BIT #RQA,$BDDAT
BEQ 40$ ;BRANCH IF RQA IS RESET
BIT #RQB,$BDDAT
BEQ 40$ ;BRANCH IF RQB IS RESET
30$:
;DRQ IS ZERO AND A REQUEST FLOP IS ON, OR, DRQ IS ONE AND
;BOTH REQUEST FLOPS ARE ON
MOV R0,$BDADR
ADD #RMMR2,$BDADR
ERROR 223 ;REQUEST FLOP NOT RESET
40$: ;END OF TEST

```

LO1

CZRMJBO RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 218
T101 RELEASE TEST

SEQ 0218

```

10515
10516
10517
10518
10519
10520 040454 000004
10521 040456 012737 000102 001226
10522
10523 040464 000240
10524 040466 012737 000024 001120
10525 040474 112737 000001 001131
10526 040502 012737 040516 001122
10527 040510 012737 040516 001124
10528 040516
10529 040516 012706 001100
10530 040522 013700 001276
10531 040526 013701 001456
10532
10533 040532 012760 000040 000010
10534 040540 111160 000010
10535
10536 040544 012760 000001 000024
10537
10538 040552 012760 000000 000014
10539
10540 040560 012760 000000 000042
10541
10542 040566 012760 001001 000024
10543
10544 040574 111102
10545 040576 042702 177770
10546 040602 116203 066242
10547 040606 042703 177400
10548
10549 040612 010360 000016
10550
10551
10552 040616 016037 000012 001142
10553 040624 042737 077777 001142
10554 040632 001411
10555 040634 010037 001136
10556 040640 062737 000012 001136
10557 040646 005037 001140
10558 040652 104224
10559 040654 000417
10560 040656
10561
10562
10563 040656 016037 000016 001142
10564 040664 005103
10565 040666 040337 001142
10566 040672 001410
10567 040674 005037 001140
10568 040700 010037 001136
10569 040704 062737 000016 001136
10570 040712 104225

;*****
;*TEST 102 WRITE ATA TEST
;*****
†ST102: SCOPE
MOV #102,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX

NOP
MOV #20,$ICNT ;20 ITERATIONS
MOV #1,$ERMAX ;ONE ERROR ALLOWED
MOV #T102,$LPADR ;LOAD LOOP ON TEST ADDRESS
MOV #T102,$LPERR ;LOAD LOOP ON ERROR ADDRESS

T102:
MOV #STACK,SP ;LOAD THE STACK POINTER
MOV $BASE,R0 ;R0 = UNIBUS ADDRESS OF UUT
MOV TSTQUE,R1 ;R1 = POINTER TO DEVICE
;CLEAR THE DEVICE, SET DIAGNOSTIC MODE THEN SET UNIT READY
MOV #CLR,RMCS2(R0) ;CLEAR THE MASSBUS
MOVB (R1),RMCS2(R0) ;SELECT UNIT

MOV #DMD,RMMR1(R0) ;LOAD RMMR1
MOV #0,RMER1(R0) ;LOAD RMER1
MOV #0,RMER2(R0) ;LOAD RMER2

MOV #DMD!MUR,RMMR1(R0) ;LOAD RMMR1
;WRITE THE ATTENTION SUMMARY REGISTER
MOVB (R1),R2 ;ASSEMBLE THE ATA BIT IN R3
BIC #!CUNTMSK,R2
MOVB ATNTBL(R2),R3
BIC #!CATNMSK,R3

MOV R3,RMAS(R0) ;LOAD RMAS
;READ RMDS AND VERIFY THAT ATA IS RESET

MOV RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
BIC #!CATA,$BDDAT
BEQ 10$ ;BRANCH IF ATA IS RESET
MOV R0,$BDADR
ADD #RMDS,$BDADR
CLR $GDDAT
ERROR 224 ;CANT CLEAR ATA
BR 20$

10$:
;READ RMAS AND VERIFY ATA IS RESET

MOV RMAS(R0),$BDDAT ;STORE RMAS AT $BDDAT
COM R3
BIC R3,$BDDAT
BEQ 20$ ;BRANCH IF ATA IS RESET
CLR $GDDAT
MOV R0,$BDADR
ADD #RMAS,$BDADR
ERROR 225 ;RMAS ATA NOT ZERO

```

MO1

CZRMJBO RM03/2 OSA ~ DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 219
T102 WRITE ATA TEST

SEQ 0219

```

10571 040714          20$:                               ;END OF TEST
10572
10573                ;*****
10574                ;*TEST 103      RESET ATA BY GO TEST
10575                ;*****
10576                ;*****
10577 040714 000004    †ST103: SCOPE
10578 040716 012737 000103 001226      MOV      #103,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
10579
10580 040724 000240          NOP
10581 040726 012737 000024 001120      MOV      #20,$ICNT      ;20 ITERATIONS
10582 040734 112737 000001 001131      MOVVB   #1,$ERMAX      ;ONE ERROR ALLOWED
10583 040742 012737 040756 001122      MOV      #T103,$LPADR   ;LOAD LOOP ON TEST ADDRESS
10584 040750 012737 040756 001124      MOV      #T103,$LPERR   ;LOAD LOOP ON ERROR ADDRESS
10585 040756
10586 040756 012706 001100      T103:   MOV      #STACK,SP      ;LOAD THE STACK POINTER
10587 040762 013700 001276      MOV      $BASE,R0      ;R0 = UNIBUS ADDRESS OF UUT
10588 040766 013701 001456      MOV      †STACK,R1     ;R1 = POINTER TO DEVICE
10589 040772 012760 000040 000010      MOV      #CLR,RMCS2(R0) ;CLEAR THE MASSBUS
10590 041000 111160 000010      MOVVB   (R1),RMCS2(R0) ;SELECT UNIT
10591
10592 041004 012760 000001 000024      MOV      #DMD,RMMR1(R0) ;LOAD RMMR1
10593
10594 041012 012760 000000 000014      MOV      #0,RMER1(R0)  ;LOAD RMER1
10595
10596 041020 012760 000000 000042      MOV      #0,RMER2(R0)  ;LOAD RMER2
10597
10598 041026 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
10599                ;WITH DEBUG CLOCK ENABLED, SET GO AND VERIFY THAT ATA IS RESET
10600
10601 041034 012760 000001 000000      MOV      #GO,RMCS1(R0) ;LOAD RMCS1
10602
10603 041042 016037 000012 001142      MOV      RMDS(R0),$BODAT ;STORE RMDS AT $BODAT
10604 041050 042737 077777 001142      BIC     #†CATA,$BODAT
10605 041056 001410          BEQ     10$              ;BRANCH IF ATA IS RESET
10606 041060 005037 001140          CLR     $GDDAT
10607 041064 010037 001136          MOV     R0,$BODADR
10608 041070 062737 000012 001136          ADD     #RMDS,$BODADR
10609 041076 104226          ERROR    226            ;CANT RESET ATA WITH GO
10610
10611 041100          10$:                               ;END OF TEST
10612
10613                ;*****
10614                ;*TEST 104      UNIT READY ATA TEST
10615                ;*****
10616                ;*****
10617 041100 000004    †ST104: SCOPE
10618 041102 012737 000104 001226      MOV      #104,$TESTN   ;;SET TEST NUMBER IN APT MAIL BOX
10619
10620 041110 000240          NOP
10621 041112 012737 000024 001120      MOV      #20,$ICNT     ;20 ITERATIONS
10622 041120 112737 000001 001131      MOVVB   #1,$ERMAX     ;ONE ERROR ALLOWED
10623 041126 012737 041142 001122      MOV      #T104,$LPADR  ;LOAD LOOP ON TEST ADDRESS
10624 041134 012737 041142 001124      MOV      #T104,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
10625 041142
10626 041142 012706 001100      T104:   MOV      #STACK,SP      ;LOAD THE STACK POINTER

```

```

10627 041146 013700 001276      MOV      $BASE,RO      ;RO = UNIBUS ADDRESS OF UUT
10628 041152 013701 001456      MOV      TSTQUE,R1     ;R1 = POINTER TO DEVICE
10629                                     ;SET DIAGNOSTIC MODE AND CLEAR ATA
10630 041156 012760 000040 000010  MOV      #CLR,RMCS2(RO) ;CLEAR THE MASSBUS
10631 041164 111160 000010      MOV      (R1),RMCS2(RO) ;SELECT UNIT
10632
10633 041170 012760 000001 000024  MOV      #DMD,RMMR1(RO) ;LOAD RMMR1
10634
10635 041176 012760 000000 000014  MOV      #0,RMER1(RO)   ;LOAD RMER1
10636
10637 041204 012760 000000 000042  MOV      #0,RMER2(RO)   ;LOAD RMER2
10638
10639 041212 012760 177777 000016  MOV      #-1,RMAS(RO)   ;LOAD RMAS
10640                                     ;SET UNIT READY AND VERIFY ATA IS SET
10641
10642 041220 012760 001001 000024  MOV      #DMD!MUR,RMMR1(RO) ;LOAD RMMR1
10643
10644 041226 016037 000012 001142  MOV      RMDS(RO), $BDDAT ;STORE RMDS AT $BDDAT
10645 041234 042737 077777 001142  BIC      #+CATA,$BDDAT
10646 041242 001011                                     ;BRANCH IF ATA IS SET
10647 041244 010037 001136      BNE      10$
10648 041250 062737 000012 001136  MOV      RO,$BDADR
10649 041256 012737 100000 001140  ADD      #RMDS,$BDADR
10650 041264 104227                                     ;ATA NOT SET BY 01 READY
10651 041266                                     ;ATA NOT SET BY 01 READY
10652
10653                                     10$:
10654 041266 012760 177777 000016  ;CLEAR ATA, RESET UNIT READY, AND VERIFY ATA IS SET
10655
10656 041274 012760 000001 000024  MOV      #-1,RMAS(RO)   ;LOAD RMAS
10657
10658 041302 016037 000012 001142  MOV      RMDS(RO), $BDDAT ;STORE RMDS AT $BDDAT
10659 041310 042737 077777 001142  BIC      #+CATA,$BDDAT
10660 041316 001011                                     ;BRANCH IF 10 READY
10661 041320 010037 001136      BNE      20$
10662 041324 062737 000012 001136  MOV      RO,$BDADR
10663 041332 012737 100000 001140  ADD      #RMDS,$BDADR
10664 041340 104230                                     ;ATA NOT SET BY 10 READY
10665
10666 041342                                     20$:
10667                                     ;END OF TEST
10668
10669                                     ;*****
10670                                     ;*TEST 105 ERROR ATA TEST
10671
10672                                     ;*****
10673 041342 000004      ;ST105: SCOPE
10674 041344 012737 000105 001226  MOV      #105,$TESTN   ;; SET TEST NUMBER IN APT MAIL BOX
10675
10676 041352 000240      NOP
10677 041354 012737 000024 001120  MOV      #20,$ICNT     ;20 ITERATIONS
10678 041362 112737 000001 001131  MOV      #1,$EMAX      ;ONE ERROR ALLOWED
10679 041370 012737 041404 001122  MOV      #T105,$LPADR  ;LOAD LOOP ON TEST ADDRESS
10680 041404 012737 041404 001124  MOV      #T105,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
10681 041404 012706 001100      05:
10682 041410 013700 001276      MOV      #STACK,SP     ;LOAD THE STACK POINTER
                                MOV      $BASE,RO      ;RO = UNIBUS ADDRESS OF UUT

```

```

10683 041414 013701 001456      MOV     TSTQUE,R1      ;R1 = POINTER TO DEVICE
10684      ;CLEAR THE DEVICE AND RESET ATTENTION
10685 041420 012760 000040 000010  MOV     #CLR,RMCS2(RO) ;CLEAR THE MASSBUS
10686 041426 111160 000010      MOV     (R1),RMCS2(RO) ;SELECT UNIT
10687
10688 041432 012760 000001 000024  MOV     #DMD,RMMR1(RO) ;LOAD RMMR1
10689
10690 041440 012760 000000 000014  MOV     #0,RMER1(RO)   ;LOAD RMER1
10691
10692 041446 012760 000000 000042  MOV     #0,RMER2(RO)   ;LOAD RMER2
10693
10694 041454 012760 177777 000016  MOV     #-1,RMAS(RO)   ;LOAD RMAS
10695      ;WRITE ONES IN ERROR REGISTER 1 AND VERIFY ATA IS SET
10696
10697 041462 012760 177777 000014  MOV     #-1,RMER1(RO)  ;LOAD RMER1
10698
10699 041470 016037 000012 001142  MOV     RMDS(RO),%BDDAT ;STORE RMDS AT %BDDAT
10700 041476 042737 077777 001142  BIC     #ICATA,%BDDAT
10701 041504 001011      BNE     10$
10702 041506 012737 100000 001140  MOV     #ATA,%GDDAT
10703 041514 010037 001136      MOV     RO,%BDAOR
10704 041520 062737 000012 001136  ADD     #RMDS,%BCADR
10705 041526 104231      ERROR   231           ;ATA NOT SET BY COMP ERROR
10706
10707 041530      10$:
10708      ;END OF TEST
10709
10710      ;*****
10711      ;*TEST 106 REGISTER TRANSFER ATA TEST
10712      ;*****
10713 041530 000004      †ST106: SCOPE
10714 041532 012737 000106 001226  MOV     #106,$TESTN   ;;SET TEST NUMBER IN APT MAIL BOX
10715 041540 000240      NOP
10716 041542 012737 000024 001120  MOV     #20,$ICNT     ;20 ITERATIONS
10717 041550 112737 000001 001131  MOV     #1,$ERMAX     ;ONE ERROR ALLOWED
10718 041556 012737 041572 001122  MOV     #T106,$LPADR  ;LOAD LOOP ON TEST ADDRESS
10719 041564 012737 041572 001124  MOV     #T106,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
10720
10721 041572 012706 001100      T106:
10722 041576 013700 001276  MOV     #STACK,SP     ;LOAD THE STACK POINTER
10723 041602 013701 001456  MOV     %BASE,RO      ;RO = UNIBUS ADDRESS OF UUT
10724 041606 012702 041742  MOV     TSTQUE,R1     ;R1 = POINTER TO DEVICE
10725      MOV     #100,$A2     ;INITIALIZE TABLE ADDRESS
10726 041612      ;FORCE COMPOSITE ERROR AND RESET ATA
10727 041612 012760 000040 000010  5$:
10728 041620 111160 000010      MOV     #CLR,RMCS2(RO) ;CLEAR THE MASSBUS
10729      MOV     (R1),RMCS2(RO) ;SELECT UNIT
10730 041624 012760 000001 000024  MOV     #DMD,RMMR1(RO) ;LOAD RMMR1
10731
10732 041632 012760 177777 000014  MOV     #-1,RMER1(RO)  ;LOAD RMER1
10733
10734 041640 012760 177777 000016  MOV     #-1,RMAS(RO)   ;LOAD RMAS
10735      ;WRITE THE TEST REGISTER AND VERIFY ATA
10736 041646 011203  MOV     (R2),R3
10737 041650 060003  ADD     RO,R3         ;GENERATE REGISTER ADDRESS
10738 041652 005013  CLR     (R3)

```

```

10739
10740 041654 016037 000012 001142      MOV      RMDS(RO), $BDDAT ;STORE RMDS AT $BDDAT
10741 041662 042737 077777 001142      BIC      #1,CATA, $BDDAT
10742 041670 016237 000002 001140      MOV      2(R2), $GDDAT
10743 041676 023737 001140 001142      CMP      $GDDAT, $BDDAT
10744 041704 001410                BEQ      10$                ;BRANCH IF ATA IS OK
10745 041706 010337 001174      MOV      R3, $TMPD
10746 041712 010037 001136      MOV      RO, $BODR
10747 041716 062737 000012 001136      ADD      #RMDS, $BDADR
10748 041724 104232                ERROR      232                ;ATA INCORRECT FOR REG WRITTEN
10749 041726
10750
10751 041726 062702 000004      10$:
;MOVE TABLE POINTER - EXIT IF DONE
10752 041732 005712                ADD      #4, R2
10753 041734 100401                TST      (R2)
10754 041736 000725                BMI      20$                ;EXIT IF ENTRY MINUS
10755 041740 000410                BR       5$
10756
10757 041742                20$: BR       200$                ;JUMP OVER TABLE
10758
10759 041742 000016                100$:
;TABLE OF REGISTER ADDRESSES AND ATA BITS
10760 041744 000000                .WORD   RMAS
10761
10762 041746 000006                .WORD   RMDA
10763 041750 100000                .WORD   ATA
10764
10765 041752 000000                .WORD   RMCS1
10766 041754 000000                .WORD
10767
10768 041756 177777                .WORD   -1                ;END OF TABLE
10769 041760 000000                .WORD
10770
10771 041762                200$:
;END OF TEST
10772
10773
10774
10775
10776
10777
10778 041762 000004      ;*****
;TEST 107 P SET ATA TEST
10779 041764 012737 000107 001226      ;*****
;ST107: SCOPE
MOV      #107, $TESTN ;;SET TEST NUMBER IN APT MAIL BOX
10780
10781 041772 000240      NOP
10782 041774 012737 000024 001120      MOV      #20, $ICNT ;20 ITERATIONS
10783 042002 112737 000001 001131      MOV      #1, $ERMAX ;ONE ERROR ALLOWED
10784 042010 012737 042024 001122      MOV      #T107, $LPADR ;LOAD LOOP ON TEST ADDRESS
10785 042016 012737 042024 001124      MOV      #T107, $LPERA ;LOAD LOOP ON ERROR ADDRESS
10786 042024
10787 042024 012706 001100      T107: MOV      #STACK, SP ;LOAD THE STACK POINTER
10788 042030 013700 001276      MOV      $BASE, RO ;RO = UNIBUS ADDRESS OF UUT
10789 042034 013701 001456      MOV      TSTQUE, R1 ;R1 = POINTER TO DEVICE
10790 042040 012702 042202      MOV      #100$, R2 ;INITIALIZE TABLE POINTER
10791
10792 042044
10793
10794 042044 004737 060654      10$:
;SET VOLUME VALID USING SUBROUTINE
JSR      PC, SETVV ;GO SET VOLUME VALID

```



```

10795 042050 000402 BR 20$ ;BRANCH TO 20$ IF NO ERROR
10796 042052 104000 ERROR ;RETURN HERE IF ERROR
10797 042054 000451 BR 50$
10798 042056
10799
20$:
;EXECUTE THE COMMAND FROM THE TABLE
10800
10801 042056 012760 041001 000024 MOV #DMD:MUR:DBEN,RMMR1(RO) ;LOAD RMMR1
10802 042064 011203 MOV (R2),R3 ;GET FUNCTION CODE
10803 042066 052703 000001 BIS #GO,R3
10804
10805 042072 010360 000000 MOV R3, RMCS1(RO) ;LOAD RMCS1
10806 042076 012703 000003 MOV #3,R3
10807 042102
30$:
10808
10809 042102 012760 141001 000024 MOV #DMD:MUR:DBEN:DECK,RMMR1(RO) ;LOAD RMMR1
10810
10811 042110 012760 041001 000024 MOV #DMD:MUR:DBEN,RMMR1(RU) ;LOAD RMMR1
10812 042116 005303 DEC R3
10813 042120 001370 BNE 30$
10814 ;VERIFY THAT ATA IS SET
10815
10816 042122 016037 000012 001142 MOV RMDS(RO), $BDDAT ;STORE RMDS AT $BDDAT
10817 042130 042737 077777 001142 BIC #1CATA,$BDDAT
10818 042136 001013 BNE 40$
10819 042140 010037 001136 MOV RO,$BDADR
10820 042144 062737 000012 001136 ADD #RMDS,$BDADR
10821 042152 012737 100000 001140 MOV #ATA,$GDDAT
10822 042160 011237 001174 MOV (R2),$TMPO
10823 042164 104233 ERROR 233 ;ATA NOT SET BY SEQUENCER
10824 042166
40$:
;ADVANCE TABLE POINTER-EXIT IF DONE
10825
10826 042166 062702 000002 ADD #2,R2
10827 042172 005712 TST (R2)
10828 042174 100401 BMI 50$
10829 042176 000722 BR 10$
10830 042200 000403 50$: BR 200$ ;JUMP OVER TABLE
10831
10832 042202
100$:
;TABLE OF FUNCTION CODES
10833
10834
10835 042202 000014 .WORD OFFSET
10836
10837 042204 000016 .WORD RTC
10838
10839 042206 177777 .WORD -1 ;END OF TABLE
10840 042210
200$:
;*****
; *TEST 110 SET WLE TEST
;*****
10841
10842
10843
10844
10845
10846 042210 000004 †ST110: SCOPE
10847 042212 012737 000110 001226 MOV #110,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
10848
10849 042220 000240 NOP
10850 042222 012737 000024 001120 MOV #20.,$ICNT ;20 ITERATIONS

```

E02

CZRMJBO RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 224
T110 SET WLE TEST

SEQ 0224

10851	042230	112737	000001	001131	MOV	#1, \$ERMAX	; ONE ERROR ALLOWED
10852	042236	012737	042252	001122	MOV	#T110, \$LPADR	; LOAD LOOP ON TEST ADDRESS
10853	042244	012737	042252	001124	MOV	#T110, \$LPERR	; LOAD LOOP ON ERROR ADDRESS
10854	042252				T110:		
10855	042252	012706	001100		MOV	#STACK, SP	; LOAD THE STACK POINTER
10856	042256	013700	001276		MOV	\$BASE, R0	; R0 = UNIBUS ADDRESS OF UUT
10857	042262	013701	001456		MOV	TSTQUE, R1	; R1 = POINTER TO DEVICE
10858	042266	012702	042462		MOV	#100\$, R2	; INITIALIZE TABLE POINTER
10859	042272				10\$:		
10860					; SET VOLUME VALID USING SUBROUTINE		
10861	042272	004737	060654		JSR	PC, SETVV	; GO SET VOLUME VALID
10862	042276	000402			BR	20\$; BRANCH TO 20\$ IF NO ERROR
10863	042300	104000			ERROR		; RETURN HERE IF ERROR
10864	042302	000466			BR	50\$	
10865	042304				20\$:		
10866					; ENABLE DEBUG CLOCK AND SET WRITE PROTECT ACCORDING TO TABLE		
10867	042304	016203	000002		MOV	2(R2), R3	; GET WRITE PROTECT FROM TABLE
10868	042310	052703	041001		BIS	#DMD!MUR!DBEN, R3	; SET OTHER MAINT BITS
10869							
10870	042314	010360	000024		MOV	R3, RMMR1(R0)	; LOAD RMMR1
10871					; LOAD AND EXECUTE THE COMMAND FROM THE TABLE		
10872	042320	011204			MOV	(R2), R4	; GET FUNCTION CODE FROM TABLE
10873	042322	052704	000001		BIS	#GO, R4	; SET GO
10874							
10875	042326	010460	000000		MOV	R4, RMCS1(R0)	; LOAD RMCS1
10876	042332	012705	000002		MOV	#2, R5	; R5=CLOCK COUNT
10877	042336	052703	100000		30\$:	BIS	#DBCK, R3
10878							; SET CLOCK
10879	042342	010360	000024		MOV	R3, RMMR1(R0)	; LOAD RMMR1
10880	042346	042703	100000		BIC	#DBCK, R3	; RESET CLOCK
10881							
10882	042352	010360	000024		MOV	R3, RMMR1(R0)	; LOAD RMMR1
10883	042356	005305			DEC	R5	
10884	042360	001366			BNE	30\$; ISSUE 2 CLOCKS
10885					; VERIFY THAT WRITE LOCK ERROR IS ACCORDING TO TABLE		
10886							
10887	042362	016037	000014	001142	MOV	RMR1(R0), \$BDDAT	; STORE RMR1 AT \$BDDAT
10888	042370	042737	173777	001142	BIC	#1CWLE, \$BDDAT	
10889	042376	026237	000004	001142	CMP	4(R2), \$BDDAT	
10890	042404	001417			BEG	40\$; BRANCH IF WLE IS OK
10891	042406	016237	000004	001140	MOV	4(R2), \$GDDAT	; SAVE DATA FOR ERROR MSG
10892	042414	010037	001136		MOV	R0, \$B0ADR	
10893	042420	062737	000014	001136	ADD	#RMR1, \$B0ADR	

```

10894 042426 011237 001174      MOV      (R2),STMP0      ;STMP0=FUNCTION CODE
10895 042432 016237 000002 001176  MOV      2(R2),STMP1    ;STMP1=WRITE PROTECT
10896 042440 104234      ERROR    234            ;WLE INCORRECT
10897 042442 000406      BR       50$
10898 042444
10899
10900 042444 062702 000006      40$:
;ADVANCE TABLE POINTER TO NEXT FUNCTION CODE-EXIT IF DONE
10901 042450 005762 000002      ADD      #6,R2
10902 042454 100401      TST     2(R2)
10903 042456 000705      BMI     50$
10904
10905 042460 000416      BR       10$           ;REPEAT TEST
10906
10907 042462      50$: BR       200$     ;JUMP OVEP TABLE
10908
10909
10910      100$:
;TABLE OF FUNCTION CODES, WRITE PROTECT AND WRITE LOCK ERRORS
10911 042462 000060      .WORD   WD             ;WRITE DATA COMMAND
10912 042464 000010      .WORD   MWP            ;WRITE PROTECT ON
10913 042466 004000      .WORD   WLE            ;WRITE LOCK ERROR ONE
10914
10915 042470 000060      .WORD   WD             ;WRITE DATA COMMAND
10916 042472 000000      .WORD   MWP            ;MWP OFF
10917 042474 000000      .WORD   WLE            ;WLE OFF
10918
10919 042476 000070      .WORD   RD             ;READ DATA COMMAND
10920 042500 000010      .WORD   MWP            ;MWP ON
10921 042502 000000      .WORD   WLE            ;WLE OFF
10922
10923 042504 000020      .WORD   RIP            ;READ IN PRESET COMMAND
10924 042506 000010      .WORD   MWP            ;MWP ON
10925 042510 000000      .WORD   WLE            ;WLE OFF
10926
10927 042512 000000      .WORD   -1             ;END OF TABLE
10928 042514 177777      .WORD
10929
10930 042516      200$:
;END OF TEST
10931
10932      ;*****
10933      ;*TEST 111 EXCEPTION TEST
10934
10935      ;*****
10936 042516 000004      †ST111: SCOPE
10937 042520 012737 000111 001226  MOV      #111,$TESTN    ;:SET TEST NUMBER IN APT MAIL BOX
10938
10939      ;WITH OCCUPIED SET, EACH OF THE FOLLOWING ERRORS SHOULD CAUSE AN
10940      ;EXCEPTION:
10941      ;
10942      ;PAR, IF RUN AND GO IS SET
10943      ;RMR, IF RUN AND GO IS SET
10944      ;ILR, IF RUN AND GO IS SET
10945      ;DCK
10946      ;HCE
10947      ;HCRC
10948      ;FER
10949      ;OPI

```

10950									
10951	042526	000240							NOP
10952	042530	012737	000024	001120					MOV #20, \$ICNT ;20 ITERATIONS
10953	042536	112737	000001	001131					MOV #1, \$ERMAX ;ONE ERROR ALLOWED
10954	042544	012737	042560	001122					MOV #T111, \$LPADR ;LOAD LOOP ON TEST ADDRESS
10955	042552	012737	042560	001124					MOV #T111, \$LPERR ;LOAD LOOP ON ERROR ADDRESS
10956	042560								T111:
10957	042560	012706	001100						MOV #STACK, SP ;LOAD THE STACK POINTER
10958	042564	013700	001276						MOV \$BASE, R0 ;R0 = UNIBUS ADDRESS OF UUT
10959	042570	013701	001456						MOV TSTQUE, R1 ;R1 = POINTER TO DEVICE
10960	042574	012702	043054						MOV #100\$, R2 ;INITIALIZE TABLE POINTER
10961	042600								10\$:
10962									;INITIALIZE AND VERIFY THAT EXCEPTION IS RESET
10963	042600	012760	000040	000010					MOV #CLR, RMCS2(R0) ;CLEAR THE MASSBUS
10964	042606	111160	000010						MOVB (R1), RMCS2(R0) ;SELECT UNIT
10965									
10966	042612	016037	000024	001142					MOV RMMR1(R0), \$BDDAT ;STORE RMMR1 AT \$BDDAT
10967	042620	042737	167777	001142					BIC #CREX, \$BDDAT
10968	042626	001410							BEQ 20\$;BRANCH IF EXCEPTION IS 0
10969	042630	010037	001136						MOV R0, \$BDDADR
10970	042634	062737	000024	001136					ADD #RMMR1, \$BDDADR
10971	042642	005037	001140						CLR \$GDDAT
10972	042646	104235							ERROR 235 ;CANT CLEAR EXCEPTION
10973	042650								20\$:
10974									;SET VOLUME VALID USING SUBROUTINE
10975	042650	004737	060654						JSR PC, SETVV ;GO SET VOLUME VALID
10976	042654	000402							BR 30\$;BRANCH TO 30\$ IF NO ERROR
10977	042656	104000							ERROR ;RETURN HERE IF ERROR
10978	042660	000474							BR 60\$
10979	042662								30\$:
10980									;EXECUTE A WRITE DATA COMMAND TO SET OCCUPIED, RUN AND GO
10981									
10982	042662	012760	000000	000006					MOV #0, RMDA(R0) ;LOAD RMDA
10983									
10984	042670	012760	000000	000034					MOV #0, RMDC(R0) ;LOAD RMDC
10985									
10986	042676	012760	041001	000024					MOV #DMD!MUR!DBEN, RMMR1(R0) ;LOAD RMMR1
10987									
10988	042704	012760	000061	000000					MOV #WD!GO, RMCS1(R0) ;LOAD RMCS1
10989	042712	012703	000002						MOV #2, R3
10990	042716								40\$:
10991									
10992	042716	012760	141001	000024					MOV #DMD!MUR!DBEN!DBCK, RMMR1(R0) ;LOAD RMMR1
10993									
10994	042724	012760	041001	000024					MOV #DMD!MUR!DBEN, RMMR1(R0) ;LOAD RMMR1
10995	042732	005303							DEC R3
10996	042734	001370							BNE 40\$;ISSUE 2 CLOCKS
10997									;LOAD ERROR REGISTER WITH ENTRY FROM TABLE AND VERIFY THAT EXCEPTION IS SET
10998									
10999	042736	011260	000014						MOV (R2), RMER1(R0) ;LOAD RMER1
11000	042742	012737	000200	001524					MOV #200, WATCH ;SET WATCHDOG TIMER VALUE
11001	042750	004777	136552						JSR PC, \$CLOCK ;START THE CLOCK
11002	042754								45\$:
11003									
11004	042754	016037	000024	001142					MOV RMMR1(R0), \$BDDAT ;STORE RMMR1 AT \$BDDAT
11005	042762	042737	167777	001142					BIC #CREX, \$BDDAT

11006	042770	001021			BNE	50\$		
11007	042772	005737	001524		TST	WATCH		; HAS CLOCK EXPIRED ??
11008	042776	001366			BNE	45\$; NO - TAKE ANOTHER SAMPLE
11009	043000	004777	136524		JSR	PC, 2STOP		; STOP THE CLOCK
11010	043004	012737	010000	001140	MOV	#R6X, \$GDDAT		
11011	043012	010037	001136		MOV	RO, \$BDADR		
11012	043016	062737	000024	001136	ADD	#RMR1, \$BDAOR		
11013	043024	011237	001174		MOV	(R2), \$TMP0		
11014	043030	104236			ERROR	236		; CANT SET EXCEPTION
11015	043032	000407			BR	60\$		
11016	043034							
11017					50\$:			
11018	043034	004777	136470		;ADVANCE TABLE POINTER-EXIT IF DONE			
11019	043040	062702	0J0002		JSR	PC, 2STOP		; STOP THE CLOCK
11020	043044	005712			ADD	#2, R2		
11021	043046	001401			TST	(R2)		
11022	043050	000653			BEG	60\$		
11023	043052	000402			BR	10\$		
11024					60\$:	BR	200\$; JUMP OVER TABLE
11025	043054							
11026					100\$:			
11027					;PRESENTLY, THE TABLE HAS ONLY ONE ENTRY, THE TEST USING RMR. THE			
11028					;TABLE SHOULD BE EXPANDED TO TEST ALL THE CONDITIONS LISTED ABOVE IF			
11029					;A HARDWARE CHANGE IS MADE SUCH THAT IT IS POSSIBLE TO WRITE ERROR			
11030					;REGISTER 1 WITH GO SET.			
11031	043054	0J0004			.WORD	RMR		;RMR IS CAUSED BY HARDWARE
11032								
11033	043054	000000			.WORD			;END OF TABLE
11034	043060				200\$:			;END OF TEST
11035								
11036					::*****			
11037					;*TEST 112	SET OPI TEST		
11038					::*****			
11039					†ST112: SCOPE			
11040	043060	000004			MOV	#112, \$TESTN		; ;SET TEST NUMBER IN APT MAIL BOX
11041	043062	012737	000112	001226				
11042								
11043	043070	000240			NOP			
11044	043072	012737	000024	001120	MOV	#20, \$ICNT		; 20 ITERATIONS
11045	043100	112737	000001	001131	MOVB	#1, \$ERMAX		; ONE ERROR ALLOWED
11046	043106	012737	043122	001122	MOV	#T112, \$LADR		; LOAD LOOP ON TEST ADDRESS
11047	043114	012737	043122	001124	MOV	#T112, \$LERR		; LOAD LOOP ON ERROR ADDRESS
11048	043122				T112:			
11049	043122	012706	001100		MOV	#STACK, SP		; LOAD THE STACK POINTER
11050	043126	013700	001276		MOV	\$BASE, RO		; RO = UNIBUS ADDRESS OF UUT
11051	043132	013701	001456		MOV	ISTQUE, R1		; R1 = POINTER TO DEVICE
11052	043136	005002			CLR	R2		; INITIALIZE FUNCTION CODE
11053	043140	010037	001136		MOV	RO, \$BDADR		; SETUP ERROR MSG
11054	043144	062737	000014	001136	ADD	#RMR1, \$BDAOR		
11055	043152				10\$:			
11056					;CLEAR AND VERIFY OPI IS RESET			
11057	043152	012760	000040	000010	MOV	#CLR, RMCS2(RO)		; CLEAR THE MASSBUS
11058	043160	111160	000010		MOVB	(R1), RMCS2(RO)		; SELECT UNIT
11059								
11060	043164	016037	000014	001142	MOV	RMR1(RO), \$BDDAT		; STORE RMR1 AT \$BDDAT
11061	043172	042737	157777	001142	BIC	#ICOP1, \$BDDAT		

```

11062 043200 001403          BEQ      20$          ;BRANCH IF OPI IS ZERO
11063 043202 005037 001140    CLR      $GDDAT
11064 043206 104237          ERROR    237          ;CANT CLEAR OPI
11065 043210          20$:
11066          ;EXECUTE THE FUNCTION CODE AND VERIFY OPI IS SET
11067
11068 043210 012760 000001 000024    MOV      #DMD,RMMR1(RO) ;LOAD RMMR1
11069
11070 043216 012760 040001 000024    MOV      #DMD!DBEN,RMMR1(RO) ;LOAD RMMR1
11071 043224 010203          MOV      R2,R3          ;SET GO
11072 043226 052703 000001          BIS      #GO,R3
11073
11074 043232 010360 000000          MOV      R3,RMCS1(RO) ;LOAD RMCS1
11075 043236 012704 000004          MOV      #4,R4          ;R4=CLOCK COUNT
11076 043242          30$:
11077
11078 043242 012760 140001 000024    MOV      #DMD!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1
11079
11080 043250 012760 040001 000024    MOV      #DMD!DBEN,RMMR1(RO) ;LOAD RMMR1
11081 043256 005304          DEC      R4
11082 043260 001370          BNE     30$          ;ISSUE 4 CLOCKS
11083
11084 043262 016037 000014 001142    MOV      RMER1(RO), $BDDAT ;STORE RMER1 AT $BDDAT
11085 043270 042737 157777 001142    BIC     #1COPI, $BDDAT
11086 043276 001007          BNE     40$
11087 043300 012737 020000 001140    MOV      #OPI, $GDDAT
11088 043306 010237 001174          MOV      R2, $TMP0
11089 043312 104240          ERROR    240          ;CANT SET OPI
11090 043314 000406          BR      50$
11091 043316          40$:
11092          ;ADVANCE FUNCTION CODE-EXIT IF DONE
11093 043316 062702 000002          ADD     #2,R2
11094 043322 022702 000076          CMP     #1LF76,R2
11095 043326 103401          BLO     50$
11096 043330 000710          BR      10$          ;REPEAT FOR NEXT CODE
11097
11098 043332          50$:
11099          ;END OF TEST
11100          ;;*****
11101          ;*TEST 113 RECALIBRATE TEST
11102          ;;*****
11103          ;*****
11104 043332 000004          †ST113: SCOPE
11105 043334 012737 000113 001226    MOV     #113,$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
11106
11107 043342 000240          NOP
11108 043344 012737 000024 001120    MOV     #20,$ICNT ;20 ITERATIONS
11109 043352 112737 000001 001131    MOVB   #1,$ERMAX ;ONE ERROR ALLOWED
11110 043360 012737 043374 001122    MOV     #T113,$LPADR ;LOAD LOOP ON TEST ADDRESS
11111 043366 012737 043374 001124    MOV     #T113,$LPERR ;LOAD LOOP ON ERROR ADDRESS
11112 043374          T113:
11113 043374 012706 001100          MOV     #STACK,SP ;LOAD THE STACK POINTER
11114 043400 013700 001276          MOV     $BASE,R0 ;R0 = UNIBUS ADDRESS OF UUT
11115 043404 013701 001456          MOV     TSTQUE,R1 ;R1 = POINTER TO DEVICE
11116          ;;*****
11117          ;*****

```

```

11118 ;VERIFY THAT OPI SETS IF UNIT READY DROPS AFTER DECODE
11119
11120 ;SET VOLUME VALID USING SUBROUTINE
11121 043410 004737 060654 JSR PC SETVV ;GO SET VOLUME VALID
11122 043414 000403 BR 10$ ;BRANCH TO 10$ IF NO ERROR
11123 043416 104000 ERROR ;RETURN HERE IF ERROR
11124 043420 000137 044716 JMP 330$
11125 043424 10$:
11126 ; ENABLE DEBUG CLOCK AND LOAD RECALIBRATE COMMAND
11127
11128 043424 012760 041001 000024 MOV #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
11129
11130 043432 012760 000000 000014 MOV #0,RMER1(RO) ;LOAD RMER1
11131
11132 043440 012760 000000 000042 MOV #0,RMER2(RO) ;LOAD RMER2
11133
11134 043446 012760 000007 000000 MOV #RECAL!GO,RMCS1(RO) ;LOAD RMCS1
11135 ; STEP COMMAND SEQUENCER TO RECAL COM (2 CLOCKS)
11136 043454 012702 000002 MOV #2,R2
11137 043460 20$:
11138
11139 043460 012760 141001 000024 MOV #DMD!MUR!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1
11140
11141 043466 012760 041001 000024 MOV #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
11142 043474 005302 DEC R2
11143 043476 001370 BNE 20$
11144 ; DROP UNIT READY AND STEP COMMAND SEQUENCER (2 CLOCKS)
11145
11146 043500 012760 040001 000024 MOV #DMD!DBEN,RMMR1(RO) ;LOAD RMMR1
11147 043506 012702 000002 MOV #2,R2
11148 043512 30$:
11149
11150 043512 012760 140001 000024 MOV #DMD!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1
11151
11152 043520 012760 040001 000024 MOV #DMD!DBEN,RMMR1(RO) ;LOAD RMMR1
11153 043526 005302 DEC R2
11154 043530 001370 BNE 30$
11155 ; VERIFY THAT OPI IS SET
11156
11157 043532 016037 000014 001142 MOV RMER1(RO),SBDAT ;STORE RMER1 AT SBDAT
11158 043540 042737 157777 001142 BIC #!COPI,SBDAT
11159 043546 001011 BNE 40$
11160 043550 012737 020000 001140 MOV #OPI,SDDAT
11161 043556 010037 001136 MOV RO,SBDADR
11162 043562 062737 000014 001136 ADD #RMER1,SBDADR
11163 043570 104241 ERROR 241 ;OPI NOT SET DURING RECAL
11164
11165 043572 012737 043600 001124 40$: MOV #50$,SLPERR ;CHANGE LOOP ON ERROR ADDRESS
11166
11167 ;*****
11168 ;VERIFY THAT RECALIBRATE ABORTS DURING EXECUTION
11169
11170 043600 50$:
11171 ;SET VOLUME VALID USING SUBROUTINE
11172 043600 004737 060654 JSR PC SETVV ;GO SET VOLUME VALID
11173 043604 000403 BR 60$ ;BRANCH TO 60$ IF NO ERROR

```

```

11174 043606 104000          ERROR          ;RETURN HERE IF ERROR
11175 043610 000137 044716  JMP          330$
11176
11177 043614          60$:
11178 ;          ENABLE DEBUG CLOCK AND LOAD RECALIBRATE COMMAND
11179
11180 043614 012760 041001 000024  MOV          #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
11181
11182 043622 012760 000000 000014  MOV          #0,RMER1(RO) ;LOAD RMER1
11183
11184 043630 012760 000000 000042  MOV          #0,RMER2(RO) ;LOAD RMER2
11185
11186 043636 012760 000007 000000  MOV          #RECAL!GO,RMCS1(RO) ;LOAD RMCS1
11187 ;          STEP THE COMMAND SEQUENCER TO FIRST TEST FOR ABORT (3 CLOCKS)
11188 043644 012702 000003          MOV          #3,R2
11189 043650          70$:
11190
11191 043650 012760 141001 000024  MOV          #DMD!MUR!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1
11192
11193 043656 012760 041001 000024  MOV          #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
11194 043664 005302          DEC          R2
11195 043666 001370          BNE          70$
11196 ;          SET DRIVE FAULT TO CAUSE ABORT CONDITION
11197
11198 043670 012760 041101 000024  MOV          #DMD!MUR!DBEN!MDF,RMMR1(RO) ;LOAD RMMR1
11199 ;          STEP 2 CLOCKS AND VERIFY GO IS RESET
11200 043676 012702 000002          MOV          #2,R2
11201 043702          80$:
11202
11203 043702 012760 141101 000024  MOV          #DMD!MUR!DBEN!MDF!DBCK,RMMR1(RO) ;LOAD RMMR1
11204
11205 043710 012760 041101 000024  MOV          #DMD!MUR!DBEN!MDF,RMMR1(RO) ;LOAD RMMR1
11206 043716 005302          DEC          R2
11207 043720 001370          BNE          80$
11208
11209 043722 016037 000000 001142  MOV          RMCS1(RO),SBDDAT ;STORE RMCS1 AT SBDDAT
11210 043730 042737 177776 001142  BIC          #1CGO,SBDDAT
11211 043736 001405          BEQ          90$
11212 043740 010037 001136          MOV          RO,$BDADR
11213 043744 005037 001140          CLR          $GDADR
11214 043750 104242          ERROR          242 ;GO NOT RESET DUE TO ABORT
11215
11216 043752 012737 043760 001124  90$: MOV          #100$,SLPERR ;CHANGE LOOP ON ERROR ADDRESS
11217
11218 ;*****
11219 ;VERIFY OPI SETS IF ON CYLINDER LATCH DOESNT CLEAR
11220
11221 043760          100$:
11222 ;SET VOLUME VALID USING SUBROUTINE
11223 043760 004737 060654          JSR          PC,SETVV ;GO SET VOLUME VALID
11224 043764 000403          BR          110$ ;BRANCH TO 110$ IF NO ERROR
11225 043766 104000          ERROR          ;RETURN HERE IF ERROR
11226 043770 000137 044716          JMP          330$
11227 043774          110$:
11228 ;          ENABLE DEBUG CLOCK AND LOAD RECALIBRATE COMMAND
11229

```


L02

CZRMJBO RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 231
T113 RECALIBRATE TEST

SEG 0231

```

11230 043774 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
11231
11232 044002 012760 000000 000014      MOV      #0,RMER1(RO) ;LOAD RMER1
11233
11234 044010 012760 000000 000042      MOV      #0,RMER2(RO) ;LOAD RMER2
11235
11236 044016 012760 000007 000000      MOV      #RECAL!GO, RMCS1(RO) ;LOAD RMCS1
11237 ; STEP THE COMMAND SEQUENCER
11238 044024 012702 000017      MOV      #15.,R2
11239 044030      120$:
11240
11241 044030 012760 141001 000024      MOV      #DMD!MUR!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1
11242
11243 044036 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
11244 044044 005302      DEC      R2
11245 044046 001370      BNE     120$
11246 ; VERIFY THAT OPI IS SET
11247
11248 044050 016037 000014 001142      MOV      RMER1(RO), $BDDAT ;STORE RMER1 AT $BDDAT
11249 044056 042737 157777 001142      BIC     #1COPI, $BDDAT
11250 044064 001011      BNE     130$
11251 044066 010037 001136      MOV      RO, $BDADR
11252 044072 062737 000014 001136      ADD     #RMER1, $BDADR
11253 044100 012737 020000 001140      MOV      #OPI, $GDDAT
11254 044106 104243      ERROR   243 ;OPI NOT SET DUE TO ON LATCH
11255
11256 044110 012737 044116 001124 130$: MOV      #150$, $LPERR ;CHANGE LOOP ON ERROR ADDRESS
11257
11258 ;*****
11259 ;VERIFY ATA SETS IF DRIVE COMPLETES RECALIBRATE (ON CYLINDER SETS)
11260
11261 044116      150$:
11262 ;SET VOLUME VALID USING SUBROUTINE
11263 044116 004737 060654      JSR     PC, SETVV ;GO SET VOLUME VALID
11264 044122 000403      BR     160$ ;BRANCH TO 160$ IF NO ERROR
11265 044124 104000      ERROR
11266 044126 000137 044716      JMP     330$ ;RETURN HERE IF ERROR
11267 044132      160$:
11268 ; ENABLE DEBUG CLOCK AND LOAD RECALIBRATE COMMAND
11269
11270 044132 012760 041401 000024      MOV      #DMD!MUR!DBEN!MOC,RMMR1(RO) ;LOAD RMMR1
11271
11272 044140 012760 000000 000014      MOV      #0,RMER1(RO) ;LOAD RMER1
11273
11274 044146 012760 000000 000042      MOV      #0,RMER2(RO) ;LOAD RMER2
11275
11276 044154 012760 000007 000000      MOV      #RECAL!GO, RMCS1(RO) ;LOAD RMCS1
11277 ; STEP COMMAND SEQUENCER TO FIRST ON LATCH TEST (13 CLOCKS)
11278 044162 012702 000015      MOV      #13.,R2
11279 044166      170$:
11280
11281 044166 012760 141401 000024      MOV      #DMD!MUR!DBEN!DBCK!MOC,RMMR1(RO) ;LOAD RMMR1
11282
11283 044174 012760 041401 000024      MOV      #DMD!MUR!DBEN!MOC,RMMR1(RO) ;LOAD RMMR1
11284 044202 005302      DEC     R2
11285 044204 001370      BNE     170$

```

M02

CZRMJBO RM03/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 232
T113 RECALIBRATE TEST

SEQ 0232

```

11286 ; DROP ON CYLINDER TO RESET LATCH, THEN SET ON CYLINDER
11287
11288 044206 012760 041001 000024 MOV #DMD:MUR:DBEN,RMMR1(RO) ;LOAD RMMR1
11289
11290 044214 012760 041401 000024 MOV #DMD:MUR:DBEN:MOC,RMMR1(RO) ;LOAD RMMR1
11291 STEP COMMAND SEQUENCER TO SET ATTENTION (3 CLOCKS)
11292 044222 012702 000003 ;
11293 044226 180$: MOV #3,R2
11294
11295 044226 012760 141401 000024 MOV #DMD:MUR:DBEN:MOC:DBCK,RMMR1(RO) ;LOAD RMMR1
11296
11297 044234 012760 041401 000024 MOV #DMD:MUR:DBEN:MOC,RMMR1(RO) ;LOAD RMMR1
11298 044242 005302 DEC R2
11299 044244 001370 BNE 180$
11300 ; VERIFY ATA IS SET
11301
11302 044246 016037 000012 001142 MOV RMDS(RO),SBODAT ;STORE RMDS AT SBODAT
11303 044254 042737 077777 001142 BIC #1CATA,SBODAT
11304 044262 001011 BNE 190$
11305 044264 012737 100000 001140 MOV #ATA,SCDDAT
11306 044272 010037 001136 MOV RO,SBODADR
11307 044276 062737 000012 001136 ADD #RMDS,SBODADR
11308 044304 104244 ERROR 244 ;ATA NOT SET BY RECAL
11309
11310 044306 012737 044314 001124 190$: MOV #200$,SLPERR ;CHANGE LOOP ON ERROR ADDRESS
11311
11312 ;*****
11313 ;VERIFY THAT RECALIBRATE ABORTS AFTER EXECUTION DURING WAIT LOOP
11314
11315 044314 200$:
11316 ;SET VOLUME VALID USING SUBROUTINE
11317 044314 004737 060654 JSR PC,SETVV ;GO SET VOLUME VALID
11318 044320 000402 BR 210$ ;BRANCH TO 210$ IF NO ERROR
11319 044322 104000 ERROR ;RETURN HERE IF ERROR
11320 044324 000574 BR 330$
11321 044326 210$:
11322 ; ENABLE DEBUG CLOCK AND LOAD RECALIBRATE COMMAND
11323
11324 044326 012760 041401 000024 MOV #DMD:MUR:DBEN:MOC,RMMR1(RO) ;LOAD RMMR1
11325
11326 044334 012760 000000 000014 MOV #0,RMER1(RO) ;LOAD RMER1
11327
11328 044342 012760 000000 000042 MOV #0,RMER2(RO) ;LOAD RMER2
11329
11330 044350 012760 000007 000000 MOV #RECAL!GO,RMCS1(RO) ;LOAD RMCS1
11331 ; STEP COMMAND SEQUENCER TO FIRST ON LATCH TEST (13 CLOCKS)
11332 044356 012702 000015 ;
11333 044362 220$: MOV #13.,R2
11334
11335 044362 012760 141401 000024 MOV #DMD:MUR:DBEN:MOC:DBCK,RMMR1(RO) ;LOAD RMMR1
11336
11337 044370 012760 041401 000024 MOV #DMD:MUR:DBEN:MOC,RMMR1(RO) ;LOAD RMMR1
11338 044376 005302 DEC R2
11339 044400 001370 BNE 220$
11340 ; DROP ON CYLINDER TO RESET LATCH, LEAVE ON CYLINDER C
11341

```

```

11342 044402 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
11343                                     ; STEP COMMAND SEQUENCER THROUGH WAIT LOOP (7 CLOCKS)
11344 044410 012702 000007                MOV      #7,R2
11345 044414                                230$:
11346
11347 044414 012760 041001 000024      MOV      #DMD!MUR!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1
11348
11349 044422 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
11350 044430 005302                        DEC      R2
11351 044432 001370                        BNE     230$
11352                                     ; VERIFY THAT GO IS STILL SET
11353
11354 044434 016037 000000 001142      MOV      RMCS1(RO), $BDDAT ;STORE RMCS1 AT $BDDAT
11355 044442 042737 177776 001142      BIC     #1CGO,$BDDAT
11356 044450 001006                        BNE     240$
11357 044452 012737 000001 001140      MOV      #GO,$GDDAT
11358 044460 010037 001136      MOV      RO,$BDADR
11359 044464 104245                        ERROR   245 ;GO RESET EARLY DURING RECAL
11360                                     ; SET SEEK INCOMPLETE ERROR
11361 044466                                240$:
11362
11363 0444. 012760 041201 000024      MOV      #DMD!MUR!DBEN!MSER,RMMR1(RO) ;LOAD RMMR1
11364                                     ; STEP COMMAND SEQUENCER AND VERIFY GO RESETS (3 CLOCKS)
11365 044474 012702 000003                MOV      #3,R2
11366 044500                                250$:
11367
11368 044500 012760 041201 000024      MOV      #DMD!MUR!DBEN!MSER!DBCK,RMMR1(RO) ;LOAD RMMR1
11369
11370 044506 012760 041201 000024      MOV      #DMD!MUR!DBEN!MSER,RMMR1(RO) ;LOAD RMMR1
11371 044514 005302                        DEC      R2
11372 044516 001370                        BNE     250$
11373
11374 044520 016037 000000 001142      MOV      RMCS1(RO), $BDDAT ;STORE RMCS1 AT $BDDAT
11375 044526 042737 177776 001142      BIC     #1CGO,$BDDAT
11376 044534 001405                        BEQ     260$
11377 044536 010037 001136      MOV      RO,$BDADR
11378 044542 005037 001140      CLR     $GDDAT
11379 044546 104246                        ERROR   246 ;GO NOT RESET DUR TO WAIT ABORT
11380
11381 044550 012737 044556 001124 260$: MOV      #300$,$LPERR ;CHANGE LOOP ON ERROR ADDRESS
11382
11383                                     ;*****
11384                                     ;VERIFY THE TAG BUS DURING RECALIBRATE
11385
11386 044556                                300$:
11387                                     ;SET VOLUME VALID USING SUBROUTINE
11388 044556 004737 060654                JSR     PC SETVV ;GO SET VOLUME VALID
11389 044562 000402                        BR      310$ ;BRANCH TO 310$ IF NO ERROR
11390 044564 104000                        ERROR   ;RETURN HERE IF ERROR
11391 044566 000453                        BR      330$
11392 044570                                310$:
11393                                     ; ENABLE DEBUG CLOCK AND LOAD RECALIBRATE COMMAND
11394
11395 044570 012760 041401 000024      MOV      #DMD!MUR!DBEN!MOC,RMMR1(RO) ;LOAD RMMR1
11396
11397 044576 012760 000000 000014      MOV      #0,RMER1(RO) ;LOAD RMER1

```

```

11398
11399 044604 012760 000000 000042      MOV      #0,RMR2(RO)      ;LOAD RMR2
11400
11401 044612 012760 000007 000000      MOV      #RECAL!GO, RMCS1(RO) ;LOAD RMCS1
11402 044620 012702 044720      MOV      #400$ R2        ;INITIALIZE TABLE POINTER
11403                                VERIFY TAG BUS ACCORDING TO TABLE
11404 044624                                315$:
11405
11406 044624 016037 000040 001142      MOV      RMMR2(RO), $BDDAT      ;STORE RMMR2 AT $BDDAT
11407 044632 042737 150000 001142      BIC      #RQA!RQB!TST, $BDDAT
11408 044640 021237 001142      CMP      (R2), $BDDAT
11409 044644 001411      BEQ      320$
11410 044646 011237 001140      MOV      (R2), $GDDAT
11411 044652 010037 001136      MOV      RO, $BDAOR
11412 044656 062737 000040 001136      ADD      #RMR2, $BDAOR
11413 044664 104247      ERROR    247                ;INCORRECT TAG BUS DURING RECAL
11414 044666 000413      BR       330$
11415 044670                                320$:
11416                                ;
11417 044670 062702 000002      ADVANCE TO NEXT ENTRY IN TABLE-EXIT IF DONE
11418 044674 005712      ADD      #2, R2
11419 044676 100407      TST      (R2)
11420                                BMI      330$                ;EXIT IF ENTRY NEGATIVE
11421                                ;
11422                                STEP THE COMMAND SEQUENCER AND REPEAT VERIFICATION
11422 044700 012760 151001 000024      MOV      #DMO!DBEN!MUR!MOV!DBCK, RMMR1(RO) ;LOAD RMMR1
11423
11424 044706 012760 041401 000024      MOV      #DMO!DBEN!MUR!MOC, RMMR1(RO) ;LOAD RMMR1
11425 044714 000743      BR       315$
11426 044716 000416      330$: BR       500$                ;JUMP OVER TABLE
11427
11428 044720                                400$:
11429
11430                                ;TABLE OF TAG BUS CONTROL AND BIT VALUES
11431
11432 044720 001777      .WORD    1777                ;BUS BITS AT HIGH IMPEDANCE STATE
11433 044722 001777      .WORD    1777
11434 044724 006100      .WORD    CC!CH!8806        ;CONTROL BITS ENABLED, BIT 6 ON
11435 044726 006100      .WORD    CC!CH!8806
11436 044730 026100      .WORD    TAG!CC!CH!8806    ;TAG COMES ON
11437 044732 026100      .WORD    TAG!CC!CH!8806
11438 044734 026100      .WORD    TAG!CC!CH!8806
11439 044736 026100      .WORD    TAG!CC!CH!8806
11440 044740 026100      .WORD    TAG!CC!CH!8806
11441 044742 026100      .WORD    TAG!CC!CH!8806
11442 044744 006100      .WORD    CC!CH!8806        ;TAG GOES OFF
11443 044746 006100      .WORD    CC!CH!8806
11444 044750 001777      .WORD    1777                ;CONTROL BITS DISABLED
11445
11446 044752 177777      .WORD    -1                ;END OF TABLE
11447
11448 044754                                500$:
11449                                ;END OF TEST
11450
11451                                ;*****
11452                                ;#TEST 114      SEEK TEST
11453                                ;*****

```

```

11454 044754 000004 TST114: SCOPE
11455 044756 012737 000114 001226 MOV #114,STESTN ;;SET TEST NUMBER IN APT MAIL BOX
11456
11457 044764 000240 NOP
11458 044766 012737 000024 001120 MOV #20,SICNT ;20 ITERATIONS
11459 044774 112737 000001 001131 MOV #1,SERMAX ;ONE ERROR ALLOWED
11460 045002 012737 045016 001122 MOV #T114,SLPADR ;LOAD LOOP ON TEST ADDRESS
11461 045010 012737 045016 001124 MOV #T114,SLPERR ;LOAD LOOP ON ERROR ADDRESS
11462 045016 T114:
11463 045016 012706 001100 MOV #STACK,SP ;LOAD THE STACK POINTER
11464 045022 013700 001276 MOV $BASE,R0 ;R0 = UNIBUS ADDRESS OF UUT
11465 045026 013701 001456 MOV TSTQUE,R1 ;R1 = POINTER TO DEVICE
11466
11467 ;*****
11468 ;VERIFY THAT OPI SETS IF UNIT READY DROPS DURING COMMAND EXECUTION
11469
11470 ;SET VOLUME VALID USING SUBROUTINE
11471 045032 004737 060654 JSR PC,SETVV ;GO SET VOLUME VALID
11472 045036 000403 BR 10$ ;BRANCH TO 10$ IF NO ERROR
11473 045040 104000 ERROR ;RETURN HERE IF ERROR
11474 045042 000137 046462 JMP 330$
11475 045046
11476 10$:
11477 ; ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
11478 ; ADDRESS, AND LOAD SEEK COMMAND
11479 045046 012760 041001 000024 MOV #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
11480
11481 045054 012760 000000 000006 MOV #0,RMDA(R0) ;LOAD RMDA
11482
11483 045062 012760 000000 000034 MOV #0,RMDC(R0) ;LOAD RMDC
11484
11485 045070 012760 000000 000014 MOV #0,RMER1(R0) ;LOAD RMER1
11486
11487 045076 012760 000000 000042 MOV #0,RMER2(R0) ;LOAD RMER2
11488
11489 045104 012760 000005 000000 MOV #SEEK!GO,RMCS1(R0) ;LOAD RMCS1
11490 ; STEP COMMAND SEQUENCER TO SEEK COM (2 CLOCKS)
11491 045112 012702 000002 MOV #2,R2
11492
11493 20$:
11494 045116 012760 141001 000024 MOV #DMD!MUR!DBEN!DBCK,RMMR1(R0) ;LOAD RMMR1
11495
11496 045124 012760 041001 000024 MOV #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
11497 045132 005302 DEC R2
11498 045134 001370 BNE 20$
11499 ; DROP UNIT READY AND STEP COMMAND SEQUENCER (2 CLOCKS)
11500
11501 045136 012760 040001 000024 MOV #DMD!DBEN,RMMR1(R0) ;LOAD RMMR1
11502 045144 012702 000002 MOV #2,R2
11503
11504 30$:
11505 045150 012760 140001 000024 MOV #DMD!DBEN!DBCK,RMMR1(R0) ;LOAD RMMR1
11506
11507 045156 012760 040001 000024 MOV #DMD!DBEN,RMMR1(R0) ;LOAD RMMR1
11508 045164 005302 DEC R2
11509 045166 001370 BNE 30$

```

```

11510 ; VERIFY THAT OPI IS SET
11511
11512 045170 016037 000014 001142 MOV RMER1(RO), $BDDAT ; STORE RMER1 AT $BDDAT
11513 045176 042737 157777 001142 BIC #1COPI, $BDDAT
11514 045204 001011 BNE 40$
11515 045206 012737 020000 001140 MOV #OPI, $GDDAT
11516 045214 010037 001136 MOV RO, $BDAOR
11517 045220 062737 000014 001136 ADD #RMER1, $BDAOR
11518 045226 104250 ERROR 250 ; OPI NOT SET DURING SEEK
11519
11520 045230 012737 045236 001124 40$: MOV #50$, $LPERR ; CHANGE LOOP ON ERROR ADDRESS
11521
11522 ; *****
11523 ; VERIFY THAT SEEK ABORTS DURING EXECUTION
11524
11525 045236 50$:
11526 ; SET VOLUME VALID USING SUBROUTINE
11527 045236 004737 060654 JSR PC, SETVV ; GO SET VOLUME VALID
11528 045242 000403 BR 60$ ; BRANCH TO 60$ IF NO ERROR
11529 045244 104000 ERROR ; RETURN HERE IF ERROR
11530 045246 000137 046462 JMP 330$
11531
11532 045252 60$:
11533 ; ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
11534 ; ADDRESS, AND LOAD SEEK COMMAND
11535
11536 045252 012760 041001 000024 MOV #DMD!MUR!DBEN, RMMR1(RO) ; LOAD RMMR1
11537
11538 045260 012760 000000 000006 MOV #0, RMDA(RO) ; LOAD RMDA
11539
11540 045266 012760 000000 000034 MOV #0, RMDC(RO) ; LOAD RMDC
11541
11542 045274 012760 000000 000014 MOV #0, RMER1(RO) ; LOAD RMER1
11543
11544 045302 012760 000000 000042 MOV #0, RMER2(RO) ; LOAD RMER2
11545
11546 045310 012760 000005 000000 MOV #SEEK!GO, RMCS1(RO) ; LOAD RMCS1
11547 ; STEP THE COMMAND SEQUENCER TO FIRST TEST FOR ABORT (3 CLOCKS)
11548 045316 012702 000003 ; MOV #_, R2
11549 045322 70$:
11550
11551 045322 012760 141001 000024 MOV #DMD!MUR!DBEN!DBCK, RMMF1(RO) ; LOAD RMMR1
11552
11553 045330 012760 041001 000024 MOV #DMD!MUR!DBEN, RMMR1(RO) ; LOAD RMMR1
11554 045336 005302 DEC R2
11555 045340 001370 BNE 70$
11556 ; SET DRIVE FAULT TO CAUSE ABORT CONDITION
11557
11558 045342 012760 041101 000024 MOV #DMD!MUR!DBEN!MDF, RMMR1(RO) ; LOAD RMMR1
11559 ; STEP 2 CLOCKS AND VERIFY GO IS RESET
11560 045350 012702 000002 ; MOV #2, R2
11561 045354 80$:
11562
11563 045354 012760 141101 000024 MOV #DMD!MUR!DBEN!MDF!DBCK, RMMR1(RO) ; LOAD RMMR1
11564
11565 045362 012760 041101 000024 MOV #DMD!MUR!DBEN!MDF, RMMR1(RO) ; LOAD RMMR1

```

E03

CZRMJBO RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 237
T114 SEEK TEST

SEQ 0237

```

11566 045370 005302          DEC      R2
11567 045372 001370          BNE      80$
11568
11569 045374 016037 000000 001142  MOV     RMCS1(RO), $BDDAT      ;STORE RMCS1 AT $BDDAT
11570 045402 042737 177776 001142  BIC     #1CGO, $BDDAT
11571 045410 001405          BEQ     90$
11572 045412 010037 001136  MOV     RO, $BDADR
11573 045416 005037 001140  CLR     $GDDAT
11574 045422 104251          ERROR   251      ;GO NOT RESET DUE TO ABORT
11575
11576 045424 012737 045432 001124 90$:  MOV     #100$, $LPERR      ;CHANGE LOOP ON ERROR ADDRESS
11577
11578 ;*****
11579 ;VERIFY OPI SETS IF ON CYLINDER LATCH DOESNT CLEAR
11580
11581 045432          100$:
11582 ;SET VOLUME VALID USING SUBROUTINE
11583 045432 004737 060654          JSR     PC, SETVV      ;GO SET VOLUME VALID
11584 045436 000403          BR     110$      ;BRANCH TO 110$ IF NO ERROR
11585 045440 104000          ERROR
11586 045442 000137 046462          JMP     330$      ;RETURN HERE IF ERROR
11587 045446
11588          110$:
11589 ;
11590 ;ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
11591 ;ADDRESS, AND LOAD SEEK COMMAND
11591 045446 012760 041001 000024  MOV     #DMD!MUR!DBEN, RMMR1(RO) ;LOAD RMMR1
11592
11593 045454 012760 000000 000006  MOV     #0, RMDA(RO)      ;LOAD RMDA
11594
11595 045462 012760 000000 000034  MOV     #0, RMDC(RO)      ;LOAD RMDC
11596
11597 045470 012760 000000 000014  MOV     #0, RMER1(RO)     ;LOAD RMER1
11598
11599 045476 012760 000000 000042  MOV     #0, RMER2(RO)     ;LOAD RMER2
11600
11601 045504 012760 000005 000000  MOV     #SEEK!GO, RMCS1(RO) ;LOAD RMCS1
11602 ;
11603 045512 012702 000017          ;STEP THE COMMAND SEQUENCER
11604 045516          120$:
11605
11606 045516 012760 141001 000024  MOV     #DMD!MUR!DBEN!DBCK, RMMR1(RO) ;LOAD RMMR1
11607
11608 045524 012760 041001 000024  MOV     #DMD!MUR!DBEN, RMMR1(RO) ;LOAD RMMR1
11609 045532 005302          DEC     R2
11610 045534 001370          BNE     120$
11611 ;
11612 ;VERIFY THAT OPI IS SET
11613 045536 016037 000014 001142  MOV     RMER1(RO), $BDDAT      ;STORE RMER1 AT $BDDAT
11614 045544 042737 157777 001142  BIC     #1COPI, $BDDAT
11615 045552 001011          BNE     130$
11616 045554 010037 001136  MOV     RO, $BDADR
11617 045560 062737 000014 001136  ADD     #RMER1, $BDADR
11618 045566 012737 020000 001140  MOV     #OPI, $GDDAT
11619 045574 104252          ERROR   252      ;OPI NOT SET DUE TO ON LATCH
11620
11621 045576 012737 045604 001124 130$:  MOV     #150$, $LPERR      ;CHANGE LOOP ON ERROR ADDRESS

```

F03

CZRMJBD RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 238
T:14 SEEK TEST

SEG 0238

```

11622
11623
11624
11625
11626 045604
11627
11628 045604 004737 060654
11629 045610 000403
11630 045612 104000
11631 045614 000137 046462
11632 045620
11633
11634
11635
11636 045620 012760 041401 000024
11637
11638 045626 012760 000000 000006
11639
11640 045634 012760 000000 000034
11641
11642 045642 012760 000000 000014
11643
11644 045650 012760 000000 000042
11645
11646 045656 012760 000005 000000
11647
11648 045664 012702 000015
11649 045670
11650
11651 045670 012760 141401 000024
11652
11653 045676 012760 041401 000024
11654 045704 005302
11655 045706 001370
11656
11657
11658 045710 012760 041001 000024
11659
11660 045716 012760 041401 000024
11661
11662 045724 012702 000003
11663 045730
11664
11665 045730 012760 141401 000024
11666
11667 045736 012760 041401 000024
11668 045744 005302
11669 045746 001370
11670
11671
11672 045750 016037 000012 001142
11673 045756 042737 077777 001142
11674 045764 001011
11675 045766 012737 100000 001140
11676 045774 010037 001136
11677 046000 062737 000012 001136

```

```

;*****
;VERIFY ATA SETS IF DRIVE COMPLETES SEEK (ON CYLINDER SETS)
150$:
;SET VOLUME VALID USING SUBROUTINE
JSR PC,SETVV ;GO SET VOLUME VALID
BR 160$ ;BRANCH TO 160$ IF NO ERROR
ERROR ;RETURN HERE IF ERROR
JMP 330$
160$:
ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
ADDRESS, AND LOAD SEEK COMMAND
MOV #DMD!MUR!DBEN!MOC,RMMR1(RO) ;LOAD RMMR1
MOV #0,RMDA(RO) ;LOAD RMDA
MOV #0,RMDC(RO) ;LOAD RMDC
MOV #0,RMER1(RO) ;LOAD RMER1
MOV #0,RMER2(RO) ;LOAD RMER2
MOV #SEEK!GO,RMCS1(RO) ;LOAD RMCS1
STEP COMMAND SEQUENCER TO FIRST ON LATCH TEST (13 CLOCKS)
MOV #13,R2
170$:
MOV #DMD!MUR!DBEN!DBCK!MOC,RMMR1(RO) ;LOAD RMMR1
MOV #DMD!MUR!DBEN!MOC,RMMR1(RO) ;LOAD RMMR1
DEC R2
BNE 170$
DROP ON CYLINDER TO RESET LATCH, THEN SET ON CYLINDER
MOV #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
MOV #DMD!MUR!DBEN!MOC,RMMR1(RO) ;LOAD RMMR1
STEP COMMAND SEQUENCER TO SET ATTENTION (3 CLOCKS)
MOV #3,R2
180$:
MOV #DMD!MUR!DBEN!MOC!DBCK,RMMR1(RO) ;LOAD RMMR1
MOV #DMD!MUR!DBEN!MOC,RMMR1(RO) ;LOAD RMMR1
DEC R2
BNE 180$
VERIFY ATA IS SET
MOV RMDS(RO), $BDDAT ;STORE RMDS AT $BDDAT
BIC #1,CATA,$BDDAT
BNE 190$
MOV #ATA,$GDDAT
MOV RO,$BDADR
ADD #RMDS,$BDADR

```


G03

CZRMJ80 RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 239
T114 SEEK TEST

SEQ 0239

```

11678 046006 104253          ERROR 253          ;ATA NOT SET BY SEEK
11679
11680 046010 012737 046016 001124 190$:  MOV      #200$,SLPERR  ;CHANGE LOOP ON ERROR ADDRESS
11681
11682          ;*****
11683          ;VERIFY THAT SEEK ABORTS AFTER EXECUTION DURING WAIT LOOP
11684
11685 046016          200$:
11686          ;SET VOLUME VALID USING SUBROUTINE
11687 046016 004737 060654      JSR      PC,SETVV      ;GO SET VOLUME VALID
11688 046022 000403          BR       210$          ;BRANCH TO 210$ IF NO ERROR
11689 046024 104000          ERROR          ;RETURN HERE IF ERROR
11690 046026 000137 046462      JMP      330$
11691 046032
11692          210$:
11693          ;
11694          ;ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
11695          ;ADDRESS AND LOAD SEEK COMMAND
11695 046032 012760 041401 000024      MOV      #DMD!MUR!DBEN!MOC,RMMR1(RO) ;LOAD RMMR1
11696
11697 046040 012760 000000 000006      MOV      #0,RMDA(RO) ;LOAD RMDA
11698
11699 046046 012760 000000 000034      MOV      #0,RMDC(RO) ;LOAD RMDC
11700
11701 046054 012760 000000 000014      MOV      #0,RMER1(RO) ;LOAD RMER1
11702
11703 046062 012760 000000 000042      MOV      #0,RMER2(RO) ;LOAD RMER2
11704
11705 046070 012760 000005 000000      MOV      #SEEK!GO,RMCS1(RO) ;LOAD RMCS1
11706          ;
11707 046076 012702 000015          STEP COMMAND SEQUENCER TO FIRST ON LATCH TEST (13 CLOCKS)
11708 046102          MOV      #13.,R2
11709          220$:
11710 046102 012760 141401 000024      MOV      #DMD!MUR!DBEN!MOC!DBCK,RMMR1(RO) ;LOAD RMMR1
11711
11712 046110 012760 041401 000024      MOV      #DMD!MUR!DBEN!MOC,RMMR1(RO) ;LOAD RMMR1
11713 046116 005302          DEC      R2
11714 046120 001370          BNE     220$
11715          ;
11716          ;DROP ON CYLINDER TO RESET LATCH, LEAVE ON CYLINDER 0
11717 046122 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
11718          ;
11719 046130 012702 000007          STEP COMMAND SEQUENCER THROUGH WAIT LOOP (7 CLOCKS)
11720 046134          MOV      #7.,R2
11721          230$:
11722 046134 012760 141001 000024      MOV      #DMD!MUR!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1
11723
11724 046142 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
11725 046150 005302          DEC      R2
11726 046152 001370          BNE     230$
11727          ;
11728          ;VERIFY THAT GO IS STILL SET
11729 046154 016037 000000 001142      MOV      RMCS1(RO), $BDDAT ;STORE RMCS1 AT $BDDAT
11730 046162 042737 177776 001142      BIC     #1CGO,$BDDAT
11731 046170 001006          BNE     240$
11732 046172 012737 000001 001140      MOV      #GO,$GDDAT
11733 046200 010037 0C1136          MOV      RO,$BDADR

```

H03

CZRMJBO RM03/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 240
T114 SEEK TEST

SEQ 0240

```

11734 046204 104254          ERROR 254          ;GO RESET EARLY DURING SEEK
11735          :          SET SEEK INCOMPLETE ERROR
11736 046206          240$:
11737
11738 046206 012760 041201 000024      MOV      #DMD!MUR!DBEN!MSER,RMMR1(RO)      ;LOAD RMMR1
11739          :          STEP COMMAND SEQUENCER AND VERIFY GO RESETS (3 CLOCKS)
11740 046214 012702 000003      MOV      #3,R2
11741 046220          250$:
11742
11743 046220 012760 141201 000024      MOV      #DMD!MUR!DBEN!MSER!DBCK,RMMR1(RO)      ;LOAD RMMR1
11744          :
11745 046226 012760 041201 000024      MOV      #DMD!MUR!DBEN!MSER,RMMR1(RO)      ;LOAD RMMR1
11746 046234 005302          DEC      R2
11747 046236 001370          BNE     250$
11748
11749 046240 016037 000000 001142      MOV      RMCS1(RO), $BDDAT          ;STORE RMCS1 AT $BDDAT
11750 046246 042737 177776 001142      BIC     #1CGO,$BDDAT
11751 046254 001405          BEQ     260$
11752 046256 010037 001136          MOV      RO,$BDADR
11753 046262 005037 001140          CLR     $GDDAT
11754 046266 104255          ERROR 255          ;GO NOT RESET DUE TO WAIT ABORT
11755
11756 046270 012737 046302 001124 260$:  MOV      #300$,$LPERR          ;CHANGE LOOP ON ERROR ADDRESS
11757 046276 012703 000001          MOV      #1,R3          ;INITIALIZE CYLINDER ADDRESS
11758
11759          ;;*****
11760          ;VERIFY THE TAG BUS DURING SEEK
11761
11762 046302          300$:
11763          ;SET VOLUME VALID USING SUBROUTINE
11764 046302 004737 060654          JSR     PC,SETVV          ;GO SET VOLUME VALID
11765 046306 000402          BR     310$          ;BRANCH TO 310$ IF NO ERROR
11766 046310 104000          ERROR          ;RETURN HERE IF ERROR
11767 046312 000463          BR     330$
11768 046314
11769          310$:
11770          ; ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
11771          ; ADDRESS AND LOAD SEEK COMMAND
11772 046314 012760 041401 000024      MOV      #DMD!MUR!DBEN!MOC,RMMR1(RO)      ;LOAD RMMR1
11773          :
11774 046322 012760 000000 000006      MOV      #0,RMDA(RO)          ;LOAD RMDA
11775          :
11776 046330 010360 000034          MOV      R3,RMDC(RO)          ;LOAD RMDC
11777          :
11778 046334 012760 000000 000014      MOV      #0,RMER1(RO)          ;LOAD RMER1
11779          :
11780 046342 012760 000000 000042      MOV      #0,RMER2(RO)          ;LOAD RMER2
11781          :
11782 046350 012760 000005 000000      MOV      #SEEK!GO,RMCS1(RO)          ;LOAD RMCS1
11783 046356 012702 046476          MOV      #400$,R2          ;INITIALIZE TABLE POINTER
11784          :          VERIFY TAG BUS ACCORDING TO TABLE AND CYLINDER IN R3
11785 046362          315$:
11786
11787 046362 016037 000040 001142      MOV      RMMR2(RO), $BDDAT          ;STORE RMMR2 AT $BDDAT
11788 046370 042737 150000 001142      BIC     #RQA!RQB!1ST,$BDDAT
11789 046376 011237 001140          MOV      .R2,$GDDAT

```

```

11790 046402 050337 001140      BIS      R3,$GDDAT      ;OR CYLINDER ADDRESS IN
11791 046406 023737 001140 001142      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED AND RECEIVED
11792 046414 001407          BEQ      320$          ;BRANCH IF TAG BUS OK
11793 046416 010037 001136      MOV      R0,$BDADR
11794 046422 062737 000040 001136      ADD      #RMMR2,$BDADR
11795 046430 104256          ERROR   256          ;INCORRECT TAG BUS DURING SEEK
11796 046432 000413          BR      330$
11797 046434          320$:
11798          ; ADVANCE TO NEXT ENTRY IN TABLE-EXIT IF DONE
11799 046434 062702 000002      ADD      #2,R2
11800 046440 005712          TST      (R2)
11801 046442 100407          BMI      330$          ;EXIT IF ENTRY NEGATIVE
11802          ; STEP THE COMMAND SEQUENCER AND REPEAT VERIFICATION
11803
11804 046444 012760 151001 000024      MOV      #DMD!DBEN!MUR!MOV!DBCK,RMMR1(R0) ;LOAD RMMR1
11805
11806 046452 012760 041401 000024      MOV      #DMD!DBEN!MUR!MOC,RMMR1(R0) ;LOAD RMMR1
11807 046460 000740          BR      315$
11808 046462          330$:
11809          ;REPEAT TAG BUS TEST FOR EACH PRIME CYLINDER, I.E.. 1,2,4,...
11810
11811 046462 006303          ASL      R3          ;SHIFT TO NEXT CYLINDER
11812 046464 020327 000512      CMP      R3,#512
11813 046470 101001          BHI      340$          ;EXIT IF 512 WAS DONE
11814 046472 000703          BR      300$          ;TEST NEXT CYLINDER
11815 046474 000416          340$: BR      500$          ;JUMP OVER TABLE
11816
11817 046476          400$:
11818
11819          ;TABLE OF TAG BUS CONTROL AND BIT VALUES
11820
11821 046476 001777          .WORD   1777          ;BUS BITS AT HIGH IMPEDANCE STATE
11822 046500 001777          .WORD   1777
11823 046502 004000          .WORD   CC           ;CONTROL BITS ENABLED, BIT 6 ON
11824 046504 004000          .WORD   CC
11825 046506 024000          .WORD   TAG!CC       ;TAG COMES ON
11826 046510 024000          .WORD   TAG!CC
11827 046512 024000          .WORD   TAG!CC
11828 046514 024000          .WORD   TAG!CC
11829 046516 024000          .WORD   TAG!CC
11830 046520 024000          .WORD   TAG!CC
11831 046522 004000          .WORD   CC           ;TAG GOES OFF
11832 046524 004000          .WORD   CC
11833 046526 001777          .WORD   1777          ;CONTROL BITS DISABLED
11834
11835 046530 177777          .WORD   -1          ;END OF TABLE
11836
11837 046532          500$:
11838          ;END OF TEST
11839          ;*****
11840          ;TEST 115 SEARCH TEST
11841          ;*****
11842          ;*****
11843 046532 000004          $T115: SCOPE
11844 046534 012737 000115 001226      MOV      #115,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
11845

```

```

11846 046542 000240          NOP
11847 046544 012737 000024 001120  MOV    #20,$ICNT      ;20 ITERATIONS
11848 046552 112737 000001 001131  MOVVB  #1,$ERMAX      ;ONE ERROR ALLOWED
11849 046560 012737 046574 001122  MOV    #T115,$LPADR  ;LOAD LOOP ON TEST ADDRESS
11850 046566 012737 046574 001124  MOV    #T115,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
11851 046574          T115:
11852 046574 012706 001100          MOV    #STACK,SP     ;LOAD THE STACK POINTER
11853 046600 013700 001276          MOV    $BASE,R0      ;R0 = UNIBUS ADDRESS OF UUT
11854 046604 013701 001456          MOV    TSTQUE,R1     ;R1 = POINTER TO DEVICE
11855
11856 ;*****
11857 ;VERIFY THAT OPI SETS IF UNIT READY DROPS DURING COMMAND EXECUTION
11858
11859 ;SET VOLUME VALID USING SUBROUTINE
11860 046610 004737 060654          JSR    PC,SETVV      ;CO SET VOLUME VALID
11861 046614 000403          BR    10$           ;BRANCH TO 10$ IF NO ERROR
11862 046616 104000          ERROR  ;RETURN HERE IF ERROR
11863 046620 000137 050632          JMP    330$
11864 046624          10$:
11865 ;
11866 ;   ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
11867 ;   ADDRESS, AND LOAD SEARCH COMMAND
11868 046624 012760 041001 000024          MOV    #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
11869
11870 046632 012760 000000 000006          MOV    #0,RMDA(R0)    ;LOAD RMDA
11871
11872 046640 012760 000000 000034          MOV    #0,RMDC(R0)    ;LOAD RMDC
11873
11874 046646 012760 000000 000014          MOV    #0,RMER1(R0)   ;LOAD RMER1
11875
11876 046654 012760 000000 000042          MOV    #0,RMER2(R0)   ;LOAD RMER2
11877
11878 046662 012760 000031 000000          MOV    #SEARCH!GO,RMCS1(R0) ;LOAD RMCS1
11879 ; STEP COMMAND SEQUENCER TO SEARCH COM (2 CLOCKS)
11880 046670 012702 000002          MOV    #2,R2
11881 046674          20$:
11882
11883 046674 012760 141001 000024          MOV    #DMD!MUR!DBEN!DBCK,RMMR1(R0) ;LOAD RMMR1
11884
11885 046702 012760 041001 000024          MOV    #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
11886 046710 005302          DEC    R2
11887 046712 001370          BNE   20$
11888 ; DROP UNIT READY AND STEP COMMAND SEQUENCER (2 CLOCKS)
11889
11890 046714 012760 040001 000024          MOV    #DMD!DBEN,RMMR1(R0) ;LOAD RMMR1
11891 046722 012702 000002          MOV    #2,R2
11892 046726          30$:
11893
11894 046726 012760 140001 000024          MOV    #DMD!DBEN!DBCK,RMMR1(R0) ;LOAD RMMR1
11895
11896 046734 012760 040001 000024          MOV    #DMD!DBEN,RMMR1(R0) ;LOAD RMMR1
11897 046742 005302          DEC    R2
11898 046744 001370          BNE   30$
11899 ; VERIFY THAT OPI IS SET
11900
11901 046746 016037 000014 001142          MOV    RMER1(R0),$BDDAT ;STORE RMER1 AT $BDDAT

```

K03

CZRMJ80 RMD3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 243
T115 SEARCH TEST

SEQ 0243

```

11902 046754 042737 157777 001142      BIC      #1COPI,$BDDAT
11903 046762 001011                      BNE      40$
11904 046764 012737 020000 001140      MOV      #OPI,$GDDAT
11905 046772 010037 001136      MOV      RO,$BDADR
11906 046776 062737 000014 001136      ADD      #RMR1,$BDADR
11907 047004 104257                      ERROR    257      ;OPI NOT SET DURING SEARCH
11908
11909 047006 012737 047014 001124 40$:      MOV      #50$,$LPERR      ;CHANGE LOOP ON ERROR ADDRESS
11910
11911      ;*****
11912      ;VERIFY THAT SEARCH ABORTS DURING EXECUTION
11913
11914 047014      50$:
11915      ;SET VOLUME VALID USING SUBROUTINE
11916 047014 004737 060654      JSR      PC,SETVV      ;GO SET VOLUME VALID
11917 047020 000403      BR      60$      ;BRANCH TO 60$ IF NO ERROR
11918 047022 104000      ERROR    ;RETURN HERE IF ERROR
11919 047024 000137 050632      JMP      330$
11920
11921 047030      60$:
11922      ;
11923      ; ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
11924      ; ADDRESS, AND LOAD SEARCH COMMAND
11925 047030 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
11926
11927 047036 012760 000000 000006      MOV      #0,RMDA(RO)      ;LOAD RMDA
11928
11929 047044 012760 000000 000034      MOV      #0,RMDC(RO)      ;LOAD RMDC
11930
11931 047052 012760 000000 000014      MOV      #0,RMER1(RO)      ;LOAD RMER1
11932
11933 047060 012760 000000 000042      MOV      #0,RMER2(RO)      ;LOAD RMER2
11934
11935 047066 012760 000031 000000      MOV      #SEARCH!GO,RMCS1(RO) ;LOAD RMCS1
11936      ; STEP THE COMMAND SEQUENCER TO FIRST TEST FOR ABORT (3 CLOCKS)
11937 047074 012702 000003      MOV      #3,R2
11938 047100      70$:
11939
11940 047100 012760 141001 000024      MOV      #DMD!MUR!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1
11941
11942 047106 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
11943 047114 005302      DEC      R2
11944 047116 001370      BNE      70$
11945      ; SET DRIVE FAULT TO CAUSE ABORT CONDITION
11946
11947 047120 012760 041101 000024      MOV      #DMD!MUR!DBEN!MDF,RMMR1(RO) ;LOAD RMMR1
11948      ; STEP 2 CLOCKS AND VERIFY GO IS RESET
11949 047126 012702 000002      MOV      #2,R2
11950 047132      80$:
11951
11952 047132 012760 141101 000024      MOV      #DMD!MUR!DBEN!MDF!DBCK,RMMR1(RO) ;LOAD RMMR1
11953
11954 047140 012760 041101 000024      MOV      #DMD!MUR!DBEN!MDF,RMMR1(RO) ;LOAD RMMR1
11955 047146 005302      DEC      R2
11956 047150 001370      BNE      80$
11957

```

```

11958 047152 016037 000000 001142      MOV      RMCS1(RO), $BDDAT      ;STORE RMCS1 AT $BDDAT
11959 047160 042737 177776 001142      BIC      #1CGO, $BDDAT
11960 047166 001405                BEQ      90$
11961 047170 010037 001136      MOV      RO, $BDADR
11962 047174 005037 001140      CLR      $GDDAT
11963 047200 104260                ERROR   260                    ;GO NOT RESET DUE TO ABORT
11964
11965                ;(THE OTHER TWO ABORT TESTS IN THE COMMAND SEQUENCER ARE TESTED
11966                ;DURING DATA COMMAND TESTS)
11967
11968 047202 012737 047210 001124 90$:      MOV      #100$, $LPERR      ;CHANGE LOOP ON ERROR ADDRESS
11969
11970                ;*****
11971                ;VERIFY OPI SETS IF ON CYLINDER LATCH DOESNT CLEAR
11972
11973 047210      100$:
11974                ;SET VOLUME VALID USING SUBROUTINE
11975 047210 004737 060654      JSR      PC SET JV          ;GO SET VOLUME VALID
11976 047214 000403                BR       110$              ;BRANCH TO 110$ IF NO ERROR
11977 047216 104000                ERROR   ;RETURN HERE IF ERROR
11978 047220 000137 050632      JMP      330$
11979 047224      110$:
11980                ;
11981                ;
11982                ;
11983 047224 012760 041001 000024      MOV      #DMD!MUR!DBEN, RMMR1(RO) ;LOAD RMMR1
11984
11985 047232 012760 000000 000006      MOV      #0, RMDA(RO)      ;LOAD RMDA
11986
11987 047240 012760 000000 000034      MOV      #0, RMDC(RO)      ;LOAD RMDC
11988
11989 047246 012760 000000 000014      MOV      #0, RMER1(RO)     ;LOAD RMER1
11990
11991 047254 012760 000000 000042      MOV      #0, RMER2(RO)     ;LOAD RMER2
11992
11993 047262 012760 000031 000000      MOV      #SEARCH!GO, RMCS1(RO) ;LOAD RMCS1
11994                ;
11995 047270 012702 000023                STEP THE COMMAND SEQUENCER
11996 047274      120$:
11997
11998 047274 012760 141001 000024      MOV      #DMD!MUR!DBEN!DBCK, RMMR1(RO) ;LOAD RMMR1
11999
12000 047302 012760 041001 000024      MOV      #DMD!MUR!DBEN, RMMR1(RO) ;LOAD RMMR1
12001 047310 005302                DEC      R2
12002 047312 001370                BNE     120$
12003                ;
12004                ;
12005 047314 016037 000014 001142      MOV      RMER1(RO), $BDDAT      ;STORE RMER1 AT $BDDAT
12006 047322 042737 157777 001142      BIC      #1COPI, $BDDAT
12007 047330 001011                BNE     130$
12008 047332 010037 001136      MOV      RO, $BDADR
12009 047336 062737 000014 001136      ADD      #RMER1, $BDADR
12010 047344 012737 020000 001140      MOV      #OPI, $GDDAT
12011 047352 104261                ERROR   261                    ;OPI NOT SET DUE TO ON LATCH
12012
12013 047354 012737 047362 001124 130$:      MOV      #150$, $LPERP      ;CHANGE LOOP ON ERROR ADDRESS

```

M03

CZRMJBO RM03/2 OSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 245
1115 SEARCH TEST

SEQ 0245

```

12014
12015
12016
12017
12018
12019 047362
12020
12021 047362 004737 060654
12022 047366 000403
12023 047370 104000
12024 047372 000137 050632
12025 047376
12026
12027
12028
12029 047376 012760 041401 000024
12030
12031 047404 012760 000000 000006
12032
12033 047412 012760 000000 000034
12034
12035 047420 012760 000000 000014
12036
12037 047426 012760 000000 000042
12038
12039 047434 012760 000031 000000
12040
12041 047442 012702 000021
12042 047446
12043
12044 047446 012760 141401 000024
12045
12046 047454 012760 041401 000024
12047 047462 005302
12048 047464 001370
12049
12050
12051
12052
12053 047466 012760 041005 000024
12054
12055
12056 047474 012760 051401 000024
12057
12058 047502 012702 000002
12059 047506
12060
12061 047506 012760 151401 000024
12062
12063 047514 012760 051401 000024
12064 047522 005302
12065 047524 001370
12066
12067
12068
12069 047526 012760 051403 000024

```

```

*****
:VERIFY ATA SETS IF DRIVE COMPLETES SEARCH (ON CYLINDER AND
:SECTOR COMPARE SETS)
150$:
:SET VOLUME VALID USING SUBROUTINE
      JSR      PC_SETVV      ;GO SET VOLUME VALID
      BR      160$          ;BRANCH TO 160$ IF NO ERROR
      ERROR   ;RETURN HERE IF ERROR
      JMP      330$
160$:
:ENABLE DEBUG CLOCK, LOAD CYLINDER, TRP < AND SECTOR
:ADDRESS, AND LOAD SEARCH COMMAND
      MOV      #DMD!MUR!DBEN!MOC,RMMR1(RO) ;LOAD RMMR1
170$:
      MOV      #0,RMDA(RO) ;LOAD RMDA
      MOV      #0,RMDC(RO) ;LOAD RMDC
      MOV      #0,RMER1(RO) ;LOAD RMER1
      MOV      #0,RMER2(RO) ;LOAD RMER2
      MOV      #SEARCH!GO,RMCS1(RO) ;LOAD RMCS1
      STEP COMMAND SEQUENCER TO FIRST ON LATCH TEST (17 CLOCKS)
      MOV      #17,R2
180$:
      MOV      #DMD!MUR!DBEN!DBCK!MOC,RMMR1(RO) ;LOAD RMMR1
      MOV      #DMD!MUR!DBEN!MOC,RMMR1(RO) ;LOAD RMMR1
      DEC      R2
      BNE     170$
:DROP ON CYLINDER TO RESET LATCH,AND RAISE INDEX PULSE
:TO SET FORMAT CHANGE FLOP
      MOV      #DMD!MUR!DBEN!MI,RMMR1(RO) ;LOAD RMMR1
      RAISE ON CYLINDER AND INHIBIT SEARCH TIMEOUT
      MOV      #DMD!MUR!DBEN!MOC!DTO,RMMR1(RO) ;LOAD RMMR1
      STEP COMMAND SEQUENCER TO SEARCH ENABLE (2 CLOCKS)
      MOV      #2,R2
180$:
      MOV      #DMD!MUR!DBEN!MOC!DTO!DBCK,RMMR1(RO) ;LOAD RMMR1
      MOV      #DMD!MUR!DBEN!MOC!DTO,RMMR1(RO) ;LOAD RMMR1
      DEC      R2
      BNE     180$
:FORCE SECTOR COMPARE BY CLOCKING SECTOR PULSE WITH SECTOR COMPARE
:ACTIVE
      MOV      #DMD!MUR!MOC!DBEN!DTO!MSC,RMMR1(RO) ;LOAD RMMR1

```

```

12070
12071 047534 012760 051443 000024      MOV      #DMD!MUR!MOC!DBEN!DTO!MSC!MS,RMMR1(RO) ;LOAD RMMR1
12072
12073 047542 012760 051403 000024      MOV      #DMD!MUR!MOC!DBEN!DTO!MSC,RMMR1(RO) ;LOAD RMMR1
12074 ;CLOCK SEQUENCER TO SET ATA (3 CLOCKS)
12075 047550 012702 000003      MOV      #3,R2 ;R2 = CLOCK COUNT
12076 047554      185$:
12077
12078 047554 012760 151401 000024      MOV      #DMD!MUR!MOC!DBEN!DTO!DBCK,RMMR1(RO) ;LOAD RMMR1
12079
12080 047562 012760 051401 000024      MOV      #DMD!MUR!MOC!DBEN!DTO,RMMR1(RO) ;LOAD RMMR1
12081 047570 005302      DEC      R2
12082 047572 001370      BNE     185$
12083 ;
12084 ;VERIFY ATA IS SET
12085 047574 016037 000012 001142      MOV      RMDS(RO),SBDDAT ;STORE RMDS AT SBDDAT
12086 047602 042737 077777 001142      BIC     #!CATA,SBDDAT
12087 047610 001011      BNE     190$
12088 047612 012737 100000 001140      MOV      #ATA,SGDDAT
12089 047620 010037 001136      MOV      RO,SBODAR
12090 047624 062737 000012 001136      ADD     #RMDS,SBODAR
12091 047632 104262      ERROR   262 ;ATA NOT SET BY SEARCH
12092
12093 047634 012737 047642 001124 190$: MOV      #200$,SLPERR ;CHANGE LOOP ON ERROR ADDRESS
12094
12095 ;*****
12096 ;VERIFY THAT SEARCH ABORTS AFTER EXECUTION DURING SEARCH SEEK LOOP
12097
12098 047642      200$:
12099 ;SET VOLUME VALID USING SUBROUTINE
12100 047642 004737 060654      JSR     PC SETVV ;GO SET VOLUME VALID
12101 047646 000403      BR     210$ ;BRANCH TO 210$ IF NO ERROR
12102 047650 104000      ERROR ;RETURN HERE IF ERROR
12103 047652 000137 050632      JMP     330$
12104 047656      210$:
12105 ;
12106 ;ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
12107 ;ADDRESS AND LOAD SEARCH COMMAND
12108 047656 012760 041401 000024      MOV      #DMD!MUR!DBEN!MOC,RMMR1(RO) ;LOAD RMMR1
12109
12110 047664 012760 000000 000006      MOV      #0,RMDA(RO) ;LOAD RMDA
12111
12112 047672 012760 000000 000034      MOV      #0,RMDC(RO) ;LOAD RMDC
12113
12114 04770 012760 000000 000014      MOV      #0,RMER1(RO) ;LOAD RMER1
12115
12116 047706 012760 000000 000042      MOV      #0,RMER2(RO) ;LOAD RMER2
12117
12118 047714 012760 000031 000000      MOV      #SEARCH!GO,RMCS1(RO) ;LOAD RMCS1
12119 ;STEP COMMAND SEQUENCER TO FIRST ON LATCH TEST (17 CLOCKS)
12120 047722 012702 000021      MOV      #17.,R2
12121 047726      220$:
12122
12123 047726 012760 141401 000024      MOV      #DMD!MUR!DBEN!MOC!DBCK,RMMR1(RO) ;LOAD RMMR1
12124
12125 047734 012760 041401 000024      MOV      #DMD!MUR!DBEN!MOC,RMMR1(RO) ;LOAD RMMR1

```



```

12126 047742 005302          DEC      R2
12127 047744 001370          BNE      220$
12128 ;                          DROP ON CYLINDER TO RESET LATCH, LEAVE ON CYLINDER 0
12129
12130 047746 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
12131 ;                          STEP COMMAND SEQUENCER THROUGH WAIT LOOP (7 CLOCKS)
12132 047754 012702 000007          MOV      #7,R2
12133 047760          230$:
12134
12135 047760 012760 141001 000024      MOV      #DMD!MUR!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1
12136
12137 047766 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
12138 047774 005302          DEC      R2
12139 047776 001370          BNE      230$
12140 ;                          VERIFY THAT GO IS STILL SET
12141
12142 050000 016037 000000 001142      MOV      RMCS1(RO), $BDDAT ;STORE RMCS1 AT $BDDAT
12143 050006 042737 177776 001142      BIC      #1CGO,$BDDAT
12144 050014 001006          BNE      240$
12145 050016 012737 000001 001140      MOV      #GO,$GDDAT
12146 050024 010037 001136          MOV      RO,$BDADR
12147 050030 104263          ERROR    263 ;GO RESET EARLY DURING SEARCH
12148 ;                          SET SEEK INCOMPLETE ERROR
12149          240$:
12150
12151 050032 012760 041201 000024      MOV      #DMD!MUR!DBEN!MSER,RMMR1(RO) ;LOAD RMMR1
12152 ;                          STEP COMMAND SEQUENCER AND VERIFY GO RESETS (3 CLOCKS)
12153 050040 012702 000003          MOV      #3,R2
12154 050044          250$:
12155
12156 050044 012760 141201 000024      MOV      #DMD!MUR!DBEN!MSER!DBCK,RMMR1(RO) ;LOAD RMMR1
12157
12158 050052 012760 041201 000024      MOV      #DMD!MUR!DBEN!MSER,RMMR1(RO) ;LOAD RMMR1
12159 050060 005302          DEC      R2
12160 050062 001370          BNE      250$
12161
12162 050064 016037 000000 001142      MOV      RMCS1(RO), $BDDAT ;STORE RMCS1 AT $BDDAT
12163 050072 042737 177776 001142      BIC      #1CGO,$BDDAT
12164 050100 001405          BEQ      260$
12165 050102 010037 001136          MOV      RO,$BDADR
12166 050106 005037 001140          CLR      $GDDAT
12167 050112 104264          ERROR    264 ;GO NOT RESET DUE TO WAIT ABORT
12168
12169 050114 012737 050122 001124 260$: MOV      #265$, $LPERR ;CHANGE LOOP ON ERROR ADDRESS
12170
12171 ;*****
12172 ;VERIFY THAT SEARCH ABORTS DURING SECTOR COMPARE LOOP
12173
12174 050122          265$:
12175 ;SET VOLUME VALID USING SUBROUTINE
12176 050122 004737 060654          JSR      PC SETVV ;GO SET VOLUME VALID
12177 050126 000403          BR       270$ ;BRANCH TO 270$ IF NO ERROR
12178 050130 104000          ERROR   ;RETURN HERE IF ERROR
12179 050132 000137 050632          JMP      330$
12180 050136          270$:
12181 ;                          ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR

```

```

12182 ; ADDRESS, AND LOAD SEARCH COMMAND
12183
12184 050136 012760 041401 000024 MOV #DMD!MUR!DBEN!MOC,RMMR1(RO) ;LOAD RMMR1
12185
12186 050144 012760 000000 000006 MOV #0,RMDA(RO) ;LOAD RMDA
12187
12188 050152 012760 000000 000034 MOV #0,RMDC(RO) ;LOAD RMDC
12189
12190 050160 012760 000000 000014 MOV #0,RMER1(RO) ;LOAD RMER1
12191
12192 050166 012760 000000 000042 MOV #0,RMER2(RO) ;LOAD RMER2
12193
12194 050174 012760 000031 000000 MOV #SEARCH!GO,RMCS1(RO) ;LOAD RMCS1
12195 ; STEP COMMAND SEQUENCER TO FIRST ON LATCH TEST (17 CLOCKS)
12196 050202 012702 000021 ; MOV #17,R2
12197 050206 275$:
12198
12199 050206 012760 141401 000024 MOV #DMD!MUR!DBEN!DBCK!MOC,RMMR1(RO) ;LOAD RMMR1
12200
12201 050214 012760 041401 000024 MOV #DMD!MUR!DBEN!MOC,RMMR1(RO) ;LOAD RMMR1
12202 050222 005302 DEC #5
12203 050224 001370 BNE #5$
12204 ; DROP ON CYLINDER TO RESET LATCH, AND RAISE INDEX PULSE
12205 ; TO SET FORMAT CHANGE FLOP
12206
12207
12208 050226 012760 041005 000024 MOV #DMD!MUR!DBEN!MI,RMMR1(RO) ;LOAD RMMR1
12209 ; RAISE ON CYLINDER AND INHIBIT SEARCH TIMEOUT
12210
12211 050234 012760 051401 000024 MOV #DMD!MUR!DBEN!MOC!DTO,RMMR1(RO) ;LOAD RMMR1
12212 ; STEP COMMAND SEQUENCER TO SEARCH ENABLE (2 CLOCKS)
12213 050242 012702 000002 ; MOV #2,R2
12214 050246 280$:
12215
12216 050246 012760 151401 000024 MOV #DMD!MUR!DBEN!MOC!DTO!DBCK,RMMR1(RO) ;LOAD RMMR1
12217
12218 050254 012760 051401 000024 MOV #DMD!MUR!DBEN!MOC!DTO,RMMR1(RO) ;LOAD RMMR1
12219 050262 005302 DEC R2
12220 050264 001370 BNE 280$
12221 ; VERIFY THAT SEARCH ENABLE IS ON DURING SECTOR COMPARE LOOP
12222 050266 012702 000004 ; MOV #4,R2 ;R2 = CLOCK COUNT
12223 050272 281$:
12224
12225 050272 016037 000024 001142 MOV RMMR1(RO), $BDDAT ;STORE RMMR1 AT $BDDAT
12226 050300 042737 173777 001142 BIC #CESAC,$BDUHI
12227 050306 001411 BEQ 282$ ;BRANCH IF SEARCH NOT ENABLED
12228
12229 050310 012760 151401 000024 MOV #DMD!MUR!MOC!DBEN!DTO!DBCK,RMMR1(RO) ;LOAD RMMR1
12230
12231 050316 012760 051401 000024 MOV #DMD!MUR!MOC!DBEN!DTO,RMMR1(RO) ;LOAD RMMR1
12232 050324 005302 DEC R2
12233 050326 001361 BNE 281$
12234 050330 000411 BR 283$
12235 050332 012737 004000 001140 282$: MOV #ESRC,$GDDAT
12236 050340 010037 001136 MOV RO,$BDADR
12237 050344 062737 000024 001136 ADD #RMMR1,$BDADR

```

```

12238 050352 104265          ERROR 265          ;SEARCH NOT ENABLED
12239 050354          283$:
12240          ; SET DRIVE FAULT TO CAUSE ABORT CONDITION
12241
12242 050354 012760 051501 000024      MOV      #DMD!MUR!DBEN!MOC!DTO!MDF,RMMR1(RO) ;LOAD RMMR1
12243          ; STEP 2 CLOCKS AND VERIFY GO IS RESET
12244 050362 012702 000002      MOV      #2,R2          ;R2 = CLOCK COUNT
12245 050366          285$:
12246
12247 050366 012760 151501 000024      MOV      #DMD!MUR!DBEN!MOC!DTO!MDF!DBCK,RMMR1(RO) ;LOAD RMMR1
12248
12249 050374 012760 051501 000024      MOV      #DMD!MUR!DBEN!MOC!DTO!MDF,RMMR1(RO) ;LOAD RMMR1
12250 050402 005302
12251 050404 001370      DEC      R2
12252          BNE      285$
12253 050406 016037 000000 001142      MOV      RMCS1(RO), $BDDAT          ;STORE RMCS1 AT $BDDAT
12254 050414 042737 177776 001142      BIC      #1CGO,$BDDAT
12255 050422 001406
12256 050424 012737 000000 001140      BEQ      290$
12257 050432 010037 001136      MOV      #0,$GDDAT
12258 050436 104260      MOV      RO,$BDADR
12259          ERROR 260          ;GO NOT RESET DURING SECTOR COMPARE
12260 050440 012737 050452 001124 290$: MOV      #300$, $LPERR          ;CHANGE LOOP ON ERROR ADDRESS
12261 050446 012703 000001      MOV      #1,R3          ;INITIALIZE CYLINDER ADDRESS
12262
12263          ;*****
12264          ;VERIFY THE TAG BUS DURING SEARCH
12265
12266 050452          300$:
12267          ;SET VOLUME VALID USING SUBROUTINE
12268 050452 004737 060654      JSR      PC,SETVV          ;GO SET VOLUME VALID
12269 050456 000402      BR      310$          ;BRANCH TO 310$ IF NO ERROR
12270 050460 104000      ERROR          ;RETURN HERE IF ERROR
12271 050462 000463      BR      330$
12272 050464          310$:
12273          ; ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
12274          ; ADDRESS AND LOAD SEARCH COMMAND
12275
12276 050464 012760 041401 000024      MOV      #DMD!MUR!DBEN!MOC,RMMR1(RO) ;LOAD RMMR1
12277
12278 050472 012760 000000 000006      MOV      #0,RMDA(RO) ;LOAD RMDA
12279
12280 050500 010360 000034      MOV      R3,RMDC(RO) ;LOAD RMDC

```

E04

CZRMJ80 RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 250
T115 SEARCH TEST

SEG 0250

```

12281
12282 050504 012760 000000 000014      MOV      #0,RMER1(RO)      ;LOAD RMER1
12283
12284 050512 012760 000000 000042      MOV      #0,RMER2(RO)      ;LOAD RMER2
12285
12286 050520 012760 000031 000000      MOV      #SEARCH!GO,RMCS1(RO) ;LOAD RMCS1
12287 050526 012702 050646      MOV      #400$,R2          ;INITIALIZE TABLE POINTER
12288                                VERIFY TAG BUS ACCORDING TO TABLE AND CYLINDER IN R3
12289 050532                                ;315$:
12290
12291 050532 016037 000040 001142      MOV      RMMR2(RO), $BDDAT ;STORE RMMR2 AT $BDDAT
12292 050540 042737 150000 001142      BIC      #RQA!RQB!TST,$BDDAT
12293 050546 011237 001140      MOV      (R2), $GDDAT
12294 050552 050337 001140      BIS      R3,$GDDAT        ;OR CYLINDER ADDRESS IN
12295 050556 023737 001140 001142      CMP      $GDDAT,$BDDAT    ;COMPARE EXPECTED AND RECEIVED
12296 050564 001407 001136      BEQ      320$             ;BRANCH IF TAG BUS OK
12297 050566 010037 001136      MOV      R0,$BDADR
12298 050572 062737 000040 001136      ADD      #RMMR2,$BDADR
12299 050600 104266 000040      ERROR   266             ;INCORRECT TAG BUS DURING SEARCH
12300 050602 000420      BR       340$
12301 050604                                ;320$:
12302                                ;
12303 050604 062702 000002      ADVANCE TO NEXT ENTRY IN TABLE-EXIT IF DONE
12304 050610 005712      ADD      #2,R2
12305 050612 100407      TST      (R2)
12306                                BMI      330$             ;EXIT IF ENTRY NEGATIVE
12307                                ;
12308                                STEP THE COMMAND SEQUENCER AND REPEAT VERIFICATION
12308 050614 012760 141401 000024      MOV      #DMD!DBEN!MUR!MOC!DBCK,RMMR1(RO) ;LOAD RMMR1
12309
12310 050622 012760 041401 000024      MOV      #DMD!DBEN!MUR!MOC,RMMR1(RO) ;LOAD RMMR1
12311 050630 000740      BR       315$
12312 050632                                ;330$:
12313                                ;REPEAT TAG BUS TEST FOR EACH PRIME CYLINDER, I.E., 1,2,4,...
12314
12315 050632 006303 000512      ASL      R3              ;SHIFT TO NEXT CYLINDER
12316 050634 020327 000512      CMP      R3,#512
12317 050640 101001 000512      BHI      340$             ;EXIT IF 512 WAS DONE
12318 050642 000703 000512      BR       300$             ;TEST NEXT CYLINDER
12319 050644 000424 000512      BR       500$             ;JUMP OVER TABLE
12320
12321 050646                                ;400$:
12322                                ;
12323                                ;TABLE OF TAG BUS CONTROL AND BIT VALUES
12324
12325 050646 001777      .WORD   1777             ;BUS BITS AT HIGH IMPEDANCE STATE
12326 050650 001777      .WORD   1777
12327 050652 001777      .WORD   1777
12328 050654 001777      .WORD   1777
12329 050656 001777      .WORD   1777
12330 050660 004000      .WORD   CC              ;CONTROL BITS ENABLED, BIT 6 ON
12331 050662 004000      .WORD   CC
12332 050664 024000      .WORD   TAG!CC          ;TAG COMES ON
12333 050666 024000      .WORD   TAG!CC
12334 050670 024000      .WORD   TAG!CC
12335 050672 024000      .WORD   TAG!CC
12336 050674 024000      .WORD   TAG!CC

```

```

12337 050676 024000 .WORD TAG!CC
12338 050700 004000 .WORD CC ;TAG GOES OFF
12339 050702 004000 .WORD CC
12340 050704 001777 .WORD 1777 ;CONTROL BITS DISABLED
12341 050706 001777 .WORD 1777
12342 050710 001777 .WORD 1777
12343 050712 001777 .WORD 1777
12344
12345 050714 177777 .WORD -1 ;END OF TABLE
12346
12347 050716 500$: ;END OF TEST
12348
12349 ;*****
12350 ;*TEST 116 SEARCH TIMEOUT TEST
12351 ;*****
12352
12353 050716 000004 †ST116: SCOPE
12354 050720 012737 000116 001226 MOV #116,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
12355
12356 050726 000240 NOP
12357 050730 012737 000024 001120 MOV #20,$ICNT ;20 ITERATIONS
12358 050736 112737 000001 001131 MOVB #1,$ERMAX ;ONE ERROR ALLOWED
12359 050744 012737 050760 001122 MOV #T116,$LPADR ;LOAD LOOP ON TEST ADDRESS
12360 050752 012737 050760 001124 MOV #T116,$LPERR ;LOAD LOOP ON ERROR ADDRESS
12361 050760
12362 050760 012706 001100 T116: MOV #STACK,SP ;LOAD THE STACK POINTER
12363 050764 013700 001276 MOV $BASE,R0 ;R0 = UNIBUS ADDRESS OF UUT
12364 050770 013701 001456 MOV TSTQUE,R1 ;R1 = POINTER TO DEVICE
12365 ;SET VOLUME VALID USING SUBROUTINE
12366 050774 004737 060654 JSR PC,SETVV ;GO SET VOLUME VALID
12367 051000 000402 BR 10$ ;BRANCH TO 10$ IF NO ERROR
12368 051002 104000 ERROR ;RETURN HERE IF ERROR
12369 051004 000550 BR 90$
12370 051006
12371 10$: ;ENABLE DEBUG CLOCK AND LOAD SEARCH COMMAND
12372
12373 051006 012760 041401 000024 MOV #DMD!MUR!DBEN!MOC,RMMR1(R0) ;LOAD RMMR1
12374
12375 051014 012760 000000 000014 MOV #0,RMER1(R0) ;LOAD RMER1
12376
12377 051022 012760 000000 000042 MOV #0,RMER2(R0) ;LOAD RMER2
12378
12379 051030 012760 000000 000034 MOV #0,RMDC(R0) ;LOAD RMDC
12380
12381 051036 012760 000000 000006 MOV #0,RMDA(R0) ;LOAD RMDA
12382
12383 051044 012760 000031 000000 MOV #SEARCH!GO, RMCS1(R0) ;LOAD RMCS1
12384 ;EXECUTE SEARCH TO TEST FOR ON LATCH RESET (17 CLOCKS)
12385 051052 012702 000021 MOV #17.,R2
12386 051056
12387
12388 051056 012760 141401 000024 MOV #DMD!MUR!DBEN!MOC!DBCK,RMMR1(R0) ;LOAD RMMR1
12389
12390 051064 012760 041401 000024 MOV #DMD!MUR!DBEN!MOC,RMMR1(R0) ;LOAD RMMR1
12391 051072 005302 DEC R2
12392 051074 001370 BNE 20$

```

```

12393 ;DROP ON CYLINDER TO RESET LATCH, RAISE ON CYLINDER
12394
12395 051076 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
12396
12397 051104 012760 041401 000024      MOV      #DMD!MUR!DBEN!MOC,RMMR1(RO) ;LOAD RMMR1
12398 ;STEP COMMAND SEQUENCER TO SECTOR COMPARE LOOP (2 CLOCKS)
12399 051112 012702 000002      MOV      #2,R2
12400 051116
12401
12402 051116 012760 151401 000024      MOV      #DMD!DBEN!MUR!MOC!DBCK!DTO,RMMR1(RO) ;LOAD RMMR1
12403
12404 051124 012760 051401 000024      MOV      #DMD!DBEN!MUR!MOC!DTO,RMMR1(RO) ;LOAD RMMR1
12405 051132 005302      DEC      R2
12406 051134 001370      BNE      30$
12407 ;THE COMMAND SEQUENCER IS NOW IN THE SECTOR COMPARE LOOP. STEP
12408 ;THROUGH THE LOOP AND VERIFY SEARCH IS ENABLED.
12409 051136 012702 000005      MOV      #5,R2
12410 051142
12411
12412 051142 012760 151401 000024      MOV      #DMD!DBEN!MUR!MOC!DBCK!DTO,RMMR1(RO) ;LOAD RMMR1
12413
12414 051150 012760 051401 000024      MOV      #DMD!DBEN!MUR!MOC!DTO,RMMR1(RO) ;LOAD RMMR1
12415
12416 051156 016037 000024 001142      MOV      RMMR1(RO), $BDDAT ;STORE RMMR1 AT $BDDAT
12417 051164 042737 173777 001142      BIC      #↑CESRC,$BDDAT
12418 051172 001403      BEQ      50$ ;BRANCH IF SEARCH NOT ENABLED
12419 051174 005302      DEC      R2
12420 051176 001361      BNE      40$
12421 051200 000412      BR       60$
12422 051202 012737 004000 001140 50$: MOV      #ESRC,$GDDAT ;SETUP ERROR MSG
12423 051210 010037 001136      MOV      RO,$BDADR
12424 051214 062737 000024 001136      ADD      #RMMR1,$BDADR
12425 051222 104265      ERROR   265 ;SEARCH NOT ENABLED
12426 051224 000440      BR       90$
12427 051226
12428 60$:
12429 ;DROP DTO TO ENABLE SEARCH TIMEOUT AND WAIT FOR OPI TO SET.
12430
12431 051226 012760 041401 000024      MOV      #DMD!MUR!MOC!DBEN,RMMR1(RO) ;LOAD RMMR1
12432 051234 012737 000400 001524      MOV      #400,WATCH ;SET WATCHDOG TIMER VALUE
12433 051242 004777 130260      JSR      PC,@CLOCK ;START THE CLOCK
12434 051246
12435 051246 016037 000014 001142      MOV      RMER1(RO), $BDDAT ;STORE RMER1 AT $BDDAT
12436 051254 042737 157777 001142      BIC      #↑COPI,$BDDAT
12437 051262 001017      BNE      80$
12438 051264 005737 001524      TST      WATCH
12439 051270 001366      BNE      70$
12440 051272 004777 130232      JSR      PC,@STOP ;STOP THE CLOCK
12441 051276 012737 020000 001140      MOV      #OPI,$GDDAT ;SETUP ERROR MSG
12442 051304 010037 001136      MOV      RO,$BDADR
12443 051310 062737 000014 001136      ADD      #RMER1,$BDADR
12444 051316 104267      ERROR   267 ;OPI NOT SET BY SEARCH TIMEOUT
12445 051320 000402      BR       90$
12446 051322
12447 051322 004777 130202 80$: JSR      PC,@STOP ;STOP THE CLOCK
12448

```

```

12449 051326          90$:                               ;END OF TEST
12450
12451                ;*****
12452                ;*TEST 117      DATA COMMAND TESTS (1)
12453
12454                ;*****
12455 051326 000004      †ST117: SCOPE
12456 051330 012737 000117 001226      MOV      #117,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
12457
12458                NOP
12459 051340 012737 000024 001120      MOV      #20,$ICNT      ;20 ITERATIONS
12460 051346 112737 000001 001131      MOV      #1,$ERMAX      ;ONE ERROR ALLOWED
12461 051354 012737 051370 001122      MOV      #T117,$LPADR   ;LOAD LOOP ON TEST ADDRESS
12462 051362 012737 051370 001124      MOV      #T117,$LPERR   ;LOAD LOOP ON ERROR ADDRESS
12463 051370
12464 051370 012706 C01100      T117:    MOV      #STACK,SP      ;LOAD THE STACK POINTER
12465 051374 013700 001276      MOV      $BASE,R0      ;R0 = UNIBUS ADDRESS OF UUT
12466 051400 013701 001456      MOV      TSTQUE,R1     ;R1 = POINTER TO DEVICE
12467
12468                ;*****
12469                ;VERIFY DATA COMMAND SETS OPI IF DRIVE NOT READY
12470                ;SET VOLUME VALID USING SUBROUTINE
12471 051404 004737 060654      JSR      PC,SETVV      ;GO SET VOLUME VALID
12472 051410 000402      BR      10$           ;BRANCH TO 10$ IF NO ERROR
12473 051412 104000      ERROR
12474 051414 000471      BR      40$           ;RETURN HERE IF ERROR
12475
12476                10$:
12477                ; ENABLE DEBUG CLOCK AND LOAD READ COMMAND
12478 051416 012760 041401 000024      MOV      #DMD!MUR!MOC!DBEN,RMMR1(R0) ;LOAD RMMR1
12479 051424 012760 000000 000014      MOV      #0,RMER1(R0) ;LOAD RMER1
12480
12481 051432 012760 000000 000042      MOV      #0,RMER2(R0) ;LOAD RMER2
12482
12483 051440 012760 000000 000006      MOV      #0,RMDA(R0) ;LOAD RMDA
12484
12485 051446 012760 000000 000034      MOV      #0,RMDC(R0) ;LOAD RMDC
12486
12487 051454 012760 000071 000000      MOV      #RD!GO,RMCS1(R0) ;LOAD RMCS1
12488                ; STEP COMMAND SEQUENCER TO UNIT READY TEST (3 CLOCKS)
12489 051462 012702 000003      MOV      #3,R2
12490 051466
12491
12492 051466 012760 141401 000024      MOV      #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0) ;LOAD RMMR1
12493
12494 051474 012760 041401 000024      MOV      #DMD!MUR!MOC!DBEN,RMMR1(R0) ;LOAD RMMR1
12495 051502 005302      DEC      R2
12496 051504 001370      BNE     20$
12497                ; DROP UNIT READY
12498
12499 051506 012760 040401 000024      MOV      #DMD!MOC!DBEN,RMMR1(R0) ;LOAD RMMR1
12500                ; STEP SEQUENCER AND VERIFY OPI SETS (2 CLOCKS)
12501 051514 012702 000002      MOV      #2,R2
12502 051520
12503
12504 051520 C12760 140401 000024      MOV      #DMD!MOC!DBEN!DBCK,RMMR1(R0) ;LOAD RMMR1

```

```

12505
12506 051526 012760 040401 000024      MOV      #DMD!MOC!DBEN,RMMR1(RO) ;LOAD RMMR1
12507 051534 005302                      DEC      R2
12508 051536 001370                      BNE     30$
12509
12510 051540 016037 000014 001142      MOV      RMER1(RO), $BDDAT          ;STORE RMER1 AT $BDDAT
12511 051546 042737 157777 001142      BIC     #↑COPI,$BDDAT
12512 051554 001011                      BNE     40$
12513 051556 012737 020000 001140      MOV      #OPI,$GDDAT
12514 051564 010037 001136                      MOV      RO,$BDADR
12515 051570 062737 000014 001136      ADD     #RMER1,$BDADR
12516 051576 104270                      ERROR   270          ;OPI NOT SET DURING DATA
12517
12518 051600 012737 051606 001124 40$:      MOV      #50$,$LPERR          ;CHANGE LOOP ON ERROR TEST
12519
12520      ;*****
12521      ;VERIFY DATA COMMAND ABORTS AT LOCATION 129
12522
12523 051606      50$:
12524      ;SET VOLUME VALID USING SUBROUTINE
12525 051606 004737 060654      JSR     PC,SETV          ;GO SET VOLUME VALID
12526 051612 000402                      BR      60$          ;BRANCH TO 60$ IF NO ERROR
12527 051614 104000                      ERROR   ;RETURN HERE IF ERROR
12528 051616 000576                      BR      150$
12529 051620      60$:
12530      ;ENABLE DEBUG CLOCK AND LOAD READ COMMAND
12531
12532 051620 012760 041401 000024      MOV      #DMD!MUR!MOC!DBEN,RMMR1(RO) ;LOAD RMMR1
12533
12534 051626 012760 000000 000014      MOV      #0,RMER1(RO) ;LOAD RMER1
12535
12536 051634 012760 000000 000042      MOV      #0,RMER2(RO) ;LOAD RMER2
12537
12538 051642 012760 000000 000006      MOV      #0,RMDA(RO) ;LOAD RMDA
12539
12540 051650 012760 000000 000034      MOV      #0,RMDC(RO) ;LOAD RMDC
12541
12542 051656 012760 000071 000000      MOV      #RD!GO, RMCS1(RO) ;LOAD RMCS1
12543      ;STEP COMMAND SEQUENCER TO ABORT TEST AT LOCATION 129 (4 CLOCKS)
12544 051664 012702 000004      MOV      #4,R2
12545 051670      70$:
12546
12547 051670 012760 141401 000024      MOV      #DMD!MUR!MOC!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1
12548
12549 051676 012760 041401 000024      MOV      #DMD!MUR!MOC!DBEN,RMMR1(RO) ;LOAD RMMR1
12550 051704 005302                      DEC      R2
12551 051706 001370                      BNE     70$
12552      ;SET DEVICE FAULT TO CAUSE ABORT CONDITION
12553
12554 051710 012760 041501 000024      MOV      #DMD!MUR!MOC!DBEN!MDF,RMMR1(RO) ;LOAD RMMR1
12555      ;STEP THE SEQUENCER THROUGH THE TEST FOR ABORT (1 CLOCK)
12556 051716 012702 000001      MOV      #1,R2
12557 051722      80$:
12558
12559 051722 012760 141501 000024      MOV      #DMD!MUR!MOC!DBEN!MDF!DBCK,RMMR1(RO) ;LOAD RMMR1
12560

```



```

12561 051730 012760 041501 000024      MOV      #DMD!MUR!MOC!DBEN!MDF,RMMR1(RO) ;LOAD RMMR1
12562 051736 005302                      DEC      R2
12563 051740 001370                      BNE      80$
12564
12565      ;ABORT EBL SHOULD NOW BE ACTIVE - USE THE MAINTENANCE REGISTER TO
12566      ;FORCE BIT CLOCKS AND VERIFY THAT EBL SETS WITHIN 16 BIT CLOCKS
12567 051742 011732 000020      MOV      #16.,R2 ;MAXIMUM NUMBER OF BIT CLOCKS
12568 051746
12569 85$:
12570 051746 012760 045501 000024      MOV      #DMD!MUR!MOC!DBEN!MDF!MCLK,RMMR1(RO) ;LOAD RMMR1
12571
12572 051754 012760 041501 000024      MOV      #DMD!MUR!MOC!DBEN!MDF,RMMR1(RO) ;LOAD RMMR1
12573
12574 051762 016037 000024 001142      MOV      RMMR1(RO), $BDDAT ;STORE RMMR1 AT $BDDAT
12575 051770 042737 157777 001142      BIC      #↑CEBL,$BDDAT
12576 051776 001014                      BNE      90$ ;BRANCH IF EBL IS SET
12577
12578 052000 005302                      DEC      R2
12579 052002 001361                      BNE      85$ ;CONTINUE BIT CLOCKS IF COUNT NOT 0
12580 052004 012737 020000 001140      MOV      #EBL,$GDDAT
12581 052012 010037 001136      MOV      RO,$BDADR
12582 052016 062737 000024 001136      ADD      #RMMR1,$BDADR
12583 052024 104271                      ERROR    271 ;EBL NOT SET AT DATA ABORT
12584 052026 000472                      BR       150$
12585 052030
12586 90$:
12587 052030 012702 000002      ;STEP THE SEQUENCER THROUGH ITS TEST FOR EBL (2 CLOCKS)
12588 052034      MOV      #2,R2
12589 100$:
12590 052034 012760 141501 000024      MOV      #DMD!MUR!MOC!DBEN!MDF!DBCK,RMMR1(RO) ;LOAD RMMR1
12591
12592 052042 012760 041501 000024      MOV      #DMD!MUR!MOC!DBEN!MDF,RMMR1(RO) ;LOAD RMMR1
12593 052050 005302                      DEC      R2
12594 052052 001370                      BNE      100$
12595
12596      ;ABORT EBL SHOULD NOW BE INACTIVE - FORCE BIT CLOCK USING THE
12597      ;MAINTENANCE REGISTER TO RESET EBL (16 BIT CLOCKS)
12598 052054 012702 000020      MOV      #16.,R2 ;MAXIMUM NUMBER OF BIT CLOCKS
12599 052060
12600 110$:
12601 052060 012760 045501 000024      MOV      #DMD!MUR!MOC!DBEN!MDF!MCLK,RMMR1(RO) ;LOAD RMMR1
12602
12603 052066 012760 041501 000024      MOV      #DMD!MUR!MOC!DBEN!MDF,RMMR1(RO) ;LOAD RMMR1
12604 052074 005302                      DEC      R2
12605 052076 001370                      BNE      110$ ;ISSUE 16 BIT CLOCKS THEN TEST
12606 052100
12607 120$:
12608      ;VERIFY EBL IS NOW RESET
12609 052100 016037 000024 001142      MOV      RMMR1(RO), $BDDAT ;STORE RMMR1 AT $BDDAT
12610 052106 042737 157777 001142      BIC      #↑CEBL,$BDDAT
12611 052114 001411                      BEQ      130$ ;BRANCH IF EBL IS RESET
12612 052116 005037 001140      CLR      $GDDAT
12613 052122 010037 001136      MOV      RO,$BDADR
12614 052126 062737 000024 001136      ADD      #RMMR1,$BDADR
12615 052134 104273                      ERROR    273 ;EBL DIDNT RESET ON TIME
12616 052136 000426                      BR       150$

```

```

12617 052140
12618
12619 052140 012702 000004
12620 052144
12621
12622 052144 012760 141501 000024
12623
12624 052152 012760 041501 000024
12625 052160 005302
12626 052162 001370
12627
12628 052164 016037 000000 001142
12629 052172 042737 177776 001142
12630 052200 001405
12631 052202 005037 001140
12632 052206 010037 001136
12633 052212 104274
12634
12635 052214 012737 052222 001124
12636
12637
12638
12639
12640 052222
12641
12642 052227 004737 060654
12643 052226 000402
12644 052230 104000
12645 052232 000512
12646 052234
12647
12648
12649 052234 012760 041401 000024
12650
12651 052242 012760 000000 000014
12652
12653 052250 012760 000000 000042
12654
12655 052256 012760 000000 000006
12656
12657 052264 012760 000000 000034
12658
12659 052272 012760 000071 000000
12660
12661 052300 012702 000005
12662 052304
12663
12664 052304 012760 141401 000024
12665
12666 052312 012760 041401 000024
12667 052320 005302
12668 052322 001370
12669
12670
12671 052324 016037 000024 001142
12672 052332 042737 137777 001142

```

```

130$:
;VERIFY GO RESETS WITHIN 4 CLOCK CYCLES
MOV #4,R2
140$:
MOV #DMD!MUR!MOC!DBEN!MDF!DBCK,RMMR1(RO) ;LOAD RMMR1
MOV #DMD!MUR!MOC!DBEN!MDF,RMMR1(RO) ;LOAD RMMR1
DEC R2
BNE 140$
MOV RMCS1(RO), $BDDAT ;STORE RMCS1 AT $BDDAT
BIC #1CRG,$BDDAT
BEQ 150$
CLR $GDDAT
MOV RO,$BDADR
ERROR 274 ;GO NOT RESET DURING DATA ABORT
150$: MOV #200$,$LPERR ;CHANGE LOOP ON ERROR ADDRESS
;*****
;VERIFY SEQUENCER BRANCHES TO SEEK WHEN RUN AND GO FLOP SETS
200$:
;SET VOLUME VALID USING SUBROUTINE
JSR PC,SETVV ;GO SET VOLUME VALID
BR 210$ ;BRANCH TO 210$ IF NO ERROR
ERROR ;RETURN HERE IF ERROR
BR 250$
210$:
;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
MOV #DMD!MUR!MOC!DBEN,RMMR1(RO) ;LOAD RMMR1
MOV #0,RMER1(RO) ;LOAD RMER1
MOV #0,RMER2(RO) ;LOAD RMER2
MOV #0,RMDA(RO) ;LOAD RMDA
MOV #0,RMDC(RO) ;LOAD RMDC
MOV #RD!GO,RMCS1(RO) ;LOAD RMCS1
;MOVE SEQUENCER TO TEST FOR RUN AND GO AT LOCATION 130 (5 CLOCKS)
MOV #5,R2
220$:
MOV #DMD!MUR!MOC!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1
MOV #DMD!MUR!MOC!DBEN,RMMR1(RO) ;LOAD RMMR1
DEC R2
BNE 220$
;VERIFY RUN AND GO IS SET
MOV RMMR1(RO), $BDDAT ;STORE RMMR1 AT $BDDAT
BIC #1CRG,$BDDAT

```

```

12673 052340 001012          BNE      230$
12674 052342 012737 04G000 001140    MOV     #RG,$GDDAT
12675 052350 010037 001136          MOV     RD,$BDADR
12676 052354 062737 000024 001136    ADD     #RMMR1,$BDADR
12677 052362 104275          ERROR   27$          ;RUN AND GO NOT SET
12678 052364 000435          BR      250$
12679 052366          230$:
12680          ;VERIFY THAT CYLINDER TAG COMES UP IN ONE CLOCK CYCLE
12681 052366 012702 000001          MOV     #1,R2
12682 052372          240$:
12683
12684 052372 012760 141401 000024    MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(RD) ;LOAD RMMR1
12685
12686 052400 012760 041401 000024    MOV     #DMD!MUR!MOC!DBEN,RMMR1(RD) ;LOAD RMMR1
12687 052406 005302          DEC     R2
12688 052410 001370          BNE     240$
12689
12690 052412 016037 000040 001142    MOV     RMMR2(RD),$BDDAT ;STORE RMMR2 AT $BDDAT
12691 052420 042737 160000 001142    BIC     #RQA!RQB!TAG,$BDDAT
12692 052426 012737 004000 001140    MOV     #CC,$GDDAT
12693 052434 023737 001140 001142    CMP     $GDDAT,$BDDAT
12694 052442 001406          BEQ     250$
12695 052444 010037 001136          MOV     RD,$BDADR
12696 052450 062737 000040 001136    ADD     #RMMR2,$BDADR
12697 052456 104276          ERROR   27$          ;CYLINDER TAG NOT SET DURING DATA
12698
12699 052460 012737 052472 001124 250$:  MOV     #260$,$LPERR ;CHANGE LOOP ON ERROR ADDRESS
12700
12701          ;*****
12702          ;VERIFY DATA COMMAND ABORTS AT COMMAND SEQUENCER LOCATIONS 144, 145
12703 052466 012702 000144          MOV     #144,R2 ;INITIALIZE TEST LOCATION
12704 052472
12705          260$:
12706          ;SET VOLUME VALID USING SUBROUTINE
12707 052472 004737 060654          JSR     PC,SETVV ;GO SET VOLUME VALID
12708 052476 000402          BR      270$ ;BRANCH TO 270$ IF NO ERROR
12709 052500 104000          ERROR   270$ ;RETURN HERE IF ERROR
12710 052502 000553          BR      320$
12711          270$:
12712          ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
12713 052504 012760 041401 000024    MOV     #DMD!MUR!MOC!DBEN,RMMR1(RD) ;LOAD RMMR1
12714
12715 052512 012760 000000 000014    MOV     #0,RMER1(RD) ;LOAD RMER1
12716
12717 052520 012760 000000 000042    MOV     #0,RMER2(RD) ;LOAD RMER2
12718
12719 052526 012760 000000 000006    MOV     #0,RMDA(RD) ;LOAD RMDA
12720
12721 052534 012760 000000 000034    MOV     #0,RMDC(RD) ;LOAD RMDC
12722
12723 052542 012760 000071 000000          MOV     #RD!GO,RMCS1(RD) ;LOAD RMCS1
12724          ;WAIT FOR RUN AND GO TO SET
12725 052550 012737 000310 001524    MOV     #200,WATCH ;SET WATCHDOG TIMER VALUE
12726 052556 004777 126744          JSR     PC,@CLOCK ;START THE CLOCK
12727
12728          280$:

```

```

12729 052562 016037 000024 001142    MOV    RMMR1(RO), $BDDAT          ;STORE RMMR1 AT $BDDAT
12730 052570 042737 137777 001142    BIC    #1CRG, $BDDAT
12731 052576 001017          290$
12732 052600 005737 001524    TST    WATCH
12733 052604 001366          BNE    280$
12734 052606 004777 126716    JSR    PC, $STOP                ;STOP THE CLOCK
12735 052612 012737 040000 001140    MOV    #RG, $GDDAT
12736 052620 010037 001136    MOV    RO, $BDAOR
12737 052624 062737 000024 001136    ADD    #RMMR1, $BDAOR
12738 052632 104275          ERROR 275                      ;RUN AND GO NOT SET
12739 052634 000476          BR    320$
12740 052636          290$:
12741 052636 004777 126666    JSR    PC, $STOP                ;STOP THE CLOCK
12742          ;MOVE COMMAND SEQUENCER TO ABORT TEST (LOCATION 144 OR 145)
12743 052642 012703 000006    MOV    #6, R3                    ;SETUP CLOCK COUNT
12744 052646 022702 000144    CMP    #144, R2
12745 052652 001402          BEQ    300$
12746 052654 012703 000007    MOV    #7, R3
12747 052660          300$:
12748
12749 052660 012760 141401 000024    MOV    #DMD!MUR!MOC!DBEN!DBCK, RMMR1(RO) ;LOAD RMMR1
12750
12751 052666 012760 041401 000024    MOV    #DMD!MUR!MOC!DBEN, RMMR1(RO) ;LOAD RMMR1
12752 052674 005303          DEC    R3
12753 052676 001370          BNE    300$
12754          ;SET DRIVE FAULT TO FORCE ABORT CONDITION
12755
12756 052700 012760 041501 000024    MOV    #DMD!MUR!MOC!DBEN!MDF, RMMR1(RO) ;LOAD RMMR1
12757          ;CLOCK SEQUENCER THROUGH ITS TEST FOR ABORT (1 CLOCK)
12758 052706 012703 000001    MOV    #1, R3
12759 052712          305$:
12760
12761 052712 012760 141501 000024    MOV    #DMD!MUR!MOC!DBEN!MDF!DBCK, RMMR1(RO) ;LOAD RMMR1
12762
12763 052720 012760 041501 000024    MOV    #DMD!MUR!MOC!DBEN!MDF, RMMR1(RO) ;LOAD RMMR1
12764 052726 005303          DFC    R3
12765 052730 001370          BNE    305$                      ;ISSUE 2 CLOCKS
12766
12767          ;ABORT EBL SHOULD NOW BE ACTIVE - USE THE MAINTENANCE REGISTER TO
12768          ;FORCE BIT CLOCKS AND VERIFY THAT EBL SETS WITHIN 16 BIT CLOCKS
12769 052732 012702 000020    MOV    #16, R2                    ;MAXIMUM NUMBER OF BIT CLOCKS
12770 052736          306$:
12771
12772 052736 012760 045501 000024    MOV    #DMD!MUR!MOC!DBEN!MDF!MCLK, RMMR1(RO) ;LOAD RMMR1
12773
12774 052744 012760 041501 000024    MOV    #DMD!MUR!MOC!DBEN!MDF, RMMR1(RO) ;LOAD RMMR1
12775
12776 052752 016037 000024 001142    MOV    RMMR1(RO), $BDDAT          ;STORE RMMR1 AT $BDDAT
12777 052760 042737 157777 001142    BIC    #1CEBL, $BDDAT
12778 052766 001013          BNE    310$                      ;BRANCH IF EBL IS SET
12779
12780 052770 005302          DEC    R2
12781 052772 001361          BNE    306$                      ;CONTINUE BIT CLOCKS IF COUNT NOT 0
12782 052774 012737 020000 001140    MOV    #EBL, $GDDAT
12783 053002 010037 001136    MOV    RO, $BDAOR
12784 053006 062737 000024 001136    ADD    #RMMR1, $BDAOR

```

N04

CZRMJ80 RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 259
T117 DATA COMMAND TESTS (1)

SEQ 0259

```

12785 053014 104271
12786 053016 022702 000144
12787 053022 001003
12788 053024 012702 000145
12789 053030 000620
12790
12791 053032 012737 053044 001124 320$: MOV #330$, $LPERR ;CHANGE LOOP ON ERROR ADDRESS
12792
12793 ;*****
12794 ;VERIFY HEAD TAG DURING DATA COMMAND
12795 053040 012702 000400 MOV #400,R2 ;INITIALIZE TRACK ADDRESS
12796 053044
12797 330$:
12798 053044 004737 060654 ;SET VOLUME VALID USING SUBROUTINE
12799 053050 000402 JSR PC,SETV ;GO SET VOLUME VALID
12800 053052 104000 BR 340$ ;BRANCH TO 340$ IF NO ERROR
12801 053054 000550 ERROR ;RETURN HERE IF ERROR
12802 053056 BR 400$
12803
12804 340$:
12805 053056 012760 041401 000024 ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
12806 MOV #DMD!MUR!MOC!DBEN,RMMR1(RO) ;LOAD RMMR1
12807 053064 012760 000000 000014 MOV #0,RMER1(RO) ;LOAD RMER1
12808
12809 053072 012760 000000 000042 MOV #0,RMER2(RO) ;LOAD RMER2
12810
12811 053100 010260 000006 MOV R2,RMDA(RO) ;LOAD RMDA
12812
12813 053104 012760 000000 000034 MOV #0,RMDC(RO) ;LOAD RMDC
12814
12815 053112 012760 000071 000000 MOV #RD!GO,RMCS1(RO) ;LOAD RMCS1
12816 ;WAIT FOR RUN AND GO TO SET
12817 053120 012737 000310 001524 MOV #200,WATCH ;SET WATCHDOG TIMER VALUE
12818 053126 004777 126374 JSR PC,@CLOCK ;START THE CLOCK
12819 053132
12820
12821 053132 016037 000024 001142 350$: MOV RMMR1(RO), $BDDAT ;STORE RMMR1 AT $BDDAT
12822 053140 042737 137777 001142 BIC #1CRG,$BDDAT
12823 053146 001017 BNE 360$
12824 053150 005737 001524 TST WATCH
12825 053154 001366 BNE 350$
12826 053156 004777 126346 JSR PC,@STOP ;STOP THE CLOCK
12827 053162 012737 040000 001140 MOV #RG,$GDDAT
12828 053170 010037 001136 MOV RO,$BDADR
12829 053174 062737 000024 001136 ADD #RMMR1,$BDADR
12830 053202 104275 ERROR 275 ;RUN AND GO NOT SET
12831 053204 000474 BR 400$
12832 053206
12833 053206 004777 126316 360$: JSR PC,@STOP ;STOP THE CLOCK
12834 ;STEP COMMAND SEQUENCER TO HEAD SEQUENCE, LOCATION 156 (17 CLOCKS)
12835 053212 012703 000021 MOV #17.,R3
12836 053216
12837
12838 053216 012760 141401 000024 MOV #DMD!MUR!MOC!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1
12839
12840 053224 012760 041401 000024 MOV #DMD!MUR!MOC!DBEN,RMMR1(RO) ;LOAD RMMR1

```

```

12841 053232 005303          DEC      R3
12842 053234 001370          BNE     37CS
12843          ;DROP AND RAISE ON CYLINDER TO RESET ON LATCH
12844
12845 053236 012760 041011 000024      MOV     #DMD!MUR!CSEN,RMMR1(RO) ;LOAD RMMR1
12846
12847 053244 012760 041401 000024      MOV     #DMD!MUR!MOC!DBEN,RMMR1(RO) ;LOAD RMMR1
12848 053252 012703 053400          MOV     #450$,R3 ;INITIALIZE TABLE POINTER
12849          ;VERIFY TAG BUS ACCORDING TO TABLE AND TRACK ADDRESS IN R2
12850 053256          375$:
12851
12852 053256 016037 000040 001142      MOV     RMMR2(RO), $BDDAT ;STORE RMMR2 AT $BDDAT
12853 053264 042737 150000 001142      BIC     #RQA!RQB!TST,$BDDAT
12854 053272 011337 001140          MOV     (R3), $GDDAT
12855 053276 010204          MOV     R2,R4 ;GENERATE EXPECTED TAG BUS
12856 053300 000304          SWAB   R4
12857 053302 042704 177770          BIC     #T7,R4
12858 053306 050437 001140          BIS     R4,$GDDAT
12859 053312 023737 001140 001142      CMP     $GDDAT,$BDDAT
12860 053320 001020          BNE     390$
12861
12862 053322 012760 141401 000024      MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1
12863
12864 053330 012760 041401 000024      MOV     #DMD!MUR!MOC!DBEN,RMMR1(RO) ;LOAD RMMR1
12865          ;ADVANCE TO NEXT TABLE ENTRY
12866 053336 062703 000002          ADD     #2,R3
12867 053342 005713          TST    (R3)
12868 053344 100401          BMI    380$
12869 053346 000743          BR     375$
12870 053350          380$:
12871          ;SHIFT TO NEXT TRACK ADDRESS-EXIT LOOP IF DONE
12872 053350 006302          ASL    R2
12873 053352 020227 002000          CMP    R2,#2000
12874 053356 101007          BHI    400$
12875 053360 000631          BR     330$
12876 053362          390$:
12877          ;ERROR ON TAG BUS DURING HEAD SEQUENCE
12878 053362 010037 001136          MOV     R0,$BDADR
12879 053366 062737 000040 001136      ADD     #RMMR2,$BDADR
12880 053374 104276          ERROR  276 ;INCORRECT TAG BUS
12881
12882 053376 000440          400$: BR     500$ ;JUMP OVER TABLE
12883
12884 053400          450$:
12885          ;TABLE OF TAG BUS DURING HEAD SEQUENCE
12886 053400 002000          .WORD  CH
12887 053402 002000          .WORD  CH
12888 053404 022000          .WORD  CH!TAG
12889 053406 022000          .WORD  CH!TAG
12890 053410 022000          .WORD  CH!TAG
12891 053412 022000          .WORD  CH!TAG
12892 053414 022000          .WORD  CH!TAG
12893 053416 022000          .WORD  CH!TAG
12894 053420 002000          .WORD  CH
12895 053422 002000          .WORD  CH
12896 053424 001777          .WORD  1777

```

```

12897 053426 001777 .WORD 1777
12898 053430 001777 .WORD 1777
12899 053434 001777 .WORD 1777
12900 053438 001777 .WORD 1777
12901 053442 001777 .WORD 1777
12902 053446 001777 .WORD 1777
12903 053450 001777 .WORD 1777
12904 053454 001777 .WORD 1777
12905 053458 001777 .WORD 1777
12906 053462 001777 .WORD 1777
12907 053466 001777 .WORD 1777
12908 053470 001777 .WORD 1777
12909 053474 001777 .WORD 1777
12910 053478 001777 .WORD 1777
12911 053482 001777 .WORD 1777
12912 053486 001777 .WORD 1777
12913 053490 001777 .WORD 1777
12914 053494 001777 .WORD 1777
12915 053498 001777 .WORD 1777
12916 053502 001777 .WORD 1777
12917 053506 001777 .WORD 1777
12918 053476 177777 .WORD -1 ;END OF TABLE
12919
12920 053500 500$: ;END OF TEST
12921
12922 ;*****
12923 ;*TEST 120 DATA COMMAND TESTS (2)
12924
12925 ;*****
12926 053500 000004 T120: SCOPE
12927 053502 012737 000120 001226 MOV #120,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
12928
12929 053510 000240 NOP
12930 053512 012737 000024 001120 MOV #20,$ICNT ;20 ITERATIONS
12931 053520 112737 000001 001131 MOVB #1,$EMAX ;ONE ERROR ALLOWED
12932 053526 012737 053542 001122 MOV #T120,$LPADR ;LOAD LOOP ON TEST ADDRESS
12933 053534 012737 053542 001124 MOV #T120,$LPERR ;LOAD LOOP ON ERROR ADDRESS
12934 053542
12935 053542 012706 001100 T120: MOV #STACK,SP ;LOAD THE STACK POINTER
12936 053546 013700 001276 MOV $BASE,R0 ;R0 = UNIBUS ADDRESS OF UUT
12937 053552 013701 001456 MOV TSTQUE,R1 ;R1 = POINTER TO DEVICE
12938
12939 ;*****
12940 ;VERIFY OPI SETS IF ON CYLINDER LATCH DOESNT RESET
12941 ;SET VOLUME VALID USING SUBROUTINE
12942 053556 004737 060654 JSR PC,SETVV ;GO SET VOLUME VALID
12943 053562 000402 BR 10$ ;BRANCH TO 10$ IF NO ERROR
12944 053564 104000 ERROR ;RETURN HERE IF ERROR
12945 053566 000514 BR 60$
12946 053570
12947 10$: ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND DURING CYLINDER SEQUENCE
12948 053570 012760 041401 000024 MOV #DMD!MUR!MOC!DBEN,RMMR1(R0) ;LOAD RMMR1
12949
12950 053576 012760 000000 000014 MOV #0,RMER1(R0) ;LOAD RMER1
12951
12952 053604 012760 000000 000042 MOV #0,RMER2(R0) ;LOAD RMER2

```

```

12953
12954 053612 012760 000000 000006      MOV      #0,RMDA(RO)      ;LOAD RMDA
12955
12956 053620 012760 000000 000034      MOV      #0,RMDC(RO)      ;LOAD RMDC
12957
12958 053626 012760 000071 000000      MOV      #RD!GO,RMCSI(RO) ;LOAD RMCSI
12959 ;WAIT FOR RUN AND GO TO SET
12960 053634 012737 000310 001524      MOV      #200.,WATCH     ;SET WATCHDOG TIMER VALUE
12961 053642 004777 125660                JSR      PC,@CLOCK        ;START THE CLOCK
12962 053646
20$:
12964 053646 016037 000024 001142      MOV      RMMR1(RO), $BDDAT ;STORE RMMR1 AT $BDDAT
12965 053654 042737 137777 001142      BIC      #1CRG,$BDDAT
12966 053662 001017                BNE      30$
12967 053664 005737 001524                TST      WATCH
12968 053670 001366                BNE      20$
12969 053672 004777 125632                JSR      PC,@STOP        ;STOP THE CLOCK
12970 053676 012737 040000 001140      MOV      #RG,$GDDAT
12971 053704 010037 001136                MOV      RO,$BADDR
12972 053710 062737 000024 001136      ADD      #RMMR1,$BADDR
12973 053716 104275                ERROR    275              ;RUN AND GO DIDNT SET
12974 053720 000437                BR       60$
12975 053722
30$:
12976 053722 004777 125602                JSR      PC,@STOP        ;STOP THE CLOCK
12977 ;STEP COMMAND SEQUENCER AND VERIFY OPI SETS (19 CLOCKS)
12978 053726 012702 000023                MOV      #19.,R2
12979 053732
40$:
12981 053732 012760 141401 000024      MOV      #DMC!MUR!MOC!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1
12982
12983 053740 012760 041401 000024      MOV      #DMC!MUR!MOC!DBEN,RMMR1(RO) ;LOAD RMMR1
12984 053746 005302                DEC      R2
12985 053750 001370                BNE      40$
12986
12987 053752 016037 000014 001142      MOV      RMER1(RO), $BDDAT ;STORE RMER1 AT $BDDAT
12988 053760 042737 157777 001142      BIC      #1COPI,$BDDAT
12989 053766 001011                BNE      50$              ;BRANCH IF OPI SET
12990 053770 012737 020000 001140      MOV      #OPE,$GDDAT
12991 053776 010037 001136                MOV      RO,$BADDR
12992 054002 062737 000014 001136      ADD      #RMER1,$BADDR
12993 054010 104277                ERROR    277              ;OPI NOT SET BY ON LATCH SET
12994 054012 012737 054020 001124      MOV      #60$, $LPERR    ;CHANGE LOOP ON ERROR ADDRESS
12995 ;*****
12996 ;VERIFY DATA COMMAND ABORTS DURING SEEK WAIT LOOP
12997 054020
60$:
12998 ;SET VOLUME VALID USING SUBROUTINE
12999 054020 004737 0E0654                JSR      PC,SETVV        ;GO SET VOLUME VALID
13000 054024 000402                BR       70$              ;BRANCH TO 70$ IF NO ERROR
13001 054026 104000                ERROR    ;RETURN HERE IF ERROR
13002 054030 000567                BR       140$
13003 054032
70$:
13004 ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
13005
13006 054032 012760 041401 000024      MOV      #DMC!MUR!MOC!DBEN,RMMR1(RO) ;LOAD RMMR1
13007
13008 054040 012760 000000 000014      MOV      #0,RMER1(RO) ;LOAD RMER1

```


E05

CZRMJBO RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 263
T120 DATA COMMAND TESTS (2)

SEQ 0263

```

13009
13010 054046 012760 000000 000042      MOV      #0,RMER2(RO)      ;LOAD RMER2
13011
13012 054054 012760 000000 000034      MOV      #0,RMDC(RO)      ;LOAD RMDC
13013
13014 054062 012760 000000 000006      MOV      #0,RMDA(RO)      ;LOAD RMDA
13015
13016 054070 012760 000071 000000      MOV      #RD!GO,RMCS1(RO) ;LOAD RMCS1
13017      ;WAIT FOR RUN & GO TO SET
13018 054076 012737 000310 001524      MOV      #200,WATCH      ;SET WATCHDOG TIMER VALUE
13019 054104 004777 125416      JSR      PC,@CLOCK      ;START THE CLOCK
13020 054110
13021
13022 054110 016037 000024 001142      MOV      RMMR1(RO), $BDDAT ;STORE RMMR1 AT $BDDAT
13023 054116 042737 137777 001142      BIC      #1CRG,$BDDAT
13024 054124 001017      BNE      90$
13025 054126 005737 001524      TST      WATCH
13026 054132 001366      BNE      80$
13027 054134 004777 125370      JSR      PC,@STOP      ;STOP THE CLOCK
13028 054140 012737 040000 001140      MOV      #RG,$GDDAT
13029 054146 010037 001136      MOV      RO,$BADDR
13030 054152 062737 000024 001136      ADD      #RMMR1,$BADDR
13031 054160 104275      ERROR    275      ;RUN AND GO DIDNT SET
13032 054162 000512      BR      140$
13033 054164
13034 054164 004777 125340      JSR      PC,@STOP      ;STOP THE CLOCK
13035      ;STEP COMMAND SEQUENCER TO ON LATCH TEST AT LOCATION 156 (17 CLOCKS)
13036 054170 012702 000021      MOV      #17.,R2
13037 054174
13038
13039 054174 012760 141401 000024      MOV      #DMD!MUR!MOC!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1
13040
13041 054202 012760 041401 000024      MOV      #DMD!MUR!MOC!DBEN,RMMR1(RO) ;LOAD RMMR1
13042 054210 005302      DEC      R2
13043 054212 001370      BNE      100$
13044      ;DROP ON CYLINDER TO RESET LATCH
13045
13046 054214 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
13047      ;MOVE COMMAND SEQUENCER TO SEEK WAIT LOOP (31 CLOCKS)
13048 054222 012702 000037      MOV      #31.,R2
13049 054226
13050
13051 054226 012760 141001 000024      MOV      #DMD!MUR!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1
13052
13053 054234 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
13054 054242 005302      DEC      R2
13055 054244 001370      BNE      110$
13056      ;STEP THROUGH SEEK WAIT LOOP (6 CLOCKS) 2 TIMES
13057 054246 012702 000006      MOV      #6.,R2
13058 054252
13059
13060 054252 012760 141001 000024      MOV      #DMD!MUR!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1
13061
13062 054260 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
13063 054266 005302      DEC      R2
13064 054270 001370      BNE      120$

```

F05

CZRMJ80 RM03/2 DSKLS DIAG
CZRMJ8.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 264
T120 DATA COMMAND TESTS (2)

SEQ 0264

```

13065 ;SET SEEK INCOMPLETE ERROR TO CAUSE ABORT
13066
13067 054272 012760 041201 000024      MOV      #DMD!MUR!DBEN!MSER,RMMR1(RO) ;LOAD RMMR1
13068 ;CLOCK THE SEQUENCER THROUGH ITS TEST FOR ABORT (2 CLOCKS)
13069 054300 012702 000002      MOV      #2,R2
13070 054304
13071
13072 054304 012760 141201 000024      MOV      #DMD!MUR!DBEN!MSER!DBCK,RMMR1(RO) ;LOAD RMMR1
13073
13074 054312 012760 041201 000024      MOV      #DMD!MUR!DBEN!MSER,RMMR1(RO) ;LOAD RMMR1
13075 054320 005302
13076 054322 001370      DEC      R2
13077      BNE     130$
13078 ;ABORT EBL SHOULD NOW BE ACTIVE - USE THE MAINTENANCE REGISTER TO
13079 ;FORCE BIT CLOCKS AND VERIFY THAT EBL SETS WITHIN 16 BIT CLOCKS
13080 054324 012702 000020      MOV      #16.,R2 ;MAXIMUM NUMBER OF BIT CLOCKS
13081 054330
13082
13083 054330 012760 045501 000024      MOV      #DMD!MUR!MOC!DBEN!MDF!MCLK,RMMR1(RO) ;LOAD RMMR1
13084
13085 054336 012760 041501 000024      MOV      #DMD!MUR!MOC!DBEN!MDF,RMMR1(RO) ;LOAD RMMR1
13086
13087 054344 016037 000024 001142      MOV      RMMR1(RO), $BDDAT ;STORE RMMR1 AT $BDDAT
13088 054352 042737 157777 001142      BIC     #1CEBL,$BDDAT
13089 054360 001013      BNE     140$
13090
13091 054362 005302      DEC      R2
13092 054364 001361      BNE     135$ ;CONTINUE BIT CLOCKS IF COUNT NOT 0
13093 054366 012737 020000 001140      MOV      #EBL,$GDDAT
13094 054374 010037 001136      MOV      RO,$DADR
13095 054400 062737 000024 001136      ADD     #RMMR1,$DADR
13096 054406 104300      ERROR   300 ;EBL NOT SET DUE TO ABORT
13097 054410 012737 054416 001124 140$: MOV      #150$, $LPERR ;CHANGE LOOP ON ERROR ADDRESS
13098
13099 ;*****
13100 ;VERIFY DATA COMMAND ABORTS DURING OFFSET IF ON CYLINDER LATCH
13101 ;DOESNT RESET
13102 054416 150$:
13103 ;SET VOLUME VALID USING SUBROUTINE
13104 054416 004737 060654      JSR     PC,SETVV ;GO SET VOLUME VALID
13105 054422 000402      BR     160$ ;BRANCH TO 160$ IF NO ERROR
13106 054424 104000      ERROR   ;RETURN HERE IF ERROR
13107 054426 000536      BR     220$
13108 054430
13109 160$:
13110 ;LOAD TRACK,SETCOR,CYLINDER ADDRESS
13111 054430 012760 000000 000006      MOV      #0,RMDA(RO) ;LOAD RMDA
13112
13113 054436 012760 000000 000034      MOV      #0,RMDC(RO) ;LOAD RMDC
13114 ;SET OFFSET MODE USING SUBROUTINE
13115 054444 004737 061004      JSR     PC,SETOM ;GO SET OFFSET MODE
13116 054450 000402      BR     170$ ;BRANCH TO 170$ IF NO ERROR
13117 054452 104000      ERROR   ;RETURN HERE IF ERROR
13118 054454 000523      BR     220$
13119 054456
13120 170$:
;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND

```

G05

CZRMJBO RM03/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 265
T120 DATA COMMAND TESTS (2)

SEQ 0265

13121									
13122	054456	012760	041401	000024		MOV	#DMD!MUR!MOC!DBEN,RMMR1(RO)		;LOAD RMMR1
13123									
13124	054464	012760	000000	000014		MOV	#0,RMER1(RO)		;LOAD RMER1
13125									
13126	054472	012760	000000	000042		MOV	#0,RMER2(RU)		;LOAD RMER2
13127									
13128	054500	012760	000071	000000		MOV	#RD!GO,RMCSI(RO)		;LOAD RMCSI
13129									
13130	054506	012737	000310	001524					;WAIT FOR RUN AND GO TO SET
13131	054514	004777	125006			MOV	#200,WATCH		;SET WATCHDOG TIMER VALUE
13132	054520					JSR	PC,2CLOCK		;START THE CLOCK
13133									180\$:
13134	054520	016037	000024	001142		MOV	RMMR1(RO),SBDDAT		;STORE RMMR1 AT SBDDAT
13135	054526	042737	137777	001142		BIC	#1CRG,SBDDAT		
13136	054534	001017				BNE	190\$		
13137	054536	005737	001524			TST	WATCH		
13138	054542	001366				BNE	180\$		
13139	054544	004777	124760			JSR	PC,2STOP		;STOP THE CLOCK
13140	054550	012737	040000	001140		MOV	#RG,\$GDDAT		
13141	054556	010037	001140			MOV	RD,\$GDDAT		
13142	054562	062737	000024	001136		ADD	#RMMR1,\$BDADR		
13143	054570	104275				ERROR	275		;RUN AND GO NOT SET
13144	054572	000454				BR	220\$		
13145	054574								190\$:
13146	054574	004777	124730			JSR	PC,2STOP		;STOP THE CLOCK
13147									;STEP COMMAND SEQUENCER TO ON LATCH TEST AT LOCATION 156 (17 CLOCKS)
13148	054600	012702	000021			MOV	#17.,R2		
13149	054604								200\$:
13150									
13151	054604	012760	141401	000024		MOV	#DMD!MUR!MOC!DBEN!DBCK,RMMR1(RO)		;LOAD RMMR1
13152									
13153	054612	012760	041401	000024		MOV	#DMD!MUR!MOC!DBEN,RMMR1(RO)		;LOAD RMMR1
13154	054620	005302				DEC	R2		
13155	054622	001370				BNE	200\$		
13156									;DROP ON CYLINDER TO RESET LATCH, SET ON CYLINDER TO PASS TEST
13157									;AT LOCATION 166
13158									
13159	054624	012760	041001	000024		MOV	#DMD!MUR!DBEN,RMMR1(RO)		;LOAD RMMR1
13160									
13161	054632	012760	041401	000024		MOV	#DMD!MUR!DBEN!MOC,RMMR1(RO)		;LOAD RMMR1
13162									;MOVE SEQUENCER TO SET OPI AND EBL (39 CLOCKS)
13163	054640	012702	000047			MOV	#39.,R2		
13164	054644								210\$:
13165									
13166	054644	012760	141401	000024		MOV	#DMD!MUR!DBEN!MOC!DBCK,RMMR1(RO)		;LOAD RMMR1
13167									
13168	054652	012760	041401	000024		MOV	#DMD!MUR!DBEN!MOC,RMMR1(RO)		;LOAD RMMR1
13169	054660	005302				DEC	R2		
13170	054662	001370				BNE	210\$		
13171									;VERIFY OPI IS SET
13172									
13173	054664	016037	000014	001142		MOV	RMER1(RO),SBDDAT		;STORE RMER1 AT SBDDAT
13174	054672	042737	157777	001142		BIC	#1COPI,SBDDAT		
13175	054700	001011				BNE	220\$		
13176	054702	012737	020000	001140		MOV	#OPI,\$GDDAT		

```

13177 054710 010037 001136          MOV      R0,$BDADR
13178 054714 062737 000014 001136    ADD      #RMR1,$BDADR
13179 054722 104277          ERROR    277          ;OPI NOT SET DURING OFFSET
13180 054724 012737 054732 001124 220$:   MOV      #230$,$LPERR ;CHANGE LOOP ON ERROR ADDRESS
13181
13182          ;*****
13183          ;VERIFY DATA COMMAND ABORTS DURING OFFSET WAIT LOOP
13184
13185 054732          230$:
13186          ;SET VOLUME VALID USING SUBROUTINE
13187 054732 004737 060654          JSR      PC,SETVV    ;GO SET VOLUME VALID
13188 054736 000403          BR      240$        ;BRANCH TO 240$ IF NO ERROR
13189 054740 104000          ERROR    277          ;RETURN HERE IF ERROR
13190 054742 000137 055352          JMP      310$
13191 054746
13192          240$:
13193          ;LOAD SECTOR, TRACK AND CYLINDER ADDRESS
13194 054746 012760 000000 000006          MOV      #0,RMDA(R0) ;LOAD RMDA
13195
13196 054754 012760 000000 000034          MOV      #0,RMDC(R0) ;LOAD RMDC
13197          ;SET OFFSET MODE USING SUBROUTINE
13198 054762 004737 061004          JSR      PC,SETOM    ;GO SET OFFSET MODE
13199 054766 000402          BR      245$        ;BRANCH TO 245$ IF NO ERROR
13200 054770 104000          ERROR    277          ;RETURN HERE IF ERROR
13201 054772 000567          BR      310$
13202          ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
13203 054774          245$:
13204
13205 054774 012760 041401 000024          MOV      #DMD!MUR!MOC!DBEN,RMMR1(R0) ;LOAD RMMR1
13206
13207 055002 012760 000000 000014          MOV      #0,RMER1(R0) ;LOAD RMER1
13208
13209 055010 012760 000000 000042          MOV      #0,RMER2(R0) ;LOAD RMER2
13210
13211 055016 012760 000071 000000          MOV      #RD!GO,RMCS1(R0) ;LOAD RMCS1
13212          ;WAIT FOR RUN AND GO TO SET
13213 055024 012737 000310 001524          MOV      #200,WATCH   ;SET WATCHDOG TIMER VALUE
13214 055032 004777 124470          JSR      PC,@CLOCK    ;START THE CLOCK
13215 055036          250$:
13216
13217 055036 016037 000024 001142          MOV      RMMR1(R0),$BDDAT ;STORE RMMR1 AT $BDDAT
13218 055044 042737 137777 001142          BIC      #+CRG,$BDDAT
13219 055052 001017          BNE      260$
13220 055054 005737 001524          TST     WATCH
13221 055060 001366          BNE      250$
13222 055062 004777 124442          JSR      PC,@STOP    ;STOP THE CLOCK
13223 055066 012737 040000 001140          MOV      #RG,$GODAT
13224 055074 010037 001136          MOV      R0,$BDADR
13225 055100 062737 000024 001136          ADD      #RMMR1,$BDADR
13226 055106 104275          ERROR    275          ;RUN AND GO NOT SET
13227 055110 000520          BR      310$
13228
13229 055112 004777 124412          260$:   JSR      PC,@STOP    ;STOP THE CLOCK
13230          ;STEP SEQUENCER TO LOCATION 156 (17 CLOCKS)
13231 055114 012702 000021          MOV      #17.,R2
13232 055122          270$:

```

```

13233
13234 055122 012760 141401 000024      MOV      #DMD!MUR!MOC!DBEN!DBCK,RMMR1(RO)      ;LOAD RMMR1
13235
13236 055130 012760 041401 000024      MOV      #DMD!MUR!MOC!DBEN,RMMR1(RO)      ;LOAD RMMR1
13237 055136 005302
13238 055140 001370      DEC      R2
13239      BNE      270$
13240      ;DROP ON CYLINDER TO RESET LATCH, SET ON CYLINDER TO PASS TEST
13241      ;AT LOCATION 166
13242 055142 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
13243
13244 055150 012760 041401 000024      MOV      #DMD!MUR!MOC!DBEN,RMMR1(RO) ;LOAD RMMR1
13245      ;MOVE SEQUENCER TO LOCATION 174 (37 CLOCKS)
13246 055156 012702 000045      MOV      #37.,R2
13247 055162
13248      280$:
13249 055162 012760 141401 000024      MOV      #DMD!MUR!MOC!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1
13250
13251 055170 012760 041401 000024      MOV      #DMD!MUR!MOC!DBEN,RMMR1(RO) ;LOAD RMMR1
13252 055176 005302
13253 055200 001370      DEC      R2
13254      BNE      280$
13255      ;DROP ON CYLINDER TO RESET LATCH, LEAVE ON CYLINDER RESET
13256 055202 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
13257      ;STEP SEQUENCER THROUGH OFFSET WAIT LOOP TWICE (7 CLOCKS)
13258 055210 012702 000007      MOV      #7.,R2
13259 055214
13260      290$:
13261 055214 012760 141001 000024      MOV      #DMD!MUR!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1
13262
13263 055222 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
13264 055230 005302
13265 055232 001370      DEC      R2
13266      BNE      290$
13267      ;SET DRIVE FAULT TO CAUSE ABORT CONDITION
13268 055234 012760 041101 000024      MOV      #DMD!MUR!DBEN!MDF,RMMR1(RO) ;LOAD RMMR1
13269      ;STEP SEQUENCER THROUGH ITS TEST FOR ABORT (2 CLOCKS)
13270 055242 012702 000002      MOV      #2.,R2
13271 055246
13272      300$:
13273 055246 012760 141101 000024      MOV      #DMD!MUR!DBEN!MDF!DBCK,RMMR1(RO) ;LOAD RMMR1
13274
13275 055254 012760 041101 000024      MOV      #DMD!MUR!DBEN!MDF,RMMR1(RO) ;LOAD RMMR1
13276 055262 005302
13277 055264 001370      DEC      R2
13278      BNE      300$
13279      ;ABORT EBL SHOULD NOW BE ACTIVE - USE THE MAINTENANCE REGISTER TO
13280      ;FORCE BIT CLOCKS AND VERIFY THAT EBL SETS WITHIN 16 BIT CLOCKS
13281 055266 012702 000020      MOV      #16.,R2 ;MAXIMUM NUMBER OF BIT CLOCKS
13282 055272
13283      305$:
13284 055272 012760 045501 000024      MOV      #DMD!MUR!MOC!DBEN!MDF!MCLK,RMMR1(RO) ;LOAD RMMR1
13285
13286 055300 012760 041501 000024      MOV      #DMD!MUR!MOC!DBEN!MDF,RMMR1(RO) ;LOAD RMMR1
13287
13288 055306 016037 000024 001142      MOV      RMMR1(RO), $BDDAT ;STORE RMMR1 AT $BDDAT

```

J05

CZRMJ80 RM03/2 DSKLS CIAG MACY11 30(1046) 23-NOV-77 12:14 PAGE 268
 CZRMJB.P11 23-NOV-77 12:07 T120 DATA COMMAND TESTS (2)

SEQ 0268

```

13289 055314 042737 157777 001142      BIC      #1CEBL,$BDDAT
13290 055322 001013                      BNE      310$
13291
13292 055324 005302                      DEC      R2
13293 055326 001361                      BNE      305$ ;CONTINUE BIT CLOCKS IF COUNT NOT 0
13294 055330 012737 020000 001140      MOV      #EBL,$GDDAT
13295 055336 010037 001136      MOV      R0,$BDADR
13296 055342 062737 000024 001136      ADD      #RMMR1,$BDADR
13297 055350 104271                      ERROR    271 ;EBL NOT SET DUE TO OFFSET WAIT ABORT
13298
13299 055352 012737 055360 001124 310$:  MOV      #320$,$LPERR ;CHANGE LOOP ON ERROR ADDRESS
13300
13301 ;*****
13302 ;VERIFY THAT DATA COMMAND ABORTS DURING SECTOR WAIT LOOP AT LOCATION 179
13303
13304 055360      320$:
13305 ;SET VOLUME VALID USING SUBROUTINE
13306 055360 004737 060654      JSR      PC,SETVV ;GO SET VOLUME VALID
13307 055364 000403                      BR       330$ ;BRANCH TO 330$ IF NO ERROR
13308 055366 104000                      ERROR    ;RETURN HERE IF ERROR
13309 055370 000137 056016      JMP      420$
13310 055374      330$:
13311 ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
13312
13313 055374 012760 041401 000024      MOV      #DMD!MUR!MOC!DBEN,RMMR1(R0) ;LOAD RMMR1
13314
13315 055402 012760 000000 000014      MOV      #0,RMER1(R0) ;LOAD RMER1
13316
13317 055410 012760 000000 000042      MOV      #0,RMER2(R0) ;LOAD RMER2
13318
13319 055416 012760 000000 000006      MOV      #0,RMDA(R0) ;LOAD RMDA
13320
13321 055424 012760 000000 000034      MOV      #0,RMDC(R0) ;LOAD RMDC
13322
13323 055432 012760 000071 000000      MOV      #RD!GO,RMCS1(R0) ;LOAD RMCS1
13324 ;WAIT FOR RUN AND GO TO SET
13325 055440 012737 000310 001524      MOV      #200,WATCH ;SET WATCHDOG TIMER VALUE
13326 055446 004777 124054      JSR      PC,$CLOCK ;START THE CLOCK
13327 055452
13328
13329 055452 016037 000024 001142      MOV      RMMR1(R0),$BDDAT ;STORE RMMR1 AT $BDDAT
13330 055460 042737 137777 001142      BIC      #1CRG,$BDDAT
13331 055466 001017                      BNE      350$
13332 055470 005737 001524      TST      WATCH
13333 055474 001366                      BNE      340$
13334 055476 004777 124026      JSR      PC,$STOP ;STOP THE CLOCK
13335 055502 012737 040000 001140      MOV      #RG,$GDDAT
13336 055510 010037 001136      MOV      R0,$BDADR
13337 055514 062737 000024 001136      ADD      #RMMR1,$BDADR
13338 055522 104275                      ERROR    275 ;RUN AND GO NOT SET
13339 055524 000534                      BR       420$
13340
13341 055526 004777 123776      350$:  JSR      PC,$STOP ;STOP THE CLOCK
13342 ;STEP SEQUENCER TO LOCATION 156 (17 CLOCKS)
13343 055532 012702 000021      MOV      #17.,R2
13344 055536      360$:

```

K05

CZRMJ80 RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 269
T120 DATA COMMAND TESTS (2)

SEQ 0269

```

13345
13346 055536 012760 141401 000024      MOV      #DMD!MUR!MOC!DBEN!DBCK,RMMR1(RO)      ;LOAD RMMR1
13347
13348 055544 012760 041401 000024      MOV      #DMD!MUR!MOC!DBEN,RMMR1(RO)      ;LOAD RMMR1
13349 055552 005302
13350 055554 001370      DEC      R2
13351      BNE     360$
13352      ;DROP ON CYLINDER TO RESET LATCH, SET ON CYLINDER TO PASS TEST
13353      ;AT LOCATION 166
13354 055556 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1
13355      ;MOVE SEQUENCER TO SECTOR WAIT (34 CLOCKS)
13356 055564 012702 000042      MOV      #34.,R2
13357 055570
13358      370$:
13359 055570 012760 141401 000024      MOV      #DMD!MUR!MOC!DBEN!DBCK,RMMR1(RO)      ;LOAD RMMR1
13360
13361 055576 012760 041401 000024      MOV      #DMD!MUR!MOC!DBEN,RMMR1(RO)      ;LOAD RMMR1
13362 055604 005302
13363 055606 001370      DEC      R2
13364      BNE     370$
13365      ;STEP THROUGH SECTOR WAIT LOOP TWICE AND VERIFY SEARCH IS ENABLED
13366 055610 012702 000006      ;DURING THE LOOP (6 CLOCKS)
13367 055614      MOV      #6.,R2
13368      380$:
13369 055614 012760 141401 000024      MOV      #DMD!MUR!MOC!DBEN!DBCK,RMMR1(RO)      ;LOAD RMMR1
13370
13371 055622 012760 041401 000024      MOV      #DMD!MUR!MOC!DBEN,RMMR1(RO)      ;LOAD RMMR1
13372
13373 055630 016037 000024 001142      MOV      RMMR1(RO), $BDDAT      ;STORE RMMR1 AT $BDDAT
13374 055636 042737 173777 001142      BIC     #↑CESRC,$BDDAT
13375 055644 001403      BEQ     390$
13376 055646 005302      DEC     R2
13377 055650 001361      BNE     380$
13378 055652 000412      BR      400$
13379 055654 012737 004000 001140 390$: MOV     #ESRC,$GDDAT
13380 055662 010037 001136      MOV     RO,$B0ADR
13381 055666 062737 000024 001136      ADD     #RMMR1,$B0ADR
13382 055674 104301      ERROR  301      ;SEARCH NOT ENABLED DURING DATA
13383 055676 000447      BR      420$
13384 055700      400$:
13385      ;SET DRIVE FAULT TO CAUSE ABORT CONDITION
13386
13387 055700 012760 041501 000024      MOV      #DMD!MUR!MOC!DBEN!MDF,RMMR1(RO) ;LOAD RMMR1
13388      ;STEP SEQUENCER THROUGH ITS TEST FOR ABORT (2 CLOCKS)
13389 055706 012702 000002      MOV      #2,R2
13390 055712
13391      410$:
13392 055712 012760 141501 000024      MOV      #DMD!MUR!MOC!DBEN!MDF!DBCK,RMMR1(RO) ;LOAD RMMR1
13393
13394 055720 012760 041501 000024      MOV      #DMD!MUR!MOC!DBEN!MDF,RMMR1(RO) ;LOAD RMMR1
13395 055726 005302      DEC     R2
13396 055730 001370      BNE     410$
13397
13398      ;ABORT EBL SHOULD NOW BE ACTIVE - USE THE MAINTENANCE REGISTER TO
13399      ;FORCE BIT CLOCKS AND VERIFY THAT EBL SETS WITHIN 16 BIT CLOCKS
1400 055732 012702 000020      MOV      #16.,R2      ;MAXIMUM NUMBER OF BIT CLOCKS

```

L05

CZRMJ80 RM03/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 270
T120 DATA COMMAND TESTS (2)

SEQ 0270

```

13401 055736          415$:
13402
13403 055736 012760 045501 000024      MOV      #DMD!MUR!MOC!DBEN!MDF!MCLK,RMMR1(RO) ;LOAD RMMR1
13404
13405 055744 012760 041501 000024      MOV      #DMD!MUR!MOC!DBEN!MDF,RMMR1(RO) ;LOAD RMMR1
13406          ;VERIFY EBL IS SET
13407
13408 055752 016037 000024 001142      MOV      RMMR1(RO), $BDDAT ;STORE RMMR1 AT $BDDAT
13409 055760 042737 157777 001142      BIC      #↑CEBL,$BDDAT
13410 055766 001013          BNE      420$
13411
13412 055770 005302          DEC      R2
13413 055772 001361          BNE      415$ ;CONTINUE BIT CLOCKS IF COUNT NOT 0
13414 055774 012737 020000 001140      MOV      #EBL,$GDDAT
13415 056002 010037 001136      MOV      RO,$BDAOR
13416 056006 062737 000024 001136      ADD      #RMMR1,$BDAOR
13417 056014 104271          ERROR    271 ;NO ABORT AT SECTOR WAIT
13418
13419 056016          420$: ;END OF TEST
13420
13421          ;*****
13422          ;#TEST 121 DATA COMMAND TESTS(3)
13423
13424          ;*****
13425 056016 000004      †ST121: SCOPE
13426 056020 012737 000121 001226      MOV      #121,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
13427
13428 056026 000240      NOP
13429 056030 012737 000024 001120      MOV      #20,$ICNT ;20 ITERATIONS
13430 056036 112737 000001 001131      MOV      #1,$ERMAX ;ONE ERROR ALLOWED
13431 056044 012737 056060 001122      MOV      #T121,$LPADR ;LOAD LOOP ON TEST ADDRESS
13432 056052 012737 056060 001124      MOV      #T121,$LPERR ;LOAD LOOP ON ERROR ADDRESS
13433 056060
13434 056060 012706 001100      T121:   MOV      #STACK,$P ;LOAD THE STACK POINTER
13435 056064 013700 001276      MOV      $BASE,$RO ;RO = UNIBUS ADDRESS OF UUT
13436 056070 013701 001456      MOV      TSTQUE,$R1 ;R1 = POINTER TO DEVICE
13437
13438          ;*****
13439          ;VERIFY THE TAG BUS DURING DATA COMMAND
13440
13441          ;*****
13442          ;FIRST PART USES OFFSET FORWARD
13443
13444          ;LOAD TEST PARAMETERS IN REGISTER OUTPUT BUFFER
13445 056074 112737 000035 001410      MOV      #29,$RMDAO ;SECTOR 29
13446 056102 112737 000004 001411      MOV      #4,$RMDAO+1 ;TRACK 4
13447 056110 012737 001466 001436      MOV      #822,$RMDCO ;CYLINDER 822
13448 056116 012737 000200 001434      MOV      #OFD,$RMOFO ;FORWARD OFFSET
13449 056124 012737 000071 001402      MOV      #RD!GO,$RMC10 ;READ DATA
13450 056132 012737 177400 001404      MOV      #↑C256,+1,$RMWCO ;WORD COUNT
13451 056140 012737 107030 001406      MOV      #BUFFER,$RMBAO ;BUFFER ADDRESS
13452
13453 056146 004737 056170          ;EXECUTE COMMAND AND VERIFY TAG BUS USING SUBROUTINE
13454          JSR      PC,10$
13455
13456          ;*****
13457          ;SECOND PART USES OFFSET REVERSE

```


M05

CZRMJBO RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 271
T121 DATA COMMAND TESTS(3)

SEQ 0271

```

13457 ;LOAD TEST PARAMETERS IN REGISTER OUTPUT BUFFER
13458 056152 112737 000000 001434 MOVB #0,RMOFO ;REVERSE OFFSET
13459
13460 ;EXECUTE COMMAND AND VERIFY TAG BUS USING SUBROUTINE
13461 056160 004737 056170 JSR PC,10$
13462
13463 056164 000137 057176 JMP 300$
13464
13465 ;*****
13466 ;SUBROUTINE USED DURING TEST
13467
13468 056170 10$:
13469 ;SET VOLUME VALID USING SUBROUTINE
13470 056170 004737 060654 JSR PC,SETVV ;GO SET VOLUME VALID
13471 056174 000403 BR 20$ ;BRANCH TO 20$ IF NO ERROR
13472 056176 104000 ERROR ;RETURN HERE IF ERROR
13473 056200 000137 057014 JMP 160$
13474 056204
13475 20$:
13476 ;LOAD TRACK, SECTOR AND CYLINDER ADDRESS, LOAD OFFSET
13477 056204 013760 001410 000006 MOV RMDAO,RMDA(RO) ;LOAD RMDA
13478
13479 056212 013760 001436 000034 MOV RMDCO,RMDC(RO) ;LOAD RMDC
13480
13481 056220 013760 001434 000032 MOV RMOFO,RMOF(RO) ;LOAD RMOF
13482 ;SET OFFSET MODE USING SUBROUTINE
13483 056226 004737 061004 JSR PC,SETOM ;GO SET OFFSET MODE
13484 056232 000403 BR 30$ ;BRANCH TO 30$ IF NO ERROR
13485 056234 104000 ERROR ;RETURN HERE IF ERROR
13486 056236 000137 057014 JMP 160$
13487 056242
13488 30$:
13489 ;LOAD BUFFER ADDRESS AND WORD COUNT
13490 056242 013760 001404 000002 MOV RMWCO,RMWC(RO) ;LOAD RMWC
13491
13492 056250 013760 001406 000004 MOV RMBAO,RMBA(RO) ;LOAD RMBA
13493 ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
13494
13495 056256 012760 041401 000024 MOV #DMD!MUR!MOC!DBEN,RMMR1(RO) ;LOAD RMMR1
13496
13497 056264 012760 000000 000014 MOV #0,RMER1(RO) ;LOAD RMER1
13498
13499 056272 012760 000000 000042 MOV #0,RMER2(RO) ;LOAD RMER2
13500
13501 056300 013760 001'02 000000 MOV RMCS10,RMCS1(RO) ;LOAD RMCS1
13502 ;WAIT FOR RUN AND GO TO SET
13503 056306 012737 000310 001524 MOV #200,WATCH ;SET WATCHDOG TIMER VALUE
13504 056314 004777 123206 JSR PC,@CLOCK ;START THE CLOCK
13505 056320
13506
13507 056320 016037 000024 001142 MOV RMMR1(RO),SBDDAT ;STORE RMMR1 AT SBDDAT
13508 056326 042737 137777 001142 BIC #1CRG,SBDDAT
13509 056334 001020 BNE 50$
13510 056336 005737 001524 TST WATCH
13511 056342 001366 BNE 40$
13512 056344 004777 123160 JSR PC,@STOP ;STOP THE CLOCK

```

NOS

CZRMJBC RM03/2 DSKLS DIAG
CZRMJB P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 272
T121 DATA COMMAND TESTS(3)

SEQ 0272

13513	056350	012737	040000	001140	MOV	#RG,\$GDDAT	
13514	056356	010037	001136		MOV	RO,\$BDADR	
13515	056362	062737	000024	001136	ADD	#RMMR1,\$BDADR	
13516	056370	104275			ERROR	275	; RUN AND GO NOT SET
13517	056372	000137	057014		JMP	160\$	
13518	056376				50\$:		
13519	056376	004777	123126		JSR	PC,\$STOP	; STOP THE CLOCK
13520	056402	012704	057016		MOV	#200\$,\$R4	; R4 = TABLE POINTER
13521							
13522							
13523	056406	012705	000021				
13524	056412						
13525							
13526	056412	016037	000040	001142	MOV	RMMR2(RO),\$BDDAT	; STORE RMMR2 AT \$BDDAT
13527	056420	042737	150000	001142	BIC	#RQA!RQB!†ST,\$BDDAT	
13528	056426	012437	001140		MOV	(R4)+,\$GDDAT	
13529	056432	053737	001436	001140	BIS	RMOCO,\$GDDAT	; OR CYLINDER ADDRESS
13530	056440	023737	001140	001142	CMP	\$GDDAT,\$BDDAT	
13531	056446	001011			BNE	70\$; BRANCH IF TAG BUS WRONG
13532							
13533	056450	012760	141401	000024	MOV	#UMD!MUR!MOC!DBEN!DBCK,RMMR1(RO)	; LOAD RMMR1
13534							
13535	056456	012760	041401	000024	MOV	#DMJ!MUR!MOC!DBEN,RMMR1(RO)	; LOAD RMMR1
13536	056464	005305			DEC	R5	
13537	056466	001351			BNE	60\$	
13538	056470	000407			BR	80\$	
13539	056472	010037	001136	001136	70\$:	MOV	RO,\$BDADR
13540	056476	062737	000040	001136	ADD	#RMMR2,\$BDADR	
13541	056504	104276			ERROR	276	; INCORRECT TAG BUS DURING DATA
13542	056506	000542			BR	160\$	
13543	056510				80\$:		
13544							
13545							
13546							
13547	056510	012760	041001	000024	MOV	#DMD!MUR!DBEN,RMMR1(RO)	; LOAD RMMR1
13548							
13549	056516	012760	041401	000024	MOV	#DMD!MUR!MOC!DBEN,RMMR1(RO)	; LOAD RMMR1
13550							
13551	056524	012705	000045				
13552	056530						
13553							
13554	056530	016037	000040	001142	MOV	RMMR2(RO),\$BDDAT	; STORE RMMR2 AT \$BDDAT
13555	056536	042737	150000	001142	BIC	#RQA!RQB!†ST,\$BDDAT	
13556	056544	011437	001140		MOV	(R4)+,\$GDDAT	
13557	056550	032714	002000		BIT	#CH,(R4)	
13558	056554	001430			BEQ	110\$; BRANCH IF CONTROL/HEADER NOT ON
13559	056556	032714	004000		BIT	#CC,(R4)	
13560	056562	001416			BEQ	100\$; BRANCH IF HEADER TAG
13561							
13562	056564	052737	000010	001140			
13563	056572	032737	000200	001434	BIS	#B803,\$GDDAT	; ASSUME OFD IS NOT SET
13564	056600	001406			BIT	#OFD,\$MOFO	
13565	056602	042737	000010	001140	BEQ	95\$	
13566	056610	052737	000004	001140	BIC	#B803,\$GDDAT	; RESET BUS BIT 3 - DIRECTION IS REV
13567	056616	000407			BIS	#B802,\$GDDAT	
13568					95\$:	BR	110\$

```

13569 056620
13570
13571 056620 013703 001410
13572 056624 000303
13573 056626 042703 177770
13574 056632 050337 001140
13575 056636
13576
13577 056636 023737 001140 001142
13578 056644 001013
13579
13580 056646 012760 141401 000024
13581
13582 056654 012760 041401 000024
13583 056662 062704 000002
13584 056666 005305
13585 056670 001317
13586 056672 000407
13587 056674 010037 001136
13588 056700 062737 000040 001136
13589 056706 104276
13590 056710 000441
13591 056712
13592
13593
13594
13595 056712 012760 041001 000024
13596
13597 056720 012760 041401 000024
13598
13599 056726 012705 000010
13600 056732
13601
13602 056732 016037 000040 001142
13603 056740 042737 150000 001142
13604 056746 023737 001140 001142
13605 056754 001011
13606
13607 056756 012760 141401 000024
13608
13609 056764 012760 041401 000024
13610 056772 005305
13611 056774 001356
13612 056776 000406
13613 057000 010037 001136
13614 057004 062737 000040 001136
13615 057012 104276
13616
13617 057014 000207
13618
13619 057016
13620
13621 057016 001777
13622 057020 001777
13623 057022 001777
13624 057024 001777

100$:
;HEADER TAG SHOULD BE ON - SETUP EXPECTED TRACK ADDRESS
MOV RMDAO,R3 ;GET TRACK
SWAB R3
BIC #1C7,R3
BIS R3,$GDDAT

110$:
;COMPARE EXPECTED AND RECEIVED TAG BUS DATA
CMP $GDDAT,$BDDAT
BNE 120$

MOV #DMD!MUR!MOC!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1

MOV #DMD!MUR!MOC!DBEN,RMMR1(RO) ;LOAD RMMR1
ADD #2,R4 ;MOVE TABLE POINTER
DEC R5 ;DECREMENT CLOCK COUNT
BNE 90$
BR 130$

120$:
MOV RO,$BDADR
ADD #RMMR2,$BDADR
ERROR 276 ;INCORRECT TAG BUS DURING DATA
BR 160$

130$:
;DROP ON CYLINDER TO RESET LATCH, RAISE ON CYLINDER TO PASS TEST AT
;SEQUENCER LOCATION 175

MOV #DMD!MUR!DBEN,RMMR1(RO) ;LOAD RMMR1

MOV #DMD!MUR!MOC!DBEN,RMMR1(RO) ;LOAD RMMR1
;STEP SEQUENCER TO SECTOR WAIT LOOP (8 CLOCKS)
MOV #8.,R5

140$:
MOV RMMR2(RO),$BDDAT ;STORE RMMR2 AT $BDDAT
BIC #RQA!RQB!1ST,$BDDAT
CMP $GDDAT,$BDDAT ;GOOD DATA SAME AS LAST CMP
BNE 150$ ;BRANCH IF ERROR

MOV #DMD!MUR!MOC!DBEN!DBCK,RMMR1(RO) ;LOAD RMMR1

MOV #DMD!MUR!MOC!DBEN,RMMR1(RO) ;LOAD RMMR1
DEC R5
BNE 140$
BR 160$

150$:
MOV RO,$BDADR
ADD #RMMR2,$BDADR
ERROR 276 ;INCORRECT TAG BUS DURING DATA

160$:
RTS PC

200$:
;TABLE OF TAG BUS CONTROL AND DATA VALUES
.WORD 1777 ;LOCATION 0
.WORD 1777 ;LOCATION 25
.WORD 1777 ;LOCATION 26
.WORD 1777 ;LOCATION 128

```

13625	057026	001777	.WORD	1777	:LOCATION	129
13626	057030	001777	.WORD	1777	:LOCATION	130
13627	057032	004700	.WORD	CC	:LOCATION	144
13628	057034	004700	.WORD	CC	:LOCATION	145
13629	057036	024000	.WORD	CC:TAG	:LOCATION	146
13630	057040	024000	.WORD	CC:TAG	:LOCATION	147
13631	057042	024000	.WORD	CC:TAG	:LOCATION	148
13632	057044	024000	.WORD	CC:TAG	:LOCATION	149
13633	057046	024000	.WORD	CC:TAG	:LOCATION	150
13634	057050	024000	.WORD	CC:TAG	:LOCATION	151
13635	057052	004000	.WORD	CC	:LOCATION	152
13636	057054	004000	.WORD	CC	:LOCATION	153
13637	057056	001777	.WORD	1777	:LOCATION	154
13638	057060	002000	.WORD	CH	:LOCATION	156
13639	057062	002000	.WORD	CH	:LOCATION	157
13640	057064	022000	.WORD	CH:TAG	:LOCATION	158
13641	057066	022000	.WORD	CH:TAG	:LOCATION	159
13642	057070	022000	.WORD	CH:TAG	:LOCATION	160
13643	057072	022000	.WORD	CH:TAG	:LOCATION	161
13644	057074	022000	.WORD	CH:TAG	:LOCATION	162
13645	057076	022000	.WORD	CH:TAG	:LOCATION	163
13646	057100	002000	.WORD	CH	:LOCATION	164
13647	057102	002000	.WORD	CH	:LOCATION	165
13648	057104	001777	.WORD	1777	:LOCATION	232
13649	057106	001777	.WORD	1777	:LOCATION	233
13650	057110	001777	.WORD	1777	:LOCATION	234
13651	057112	001777	.WORD	1777	:LOCATION	235
13652	057114	001777	.WORD	1777	:LOCATION	236
13653	057116	001777	.WORD	1777	:LOCATION	237
13654	057120	001777	.WORD	1777	:LOCATION	238
13655	057122	001777	.WORD	1777	:LOCATION	239
13656	057124	001777	.WORD	1777	:LOCATION	240
13657	057126	001777	.WORD	1777	:LOCATION	241
13658	057130	001777	.WORD	1777	:LOCATION	242
13659	057132	001777	.WORD	1777	:LOCATION	243
13660	057134	001777	.WORD	1777	:LOCATION	244
13661	057136	001777	.WORD	1777	:LOCATION	245
13662	057140	001777	.WORD	1777	:LOCATION	246
13663	057142	001777	.WORD	1777	:LOCATION	247
13664	057144	001777	.WORD	1777	:LOCATION	248
13665	057146	001777	.WORD	1777	:LOCATION	249
13666	057150	001777	.WORD	1777	:LOCATION	250
13667	057152	001777	.WORD	1777	:LOCATION	251
13668	057154	001777	.WORD	1777	:LOCATION	252
13669	057156	001777	.WORD	1777	:LOCATION	166
13670	057160	006000	.WORD	CC:CH	:LOCATION	169
13671	057162	006000	.WORD	CC:CH	:LOCATION	170
13672	057164	026000	.WORD	CC:CH:TAG	:LOCATION	171
13673	057166	026000	.WORD	CC:CH:TAG	:LOCATION	172
13674	057170	026000	.WORD	CC:CH:TAG	:LOCATION	173
13675	057172	026000	.WORD	CC:CH:TAG	:LOCATION	174
13676	057174	026000	.WORD	CC:CH:TAG	:LOCATION	175, ETC
13677						
13678	057176					

300\$:

;END OF TEST

3679 057176
 13680 057176 000240
 13681 057200 013700 001456
 13682 057204 062700 000002
 13683 057210 010037 001456
 13684 057214 005710
 13685 057216 001402
 13686 057220 000137 006322
 13687 057224 012737 001460 001456
 13688
 13689
 13690
 13691
 13692
 13693
 13694
 13695
 13696
 13697 057232
 13698 057232 000240
 13699 057234 005037 001116
 13700 057240 005037 001206
 13701 057244 005237 001230
 13702 057250 042737 100000 001230
 13703 057256 005327
 13704 057260 000001
 13705 057262 003063
 13706 057264 012737
 13707 057266 000001
 13708 057270 057260
 13709 057272 104401 057300
 13710 057276 000407
 13711
 13712 057316
 13713 057316 013746 001230
 13714
 13715 057322 104405
 13716 057324 104401 057332
 13717 057330 000421
 13718
 13719 057374
 13720 057374 013746 001126
 13721
 13722 057400 104405
 13723 057402 104401 001217
 13724 057406 005037 001126
 13725 057412 013700 000042
 13726 057416 001405
 13727 057420 000005
 13728 057422 004710
 13729 057424 000240
 13730 057426 000240
 13731 057430 000240
 13732 057432
 13733 057432 000137
 13734 057434 006322

```

SEOSP:
  NOP
  MOV     TSTQUE,RO
  ADD     #2,RO
  MOV     RO,TSTQUE
  TST    (RO)
  BEQ    1$
  JMP    READY
  MOV     #TSTQUE+2,TSTQUE
1$:
.SBTTL   END OF PASS ROUTINE

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

SEOP:
  NOP
  CLR     $TSTNM           ;; ZERO THE TEST NUMBER
  CLR     $TIMES           ;; ZERO THE NUMBER OF ITERATIONS
  INC     $PASS            ;; INCREMENT THE PASS NUMBER
  BIC     #100000,$PASS    ;; DON'T ALLOW A NEG. NUMBER
  DEC     (PC)+            ;; LOOP?
SEOPCT: .WORD 1
  BGT     $DOAGN           ;; YES
  MOV     (PC)+,a(PC)+    ;; RESTORE COUNTER
SENDCT: .WORD 1
  $EOPCT
  TYPE   65$              ;; TYPE ASCIZ STRING
  BR     64$              ;; GET OVER THE ASCIZ
;;65$: .ASCIZ <12><15>/END PASS #/
64$:
  MOV     $PASS,-(SP)     ;; SAVE $PASS FOR TYPEOUT
                          ;; TYPE PASS NUMBER
                          ;; GO TYPE--DECIMAL ASCII WITH SIGN
  TYPDS
  TYPE   67$              ;; TYPE ASCIZ STRING
  BR     66$              ;; GET OVER THE ASCIZ
;;67$: .ASCIZ / TOTAL ERRORS SINCE LAST REPORT /
66$:
  MOV     $ERTTL,-(SP)    ;; SAVE $ERTTL FOR TYPEOUT
                          ;; TOTAL NUMBER OF ERRORS
                          ;; GO TYPE--DECIMAL ASCII WITH SIGN
  TYPDS
  TYPE   $SCRLF           ;; TYPE CARRIAGE RETURN, LINE FEED
  CLR     $ERTTL
  MOV     #42,RO         ;; CLEAR ERROR TOTAL
  BEQ     $DOAGN         ;; GET MONITOR ADDRESS
  RESET
                          ;; BRANCH IF NO MONITOR
  JSR    PC,(RO)        ;; CLEAR THE WORLD
  NOP
                          ;; GO TO MONITOR
  NOP
                          ;; SAVE ROOM
  NOP
                          ;; FOR
                          ;; ACT11
$DOAGN: JMP     a(PC)+
$PTNAD: .WORD  READY    ;; RETURN

```

E06

CZRMJBO RM03/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 276
END OF PASS ROUTINE

SEG 0276

13735 057436 377 377 000 \$ENULL: .BYTE -1,-1,0 ;:NULL CHARACTER STRING
13736 057442 .EVEN

```

13737
13738
13739
13740
13741
13742
13743
13744
13745
13746
13747
13748
13749
13750
13751
13752 057442
13753 057442 104414
13754 057444 032777 020000 121502
13755 057452 001402
13756 057454 000137 060172
13757
13758 057460 104401 001217
13759 057464 104401 060206
13760 057470 013746 001234
13761
13762 057474 104403
13763 057476 003
13764 057477 000
13765 057500 005037 060176
13766 057504 013737 001226 060176
13767 057512 104401 060214
13768 057516 013746 060176
13769
13770 057522 104403
13771 057524 003
13772 057525 000
13773 057526 005037 060200
13774 057532 1:3737 001130 060200
13775 057540 001406
13776 057542 104401 060224
13777 057546 013746 060200
13778
13779 057552 104403
13780 057554 003
13781 057555 000
13782 057556 104401 060233
13783 057562 013746 001132
13784
13785 057566 104403
13786 057570 006
13787 057571 001
13788
13789 057572 005737 060200
13790 057576 001575
13791 057600 104401 001217
13792 057604 105037 060204

```

```

.SBTTL SUBROUTINES
:*****
.SBTTL ERROR TYPEOUT ROUTINE
:
:THE ERROR TYPEOUT ROUTINE ASSEMBLES AND PRINTS INFORMATION
:REGARDING THE DETECTION OF AN ERROR AS FOLLOWS:
:
:
:UNIT NUMBER, TEST NUMBER, ERROR NUMBER AND PROGRAM COUNTER ARE
:PRINTED ON THE FIRST LINE;
:ERROR MESSAGE IS ASSEMBLED, FORMATTED AND PRINTED ON
:ONE OR MORE SUCCEEDING LINES;
:
:PAIRED LINES OF ERROR HEADERS AND ERROR DATA
:ARE PRINTED AFTER THE ERROR MESSAGE.
ERRTYP:
SAVREG
BIT #SW13,DSWR ;INHIBIT TYPEOUTS??
BEQ 1$ ;NO!!
JMP 21$ ;YES!!
:TYPE UNIT NUMBER, TEST NUMBER, ERROR NUMBER, AND PROGRAM COUNTER
1$:
TYPE $CRLF
TYPE $ERTY00 ;TYPE "UNT#"
MOV $UNIT,-(SP) ;SAVE $UNIT FOR TYPEOUT
;TYPE UNIT NUMBER
;GO TYPE--OCTAL ASCII
;TYPE 3 DIGIT(S)
;SUPPRESS LEADING ZEROS
;LOAD TEST NUMBER FOR
TYPOS
.BYTE 3
.BYTE 0
CLR TSTNMB
MOV $TESTN,TSTNMB
TYPE $ERTY01 ;TYPE "TST#"
MOV $TSTNMB,-(SP) ;SAVE TSTNMB FOR TYPEOUT
;TYPE TEST NUMBER
;GO TYPE--OCTAL ASCII
;TYPE 3 DIGIT(S)
;SUPPRESS LEADING ZEROS
;LOAD ERROR NUMBER FOR
;TYPEOUT
;SKIP IF NO ERROR CALLED
TYPE $ERTY02 ;TYPE "ERR#"
MOV $ERRNMB,-(SP) ;SAVE ERRNMB FOR TYPEOUT
;TYPE ERROR NUMBER
;GO TYPE--OCTAL ASCII
;TYPE 3 DIGIT(S)
;SUPPRESS LEADING ZEROS
;TYPE "PC="
;SAVE $ERRPC FOR TYPEOUT
;TYPE PROGRAM COUNTER
;GO TYPE--OCTAL ASCII
;TYPE 6 DIGIT(S)
;TYPE LEADING ZEROS
;UNLESS ERROR NUMBER IS 0
3$:
TST ERRNMB ;WAS AN ERROR CALLED?
BEQ 21$ ;NO!!
TYPE $CRLF ;YES-TYPE CRLF
CLRB BOTFLG ;CLEAR BOT FLAG

```

```

13793 057610 105037 060205          CLRB   CHRCNT      ;CLEAR CHARACTER COUNTER
13794 057614 013700 060200          MOV    ERRNMB,R0  ;R0 POINTS TO FIRST OF
13795 057620 006300                   ASL    R0         ;FOUR ENTRIES IN ERROR
13796 057622 006300                   ASL    R0         ;TABLE
13797 057624 006300                   ASL    R0
13798 057626 062700 001522          ADD    #ERRTB-B.,R0
13799 057632 011001                   MOV    (R0),R1   ;R1 POINTS TO ERROR MESSAGE
13800                                     ;TABLE
13801 057634 001507                                     ;BRANCH IF NO ERROR MESSAGE
13802                                     ;
13803 057636 012102          4$:   BEQ    12$,   ;R2=ADDRESS OF MESSAGE STRING
13804 057640 001505          MOV    (R1)+,R2  ;BRANCH IF END OF MESSAGE
13805 057642 010237 060010          BEQ    12$,     ;LOAD ADDRESS OF STRING
13806 057646 005037 060202          MOV    R2,11$   ;CLEAR BOT ADDRESS
13807 057652 112203          CLR    BOTADR   ;END OF STRING??
13808 057654 001454          5$:   MOVB   (R2)+,R3 ;YES!!
13809 057656 122703 000015          BEQ    10$,     ;CARRIAGE RETURN??
13810 057662 001003          CMPB  #CR,R3    ;NO!!
13811 057664 105037 060205          BNE    6$,     ;YES-CLEAR CHAR COUNT
13812 057670 000770          CLRB  CHRCNT   ;GET NEXT CHARACTER
13813 057672 122703 000012          BR    5$,     ;LINE FEED??
13814 057676 001765          6$:   BEQ    5$,     ;YES-GET NEXT CHARACTER
13815 057700 122703 000011          CMPB  #HT,R3   ;HORIZONTAL TAB??
13816 057704 001007          BNE    8$,     ;NO!!
13817 057706 105237 060205          INCB  CHRCNT   ;ADJUST CHARACTER COUNT
13818 057712 132737 000007 060205 7$:   BITB  #7,CHRCNT
13819 057720 001372          BNE    7$,     ;
13820 057722 000407          BR    9$,     ;
13821 057724 105237 060205          INCB  CHRCNT   ;INCREMENT CHARACTER COUNT
13822 057730 122703 000040          CMPB  #' ,R3   ;SPACE??
13823 057734 001002          BNE    9$,     ;NO!!
13824 057736 010237 060202          MOV    R2,BOTADR ;SAVE ADDRESS OF SPACE
13825 057742 122737 000100 060205 9$:   CMPB  #64.,CHRCNT ;END OF LINE??
13826 057750 103340          BHS    5$,     ;NO!!
13827 057752 013704 060202          MOV    BOTADR,R4 ;GET ADDRESS OF LAST SPACE
13828 057756 001007          BNE    90$,   ;BRANCH IF SPACE DETECTED
13829 057760 104401 001217          TYPE %CRLF    ;TYPE CRLF
13830 057764 105037 060205          CLRB  CHRCNT   ;CLEAR CHARACTER COUNT
13831 057770 013702 060010          MOV    11$,R2  ;SET UP R2 FOR TESTING
13832 057774 000726          BR    5$,     ;
13833 057776 105044          90$:  CLRB  -(R4)   ;REPLACE SPACE
13834 060000 112737 177777 060204 10$:  MOVB  #-1,BOTFLG ;SET BOT FLAG
13835 060006 104401          TYPE %CRLF    ;TYPE ERROR MESSAGE STRING
13836 060010 000000          11$:  .WORD ;STRING ADDRESS GOES HERE
13837 060012 105737 060204          TSTB  BOTFLG   ;WAS STRING TRUNCATED??
13838 060016 001707          BEQ    4$,     ;NO!!
13839 060020 104401          TYPE %CRLF    ;YES-TYPE CRLF
13840 060024 105037 060204          CLRB  BOTFLG   ;CLEAR BOT FLAG
13841 060030 105037 060205          CLRB  CHRCNT   ;CLEAR CHARACTER COUNT
13842 060034 013702 060202          MOV    BOTADR,R2 ;SETUP R2 FOR TESTING
13843 060040 010237 060010          MOV    R2,11$  ;SETUP 11$ FOR TYPING
13844 060044 112742 000040          MOVB  #' ,-(R2) ;RESTORE SPACE
13845 060050 105722          TSTB  (R2)+    ;RESTORE R2
13846 060052 000677          BR    5$,     ;TYPE REST OF STRING
13847 060054          12$:
13848          ;TYPE ERROR HEADER AND ERROR DATA

```



```

13901      .SBTTL  CLOCK SUBROUTINES
13902
13903      ;ROUTINE TO SIZE FOR CLOCKS (KW11-L OR KW11-P)
13904      ;*****
13905      SIZCLK: NOP
13906      MOV      ERRVEC, -(SP)      ;; PUSH ERRVEC ON STACK
13907      MOV      ERRVEC+2, -(SP)    ;; PUSH ERRVEC+2 ON STACK
13908      MOV      #10$, ERRVEC      ;LOAD 04 TRAP VECTORS
13909      MOV      #PR6, ERRVEC+2
13910
13911      ;SEE IF A KW11-P CLOCK IS PRESENT - GO TO 10$ IF NOT PRESENT
13912      TST      @SLPCSR           ;TEST FOR P CLOCK
13913      MOV      #PCLOCK, CLOCK    ;LOAD SUBROUTINE ADDRESS
13914      MOV      #PSTOP, STOP      ;LOAD STOP ADDRESS
13915      MOV      #PCOUNT, @SLPVEC  ;LOAD P CLOCK INTERRUPT VECTOR
13916      MOV      #PR6, @SLPVEC+2
13917      MOV      @SLLVEC+2, @SLLVEC; CLEAR L CLOCK INTERRUPT VECTOR
13918      CLR      @SLLVEC+2
13919      BR       30$
13920      10$:    MOV      #15$, (SP)  ;DUMMY RTI ADDRESS
13921      RTI
13922      ;RESTORE PRIORITY
13923
13924      15$:
13925      ;NO P CLOCK-SEE IF L CLOCK IS PRESENT-GO TO 20$ IF NOT PRESET
13926      MOV      #20$, ERRVEC      ;CHANGE 04 TRAP VECTOR
13927      TST      @SLLCSR           ;TEST FOR L CLOCK
13928      MOV      #LCLOCK, CLOCK    ;LOAD SUBROUTINE ADDRESS
13929      MOV      #LSTOP, STOP      ;LOAD STOP ADDRESS
13930      MOV      #LCOUNT, @SLLVEC  ;LOAD L CLOCK INTERRUPT VECTOR
13931      MOV      #PR6, @SLLVEC+2
13932      MOV      @SLPVEC+2, @SLPVEC; CLEAR P CLOCK INTERRUPT VECTOR
13933      CLR      @SLPVEC+2
13934      BR       30$
13935      20$:    MOV      #25$, (SP)  ;DUMMY RTI ADDRESS
13936      RTI
13937      ;RESTORE PRIORITY
13938
13939      25$:
13940      ;NO CLOCK AVAILABLE - AUGMENT RETURN ADDRESS
13941      CLR      CLOCK             ;CLEAR SUBROUTINE ADDRESS
13942      MOV      @SLPVEC+2, @SLPVEC; CLEAR P CLOCK INTERRUPT VECTOR
13943      CLR      @SLPVEC+2
13944      MOV      @SLLVEC+2, @SLLVEC; CLEAR L CLOCK INTERRUPT VECTOR
13945      CLR      @SLLVEC+2
13946      ADD      #2, 4(SP)         ;CHANGE RETURN ADDRESS
13947
13948      30$:
13949      MOV      (SP)+, ERRVEC+2    ;POP STACK INTO ERRVEC+2
13950      MOV      (SP)+, ERRVEC     ;; POP STACK INTO ERRVEC
13951      RTS      PC
13952      ;ROUTINES TO START THE CLOCK (KW11-L OR KW11-P)
13953      ;*****
13954      PCLOCK: MOV      #-1, @SLPCSB ;LOAD COUNT SET BUFFER
13955      MOV      #135, @SLPCSR      ;LOAD CONTROL REGISTER
13956      BR       PLCLK             ;GO TO COMMON CODE
13957
13958      LCLOCK: MOV      #100, @SLLCSR ;LOAD CONTROL REGISTER

```

```

13957 060530 005037 001522    PLCLK: CLR      TIME      ;CLEAR TIMER COUNT
13958 060534 104400                TRAP                ;: PUSH OLD PSW AND PC ON STACK
13959 060536 012605                MOV      (SP)+,R5    ;: SAVE THE PSW IN R5
13960 060540 010537 001520    MOV      R5,SPSW    ;: SAVE PRIORITY
13961 060544 042705 177437    BIC      #1CPR7,R5  ;: MASK X
13962 060550 022705 000300    CMP      #PR6,R5    ;: IS PRIORITY TOO HIGH??
13963 060554 101005                BHI      40$        ;: NO!!
13964 060556 012746 000240    MOV      #PR5,-(SP) ;: PUT NEW PS ON STACK
13965 060562 012746 060570    MOV      #30$,-(SP) ;: PUT NEW PC ON STACK
13966 060566 000002                RTI                ;: POP NEW PC AND PS
13967 060570                30$:
13968 060570 000207                40$: RTS      PC
13969
13970                ;ROUTINES TO HANDLE CLOCK INTERRUPTS (KW11-L OR KW11-P)
13971
13972 060572                PCOUNT:
13973 060572                LCOUNT:
13974 060572 062737 000021 001522    ADD      #17.,TIME  ;: ADD 17MS TO ELAPSED TIME
13975 060600 103003                BCC      10$        ;: BRANCH IF NO OVERFLOW
13976 060602 012737 177777 001522    MOV      #-1.,TIME  ;: RESTORE MAXIMUM COUNT
13977 060610 162737 000021 001524    10$: SUB      #17.,WATCH ;: DECREMENT REMAINING TIME
13978 060616 100002                BPL      20$        ;: BRANCH IF POSITIVE
13979 060620 005037 001524    CLR      WATCH      ;: CLEAR REMAINING TIME
13980 060624 000002                20$: RTI                ;: RETURN TO USER
13981
13982                ;ROUTINES TO STOP THE CLOCK (KW11-L OR KW11-P)
13983
13984 060626 005077 120650    PSTOP: CLR      #SLPCSR ;: STOP P CLOCK
13985 060632 000402                BR      PLSTP      ;: GO TO COMMON STOP CODE
13986
13987 060634 005077 120652    LSTOP: CLR      #SLLCSR ;: STOP L CLOCK
13988
13989 060640                PLSTP:
13990 060640 013746 001520    MOV      $PSW,-(SP) ;: PUT NEW PS ON STACK
13991 060644 012746 060652    MOV      #10$,-(SP) ;: PUT NEW PC ON STACK
13992 060650 000002                RTI                ;: POP NEW PC AND PS
13993 060652
13994 060652 000207                10$: RTS      PC
13995
13996                .SBTTL SET VOLUME VALID SUBROUTINE
13997
13998                ; THIS SUBROUTINE INITIALIZES THE SUBSYSTEM AND SETS VOLUME VALID,
13999                ; RETURNING WITH THE DRIVE STILL IN DIAGNOSTIC MODE. THE SUBROUTINE
14000                ; RETURNS TO THE WORD FOLLOWING THE CALL, EXCEPT WHEN AN ERROR IS
14001                ; DETECTED, IN WHICH CASE IT RETURNS TO THE SECOND WORD FOLLOWING THE
14002                ; CALL.
14003
14004                ;CALL: JSR      PC,SETVV      JUMP TO SUBROUTINE
14005                ;      BR      ??              RETURN HERE IF NO ERROR
14006                ;      ERROR            RETURN HERE IF ERROR
14007
14008 060654                SETVV:
14009 060654 012760 000040 000010    MOV      #CLR,RMCS2(RO) ;: CLEAR THE MASSBUS
14010 060662 111160 000010                MOV      (R1),RMCS2(RO) ;: SELECT UNIT
14011
14012 060666 012760 000001 000024    MOV      #DMD,RMMR1(RO) ;: LOAD RMMR1

```

```

14013
14014 060674 012760 001001 000024      MOV      #DMD!MUR,RMMR1(RO)      ;LOAD RMMR1
14015
14016 060702 012760 000000 000014      MOV      #0,RMER1(RO)           ;LOAD RMER1
14017
14018 060710 012760 000000 000042      MOV      #0,RMER2(RO)           ;LOAD RMER2
14019
14020 060716 012760 000023 000000      MOV      #PACACK!GO,RMCS1(RO)   ;LOAD RMCS1
14021
14022 060724 016037 000012 001142      MOV      RMDS(RO), $BDDAT ;STORE RMDS AT $BDDAT
14023 060732 042737 177677 001142      BIC      #1CVV,$BDDAT
14024 060740 001020                BNE      10$                    ;BRANCH IF VOLUME VALID SET
14025 060742 010037 001136                MOV      RO,$BDADR              ;SETUP FOR ERROR MSG
14026 060746 062737 000012 001136      ADD      #RMDS,$BDADR
14027 060754 012737 000100 001140      MOV      #VV,$GDDAT
14028 060762 062716 000002                ADD      #2,(SP)                ;MOVE RETURN ADDRESS TO ERROR
14029 060766 112776 000170 000000      MOV      #170,2(SP)            ;WRITE ERROR NUMBER
14030 060774 012737 000022 001174      MOV      #PACACK,$TMPD
14031 061002 000207      10$:   RTS      PC                ;RETURN
14032
14033      .SBTTL  SET OFFSET MODE SUBROUTINE
14034
14035      ;THIS SUBROUTINE EXECUTES AN OFFSET COMMAND AND VERIFIES THAT OFFSET
14036      ;MODE SETS. THE DRIVE SHOULD BE IN DIAGNOSTIC MODE WHEN CALLING THE
14037      ;SUBROUTINE, WHICH WILL LEAVE DMD ON. THE SUBROUTINE RETURNS TO THE
14038      ;WORD FOLLOWING THE CALL UNLESS THERE IS AN ERROR, IN WHICH CASE IT
14039      ;RETURNS TO THE SECOND WORD FOLLOWING THE CALL
14040
14041      ;CALL:  JSR      PC,SETOM      JUMP TO SUBROUTINE
14042      ;        BR      ??           RETURN HERE IF NO ERROR
14043      ;        ERROR      RETURN HERE IF ERROR
14044
14045      SETOM:
14046
14047 061004 012760 001001 000024      MOV      #DMD!MUR,RMMR1(RO)   ;LOAD RMMR1
14048
14049 061012 012760 000000 000014      MOV      #0,RMER1(RO)         ;LOAD RMER1
14050
14051 061020 012760 000000 000042      MOV      #0,RMER2(RO)         ;LOAD RMER2
14052
14053 061026 012760 000015 000000      MOV      #OFFSET!GO,RMCS1(RO) ;LOAD RMCS1
14054
14055 061034 016037 000012 001142      MOV      RMDS(RO), $BDDAT ;STORE RMDS AT $BDDAT
14056 061042 042737 177776 001142      BIC      #1COM,$BDDAT
14057 061050 001015                BNE      10$                    ;BRANCH IF OFFSET ON
14058 061052 012737 000001 001140      MOV      #OM,$GDDAT
14059 061060 010037 001136                MOV      RO,$BDADR
14060 061064 062737 000012 001136      ADD      #RMDS,$BDADR
14061 061072 062716 000002                ADD      #2,(SP)                ;MOVE RETURN ADDRESS TO ERROR
14062 061076 112776 000200 000000      MOV      #200,2(SP)           ;WRITE ERROR NUMBER
14063 061104
14064 061104 000207      10$:   RTS      PC                ;RETURN TO USER
14065

```

```

14066
14067
14068
14069
14070
14071
14072
14073
14074
14075
14076
14077
14078
14079
14080
14081
14082
14083 061106
14084 061106 010046
14085 061110 010146
14086 061112 010246
14087 061114 010346
14088 061116 010446
14089 061120 010546
14090 061122 016646 000022
14091 061126 016646 000022
14092 061132 016646 000022
14093 061136 016646 000022
14094 061142 000002
14095
14096
14097
14098
14099 061144
14100 061144 012666 000022
14101 061150 012666 000022
14102 061154 012666 000022
14103 061160 012666 000022
14104 061164 012605
14105 061166 012604
14106 061170 012603
14107 061172 012602
14108 061174 012601
14109 061176 012600
14110 061200 000002
14111
14112
14113
14114
14115
14116
14117
14118
14119
14120 061202 010146
14121 061204 016601 000006

```

```

.SBTTL SAVE AND RESTORE RO-R5 ROUTINES
;*****
;SAVE RO-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
$SAVREG:
MOV RO,-(SP) ;: PUSH RO ON STACK
MOV R1,-(SP) ;: PUSH R1 ON STACK
MOV R2,-(SP) ;: PUSH R2 ON STACK
MOV R3,-(SP) ;: PUSH R3 ON STACK
MOV R4,-(SP) ;: PUSH R4 ON STACK
MOV R5,-(SP) ;: PUSH R5 ON STACK
MOV 22(SP),-(SP) ;: SAVE PS OF MAIN FLOW
MOV 22(SP),-(SP) ;: SAVE PC OF MAIN FLOW
MOV 22(SP),-(SP) ;: SAVE PS OF CALL
MOV 22(SP),-(SP) ;: SAVE PC OF CALL
RTI

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

.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
$TYPBN:
MOV R1,-(SP) ;: SAVE R1 ON THE STACK
MOV 6(SP),R1 ;: GET THE INPUT NUMBER

```

M06

CZRMJBO RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 284
BINARY TO ASCII AND TYPE ROUTINE

SEQ 0284

14122	061210	000261				SEC			;; SET "C" SO CAN KEEP TRACK OF THE NUMBER OF BITS
14123	061212	112737	000060	061254	1\$:	MOVB	#'0,\$BIN		;; SET CHARACTER TO AN ASCII "0".
14124	061220	006101				ROL	R1		;; GET THIS BIT
14125	061222	001406				BEQ	2\$;; DONE?
14126	061224	105537	061254			ADCB	\$BIN		;; NO--SET THE CHARACTER EQUAL TO THIS BIT
14127	061230	104401	061254			TYPE	,\$BIN		;; GO TYPE THIS BIT
14128	061234	000241				CLC			;; CLEAR "C" SO CAN KEEP TRACK OF BITS
14129	061236	000765				BR	1\$;; GO DO THE NEXT BIT
14130	061240	012601			2\$:	MOV	(SP)+,R1		;; POP THE STACK INTO R1
14131	061242	016666	000002	000004		MOV	2(SP),4(SP)		;; ADJUST THE STACK
14132	061250	012616				MOV	(SP)+,(SP)		
14133	061252	000002				RTI			;; RETURN TO USER
14134	061254	000	000			\$BIN:	.BYTE 0,0		;; STORAGE FOR ASCII CHAR. AND TERMINATOR
14135						.SBTTL	CONVERT BINARY TO DECIMAL AND TYPE ROUTINE		
14136									
14137									
14138									
14139									
14140									
14141									
14142									
14143									
14144									
14145									
14146									
14147	061256					\$TYPDS:			
14148	061256	010046				MOV	R0,-(SP)		;; PUSH R0 ON STACK
14149	061260	010146				MOV	R1,-(SP)		;; PUSH R1 ON STACK
14150	061262	010246				MOV	R2,-(SP)		;; PUSH R2 ON STACK
14151	061264	010346				MOV	R3,-(SP)		;; PUSH R3 ON STACK
14152	061266	010546				MOV	R5,-(SP)		;; PUSH R5 ON STACK
14153	061270	012746	020200			MOV	#20200,-(SP)		;; SET BLANK SWITCH AND SIGN
14154	061274	016605	000020			MOV	20(SP),R5		;; GET THE INPUT NUMBER
14155	061300	100004				BPL	1\$;; BR IF INPUT IS POS.
14156	061302	005405				NEG	R5		;; MAKE THE BINARY NUMBER POS.
14157	061304	112766	000055	000001		MOVB	#',1(SP)		;; MAKE THE ASCII NUMBER NEG.
14158	061312	005000			1\$:	CLR	R0		;; ZERO THE CONSTANTS INDEX
14159	061314	012703	061472			MOV	#\$DBLK,R3		;; SETUP THE OUTPUT POINTER
14160	061320	112723	000040			MOVB	#',(R3)+		;; SET THE FIRST CHARACTER TO A BLANK
14161	061324	005002			2\$:	CLR	R2		;; CLEAR THE BCD NUMBER
14162	061326	016001	061462			MOV	\$DTBL(R0),R1		;; GET THE CONSTANT
14163	061332	160105			3\$:	SUB	R1,R5		;; FORM THIS BCD DIGIT
14164	061334	002402				BLT	4\$;; BR IF DONE
14165	061336	005202				INC	R2		;; INCREASE THE BCD DIGIT BY 1
14166	061340	000774				BR	3\$		
14167	061342	060105			4\$:	ADD	R1,R5		;; ADD BACK THE CONSTANT
14168	061344	005702				TST	R2		;; CHECK IF BCD DIGIT=0
14169	061346	001002				BNE	5\$;; FALL THROUGH IF 0
14170	061350	105716				TSTB	(SP)		;; STILL DOING LEADING 0'S?
14171	061352	100407				BMI	7\$;; BR IF YES
14172	061354	106316			5\$:	ASLB	(SP)		;; MSD?
14173	061356	103003				BCC	6\$;; BR IF NO
14174	061360	116663	000001	177777		MOVB	1(SP),-1(R3)		;; YES--SET THE SIGN
14175	061366	052702	000060		6\$:	BIS	#'0,R2		;; MAKE THE BCD DIGIT ASCII
14176	061372	052702	000040		7\$:	BIS	#',R2		;; MAKE IT A SPACE IF NOT ALREADY A DIGIT
14177	061376	110223				MOVB	R2,(R3)+		;; PUT THIS CHARACTER IN THE OUTPUT BUFFER

N06

CZRMJBO RM03/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 285
CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

SEQ 0285

14178	061400	005720			TST	(R0)+	;; JUST INCREMENTING	
14179	061402	020027	000010		CMP	R0,#10	;; CHECK THE TABLE INDEX	
14180	061406	002746			BLT	2\$;; GO DO THE NEXT DIGIT	
14181	061410	003002			BGT	8\$;; GO TO EXIT	
14182	061412	010502			MOV	R5,R2	;; GET THE LSD	
14183	061414	000764			BR	6\$;; GO CHANGE TO ASCII	
14184	061416	105726			8\$:	TSTB (SP)+	;; WAS THE LSD THE FIRST NON-ZERO?	
14185	061420	100003			BPL	9\$;; BR IF NO	
14186	061422	116663	177777	177776	MOV	-1(SP),2(R3)	;; YES--SET THE SIGN FOR TYPING	
14187	061430	105013			9\$:	CLRB (R3)	;; SET THE TERMINATOR	
14188	061432	012605			MOV	(SP)+,R5	;; POP STACK INTO R5	
14189	061434	012603			MOV	(SP)+,R3	;; POP STACK INTO R3	
14190	061436	012602			MOV	(SP)+,R2	;; POP STACK INTO R2	
14191	061440	012601			MOV	(SP)+,R1	;; POP STACK INTO R1	
14192	061442	012600			MOV	(SP)+,R0	;; POP STACK INTO R0	
14193	061444	104401	061472		TYPE	\$DBLK	;; NOW TYPE THE NUMBER	
14194	061450	016666	000002	000004	MOV	2(SP),4(SP)	;; ADJUST THE STACK	
14195	061456	012616			MOV	(SP)+,(SP)		
14196	061460	000002			RTI		;; RETURN TO USER	
14197	061462	023420			\$DTBL:	10000.		
14198	061464	001750				1000.		
14199	061466	000144				100.		
14200	061470	000012				10.		
14201	061472	000004			\$DBLK:	.BLKW 4		
14202					.SBTTL	BINARY TO OCTAL (ASCII) AND TYPE		
14203								
14204								
14205							*****	
14206							*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT	
14207							*OCTAL (ASCII) NUMBER AND TYPE IT.	
14208							*\$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE	
14209							*CALL:	
14210						MOV	NUM,-(SP) ;: NUMBER TO BE TYPED	
14211						TYPOS	;; CALL FOR TYPEOUT	
14212						.BYTE	N ;: N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE	
14213						.BYTE	M ;: M=1 OR 0	
14214							;; 1=TYPE LEADING ZEROS	
14215							;; 0=SUPPRESS LEADING ZEROS	
14216							*\$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST	
14217							*\$TYPOS OR \$TYPOC	
14218							*CALL:	
14219						MOV	NUM,-(SP) ;: NUMBER TO BE TYPED	
14220						TYPON	;; CALL FOR TYPEOUT	
14221							*\$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER	
14222							*CALL:	
14223						MOV	NUM,-(SP) ;: NUMBER TO BE TYPED	
14224						TYPOC	;; CALL FOR TYPEOUT	
14225								
14226								
14227	061502	017646	000000		\$TYPOS:	MOV	2(SP),-(SP) ;: PICKUP THE MODE	
14228	061506	116637	000001	061725		MOVB	1(SP),\$OFILL ;: LOAD ZERO FILL SWITCH	
14229	061514	112637	061727			MOVB	(SP)+,\$OMODE+1 ;: NUMBER OF DIGITS TO TYPE	
14230	061520	062716	000002			ADD	#2,(SP) ;: ADJUST RETURN ADDRESS	
14231	061524	000406				BR	\$TYPON	
14232	061526	112737	000001	061725	\$TYPOC:	MOVB	#1,\$OFILL ;: SET THE ZERO FILL SWITCH	
14233	061534	112737	000006	061727		MOVB	#6,\$OMODE+1 ;: SET FOR SIX(6) DIGITS	

```

14234 061542 112737 000005 061724 $TYPON: MOVB #5,$OCNT      ;; SET THE ITERATION COUNT
14235 061550 010346      MOV R3,-(SP)      ;; SAVE R3
14236 061552 010446      MOV R4,-(SP)      ;; SAVE R4
14237 061554 010546      MOV R5,-(SP)      ;; SAVE R5
14238 061556 113704 061727 MOVB $OMODE+1,R4  ;; GET THE NUMBER OF DIGITS TO TYPE
14239 061562 005404      NEG R4
14240 061564 052704 000006 ADD #6,R4        ;; SUBTRACT IT FOR MAX. ALLOWED
14241 061570 110437 061726 MOVB R4,$OMODE  ;; SAVE IT FOR USE
14242 061574 113704 061725 MOVB $OFILL,R4   ;; GET THE ZERO FILL SWITCH
14243 061600 016605 000012 MOV 12(SP),R5   ;; PICKUP THE INPUT NUMBER
14244 061604 005003      CLR R3          ;; CLEAR THE OUTPUT WORD
14245 061606 006105      1$: ROL R5     ;; ROTATE MSB INTO "C"
14246 061610 000404      BR 3$         ;; GO DO MSB
14247 061612 006105      2$: ROL R5     ;; FORM THIS DIGIT
14248 061614 006105      ROL R5
14249 061616 006105      ROL R5
14250 061620 010503      MOV R5,R3
14251 061622 006103      3$: ROL R3     ;; GET LSB OF THIS DIGIT
14252 061624 105337 061726 DECB $OMODE     ;; TYPE THIS DIGIT?
14253 061630 100016      BPL 7$        ;; BR IF NO
14254 061632 042703 177770 BIC #177770,R3 ;; GET RID OF JUNK
14255 061636 001002      BNE 4$        ;; TEST FOR 0
14256 061640 005704      TST R4       ;; SUPPRESS THIS 0?
14257 061642 001403      BEQ 5$        ;; BR IF YES
14258 061644 005204      4$: INC R4     ;; DON'T SUPPRESS ANYMORE 0'S
14259 061646 052703 000060 BIS #'0,R3     ;; MAKE THIS DIGIT ASCII
14260 061652 052703 000040 BIS #' ,R3     ;; MAKE ASCII IF NOT ALREADY
14261 061656 110337 061722 MOVB R3,8$     ;; SAVE FOR TYPING
14262 061662 104401 061722 TYPE 8$        ;; GO TYPE THIS DIGIT
14263 061666 105337 061724 7$: DECB $OCNT  ;; COUNT BY 1
14264 061672 003347      BGT 2$        ;; BR IF MORE TO DO
14265 061674 002402      BLT 6$        ;; BR IF DONE
14266 061676 005204      INC R4       ;; INSURE LAST DIGIT ISN'T A BLANK
14267 061700 000744      BR 2$        ;; GO DO THE LAST DIGIT
14268 061702 012605      6$: MOV (SP)+,R5 ;; RESTORE R5
14269 061704 012604      MOV (SP)+,R4 ;; RESTORE R4
14270 061706 012603      MOV (SP)+,R3 ;; RESTORE R3
14271 061710 016666 000002 000004 MOV 2(SP),4(SP) ;; SET THE STACK FOR RETURNING
14272 061716 012616      MOV (SP)+,(SP)
14273 061720 000002      RTI         ;; RETURN
14274 061722 000      8$: .BYTE 0   ;; STORAGE FOR ASCII DIGIT
14275 061723 000      .BYTE 0   ;; TERMINATOR FOR TYPE ROUTINE
14276 061724 000      $OCNT: .BYTE 0 ;; OCTAL DIGIT COUNTER
14277 061725 000      $OFILL: .BYTE 0 ;; ZERO FILL SWITCH
14278 061726 000000      $OMODE: .WORD 0 ;; NUMBER OF DIGITS TO TYPE
14279
14280
14281
14282
14283
14284
14285
14286
14287
14288
14289

```

```

*****
*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 ROUTINE
;*      TYPE      ,MESADR      ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
;*OR
;*      TYPE
;*      MESADR
;*
14290
14291
14292
14293
14294
14295
14296 061730 105737 001173 $TYPE: TSTB $TPFLG      ;; IS THERE A TERMINAL?
14297 061734 100002      BPL 1$      ;; BR IF YES
14298 061736 000000      HALT      ;; HALT HERE IF NO TERMINAL
14299 061740 000430      BR      ;; LEAVE
14300 061742 010046 1$: MOV R0,-(SP)      ;; SAVE R0
14301 061744 017600 000002      MOV 22(SP),R0      ;; GET ADDRESS OF ASCIZ STRING
14302 061750 122737 000001 001242      CMPB #APTENV,$ENV      ;; RUNNING IN APT MODE
14303 061756 001011      BNE 62$      ;; NO GO CHECK FOR APT CONSOLE
14304 061760 132737 000100 001243      BITB #APTSPool,$ENVM      ;; SPOOL MESSAGE TO APT
14305 061766 001405      BEQ 62$      ;; NO GO CHECK FOR CONSOLE
14306 061770 010037 062000      MOV R0,61$      ;; SETUP MESSAGE ADDRESS FOR APT
14307 061774 004737 065020      JSR PC,$ATY3      ;; SPOOL MESSAGE TO APT
14308 062000 000000      .WORD 0      ;; MESSAGE ADDRESS
14309 062002 132737 000040 001243 61$: BITB #APTCSUP,$ENVM      ;; APT CONSOLE SUPPRESSED
14310 062010 001003      BNE 60$      ;; YES SKIP TYPE OUT
14311 062012 112046 2$: MOVVB (R0)+,-(SP)      ;; PUSH CHARACTER TO BE TYPED ONTO STACK
14312 062014 001005      BNE 4$      ;; BR IF IT ISN'T THE TERMINATOR
14313 062016 005726      TST (SP)+      ;; IF TERMINATOR POP IT OFF THE STACK
14314 062020 012600 60$: MOV (SP)+,R0      ;; RESTORE R0
14315 062022 062716 000002 3$: ADD #2,(SP)      ;; ADJUST RETURN PC
14316 062026 000002      RTI      ;; RETURN
14317 062030 122716 000011 4$: CMPB #HT,(SP)      ;; BRANCH IF <HT>
14318 062034 001430      BEQ 8$      ;;
14319 062036 122716 000200      CMPB #CRLF,(SP)      ;; BRANCH IF NOT <CRLF>
14320 062042 001006      BNE 5$      ;;
14321 062044 005726      TST (SP)+      ;; POP <CR><LF> EQUIV
14322 062046 104401      TYPE      ;; TYPL A CR AND LF
14323 062050 001217 $CRLF
14324 062052 105037 062206      CLRB $CHARCNT      ;; CLEAR CHARACTER COUNT
14325 062056 000755      BR 2$      ;; GET NEXT CHARACTER
14326 062060 004737 062142 5$: JSR PC,$TYPEC      ;; GO TYPE THIS CHARACTER
14327 062064 123726 001172 6$: CMPB $FILLC,(SP)+      ;; IS IT TIME FOR FILLER CHARS.?
14328 062070 001350      BNE 2$      ;; IF NO GO GET NEXT CHAR.
14329 062072 013746 001170      MOV $NULL,-(SP)      ;; GET # OF FILLER CHARS. NEEDED
14330      AND THE NULL CHAR.
14331 062076 105366 000001 7$: DECB 1(SP)      ;; DOES A NULL NEED TO BE TYPED?
14332 062102 002770      BLT 6$      ;; BR IF NO--GO POP THE NULL OFF OF STACK
14333 062104 004737 062142      JSR PC,$TYPEC      ;; GO TYPE A NULL
14334 062110 105337 062206      DECB $CHARCNT      ;; DO NOT COUNT AS A COUNT
14335 062114 000770      BR 7$      ;; LOOP
14336
14337 ;HORIZONTAL TAB PROCESSOR
14338
14339 062116 112716 000040 8$: MOVVB #'(SP)      ;; REPLACE TAB WITH SPACE
14340 062122 004737 062142 9$: JSR PC,$TYPEC      ;; TYPE A SPACE
14341 062126 132737 000007 062206      BITB #7,$CHARCNT      ;; BRANCH IF NOT AT
14342 062134 001372      BNE 9$      ;; TAB STOP
14343 062136 005726      TST (SP)+      ;; POP SPACE OFF STACK
14344 062140 000724      BR 2$      ;; GET NEXT CHARACTER
14345 062142 105777 117016 $TYPEC: TSTB 2$TPS      ;; WAIT UNTIL PRINTER IS READ

```

```

14346 062146 100375      BPL      $TYPEC
14347 062150 116677 000002 117010      MOVB     2(SP),2$TPB      ;:LOAD CHAR TO BE TYPED INTO DATA REG.
14348 062156 122766 000015 000002      CMPB     #CR,2(SP)      ;:IS CHARACTER A CARRIAGE RETURN?
14349 062164 001003      BNE     1$             ;:BRANCH IF NO
14350 062166 105037 062206      CLRB     $CHARCNT      ;:YES--CLEAR CHARACTER COUNT
14351 062172 000406      BR      $TYPEX        ;:EXIT
14352 062174 122766 000012 000002 1$:      CMPB     #LF,2(SP)     ;:IS CHARACTER A LINE FEED?
14353 062202 001402      BEQ     $TYPEX        ;:BRANCH IF YES
14354 062204 105227      INCB     (PC)+        ;:COUNT THE CHARACTER
14355 062206 000000      $CHARCNT: WORD 0      ;:CHARACTER COUNT STORAGE
14356 062210 000207      $TYPEX: RTS          PC
14357
14358      .SBTTL  SCOPE HANDLER ROUTINE
14359
14360      ;:*****
14361      ;:THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
14362      ;:AND LOAD THE TEST NUMBER($STNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
14363      ;:AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
14364      ;:THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
14365      ;:$W14=1      LOOP ON TEST
14366      ;:$W11=1      INHIBIT ITERATIONS
14367      ;:$W09=1      LOOP ON ERROR
14368      ;:$W08=1      LOOP ON TEST IN SWR<7:0>
14369      ;:$CALL
14370      ;:$*      SCOPE      ;:SCOPE=IOT
14371
14372      $SCOPE:
14373      062212 104410      CKSWR
14374 062214 032777 040000 116732 1$:      BIT     #BIT14,$SWR    ;:TEST FOR CHANGE IN SOFT-SWR
14375 062222 001131      BNE     $OVER         ;:LOOP ON PRESENT TEST?
14376      ;:*****START OF CODE FOR THE XOR TESTER*****
14377 062224 000416      $XTSTR: BR     6$     ;:IF RUNNING ON THE "XOR" TESTER CHANGE
14378      ;:THIS INSTRUCTION TO A "NOP" (NOP=240)
14379 062226 013746 000004      MOV     2$ERRVEC, -(SP) ;:SAVE THE CONTENTS OF THE ERROR VECTOR
14380 062232 012737 062252 000004      MOV     #5,$ERRVEC    ;:SET FOR TIMEOUT
14381 062240 005737 177060      TST     2$177060     ;:TIME OUT ON XOR?
14382 062244 012637 000004      MOV     (SP)+,2$ERRVEC ;:RESTORE THE ERROR VECTOR
14383 062250 000500      BR      $SVLAD        ;:GO TO THE NEXT TEST
14384 062252 022626      $S:      CMP     (SP)+,(SP)+ ;:CLEAR THE STACK AFTER A TIME OUT
14385 062254 012637 000004      MOV     (SP)+,2$ERRVEC ;:RESTORE THE ERROR VECTOR
14386 062260 000440      BR      7$           ;:LOOP ON THE PRESENT TEST
14387 062262
14388 062262 032777 000400 116664 6$:;*****END OF CODE FOR THE XOR TESTER*****
14389 062270 001421      BIT     #BIT08,$SWR   ;:LOOP ON SPEC. TEST?
14390 062272 005046      BEQ     2$           ;:BR IF NO
14391 062274 117716 116654      CLR     -(SP)        ;:CLEAR A TEMP. LOCATION
14392 062300 001414      MOVB    2$SWR,(SP)   ;:PICKUP THE DESIRED TEST NUMBER
14393 062302 022716 000121      BEQ     8$           ;:BRANCH IF BAD TEST NUMBER IN SWR
14394 062306 002411      CMP     #121,(SP)   ;:CHECK THE NUMBER IN THE SWR
14395 062310 011637 001116      BLT    8$           ;:BRANCH IF TEST NUMBER IS OUT OF RANGE
14396 062314 005316      MOV     (SP),$STNM  ;:UPDATE THE TEST NUMBER
14397 062316 006316      DEC     (SP)        ;:BACKUP BY ONE
14398 062320 062716 062524      ASL     (SP)        ;:SCALE THE TEST NUMBER AS AN INDEX
14399 062324 013637 001122      ADD     #5$W08TBL,(SP) ;:FORM THE ADDRESS OF TEST POINTER
14400 062330 000466      MOV     2(SP)+,$LPAOR ;:SET LOOP ADDRESS TO DESIRED TEST
14401 062332 005726      BR      $OVER       ;:GO LOOP ON THE TEST
14401      $S:      TST     (SP)+      ;:CLEAN THE BAD TEST NUMBER OFF OF THE STACK

```

14402	062334	105737	001117		2S:	TSTB	\$ERFLG	...	HAS AN ERROR OCCURRED?
14403	062340	001421				BEQ	3S	...	BR IF NO
14404	062342	123737	001131	001117		CMPB	\$ERMAX, \$ERFLG	...	MAX. ERRORS FOR THIS TEST OCCURRED?
14405	062350	101015				BHI	3S	...	BR IF NO
14406	062352	032777	001000	116574		BIT	#BIT09, @SWR	...	LOOP ON ERROR?
14407	062360	001404				BEQ	4S	...	BR IF NO
14408	062362	013737	001124	001122	7S:	MOV	\$LPERR, \$LPADR	...	SET LOOP ADDRESS TO LAST SCOPE
14409	062370	000446				BR	\$OVER		
14410	062372	105037	001117		4S:	CLRB	\$ERFLG	...	ZERO THE ERROR FLAG
14411	062376	005037	001206			CLR	\$TIMES	...	CLEAR THE NUMBER OF ITERATIONS TO MAKE
14412	062402	000415				BR	1S	...	ESCAPE TO THE NEXT TEST
14413	062404	032777	004000	116542	3S:	BIT	#BIT11, @SWR	...	INHIBIT ITERATIONS?
14414	062412	001011				BNE	1S	...	BR IF YES
14415	062414	005737	001230			TST	\$PASS	...	IF FIRST PASS OF PROGRAM
14416	062420	001406				BEQ	1S	...	INHIBIT ITERATIONS
14417	062422	005237	001120			INC	\$ICNT	...	INCREMENT ITERATION COUNT
14418	062426	023737	001206	001120		CMP	\$TIMES, \$ICNT	...	CHECK THE NUMBER OF ITERATIONS MADE
14419	062434	002024				BGE	\$OVER	...	BR IF MORE ITERATION REQUIRED
14420	062436	012737	000001	001120	1S:	MOV	#1, \$ICNT	...	REINITIALIZE THE ITERATION COUNTER
14421	062444	013737	062522	001206		MOV	\$MXCNT, \$TIMES	...	SET NUMBER OF ITERATIONS TO DO
14422	062452	105237	001116		\$SVLAD:	INCB	\$TSTNM	...	COUNT TEST NUMBERS
14423	062456	113737	001116	001226		MOVB	\$TSTNM, \$TESTN	...	SET TEST NUMBER IN APT MAILBOX
14424	062464	011637	001122			MOV	(SP), \$LPADR	...	SAVE SCOPE LOOP ADDRESS
14425	062470	011637	001124			MOV	(SP), \$LPERR	...	SAVE ERROR LOOP ADDRESS
14426	062474	005037	001210			CLR	\$ESCAPE	...	CLEAR THE ESCAPE FROM ERROR ADDRESS
14427	062500	112737	000001	001131		MOVB	#1, \$ERMAX	...	ONLY ALLOW ONE(1) ERROR ON NEXT TEST
14428	062506	013777	001116	116442	\$OVER:	MOV	\$TSTNM, @DISPLAY	...	DISPLAY TEST NUMBER
14429	062514	013716	001122			MOV	\$LPADR, (SP)	...	FUDGE RETURN ADDRESS
14430	062520	000002				RTI		...	FIXES PS
14431	062522	000012						...	MAX. NUMBER OF ITERATIONS
14432	062524				\$MXCNT:	10.			
14433	062524	006340			\$SWOBTBL:				
14434	062526	006714				.WORD	TST1+2	...	STARTING ADDRESS OF TEST 1
14435	062530	007134				.WORD	TST2+2	...	STARTING ADDRESS OF TEST 2
14436	062532	007350				.WORD	TST3+2	...	STARTING ADDRESS OF TEST 3
14437	062534	007536				.WORD	TST4+2	...	STARTING ADDRESS OF TEST 4
14438	062536	010674				.WORD	TST5+2	...	STARTING ADDRESS OF TEST 5
14439	062540	012054				.WORD	TST6+2	...	STARTING ADDRESS OF TEST 6
14440	062542	012234				.WORD	TST7+2	...	STARTING ADDRESS OF TEST 7
14441	062544	012356				.WORD	TST10+2	...	STARTING ADDRESS OF TEST 10
14442	062546	012654				.WORD	TST11+2	...	STARTING ADDRESS OF TEST 11
14443	062550	013266				.WORD	TST12+2	...	STARTING ADDRESS OF TEST 12
14444	062552	014114				.WORD	TST13+2	...	STARTING ADDRESS OF TEST 13
14445	062554	014650				.WORD	TST14+2	...	STARTING ADDRESS OF TEST 14
14446	062556	015006				.WORD	TST15+2	...	STARTING ADDRESS OF TEST 15
14447	062560	015362				.WORD	TST16+2	...	STARTING ADDRESS OF TEST 16
14448	062562	015522				.WORD	TST17+2	...	STARTING ADDRESS OF TEST 17
14449	062564	016044				.WORD	TST20+2	...	STARTING ADDRESS OF TEST 20
14450	062566	016220				.WORD	TST21+2	...	STARTING ADDRESS OF TEST 21
14451	062570	016404				.WORD	TST22+2	...	STARTING ADDRESS OF TEST 22
14452	062572	016714				.WORD	TST23+2	...	STARTING ADDRESS OF TEST 23
14453	062574	017252				.WORD	TST24+2	...	STARTING ADDRESS OF TEST 24
14454	062576	017732				.WORD	TST25+2	...	STARTING ADDRESS OF TEST 25
14455	062600	020072				.WORD	TST26+2	...	STARTING ADDRESS OF TEST 26
14456	062602	020416				.WORD	TST27+2	...	STARTING ADDRESS OF TEST 27
14457	062604	020770				.WORD	TST30+2	...	STARTING ADDRESS OF TEST 30
						.WORD	TST31+2	...	STARTING ADDRESS OF TEST 31

14458	062606	021316	.WORD	TST32+2	STARTING	ADDRESS	OF	TEST	32
14459	062610	022072	.WORD	TST33+2	STARTING	ADDRESS	OF	TEST	33
14460	062612	022442	.WORD	TST34+2	STARTING	ADDRESS	OF	TEST	34
14461	062614	022772	.WORD	TST35+2	STARTING	ADDRESS	OF	TEST	35
14462	062616	023352	.WORD	TST36+2	STARTING	ADDRESS	OF	TEST	36
14463	062620	023740	.WORD	TST37+2	STARTING	ADDRESS	OF	TEST	37
14464	062622	024460	.WORD	TST40+2	STARTING	ADDRESS	OF	TEST	40
14465	062624	025010	.WORD	TST41+2	STARTING	ADDRESS	OF	TEST	41
14466	062626	025252	.WORD	TST42+2	STARTING	ADDRESS	OF	TEST	42
14467	062630	025712	.WORD	TST43+2	STARTING	ADDRESS	OF	TEST	43
14468	062632	026120	.WORD	TST44+2	STARTING	ADDRESS	OF	TEST	44
14469	062634	026536	.WORD	TST45+2	STARTING	ADDRESS	OF	TEST	45
14470	062636	027304	.WORD	TST46+2	STARTING	ADDRESS	OF	TEST	46
14471	062640	027636	.WORD	TST47+2	STARTING	ADDRESS	OF	TEST	47
14472	062642	030276	.WORD	TST50+2	STARTING	ADDRESS	OF	TEST	50
14473	062644	030570	.WORD	TST51+2	STARTING	ADDRESS	OF	TEST	51
14474	062646	031034	.WORD	TST52+2	STARTING	ADDRESS	OF	TEST	52
14475	062650	031724	.WORD	TST53+2	STARTING	ADDRESS	OF	TEST	53
14476	062652	032234	.WORD	TST54+2	STARTING	ADDRESS	OF	TEST	54
14477	062654	032466	.WORD	TST55+2	STARTING	ADDRESS	OF	TEST	55
14478	062656	032730	.WORD	TST56+2	STARTING	ADDRESS	OF	TEST	56
14479	062660	033162	.WORD	TST57+2	STARTING	ADDRESS	OF	TEST	57
14480	062662	033400	.WORD	TST60+2	STARTING	ADDRESS	OF	TEST	60
14481	062664	033654	.WORD	TST61+2	STARTING	ADDRESS	OF	TEST	61
14482	062666	034052	.WORD	TST62+2	STARTING	ADDRESS	OF	TEST	62
14483	062670	034326	.WORD	TST63+2	STARTING	ADDRESS	OF	TEST	63
14484	062672	034472	.WORD	TST64+2	STARTING	ADDRESS	OF	TEST	64
14485	062674	034674	.WORD	TST65+2	STARTING	ADDRESS	OF	TEST	65
14486	062676	035162	.WORD	TST66+2	STARTING	ADDRESS	OF	TEST	66
14487	062700	035430	.WORD	TST67+2	STARTING	ADDRESS	OF	TEST	67
14488	062702	035706	.WORD	TST70+2	STARTING	ADDRESS	OF	TEST	70
14489	062704	036174	.WORD	TST71+2	STARTING	ADDRESS	OF	TEST	71
14490	062706	036436	.WORD	TST72+2	STARTING	ADDRESS	OF	TEST	72
14491	062710	036700	.WORD	TST73+2	STARTING	ADDRESS	OF	TEST	73
14492	062712	037164	.WORD	TST74+2	STARTING	ADDRESS	OF	TEST	74
14493	062714	037330	.WORD	TST75+2	STARTING	ADDRESS	OF	TEST	75
14494	062716	037512	.WORD	TST76+2	STARTING	ADDRESS	OF	TEST	76
14495	062720	037672	.WORD	TST77+2	STARTING	ADDRESS	OF	TEST	77
14496	062722	040054	.WORD	TST100+2	STARTING	ADDRESS	OF	TEST	100
14497	062724	040236	.WORD	TST101+2	STARTING	ADDRESS	OF	TEST	101
14498	062726	040456	.WORD	TST102+2	STARTING	ADDRESS	OF	TEST	102
14499	062730	040716	.WORD	TST103+2	STARTING	ADDRESS	OF	TEST	103
14500	062732	041102	.WORD	TST104+2	STARTING	ADDRESS	OF	TEST	104
14501	062734	041344	.WORD	TST105+2	STARTING	ADDRESS	OF	TEST	105
14502	062736	041532	.WORD	TST106+2	STARTING	ADDRESS	OF	TEST	106
14503	062740	041764	.WORD	TST107+2	STARTING	ADDRESS	OF	TEST	107
14504	062742	042212	.WORD	TST110+2	STARTING	ADDRESS	OF	TEST	110
14505	062744	042520	.WORD	TST111+2	STARTING	ADDRESS	OF	TEST	111
14506	062746	043062	.WORD	TST112+2	STARTING	ADDRESS	OF	TEST	112
14507	062750	043334	.WORD	TST113+2	STARTING	ADDRESS	OF	TEST	113
14508	062752	044756	.WORD	TST114+2	STARTING	ADDRESS	OF	TEST	114
14509	062754	046534	.WORD	TST115+2	STARTING	ADDRESS	OF	TEST	115
14510	062756	050720	.WORD	TST116+2	STARTING	ADDRESS	OF	TEST	116
14511	062760	051330	.WORD	TST117+2	STARTING	ADDRESS	OF	TEST	117
14512	062762	053502	.WORD	TST120+2	STARTING	ADDRESS	OF	TEST	120
14513	062764	056020	.WORD	TST121+2	STARTING	ADDRESS	OF	TEST	121

```

14514
14515
14516
14517
14518
14519
14520
14521
14522
14523
14524
14525
14526
14527
14528 062766
14529 062766 104410
14530 062770 105237 001117
14531 062774 001775
14532 062776 013777 001116 116152
14533 063004 032777 002000 116142
14534 063012 001402
14535 063014 104401 001212
14536 063020 005237 001126
14537 063024 011637 001132
14538 063030 162737 000002 001132
14539 063036 117737 116070 001130
14540 063044 032777 020000 116102
14541 063052 001004
14542 063054 004737 057442
14543 063060 104401 001217
14544 063064
14545 063064 122737 000001 001242
14546 063072 001007
14547 063074 113737 001130 063106
14548 063102 004737 065030
14549 063106 000
14550 063107 000
14551 063110 000777
14552 063112 005777 116036
14553 063116 100002
14554 063120 000000
14555 063122 104410
14556 063124 032777 001000 116022
14557 063132 001402
14558 063134 013716 001124
14559 063140 005737 001210
14560 063144 001402
14561 063146 013716 001210
14562 063152
14563 063152 022737 057422 000042
14564 063160 001001
14565 063162 000000
14566 063164
14567 063164 000002
14568
14569

```

```

.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 ERRYP 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

$ERROR:
7$: CKSWR ;;TEST FOR CHANGE IN SOFT-SWR
INCB $ERFLG ;;SET THE ERROR FLAG
BEQ 7$ ;;DON'T LET THE FLAG GO TO ZERO
MOV $STNM,$DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
BIT #BIT10,$SWR ;;BELL ON ERROR?
BEQ 1$ ;;NO - SKIP
TYPE $BELL ;;RING BELL
1$: INC $ERTTL ;;COUNT THE NUMBER OF ERRORS
MOV (SP) $ERRPC ;;GET ADDRESS OF ERROR INSTRUCTION
SUB #2,$ERRPC
MOVB #2,$ERRPC,$ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
BIT #BIT13,$SWR ;;SKIP TYPEOUT IF SET
BNE 20$ ;;SKIP TYPEOUTS
JSR PC,ERRYP ;;GO TO USER ERROR ROUTINE
TYPE $CRLF

20$: CMPB #APTENV,$ENV ;;RUNNING IN APT MODE
BNE 2$ ;;NO SKIP APT ERROR REPORT
MOVB $ITEMB,21$ ;;SET ITEM NUMBER AS ERROR NUMBER
JSR PC,$ATY4 ;;REPORT FATAL ERROR TO APT

21$: .BYTE 0
.BYTE 0

22$: BR 22$ ;;APT ERROR LOOP
2$: TST $SWR ;;HALT ON ERROR
BPL 3$ ;;SKIP IF CONTINUE
HALT ;;HALT ON ERROR!
CKSWR ;;TEST FOR CHANGE IN SOFT-SWR
3$: BIT #BIT09,$SWR ;;LOOP ON ERROR SWITCH SET?
BEQ 4$ ;;BR IF NO
MOV $LPERR,(SP) ;;FUJGE RETURN FOR LOOPING
4$: TST $ESCAPE ;;CHECK FOR AN ESCAPE ADDRESS
BEQ 5$ ;;BR IF NONE
MOV $ESCAPE,(SP) ;;FUJGE RETURN ADDRESS FOR ESCAPE

5$: CMP #SENDAD,$#42 ;;ACT-11 AUTO-ACCEPT?
BNE 6$ ;;BRANCH IF NO
HALT ;;YES

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

```

```

14570
14571
14572 063166 000000
14573 063170 000000
14574 063172 000000
14575 063174 000001
14576 063175
14577 063176
14578
14579
14580
14581
14582
14583
14584
14585
14586
14587 063176 005037 063166
14588 063202 012737 063174 063170
14589 063210 013737 063170 063172
14590 063216 012737 063246 000060
14591 063224 012737 000200 000062
14592 063232 005777 115724
14593 063236 012777 000100 115714
14594 063244 000207
14595
14596
14597
14598
14599
14600
14601
14602
14603 063246 117746 115710
14604 063252 042716 177600
14605 063256 021627 000003
14606 063262 001007
14607 063264 104401 064362
14608 063270 004737 063176
14609 063274 005726
14610 063276 000137 004542
14611 063302 021627 000007
14612 063306 001004
14613 063310 022737 000176 001154
14614 063316 001500
14615
14616 063320
14617 063320 022737 000001 063166
14618 063326 001004
14619 063330 104401 001212
14620 063334 005726
14621 063336 000451
14622 063340 021627 000023
14623 063344 001021
14624 063346 005077 115606
14625 063352 005726

```

```

*****
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
$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 (START)
$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 $CNTLC      ;; TYPE A CONTROL-C (↑C)
        JSR PC,$TKINT    ;; INIT THE KEYBOARD
        TST (SP)+        ;; CLEAN UP STACK
        JMP START        ;; 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
2$:
   CMP #1,$TKCNT      ;; IS THE QUEUE FULL?
   BNE 3$            ;; BRANCH IF NO
   TYPE $BELL         ;; RING THE TTY BELL
   TST (SP)+         ;; CLEAN CHARACTER OFF OF STACK
   BR 5$             ;; EXIT
3$: CMP (SP),#23      ;; IS IT A CONTROL-S?
   BNE 32$          ;; BRANCH IF NO
   CLR $TKS         ;; DISABLE TTY KEYBOARD INTERRUPTS
   TST (SP)+       ;; CLEAN CHAR OFF STACK

```

```

14626 063354 105777 115600      31$:  TSTB  2$TKS      ;; WAIT FOR A CHAR
14627 063360 100375                BPL  31$          ;; LOOP UNTIL ITS THERE
14628 063362 117746                MOVB 2$TKB, -(SP) ;; GET THE CHARACTER
14629 063366 042716 177600      BIC  #1C177, (SP) ;; MAKE IT 7-BIT ASCII
14630 063372 022627 000021      CMP  (SP)+, #21   ;; IS IT A CONTROL-Q?
14631 063376 001366                BNE  31$          ;; BRANCH IF NO
14632 063400 012777 000100 115552  MOV  #100, 2$TKS  ;; REENABLE TTY KEYBOARD INTERRUPTS
14633 063406 000002                RTI                    ;; RETURN
14634 063410 005237 063166      32$:  INC  $TKCNT      ;; COUNT THIS CHARACTER
14635 063414 021627 000140      CMP  (SP), #140   ;; IS IT UPPER CASE?
14636 063420 002405                BLT  4$           ;; BRANCH IF YES
14637 063422 021627 000175      CMP  (SP), #175   ;; IS IT A SPECIAL CHAR?
14638 063426 003002                BGT  4$           ;; BRANCH IF YES
14639 063430 042716 000040      BIC  #40, (SP)    ;; MAKE IT UPPER CASE
14640 063434 112677 177530      4$:  MOVB  (SP)+, 2$TKQIN ;; AND PUT IT IN QUEUE
14641 063440 005237 063170      INC  $TKQIN       ;; UPDATE THE POINTER
14642 063444 023727 063175 063175  CMP  $TKQIN, #2$TKQEND ;; GO OFF THE END?
14643 063452 001003                BNE  5$           ;; BRANCH IF NO
14644 063454 012737 063174 063170  MOV  #2$TKQSR, $TKQIN ;; RESET THE POINTER
14645 063462 000002                RTI                    ;; RETURN
14646
14647
14648
14649
14650
14651
14652 063464 022737 000176 001154  $CKSWR: CMP  #SWREG, SWR ;; IS THE SOFT-SWR SELECTED
14653 063472 001124                BNE  15$          ;; EXIT IF NOT
14654 063474 105777 115460      TSTB 2$TKS        ;; IS A CHAR WAITING?
14655 063500 100121                BPL  15$          ;; IF NOT, EXIT
14656 063502 117746 115454      MOVB 2$TKB, -(SP) ;; YES
14657 063506 042716 177600      BIC  #1C177, (SP) ;; MAKE IT 7-BIT ASCII
14658 063512 021627 000007      CMP  (SP), #7     ;; IS IT A CONTROL-G?
14659 063516 001300                BNE  2$           ;; IF NOT, PUT IT IN THE TTY QUEUE
14660
14661
14662
14663
14664
14665
14666 063520 123727 001150 000001 6$:  CMPB  $AUTOB, #1 ;; ARE WE RUNNING IN AUTO-MODE?
14667 063526 001674                BEQ  2$           ;; BRANCH IF YES
14668 063530 005726                TST  (SP)+        ;; CLEAR CONTROL-G OFF STACK
14669 063532 004737 063176      JSR  PC, $TKINT   ;; FLUSH THE TTY INPUT QUEUE
14670 063536 005077 115416      CLR  2$TKS        ;; DISABLE TTY KEYBOARD INTERRUPTS
14671 063542 112737 000001 001151  MOVB  #1, $INTAG  ;; SET INTERRUPT MODE INDICATOR
14672
14673 063550 104401 064374      TYPE  , $CNTLG    ;; ECHO THE CONTROL-G (#G)
14674 063554 104401 064401      TYPE  $MSWR       ;; TYPE CURRENT CONTENTS
14675 063560 013746 000176      MOV  SWREG, -(SP) ;; SAVE SWREG FOR TYPEOUT
14676 063564 104402                TYPOC            ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
14677 063566 104401 064412      TYPE  , $MNEW     ;; PROMPT FOR NEW SWR
14678 063572 005046      19$:  CLR  -(SP)        ;; CLEAR COUNTER
14679 063574 005046      CLR  -(SP)        ;; THE NEW SWR
14680 063576 105777 115356      7$:  TSTB  2$TKS        ;; CHAR THERE?
14681 063602 100375                BPL  7$           ;; IF NOT TRY AGAIN

```

```

14682
14683 063604 117746 115352          MOVB   2STKB, -(SP)          ;; PICK UP CHAR
14684 063610 042716 177600          BIC    #1C177, (SP)        ;; MAKE IT 7-BIT ASCII
14685
14686 063614 021627 000003          CMP    (SP), #3            ;; IS IT A CONTROL-C?
14687 063620 001015                    BNE    9$                  ;; BRANCH IF NOT
14688 063622 104401 064362          TYPE   $CNTLC              ;; YES, ECHO CONTROL-C (↑C)
14689 063626 062706 000006          ADD    #6, SP              ;; CLEAN UP STACK
14690 063632 123727 001151 000001    CMPB   $INTAG, #1          ;; REENABLE TTY KEYBOARD INTERRUPTS?
14691 063640 001003                    BNE    8$                  ;; BRANCH IF NO
14692 063642 012777 000100 115310    MOV    #100, 2STKS        ;; ALLOW TTY KEYBOARD INTERRUPTS
14693 063650 000137 004542          JMP    START               ;; CONTROL-C RESTART
14694
14695
14696 063654 021627 000025          9$:   CMP    (SP), #25        ;; IS IT A CONTROL-U?
14697 063660 001005                    BNE    10$                 ;; BRANCH IF NOT
14698 063662 104401 064367          TYPE   $CNTLU              ;; YES, ECHO CONTROL-U (↑U)
14699 063666 062706 000006          20$:  ADD    #6, SP              ;; IGNORE PREVIOUS INPUT
14700 063672 000737                    BR     19$                 ;; LET'S TRY IT AGAIN
14701
14702
14703 063674 021627 000015          10$:  CMP    (SP), #15         ;; IS IT A <CR>?
14704 063700 001022                    BNE    16$                 ;; BRANCH IF NO
14705 063702 005766 000004          TST    4(SP)                ;; YES, IS IT THE FIRST CHAR?
14706 063706 001403                    BEQ    11$                 ;; BRANCH IF YES
14707 063710 016677 000002 115236    MOV    2(SP), 2SWR          ;; SAVE NEW SWR
14708 063716 062706 000006          11$:  ADD    #6, SP              ;; CLEAN UP STACK
14709 063722 104401 001217          14$:  TYPE   $CRLF              ;; ECHO <CR> AND <LF>
14710 063726 123727 001151 000001    CMPB   $INTAG, #1          ;; RE-ENABLE TTY KBD INTERRUPTS?
14711 063734 001003                    BNE    15$                 ;; BRANCH IF NOT
14712 063736 012777 000100 115214    MOV    #100, 2STKS        ;; RE-ENABLE TTY KBD INTERRUPTS
14713 063744 000002                    RTI                               ;; RETURN
14714 063746 004737 062142          15$:  JSR    PC, $TYPEPC        ;; ECHO CHAR
14715 063752 021627 000060          16$:  CMP    (SP), #60          ;; CHAR < 0?
14716 063756 002420                    BLT    18$                 ;; BRANCH IF YES
14717 063760 021627 000067          CMP    (SP), #67          ;; CHAR > 7?
14718 063764 003015                    BGT    18$                 ;; BRANCH IF YES
14719 063766 042726 000060          BIC    #60, (SP)+         ;; STRIP-OFF ASCII
14720 063772 005766 000002          TST    2(SP)                ;; IS THIS THE FIRST CHAR
14721 063776 001403                    BEQ    17$                 ;; BRANCH IF YES
14722 064000 006316                    ASL    (SP)                ;; NO, SHIFT PRESENT
14723 064002 006316                    ASL    (SP)                ;; CHAR OVER TO MAKE
14724 064004 006316                    ASL    (SP)                ;; ROOM FOR NEW ONE.
14725 064006 005266 000002          17$:  INC    2(SP)              ;; KEEP COUNT OF CHAR
14726 064012 056616 177776          BIS    -2(SP), (SP)        ;; SET IN NEW CHAR
14727 064016 000667                    BR     7$                  ;; GET THE NEXT ONE
14728 064020 104401 001216          18$:  TYPE   $QUES              ;; TYPE ?<CR><LF>
14729 064024 000720                    BR     20$                 ;; SIMULATE CONTROL-U
14730
14731
14732
14733
14734
14735
14736
14737

```

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


```

14738 ;* ; ; WITH PARITY BIT STRIPPED OFF
14739 ;
14740 ;
14741 064026 011646 ; SRDCHR: MOV (SP), -(SP) ; PUSH DOWN THE PC AND
14742 064030 016666 000004 000002 MOV 4(SP), 2(SP) ; THE PS
14743 064036 005066 000004 CLR 4(SP) ; GET READY FOR A CHARACTER
14744 064042 005046 CLR -(SP) ; PUT NEW PS ON STACK
14745 064044 012746 064052 MOV #64$, -(SP) ; PUT NEW PC ON STACK
14746 064050 000002 RTI ; POP NEW PC AND PS
14747 064052
14748 064052 005737 063166 64$: TST $TKCNT ; ; WAIT ON A CHARACTER
14749 064056 001775 1$: BEQ 1$
14750 064060 005337 063166 DEC $TKCNT ; DECREMENT THE COUNTER
14751 064064 117766 177102 000004 MOVB $TKQOUT, 4(SP) ; GET ONE CHARACTER
14752 064072 005237 063172 INC $TKQOUT ; UPDATE THE POINTER
14753 064076 023727 063172 063175 CMP $TKQOUT, # $TKQEND ; DID IT GO OFF OF THE END?
14754 064104 001003 BNE 2$ ; BRANCH IF NO
14755 064106 012737 063174 063172 MOV # $TKQSR, $TKQOUT ; RESET THE POINTER
14756 064114 000002 2$: RTI ; RETURN
14757 ; *****
14758 ; THIS ROUTINE WILL INPUT A STRING FROM THE TTY
14759 ; *CALL:
14760 ; * RDLIN ; INPUT A STRING FROM THE TTY
14761 ; * RETURN HERE ; ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
14762 ; * ; TERMINATOR WILL BE A BYTE OF ALL 0'S
14763 ;
14764 064116 010346 ; SRDLIN: MOV R3, -(SP) ; SAVE R3
14765 064120 005046 CLR -(SP) ; CLEAR THE RUBOUT KEY
14766 064122 012703 064352 1$: MOV # $TTYIN, R3 ; GET ADDRESS
14767 064126 022703 064362 2$: CMP # $TTYIN+8., R3 ; BUFFER FULL?
14768 064132 101456 BLOS 4$ ; BR IF YES
14769 064134 104411 RDRCHR ; GO READ ONE CHARACTER FROM THE TTY
14770 064136 112613 MOVB (SP)+, (R3) ; GET CHARACTER
14771 064140 122713 000177 10$: CMPB #177, (R3) ; IS IT A RUBOUT
14772 064144 001022 BNE 5$ ; BR IF NO
14773 064146 005716 TST (SP) ; IS THIS THE FIRST RUBOUT?
14774 064150 001007 BNE 6$ ; BR IF NO
14775 064152 112737 000134 064350 MOVB #' \, 9$ ; TYPE A BACK SLASH
14776 064160 104401 064350 TYPE 9$
14777 064164 012716 177777 MOV #-1, (SP) ; SET THE RUBOUT KEY
14778 064170 005303 6$: DEC R3 ; BACKUP BY ONE
14779 064172 020327 064352 CMP R3, # $TTYIN ; STACK EMPTY?
14780 064176 103434 BLO 4$ ; BR IF YES
14781 064200 111337 064350 MOVB (R3), 9$ ; SETUP TO TYPEOUT THE DELETED CHAR.
14782 064204 104401 064350 TYPE 9$ ; GO TYPE
14783 064210 000746 BR 2$ ; GO READ ANOTHER CHAR.
14784 064212 005716 5$: TST (SP) ; RUBOUT KEY SET?
14785 064214 001406 BEQ 7$ ; BR IF NO
14786 064216 112737 000134 064350 MOVB #' \, 9$ ; TYPE A BACK SLASH
14787 064224 104401 064350 TYPE 9$
14788 064230 005016 CLR (SP) ; CLEAR THE RUBOUT KEY
14789 064232 122713 000025 7$: CMPB #25, (R3) ; IS CHARACTER A CTRL U?
14790 064236 001003 BNE 8$ ; BR IF NO
14791 064240 104401 064367 TYPE $CNTLU ; TYPE A CONTROL "U"
14792 064244 000726 1$ BR 1$ ; GO START OVER
14793 064246 122713 000022 8$: CMPB #22, (R3) ; IS CHARACTER A "↑"?

```

```

14794 064252 001011          BNE      3$          ;; BRANCH IF NO
14795 064254 105013          CLRB     (R3)       ;; CLEAR THE CHARACTER
14796 064256 104401 001217      TYPE    ,SCRLF     ;; TYPE A "CR" & "LF"
14797 064262 104401 064352      TYPE    ,STTYIN    ;; TYPE THE INPUT STRING
14798 064266 000717          BR       2$          ;; GO PICKUP ANOTHER CHAFTER
14799 064270 104401 001216      4$:    TYPE    ,SQUES  ;; TYPE A '?'
14800 064274 000712          BR       1$          ;; CLEAR THE BUFFER AND LOOP
14801 064276 111337 064350      3$:    MOVVB   (R3),9$  ;; ECHO THE CHARACTER
14802 064302 104401 064350      TYPE    ,9$
14803 064306 122723 000015      CMPB    15,(R3)+   ;; CHECK FOR RETURN
14804 064312 001305          BNE     2$          ;; LOOP IF NOT RETURN
14805 064314 105063 177777      CLRB    -1(R3)     ;; CLEAR RETURN (THE 15)
14806 064320 104401 001220      TYPE    ,SLF      ;; TYPE A LINE FEED
14807 064324 005726          TST     (SP)+      ;; CLEAN RUBOUT KEY FROM THE STACK
14808 064326 012603          MOV     (SP)+,R3   ;; RESTORE R3
14809 064330 011646          MOV     (SP)-,(SP) ;; ADJUST THE STACK AND PUT ADDRESS OF THE
14810 064332 016666 000004 000002  MOV     4(SP),2(SP) ;; FIRST ASCII CHARACTER ON IT
14811 064340 012766 064352 000004  MOV     *STTYIN,4(SP)
14812 064346 000002          RTI
14813 064350          9$:    .BYTE   0          ;; RETURN
14814 064351          .BYTE   0          ;; STORAGE FOR ASCII CHAR. TO TYPE
14815 064352 000010          $TTYIN: .BLKB   8          ;; TERMINATOR
14816 064362 041536 005015 000      $CNTLC: .ASCIZ  /?C/<15><12> ;; RESERVE 8 BYTES FOR TTY INPUT
14817 064367 136 006525 000012  $CNTLU: .ASCIZ  /?U/<15><12> ;; CONTROL "C"
14818 064374 043536 005015 000      $CNTLG: .ASCIZ  /?G/<15><12> ;; CONTROL "U"
14819 064401 015 051412 051127  $MSWR:  .ASCIZ  <15><12>/SWR = / ;; CONTROL "G"
14820 064406 036440 000040
14821 064412 020040 042516 020127  $MNEW:  .ASCIZ  / NEW = /
14822 064420 020075 000
14823 064424
14824          .EVEN
14825          .SBTTL READ AN OCTAL NUMBER FROM THE TTY
14826          ;; *****
14827          ;; *THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
14828          ;; *CHANGE IT TO BINARY.
14829          ;; *CALL:
14830          ;; *   RDOCT
14831          ;; *   RETURN HERE
14832          ;; *
14833          ;; *   READ AN OCTAL NUMBER
14834          ;; *   LOW ORDER BITS ARE ON TOP OF THE STACK
14835          ;; *   HIGH ORDER BITS ARE IN $HIOCT
14836          $RDOCT: MOV     (SP)-,(SP) ;; PROVIDE SPACE FOR THE
14837          MOV     4(SP),2(SP) ;; INPUT NUMBER
14838          MOV     R0,-(SP) ;; PUSH R0 ON STACK
14839          MOV     R1,-(SP) ;; PUSH R1 ON STACK
14840          MOV     R2,-(SP) ;; PUSH R2 ON STACK
14841          1$:  RDLIN  ;; READ AN ASCIZ LINE
14842          MOV     (SP)+,R0 ;; GET ADDRESS OF 1ST CHARACTER
14843          CLR     R1 ;; CLEAR DATA WORD
14844          CLR     R2
14845          2$:  MOVVB   (R0)+,-(SP) ;; PICKUP THIS CHARACTER
14846          BEQ    3$ ;; IF ZERO GET OUT
14847          ASL    R1 ;; *2
14848          ROL    R2 ;; *4
14849          ASL    R1 ;; *8

```

MO7

CZRMJBO RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 297
READ AN OCTAL NUMBER FROM THE TTY

SEQ 0297

14850 064470 006102
14851 064472 042716 177770
14852 064476 062601
14853 064500 000764
14854 064502 005726
14855 064504 010166 000012
14856 064510 010237 064524
14857 064514 012602
14858 064516 012601
14859 064520 012600
14860 064522 000002
14861 064524 00000C
14862
14863
14864
14865
14866
14867
14868
14869
14870 064526 016646 000002
14871 064532 042716 000020
14872 064534 012746 064544
14873 064542 000002
14874 064544 010046
14875 064546 016600 000002
14876 064552 005740
14877 064554 11100C
14878 064556 006300
14879 064560 016000 064600
14880 064564 000200
14881
14882
14883
14884
14885 064566 011646
14886 064570 016666 000004 000002
14887 064576 000002
14888
14889
14890
14891
14892
14893
14894
14895
14896 064600 064566
14897 064602 061730
14898 064604 061526
14899 064606 061502
14900 064610 061542
14901 064612 061626
14902 064614 061202
14903
14904 064616 063554
14905

```

ROL      R2
BIC      #1C7,(SP)      ;; STRIP THE ASCII JUNK
AND      (SP)+,R1      ;; ADD IN THIS DIGIT
BR       2$            ;; LOOP
3$:      TST      (SP)+      ;; CLEAN TERMINATOR FROM STACK
          MOV      R1,12(SP)  ;; SAVE THE RESULT
          MOV      R2,$HIOCT
          MOV      (SP)+,R2      ;; POP STACK INTO R2
          MOV      (SP)+,R1      ;; POP STACK INTO R1
          MOV      (SP)+,R0      ;; POP STACK INTO R0
          RTI
$HIOCT:  .WORD    0
.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.
    
```

```

$TRAP:  MOV      2(SP),-(SP)      ;; ASSUME THE STATUS OF
          BIC      #20,(SP)      ;; THE CALLER--DO NOT ALLOW
          MOV      #1$,-(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

```

$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)
          $TYPOS  ;; CALL=TYPOS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
          $TYPBN  ;; CALL=TYPN     TRAP+6(104406)  TYPE BINARY (ASCII) NUMBER
          $GTSWR  ;; CALL=GTSWR    TRAP+7(104407)  GET SOFT-SWR SETTING
    
```

```

14906 064620 063464          $CKSWR  ;;CALL=CKSWR   TRAP+10(104410) TEST FOR CHANGE IN SOFT-SWR
14907 064622 064026          $RDCHR  ;;CALL=RDCHR   TRAP+11(104411) TTY TYPEIN CHARACTER ROUTINE
14908 064624 064116          $RDLIN  ;;CALL=RDLIN   TRAP+12(104412) TTY TYPEIN STRING ROUTINE
14909 064626 064424          $RDOCT  ;;CALL=RDOCT   TRAP+13(104413) READ AN OCTAL NUMBER FROM TTY
14910 064630 061106          $SAVREG ;;CALL=SAVREG  TRAP+14(104414) SAVE R0-R5 ROUTINE
14911 064632 061144          $RESREG ;;CALL=RESREG  TRAP+15(104415) RESTORE R0-R5 ROUTINE
14912
14913 .SBTTL POWER DOWN AND UP ROUTINES
14914
14915 ::*****
14916 :POWER DOWN ROUTINE
14917 $PWRDN: MOV      $SILLUP, @PWRVEC      ;;SET FOR FAST UP
14918        MOV      @340, @PWRVEC+2  ;;PRIO:7
14919        MOV      R0, -(SP)         ;;PUSH R0 ON STACK
14920        MOV      R1, -(SP)         ;;PUSH R1 ON STACK
14921        MOV      R2, -(SP)         ;;PUSH R2 ON STACK
14922        MOV      R3, -(SP)         ;;PUSH R3 ON STACK
14923        MOV      R4, -(SP)         ;;PUSH R4 ON STACK
14924        MOV      R5, -(SP)         ;;PUSH R5 ON STACK
14925        MOV      @SWR, -(SP)       ;;PUSH @SWR ON STACK
14926        MOV      SP, $SAVR6       ;;SAVE SP
14927        MOV      @SPWRUP, @PWRVEC ;;SET UP VECTOR
14928        HALT
14929        BR        .-2              ;;HANG UP
14930
14931 ::*****
14932 :POWER UP ROUTINE
14933 $PWRUP: MOV      $SILLUP, @PWRVEC  ;;SET FOR FAST DOWN
14934        MOV      $SAVR6, SP        ;;GET SP
14935        CLR      $SAVR6           ;;WAIT LOOP FOR THE TTY
14936        IS:    INC      $SAVR6     ;;WAIT FOR THE INC
14937        BNE     IS              ;;OF WORD
14938        MOV     (SP)+, @SWR       ;;POP STACK INTO @SWR
14939        MOV     (SP)+, R5        ;;POP STACK INTO R5
14940        MOV     (SP)+, R4        ;;POP STACK INTO R4
14941        MOV     (SP)+, R3        ;;POP STACK INTO R3
14942        MOV     (SP)+, R2        ;;POP STACK INTO R2
14943        MOV     (SP)+, R1        ;;POP STACK INTO R1
14944        MOV     (SP)+, R0        ;;POP STACK INTO R0
14945        MOV     @SPWRDN, @PWRVEC ;;SET UP THE POWER DOWN VECTOR
14946        MOV     @340, @PWRVEC+2 ;;PRIO:7
14947        TYPE    $PWRMG: .WORD    $POWER ;;REPORT THE POWER FAILURE
14948        RTI    $PWRMG:          ;;POWER FAIL MESSAGE POINTER
14949        $SILLUP: HALT
14950        BR     .-2              ;;THE POWER UP SEQUENCE WAS STARTED
14951        $SAVR6: 0                ;;BEFORE THE POWER DOWN WAS COMPLETE
14952        $POWER: .ASCIZ <15><12>"POWER" ;;PUT THE SP HERE
14953
14954 .EVEN
14955 .SBTTL APT COMMUNICATIONS ROUTINE
14956
14957 ::*****
14958 $SATY1: MOV     #1, $FFLG        ;;TO REPORT FATAL ERROR
14959 $SATY3: MOV     #1, $MFLG        ;;TO TYPE A MESSAGE
14960        BR     $ATYC
14961 $SATY4: MOV     #1, $FFLG        ;;TO ONLY REPORT FATAL ERROR

```

```

14962 065036 010046 SATYC:
14963 065036 010146 MOV RO,-(SP) ;; PUSH RO ON STACK
14964 065040 010146 MOV R1,-(SP) ;; PUSH R1 ON STACK
14965 065042 105737 065254 TSTB $MFLG ;; SHOULD TYPE A MESSAGE?
14966 065046 001450 BEQ 5$ ;; IF NOT: BR
14967 065050 122737 000001 001242 CMPB #APTENV,$ENV ;; OPERATING UNDER APT?
14968 065056 001031 BNE 3$ ;; IF NOT: BR
14969 065060 132737 000100 071243 BITB #APTPOOL,$ENVM ;; SHOULD SPOOL MESSAGES?
14970 065066 001425 BEQ 3$ ;; IF NOT: BR
14971 065070 017600 000004 MOV #4(SP),RO ;; GET MESSAGE ADDR.
14972 065074 062766 000002 000004 ADD #2,4(SP) ;; BUMP RETURN ADDR.
14973 065102 005737 001222 1$: TST $MSGTYPE ;; SEE IF DONE W/ LAST XMISSION?
14974 065106 001375 BNE 1$ ;; IF NOT: WAIT
14975 065110 010037 001236 MOV RO,$MSGAD ;; PUT ADDR IN MAILBOX
14976 065114 105720 2$: TSTB (RO)+ ;; FIND END OF MESSAGE
14977 065116 001376 BNE 2$
14978 065120 163700 001236 SUB $MSGAD,RO ;; SUB START OF MESSAGE
14979 065124 006200 ASR RO ;; GET MESSAGE LGTH IN WORDS
14980 065126 010037 001240 MOV RO,$MSGLGTH ;; PUT LENGTH IN MAILBOX
14981 065132 012737 000004 001222 MOV #4,$MSGTYPE ;; TELL APT TO TAKE MSG.
14982 065140 000413 BR 5$
14983 065142 017637 000004 065166 3$: MOV #4(SP),4$ ;; PUT MSG ADDR IN JSR LINKAGE
14984 065150 062766 000002 000004 ADD #2,4(SP) ;; BUMP RETURN ADDRESS
14985 065156 013746 177776 MOV 177776,-(SP) ;; PUSH 177776 ON STACK
14986 065162 004737 061730 JSR PC,$TYPE ;; CALL TYPE MACRO
14987 065166 000000 4$: .WORD 0
14988 065170 5$:
14989 065170 105737 065256 10$: TSTB $FFLG ;; SHOULD REPORT FATAL ERROR?
14990 065174 001416 BEQ 12$ ;; IF NOT: BR
14991 065176 005737 001242 TST $ENV ;; RUNNING UNDER APT?
14992 065202 001413 BEQ 12$ ;; IF NOT: BR
14993 065204 005737 001222 11$: TST $MSGTYPE ;; FINISHED LAST MESSAGE?
14994 065210 001375 BNE 11$ ;; IF NOT: WAIT
14995 065212 017637 000004 001224 000004 MOV #4(SP),$FATAL ;; GET ERROR #
14996 065220 062766 000002 000004 ADD #2,4(SP) ;; BUMP RETURN ADDR.
14997 065226 005237 001222 INC $MSGTYPE ;; TELL APT TO TAKE ERROR
14998 065232 105037 065256 12$: CLRB $FFLG ;; CLEAR FATAL FLAG
14999 065236 105037 065255 CLRB $LFLG ;; CLEAR LOG FLAG
15000 065242 105037 065254 CLRB $MFLG ;; CLEAR MESSAGE FLAG
15001 065246 012601 MOV (SP)+,R1 ;; POP STACK INTO R1
15002 065250 012600 MOV (SP)+,RO ;; POP STACK INTO RO
15003 065252 000207 RTS PC ;; RETURN
15004 065254 000 $MFLG: .BYTE 0 ;; MESSG. FLAG
15005 065255 000 $LFLG: .BYTE 0 ;; LOG FLAG
15006 065256 000 $FFLG: .BYTE 0 ;; FATAL FLAG
15007 065260 .EVEN
15008 000200 APTSIZE=200
15009 000001 APTENV=001
15010 000100 APTPOOL=100
15011 000040 APTCSUP=040
15012

```

```

15013
15014
15015 065260 051
15016 065261 075 000
15017 065262 015 025012 000
15018 065263 077 000
15019 065271
15020 065271 015 052012 050131
15021 065276 020105 042510 050114
15022 065304 052040 054105 020124
15023 065312 054450 047440 020122
15024 065320 024516 037477 000
15025 065325
15026 065325 015 041412 040510
15027 065332 043516 020105 046522
15028 065340 031460 052440 044516
15029 065346 052502 020123 042101
15030 065374 051104 051505 020123
15031 065372 051117 053040 041505
15032 065370 047524 020122 042101
15033 065376 051104 051505 020123
15034 065404 054450 047440 020122
15035 065412 024516 041474 037122
15036 065420 037440 000077
15037 065424 005015 051525 020105
15038 065432 040523 042515 042040
15039 065440 053105 041511 051505
15040 065446 024040 020131 051117
15041 065454 047040 020051 037477
15042 065462 000
15043 065463 015 051012 030115
15044 065470 020063 052502 020123
15045 065476 042101 051104 051505
15046 065504 020123 000050
15047 065510 005015 047105 051124
15048 065516 020131 047516 020124
15049 065524 047111 044440 047457
15050 065532 050040 043501 105
15051 065537 015 040412 042104
15052 065544 042522 051523 046440
15053 065552 051525 020124 042502
15054 065560 037040 033061 030060
15055 065566 030060 000
15056 065571 015 051012 030115
15057 065576 020063 042526 052103
15058 065604 051117 040440 042104
15059 065612 042522 051523 024040
15060 065620 000
15061 065621 015 042412 052116
15062 065626 054522 047440 052125
15063 065634 047440 020106 040522
15064 065642 043516 105
15065 065645 015 040412 042104
15066 065652 042522 051523 046440
15067 065660 051525 020124 042502
15068 065666 036040 030061 030060

```

```

.SBTTL CONSOLE MESSAGES
CLSPRN: .ASCII @@@
EQUALS: .ASCIZ @=@
PROMPT: .ASCIZ <CR><LF>@*@
QSTMRK: .ASCIZ @?@
HELPOST:
.ASCIZ <CR><LF>@TYPE HELP TEXT (Y OR N)??@

UBUSQST:
.ASCIZ <CR><LF>@CHANGE RMO3 UNIBUS ADDRESS OR VECTOR ADDRESS (Y OR N)<CR> ??@

CNSLOO: .ASCIZ <CR><LF>@USE SAME DEVICES (Y OR N) ??@

CNSLO1: .ASCIZ <CR><LF>@RMO3 BUS ADDRESS (@

CNSLO2: .ASCII <CR><LF>@ENTRY NOT IN I/O PAGE@

.ASCIZ <CR><LF>@ADDRESS MUST BE >160000@

CNSLO3: .ASCIZ <CR><LF>@RMO3 VECTOR ADDRESS (@

CNSLO4: .ASCII <CR><LF>@ENTRY OUT OF RANGE@

.ASCIZ <CR><LF>@ADDRESS MUST BE <1000@

```

15069	065674	000			
15070	065675	015	051012	030115	CNSL05: .ASCIZ <CR><LF>@RM03 INTERRUPT PRIORITY @
15071	065702	020063	047111	042524	
15072	065710	051122	050125	020124	
15073	065716	051120	047511	044522	
15074	065724	054524	024040	000	
15075	065731	015	042412	052116	CNSL06: .ASCIZ <CR><LF>@ENTRY OUT OF RANGE@
15076	065736	054522	047440	052125	
15077	065744	047440	020106	040522	
15078	065752	043516	000105		
15079	065756				CNSL07:
15080	065756	005015	054524	042520	.ASCII <CR><LF>@TYPE (A) TO TEST ALL DEVICES, OR TYPE DEVICE@
15081	065764	024040	024501	052040	
15082	065772	020117	042524	052123	
15083	066000	040440	046114	042040	
15084	066006	053105	041511	051505	
15085	066014	020054	051117	052040	
15086	066022	050131	020105	042504	
15087	066030	044526	042503		
15088	066034	047040	046525	042502	.ASCII @ NUMBER(S)@
15089	066042	024122	024523		
15090	066046	005015	042524	046522	.ASCIZ <CR><LF>@TERMINATE INPUT WITH CARRIAGE RETURN@
15091	066054	047111	052101	020105	
15092	066062	047111	052520	020124	
15093	066070	044527	044124	041440	
15094	066076	051101	044522	043501	
15095	066104	020105	042522	052524	
15096	066112	047122	000		
15097	066115	015	047012	052117	NOTEX: .ASCIZ <CR><LF>/NOT EXIST DRIVE /
15098	066122	042440	044530	052123	
15099	066130	042040	044522	042526	
15100	066136	020040	000		
15101					
15102	066142				.EVEN

15103
15104 066142
15105
15106 066142 020000
15107 066144 130001
15108 066146 132000
15109 066150 130000
15110 066152 020000
15111 066154 030000
15112 066156 130000
15113 066160 130000
15114 066162 020000
15115 066164 020000
15116 066166 130001
15117 066170 130001
15118 066172 132000
15119 066174 130001
15120 066176 130001
15121 066200 130001
15122 066202 130001
15123 066204 130001
15124 066206 130001
15125 066210 130001
15126 066212 073300
15127 066214 073300
15128 066216 130001
15129 066220 130001
15130 066222 037200
15131 066224 037000
15132 066226 130001
15133 066230 130001
15134 066232 033300
15135 066234 033300
15136 066236 130001
15137 066240 130001
15138
15139 066242 001
15140 066243 002
15141 066244 004
15142 066245 010
15143 066246 020
15144 066247 040
15145 066250 100
15146 066251 200
15147
15148 066252
15149 066252 000000
15150 066254 000001
15151 066256 000033
15152 066260 000007
15153 066262 000017
15154 066264 000037
15155 066266 000077
15156 066270 000177
15157 066272 000377
15158 066274 000777

FNCDTB:

.WORD OPI
.WORD OPI:ATA:ILF:IVC
.WORD ATA:OPI:IVC:IAE
.WORD ATA:OPI:IVC
.WORD OPI
.WORD OPI:IVC
.WORD OPI:ATA:IVC
.WORD OPI:ATA:IVC
.WORD OPI
.WORD OPI
.WORD OPI:ATA:ILF:IVC
.WORD OPI:ATA:ILF:IVC
.WORD ATA:OPI:IVC:IAE
.WORD OPI:ATA:ILF:IVC
.WORD OPI:ATA:ILF:IVC
.WORD OPI:ATA:ILF:IVC
.WORD OPI:ATA:ILF:IVC
.WORD OPI:ATA:ILF:IVC
.WORD OPI:ATA:ILF:IVC
.WORD OPI:ATA:ILF:IVC
.WORD OPI:ATA:ILF:IVC
.WORD WCE:OPI:IVC:IAE:AOE:HCE:ECH
.WORD WCE:OPI:IVC:IAE:AOE:HCE:ECH
.WORD OPI:ATA:ILF:IVC
.WORD OPI:ATA:ILF:IVC
.WORD OPI:IVC:WLE:IAE:AOE:HCE
.WORD OPI:IVC:WLE:IAE:AOE
.WORD OPI:ATA:ILF:IVC
.WORD OPI:ATA:ILF:IVC
.WORD OPI:IVC:IAE:AOE:HCE:ECH
.WORD OPI:IVC:IAE:AOE:HCE:ECH
.WORD OPI:ATA:ILF:IVC
.WORD OPI:ATA:ILF:IVC

;FUNCTION CODE TABLE

:NOP
:ILLEGAL FUNCTION (2)
:SEEK
:RECALIBRATE
:DRIVE CLEAR
:RELEASE
:OFFSET
:RETURN TO CENTERLINE
:READ IN PRESET
:PACK ACKNOWLEDGE
:ILLEGAL FUNCTION (24)
:ILLEGAL FUNCTION (26)
:SEARCH
:ILLEGAL FUNCTION (32)
:ILLEGAL FUNCTION (34)
:ILLEGAL FUNCTION (36)
:ILLEGAL FUNCTION (40)
:ILLEGAL FUNCTION (42)
:ILLEGAL FUNCTION (44)
:ILLEGAL FUNCTION (46)
:WRITE CHECK DATA
:WRITE CHECK HEADER AND DATA
:ILLEGAL FUNCTION (54)
:ILLEGAL FUNCTION (56)
:WRITE DATA
:WRITE HEADER AND DATA
:ILLEGAL FUNCTION (64)
:ILLEGAL FUNCTION (66)
:READ DATA
:READ HEADER AND DATA
:ILLEGAL FUNCTION (74)
:ILLEGAL FUNCTION (76)

ATNTBL:

.BYTE 1.
.BYTE 2.
.BYTE 4.
.BYTE 8.
.BYTE 16.
.BYTE 32.
.BYTE 64.
.BYTE 128.

RGDTPT:

.WORD 0.
.WORD 1.
.WORD 3.
.WORD 7.
.WORD 15.
.WORD 31.
.WORD 63.
.WORD 127.
.WORD 255.
.WORD 511.

15159	066276	001777	.WORD	1023.
15160	066300	003777	.WORD	2047.
15161	066302	007777	.WORD	4095.
15162	066304	017777	.WORD	8191.
15163	066306	037777	.WORD	16383.
15164	066310	077777	.WORD	32767.
15165	066312	177777	ONES: .WORD	65535.
15166	066314	177777	.WORD	65535.
15167	066316	077777	.WORD	32767.
15168	066320	037777	.WORD	16383.
15169	066322	017777	.WORD	8191.
15170	066324	007777	.WORD	4095.
15171	066326	003777	.WORD	2047.
15172	066330	001777	.WORD	1023.
15173	066332	000777	.WORD	511.
15174	066334	000377	.WORD	255.
15175	066336	000177	.WORD	127.
15176	066340	000077	.WORD	63.
15177	066342	000037	.WORD	31.
15178	066344	000017	.WORD	15.
15179	066346	000007	.WORD	7.
15180	066350	000003	.WORD	3.
15181	066352	000001	.WORD	1.
15182	066354	000000	ZEROS: .WORD	0.
15183	066356	000000	.WORD	0.
15184	066360	000001	.WORD	1.
15185	066362	000002	.WORD	2.
15186	066364	000004	.WORD	4.
15187	066366	000010	.WORD	8.
15188	066370	000020	.WORD	16.
15189	066372	000040	.WORD	32.
15190	066374	000100	.WORD	64.
15191	066376	000200	.WORD	128.
15192	066400	000400	.WORD	256.
15193	066402	001000	.WORD	512.
15194	066404	002000	.WORD	1024.
15195	066406	004000	.WORD	2048.
15196	066410	010000	.WORD	4096.
15197	066412	020000	.WORD	8192.
15198	066414	040000	.WORD	16384.
15199	066416	100000	.WORD	32768.
15200	066420	100000	.WORD	32768.
15201	066422	040000	.WORD	16384.
15202	066424	020000	.WORD	8192.
15203	066426	010000	.WORD	4096.
15204	066430	004000	.WORD	2048.
15205	066432	002000	.WORD	1024.
15206	066434	001000	.WORD	512.
15207	066436	000400	.WORD	256.
15208	066440	000200	.WORD	128.
15209	066442	000100	.WORD	64.
15210	066444	000040	.WORD	32.
15211	066446	000020	.WORD	16.
15212	066450	000010	.WORD	8.
15213	066452	000004	.WORD	4.
15214	066454	000002	.WORD	2.

15215	066456	000001	.WORD	1.
15216	066460	000000	.WORD	0.
15217	066462	177777	.WORD	65535.
15218	066464	177776	.WORD	65534.
15219	066466	177774	.WORD	65532.
15220	066470	177770	.WORD	65528.
15221	066472	177760	.WORD	65520.
15222	066474	177740	.WORD	65504.
15223	066476	177700	.WORD	65472.
15224	066500	177600	.WORD	65408.
15225	066502	177400	.WORD	65280.
15226	066504	177000	.WORD	65024.
15227	066506	176000	.WORD	64512.
15228	066510	174000	.WORD	63488.
15229	066512	170000	.WORD	61440.
15230	066514	160000	.WORD	57344.
15231	066516	140000	.WORD	49152.
15232	066520	100000	.WORD	32768.
15233	066522	000000	.WORD	0.
15234	066524	000000	.WORD	0.
15235	066526	100000	.WORD	32768.
15236	066530	140000	.WORD	49152.
15237	066532	160000	.WORD	57344.
15238	066534	170000	.WORD	61440.
15239	066536	174000	.WORD	63488.
15240	066540	176000	.WORD	64512.
15241	066542	177000	.WORD	65024.
15242	066544	177400	.WORD	65280.
15243	066546	177600	.WORD	65408.
15244	066550	177700	.WORD	65472.
15245	066552	177740	.WORD	65504.
15246	066554	177760	.WORD	65520.
15247	066556	177770	.WORD	65528.
15248	066560	177774	.WORD	65532.
15249	066562	177776	.WORD	65534.
15250	066564	177777	.WORD	65535.
15251	066566	125252	.WORD	43690.
15252	066570	152525	.WORD	43690. /2
15253	066572	125252	.WORD	43690.
15254	066574	177777	.WORD	65535.
15255	066576	177776	.WORD	65534.
15256	066600	177775	.WORD	65533.
15257	066602	177773	.WORD	65531.
15258	066604	177767	.WORD	65527.
15259	066606	177757	.WORD	65519.
15260	066610	177737	.WORD	65503.
15261	066612	177677	.WORD	65471.
15262	066614	177577	.WORD	65407.
15263	066616	177377	.WORD	65279.
15264	066620	176777	.WORD	65023.
15265	066622	175777	.WORD	64511.
15266	066624	173777	.WORD	63487.
15267	066626	167777	.WORD	61439.
15268	066630	157777	.WORD	57343.
15269	066632	137777	.WORD	49151.
15270	066634	077777	.WORD	32767.

15271	066636	077777	.WORD	32767.
15272	066640	137777	.WORD	49151.
15273	066642	157777	.WORD	57343.
15274	066644	167777	.WORD	61439.
15275	066646	173777	.WORD	63487.
15276	066650	175777	.WORD	64511.
15277	066652	176777	.WORD	65023.
15278	066654	177377	.WORD	65279.
15279	066656	177577	.WORD	65407.
15280	066660	177677	.WORD	65471.
15281	066662	177737	.WORD	65503.
15282	066664	177757	.WORD	65519.
15283	066666	177767	.WORD	65527.
15284	066670	177773	.WORD	65531.
15285	066672	177775	.WORD	65533.
15286	066674	17777E	.WORD	65534.
15287	066676	177777	.WORD	65535.
15288	066700			

ENRGDT:

```

15289          .EVEN
15290
15291 066700 102051 074726 000000 EMT1: .WORD EMS300,EMS1,0
15292 066706 102067 102112 102137 EMT2: .WORD EMS301,EMS302,EMS303,EMS1,EMS304
15293 066714 074726 102150
15294 066720 105055 104276 104436 .WORD EMS511,EMS500,EMS501,EMS502,EMS503,0
15295 066726 104463 104532 000000
15296 066734 102067 102157 102112 EMT3: .WORD EMS301,EMS306,EMS302
15297 066742 105055 104602 104436 .WORD EMS511,EMS505,EMS501,EMS502,0
15298 066750 104463 000000
15299 066754 102051 102112 102173 EMT4: .WORD EMS300,EMS302,EMS307,EMS2
15300 066762 074777
15301 066764 105055 104463 104436 .WORD EMS511,EMS502,EMS501,EMS503,0
15302 066772 104532 000000
15303 066776 102067 102234 102251 EMT5: .WORD EMS301,EMS310,EMS311
15304 067004 105055 104463 104436 .WORD EMS511,EMS502,EMS501,EMS503,EMS504
15305 067012 104532 104556
15306 067016 102315 000000
15307 067022 102067 102157 102251 EMT6: .WORD EMS301,EMS306,EMS311
15308 067030 105055 104463 104436 .WORD EMS511,EMS502,EMS501,EMS503,EMS504,0
15309 067036 104532 104556 000000
15310 067044 102067 102353 102112 EMT7: .WORD EMS301,EMS313,EMS302
15311 067052 105055 104436 104463 .WORD EMS511,EMS501,EMS502,EMS504,EMS503,0
15312 067060 104556 104532 000000
15313 067066 102461 102502 102404 EMT10: .WORD EMS316,EMS317,EMS314
15314 067074 105055 104436 104463 .WORD EMS511,EMS501,EMS502,0
15315 067102 000000
15316 067104 102461 102502 102433 EMT11: .WORD EMS316,EMS317,EMS315
15317 067112 105055 104436 104463 .WORD EMS511,EMS501,EMS502,0
15318 067120 000000
15319 067122 102461 102522 102404 EMT12: .WORD EMS316,EMS320,EMS314
15320 067130 105055 104436 104463 .WORD EMS511,EMS501,EMS502,0
15321 067136 000000
15322 067140 102461 102522 102433 EMT13: .WORD EMS316,EMS320,EMS315
15323 067146 105055 104436 104463 .WORD EMS511,EMS501,EMS502,0
15324 067154 000000
15325 067156 102461 102542 102404 EMT14: .WORD EMS316,EMS321,EMS314
15326 067164 105055 104436 104463 .WORD EMS511,EMS501,EMS502,0
15327 067172 000000
15328 067174 102461 102542 102433 EMT15: .WORD EMS316,EMS321,EMS315
15329 067202 105055 104436 104463 .WORD EMS511,EMS501,EMS502,0
15330 067210 000000
15331 067212 102461 102562 102404 EMT16: .WORD EMS316,EMS322,EMS314
15332 067220 105055 104436 104463 .WORD EMS511,EMS501,EMS502,0
15333 067226 000000
15334 067230 102461 102562 102433 EMT17: .WORD EMS316,EMS322,EMS315
15335 067236 105055 104436 104463 .WORD EMS511,EMS501,EMS502,0
15336 067244 000000
15337 067246 102067 102621 101337 EMT20: .WORD EMS301,EMS324,EMS250
15338 067254 105055 104556 .WORD EMS511,EMS504
15339 067260 102315 102652 000000 .WORD EMS312,EMS326,0
15340 067266 102067 102602 101337 EMT21: .WORD EMS301,EMS323,EMS250
15341 067274 105055 104556 .WORD EMS511,EMS504
15342 067300 102315 102641 000000 .WORD EMS312,EMS325,0
15343 067306 102067 102353 101337 EMT22: .WORD EMS301,EMS313,EMS250
15344 067314 105055 104556 000000 .WORD EMS511,EMS504,0

```

JOB

CZRMJBO RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 307
CONSOLE MESSAGES

SEQ 0307

15345	067322	102067	102621	101375	EMT23:	.WORD	EMS301,EMS324,EMS251
15346	067330	105055	104436			.WORD	EMS511,EMS501
15347	067334	102315	102652	000000		.WORD	EMS312,EMS326,0
15348	067342	102067	102602	101375	EMT24:	.WORD	EMS301,EMS323,EMS251
15349	067350	105055	104436			.WORD	EMS511,EMS501
15350	067354	102315	102641	000000		.WORD	EMS312,EMS325,0
15351	067362	102067	102353	101375	EMT25:	.WORD	EMS301,EMS313,EMS251
15352	067370	105055	104436	000000		.WORD	EMS511,EMS501,0
15353	067376	102067	102157	101441	EMT26:	.WORD	EMS301,EMS306,EMS252,EMS253,EMS327,EMS254
15354	067404	101474	102662	101527			
15355	067412	105055	104532	104436		.WORD	EMS511,EMSS03,EMS501,EMS502
15356	067420	104463					
15357	067422	102670	102404	000000		.WORD	EMS330,EMS314,0
15358	067430	102051	101441		EMT27:	.WORD	EMS300,EMS252
15359	067434	105055	104436	000000		.WORD	EMS511,EMS501,0
15360	067442	102051	101441		EMT30:	.WORD	EMS300,EMS252
15361	067446	105055	104436	104556		.WORD	EMS511,EMS501,EMS504,0
15362	067454	000000					
15363	067456	102051	101441		EMT31:	.WORD	EMS300,EMS252
15364	067462	105055	104436	104532		.WORD	EMS511,EMS501,EMS503,0
15365	067470	000000					
15366	067472	102067	102621	101441	EMT32:	.WORD	EMS301,EMS324,EMS252
15367	067500	105055	104436	000000		.WORD	EMS511,EMS501,0
15368	067506	102067	102621	101441	EMT33:	.WORD	EMS301,EMS324,EMS252
15369	067514	105055	104436	104556		.WORD	EMS511,EMS501,EMS504,0
15370	067522	000000					
15371	067524	102067	102621	101441	EMT34:	.WORD	EMS301,EMS324,EMS252
15372	067532	105055	104436	104532		.WORD	EMS511,EMS501,EMS503,0
15373	067540	000000					
15374	067542	102067	102602	101441	EMT35:	.WORD	EMS301,EMS323,EMS252
15375	067550	105055	104436	000000		.WORD	EMS511,EMS501,0
15376	067556	102067	102353	101441	EMT36:	.WORD	EMS301,EMS313,EMS252
15377	067554	105055	104436	000000		.WORD	EMS511,EMS501,0
15378	067572	102067	102621	101570	EMT37:	.WORD	EMS301,EMS324,EMS255
15379	067600	102315	102652	000000		.WORD	EMS312,EMS326,0
15380	067606	102067	102602	101570	EMT40:	.WORD	EMS301,EMS323,EMS255
15381	067614	105055	104556			.WORD	EMS511,EMS504
15382	067620	102315	102641	000000		.WORD	EMS312,EMS325,0
15383	067626	102067	102353	101570	EMT41:	.WORD	EMS301,EMS313,EMS255
15384	067634	105055	104556	000000		.WORD	EMS511,EMS504,0
15385	067642	102067	102602	101337	EMT42:	.WORD	EMS301,EMS323,EMS250,EMS327,EMS255
15386	067650	102662	101570				
15387	067654	105055	104556	104436		.WORD	EMS511,EMS504,EMS501,EMS503,0
15388	067662	104532	000000				
15389	067666	102051	075046		EMT43:	.WORD	EMS300,EMS3
15390	067672	105055	104436	000000		.WORD	EMS511,EMS501,0
15391	067700	102707	075113	101474	EMT44:	.WORD	EMS331,EMS4,EMS253
15392	067706	105055	104436	000000		.WORD	EMS511,EMS501,0
15393							
15394	067714	102051	101474		EMT45:	.WORD	EMS300,EMS253
15395	067720	105055	104436	104532		.WORD	EMS511,EMS501,EMS503,0
15396	067726	000000					
15397							
15398	067730	102051	101474		EMT46:	.WORD	EMS300,EMS253
15399	067734	105055	104436	104556		.WORD	EMS511,EMS501,EMS504,0
15400	067742	000000					

K08

CZRMJBO RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 308
CONSOLE MESSAGES

SEQ 0308

15401									
15402	067744	102067	102621	101474	EMT47:	.WORD	EMS301,EMS324,EMS253		
15403	067752	105055	104436	104532		.WORD	EMS511,EMS501,EMS503		
15404	067760	102315	102652	000000		.WORD	EMS312,EMS326,0		
15405									
15406	067766	102067	102621	101474	EMT50:	.WORD	EMS301,EMS324,EMS253		
15407	067774	105055	104436	104556		.WORD	EMS511,EMS501,EMS504		
15408	070002	102315	102652	000000		.WORD	EMS312,EMS326,0		
15409									
15410	070010	102067	102602	101474	EMT51:	.WORD	EMS301,EMS323,EMS253		
15411	070016	105055	104436			.WORD	EMS511,EMS501		
15412	070022	102315	102641	000000		.WORD	EMS312,EMS325,0		
15413									
15414	070030	102067	102353	101474	EMT52:	.WORD	EMS301,EMS313,EMS253		
15415	070036	105055	104436	000000		.WORD	EMS511,EMS501,0		
15416									
15417	070044	102707	075113	101632	EMT53:	.WORD	EMS331,EMS4,EMS256		
15418	070052	105055	104436	000000		.WORD	EMS511,EMS501,0		
15419									
15420	070060	102067	102621	101632	EMT54:	.WORD	EMS301,EMS324,EMS256		
15421	070066	105055	104436			.WORD	EMS511,EMS501		
15422	070072	102315	102652	000000		.WORD	EMS312,EMS326,0		
15423									
15424	070100	102067	102602	101632	EMT55:	.WORD	EMS301,EMS323,EMS256		
15425	070106	105055	104436			.WORD	EMS511,EMS501		
15426	070112	102315	102641	000000		.WORD	EMS312,EMS325,0		
15427									
15428	070120	102067	102353	101632	EMT56:	.WORD	EMS301,EMS313,EMS256		
15429	070126	105055	104436	000000		.WORD	EMS511,EMS501,0		
15430									
15431	070134	101662	102737		EMT57:	.WORD	EMS257,EMS332		
15432	070140	105055	104655	104436		.WORD	EMS511,EMS506,EMS501,0		
15433	070146	000000							
15434									
15435	070150	075144	102755		EMT60:	.WORD	EMS5,EMS333		
15436	070154	105055	104712	104436		.WORD	EMS511,EMS507,EMS501,0		
15437	070162	000000							
15438									
15439	070164	102067	102621	101716	EMT61:	.WORD	EMS301,EMS324,EMS260		
15440	070172	105055	104436			.WORD	EMS511,EMS501		
15441	070176	102315	102652	000000		.WORD	EMS312,EMS326,0		
15442									
15443	070204	102067	102602	101716	EMT62:	.WORD	EMS301,EMS323,EMS260		
15444	070212	105055	104436			.WORD	EMS511,EMS501		
15445	070216	102315	102641	000000		.WORD	EMS312,EMS325,0		
15446									
15447	070224	102067	102353	101716	EMT63:	.WORD	EMS301,EMS313,EMS260		
15448	070232	105055	104436	000000		.WORD	EMS511,EMS501,0		
15449									
15450	070240	102051	075465		EMT64:	.WORD	EMS300,EMS12		
15451	070244	105055	104436	000000		.WORD	EMS511,EMS501,0		
15452									
15453	070252	075465	103001	103077	EMT65:	.WORD	EMS12,EMS335,EMS342		
15454	070260	105055	104436	000000		.WORD	EMS511,EMS501,0		
15455									
15456	070266	075465	103024	103077	EMT66:	.WORD	EMS12,EMS336,EMS342		

15457	070274	105055	104436	000000		.WORD	EMSS11,EMSS01,0
15458							
15459	070302	102051	075214		EMT67:	.WORD	EMS300,EMS6
15460	070306	105055	104436			.WORD	EMSS11,EMSS01
15461	070312	075260	102755	000000		.WORD	EMS7,EMS333,0
15462							
15463	070320	102051	075214	102662	EMT70:	.WORD	EMS300,EMS6,EMS327,EMS7
15464	070326	075260					
15465	070330	105055	104436	104556		.WORD	EMSS11,EMSS01,EMSS04,0
15466	070336	000000					
15467							
15468	070340	075214	103001	103057	EMT71:	.WORD	EMS6,EMS335,EMS340,EMS10,EMS333,EMS342
15469	070346	075331	102755	103077			
15470	070354	105055	104436	104463		.WORD	EMSS11,EMSS01,EMSS02,0
15471	070362	000000					
15472							
15473	070364	075214	103024	103057	EMT72:	.WORD	EMS6,EMS336,EMS340,EMS10,EMS334,EMS342
15474	070372	075331	102771	103077			
15475	070400	105055	104436	104463		.WORD	EMSS11,EMSS01,EMSS02,0
15476	070406	000000					
15477							
15478	070410	102067	101716	102137	EMT73:	.WORD	EMS301,EMS260,EMS303,EMS11
15479	070416	075374					
15480	070420	105055	104436	104463		.WORD	EMSS11,EMSS01,EMSS02,0
15481	070426	000000					
15482							
15483	070430	103131	103145	103077	EMT74:	.WORD	EMS343,EMS344,EMS342,0
15484	070436	000000					
15485							
15486	070440	102051	075543		EMT75:	.WORD	EMS300,EMS13
15487	070444	105055	104532	000000		.WORD	EMSS11,EMSS03,0
15488							
15489	070452	103222	075543	103174	EMT76:	.WORD	EMS346,EMS13,EMS345
15490	070460	105055	104532	000000		.WORD	EMSS11,EMSS03,0
15491							
15492	070466	103043	075543	103174	EMT77:	.WORD	EMS337,EMS13,EMS345
15493	070474	105055	104532	000000		.WORD	EMSS11,EMSS03,0
15494							
15495	070502	102067	102353	101527	EMT100:	.WORD	EMS301,EMS313,EMS254,EMS347,EMS13
15496	070510	103240	075543				
15497	070514	105055	104532	000000		.WORD	EMSS11,EMSS03,0
15498							
15499	070522	103222	075612	103070	EMT101:	.WORD	EMS346,EMS14,EMS341,EMS15
15500	070530	075657					
15501	070532	105055	104532	104436		.WORD	EMSS11,EMSS03,EMSS01,0
15502	070540	000000					
15503							
15504	070542	103043	075612	103070	EMT102:	.WORD	EMS337,EMS14,EMS341,EMS15
15505	070550	075657					
15506	070552	105055	104532	104436		.WORD	EMSS11,EMSS03,EMSS01,0
15507	070560	000000					
15508							
15509	070562	102067	102353	101527	EMT103:	.WORD	EMS301,EMS313,EMS254,EMS347,EMS15
15510	070570	103240	075657				
15511	070574	105055	104532			.WORD	EMSS11,EMSS03
15512	070600	075612	102737	000000		.WORD	EMS14,EMS332,0

MOB

CZRMJBO RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 310
CONSOLE MESSAGES

SEQ 0310

15513							
15514	070606	103222	076016	103070	EMT104:	.WORD	EMS346,EMS17,EMS341,EMS16
15515	070614	075735					
15516	070616	105055	104532	104436		.WORD	EMS511,EMS503,EMS501,0
15517	070624	000000					
15518							
15519	070626	103043	076016	103070	EMT105:	.WORD	EMS337,EMS17,EMS341,EMS16
15520	070634	075735					
15521	070636	105055	104532	104436		.WORD	EMS511,EMS503,EMS501,0
15522	070644	000000					
15523							
15524	070646	102067	102353	101527	EMT106:	.WORD	EMS301,EMS313,EMS254,EMS347,EMS16
15525	070654	103240	075735				
15526	070660	105055	104532			.WORD	EMS511,EMS503
15527	070664	076016	102737	000000		.WORD	EMS17,EMS332,0
15528							
15529	070672	103222	076057	103070	EMT107:	.WORD	EMS346,EMS20,EMS341,EMS21
15530	070700	076123					
15531	070702	105055	104532	104436		.WORD	EMS511,EMS503,EMS501,0
15532	070710	000000					
15533							
15534	070712	076057	103266	076123	EMT110:	.WORD	EMS20,EMS351,EMS21,EMS350,EMS22,EMS315
15535	070720	103261	076202	102433			
15536	070726	105055	104436	000000		.WORD	EMS511,EMS501,0
15537							
15538	070734	076057	102771	103261	EMT111:	.WORD	EMS20,EMS334,EMS350,EMS22,EMS333
15539	070742	076202	102755				
15540	070746	105055	104436	000000		.WORD	EMS511,EMS501,0
15541							
15542	070754	103043	076057	103070	EMT112:	.WORD	EMS337,EMS20,EMS341,EMS21
15543	070762	076123					
15544	070764	105055	104532	104436		.WORD	EMS511,EMS503,EMS501,0
15545	070772	000000					
15546							
15547	070774	103043	076057	103070	EMT113:	.WORD	EMS337,EMS20,EMS341,EMS21,EMS350,EMS22,EMS334
15548	071002	076123	103261	076202			
15549	071010	102771					
15550	071012	105055	104436	000000		.WORD	EMS511,EMS501,0
15551							
15552	071020	076057	103304	076123	EMT114:	.WORD	EMS20,EMS352,EMS21,EMS350,EMS22,EMS333
15553	071026	103261	076202	102755			
15554	071034	105055	104436	000000		.WORD	EMS511,EMS501,0
15555							
15556	071042	102067	102353	101527	EMT115:	.WORD	EMS301,EMS313,EMS254,EMS347,EMS21
15557	071050	103240	076123				
15558	071054	105055	104532			.WORD	EMS511,EMS503
15559	071060	076057	102737	000000		.WORD	EMS20,EMS332,0
15560							
15561	071066	103222	076247	103070	EMT116:	.WORD	EMS346,EMS23,EMS341,EMS24
15562	071074	076325					
15563	071076	105055	104532	104436		.WORD	EMS511,EMS503,EMS501,0
15564	071104	000000					
15565							
15566	071106	103043	076247	103070	EMT117:	.WORD	EMS337,EMS23,EMS341,EMS24
15567	071114	076325					
15568	071116	105055	104532	104436		.WORD	EMS511,EMS503,EMS501,0

NO8

CZRMJ80 RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 311
CONSOLE MESSAGES

SEQ 0311

15569	071124	000000					
15570							
15571	071126	102067	102353	101527	EMT120:	.WORD	EMS301,EMS313,EMS254,EMS347,EMS24
15572	071134	103240	076325				
15573	071140	105055	104532			.WORD	EMS511,EMS503
15574	071144	076247	102737	000000		.WORD	EMS23,EMS332,0
15575							
15576	071152	103222	076404	103070	EMT121:	.WORD	EMS346,EMS25,EMS341,EMS26
15577	071160	076462					
15578	071162	105055	104532	104436		.WORD	EMS511,EMS503,EMS501,0
15579	071170	000000					
15580							
15581	071172	103043	076404	103070	EMT122:	.WORD	EMS337,EMS25,EMS341,EMS26
15582	071200	076462					
15583	071202	105055	104532	104436		.WORD	EMS511,EMS503,EMS501,0
15584	071210	000000					
15585							
15586	071212	102067	102353	101527	EMT123:	.WORD	EMS301,EMS313,EMS254,EMS347,EMS26
15587	071220	103240	076462				
15588	071224	105055	104532			.WORD	EMS511,EMS503
15589	071230	076404	102737	000000		.WORD	EMS25,EMS332,0
15590							
15591	071236	102051	076541	102173	EMT124:	.WORD	EMS300,EMS27,EMS307,EMS2
15592	071244	074777					
15593	071246	105055	104532	000000		.WORD	EMS511,EMS503,0
15594							
15595	071254	102707	076541	103320	EMT125:	.WORD	EMS331,EMS27,EMS353
15596	071262	105055	104532			.WORD	EMS511,EMS503
15597	071266	076605	102433	000000		.WORD	EMS30,EMS315,0
15598							
15599	071274	103043	076541	103070	EMT126:	.WORD	EMS337,EMS27,EMS341,EMS30
15600	071302	076605					
15601	071304	105055	104532	000000		.WORD	EMS511,EMS503,0
15602							
15603	071312	102067	102353	101527	EMT127:	.WORD	EMS301,EMS313,EMS254,EMS347,EMS30
15604	071320	103240	076605				
15605	071324	105055	104532			.WORD	EMS511,EMS503
15606	071330	076541	102737	000000		.WORD	EMS27,EMS332,0
15607							
15608	071336	076665	103344	101337	EMT130:	.WORD	EMS31,EMS354,EMS250
15609	071344	105055	104556	104532		.WORD	EMS511,EMS504,EMS503,0
15610	071352	000000					
15611							
15612	071354	076736	103344	101337	EMT131:	.WORD	EMS32,EMS354,EMS250
15613	071362	105055	104556	104532		.WORD	EMS511,EMS504,EMS503,0
15614	071370	000000					
15615							
15616	071372	103377	077016	101337	EMT132:	.WORD	EMS355,EMS33,EMS250,EMS341,EMS30
15617	071400	103070	076605				
15618	071404	105055	104556	000000		.WORD	EMS511,EMS504,0
15619							
15620	071412	103377	077055	101337	EMT133:	.WORD	EMS355,EMS34,EMS250,EMS341,EMS30
15621	071420	103070	076605				
15622	071424	105055	104556	000000		.WORD	EMS511,EMS504,0
15623							
15624	071432	102707	075113	101570	EMT134:	.WORD	EMS331,EMS4,EMS255

15625	071440	105055	104556	000000		.WORD	EMS511,EMS504,0
15626							
15627	071446	077113	103441	103463	EMT135:	.WORD	EMS35,EMS357,EMS360,EMS15
15628	071454	075657					
15629	071456	105055	104436	000000		.WORD	EMS511,EMS501,0
15630							
15631	071464	101747	103537		EMT136:	.WORD	EMS261,EMS362
15632	071470	105055	104532	000000		.WORD	EMS511,EMS503,0
15633							
15634	071476	102051	077155	102173	EMT137:	.WORD	EMS300,EMS36,EMS307,EMS2
15635	071504	074777					
15636	071506	105055	104436	000000		.WORD	EMS511,EMS501,0
15637							
15638	071514	103377	077205	101570	EMT140:	.WORD	EMS355,EMS37,EMS255,EMS341,EMS30
15639	071522	103070	076605				
15640	071526	105055	104556	000000		.WORD	EMS511,EMS504,0
15641							
15642	071534	103222	077245	103174	EMT141:	.WORD	EMS346,EMS40,EMS345
15643	071542	105055	104556	000000		.WORD	EMS511,EMS504,0
15644							
15645	071550	103043	077245	103070	EMT142:	.WORD	EMS337,EMS40,EMS341,EMS30
15646	071556	076605					
15647	071560	105055	104556	000000		.WORD	EMS511,EMS504,0
15648							
15649	071566	103560	102234	077323	EMT143:	.WORD	EMS363,EMS310,EMS41
15650	071574	105055	104436	000000		.WORD	EMS511,EMS501,0
15651							
15652	071602	103043	077323	103070	EMT144:	.WORD	EMS337,EMS41,EMS341,EMS252,EMS327,EMS253
15653	071610	101441	102662	101474			
15654	071616	105055	104436	000000		.WORD	EMS511,EMS501,0
15655							
15656	071624	077323	103344	103575	EMT145:	.WORD	EMS41,EMS354,EMS364,EMS252,EMS365,EMS253
15657	071632	101441	103616	101474			
15658	071640	105055	104436	000000		.WORD	EMS511,EMS501,0
15659							
15660	071646	102067	102157	077155	EMT146:	.WORD	EMS301,EMS306,EMS36
15661	071654	105055	104436	104532		.WORD	EMS511,EMS501,EMS503,0
15662	071662	000000					
15663							
15664	071664	103623	077371		EMT147:	.WORD	EMS366,EMS42
15665	071670	105055	104532	000000		.WORD	EMS511,EMS503,0
15666							
15667	071676	103646	103320	103616	EMT150:	.WORD	EMS367,EMS353,EMS365,EMS42,EMS354,EMS3
15668	071704	077371	103344	075046			
15669	071712	105055	104532	000000		.WORD	EMS511,EMS503,0
15670							
15671	071720	103043	077155		EMT151:	.WORD	EMS337,EMS36
15672	071724	105055	104436	000000		.WORD	EMS511,EMS501,0
15673							
15674	071732	077453	103344	077155	EMT152:	.WORD	EMS43,EMS354,EMS36
15675	071740	105055	104436	000000		.WORD	EMS511,EMS501,0
15676							
15677	071746	103646	103320	103616	EMT153:	.WORD	EMS367,EMS353,EMS365,EMS36,EMS370
15678	071754	077155	103716				
15679	071760	105055	104532	000000		.WORD	EMS511,EMS503,0
15680							

15681	071766	103646	103320	103616	EMT154: .WORD	EMS367, EMS353, EMS365, EMS36, EMS371
15682	071774	077155	103733			
15683	072000	105055	104532	000000	.WORD	EMS511, EMS503, 0
15684						
15685	072006	102051	077524	102173	EMT155: .WORD	EMS300, EMS44, EMS307, EMS2
15686	072014	074777				
15687	072016	105055	104532	000000	.WORD	EMS511, EMS503, 0
15688						
15689	072024	103646	103320	103616	EMT156: .WORD	EMS367, EMS353, EMS365, EMS44, EMS354, EMS3
15690	072032	077524	103344	075046		
15691	072040	105055	104532	000000	.WORD	EMS511, EMS503, 0
15692						
15693	072046	102051	077565	102173	EMT157: .WORD	EMS300, EMS45, EMS307, EMS2
15694	072054	074777				
15695	072056	105055	104532	000000	.WORD	EMS511, EMS503, 0
15696						
15697	072064	103646	103320	103616	EMT160: .WORD	EMS367, EMS353, EMS365, EMS45, EMS354, EMS3
15698	072072	077565	103344	075046		
15699	072100	105055	104532	104436	.WORD	EMS511, EMS503, EMS501
15700	072106	077113	102755	000000	.WORD	EMS35, EMS333, 0
15701						
15702	072114	102051	077642	102173	EMT161: .WORD	EMS300, EMS46, EMS307, EMS2
15703	072122	074777				
15704	072124	105055	104532	000000	.WORD	EMS511, EMS503, 0
15705						
15706	072132	103043	077642	103320	EMT162: .WORD	EMS337, EMS46, EMS353
15707	072140	105055	104532	104436	.WORD	EMS511, EMS503, EMS501, 0
15708	072146	000000				
15709						
15710	072150	077113	103001	103043	EMT163: .WORD	EMS35, EMS335, EMS337, EMS41, EMS334, EMS372
15711	072156	077323	102771	103762		
15712	072164	105055	104436	000000	.WORD	EMS511, EMS501, 0
15713						
15714	072172	077724	103001	103043	EMT164: .WORD	EMS47, EMS335, EMS337, EMS41, EMS335, EMS372
15715	072200	077323	103001	103762		
15716	072206	105055	104436	000000	.WORD	EMS511, EMS501, 0
15717						
15718	072214	103043	077113	102662	EMT165: .WORD	EMS337, EMS35, EMS327, EMS47
15719	072222	077724				
15720	072224	105055	104436		.WORD	EMS511, EMS501
15721	072230	077323	102755	103762	.WORD	EMS41, EMS333, EMS372, 0
15722	072236	000000				
15723						
15724	072240	102051	077724	102173	EMT166: .WORD	EMS300, EMS47, EMS307, EMS2
15725	072246	074777				
15726	072250	105055	104436	104532	.WORD	EMS511, EMS501, EMS503, 0
15727	072256	000000				
15728						
15729	072260	077764	103001	103057	EMT167: .WORD	EMS50, EMS335, EMS340, EMS36, EMS333
15730	072266	077155	102755			
15731	072272	105055	104436	104532	.WORD	EMS511, EMS501, EMS503, 0
15732	072300	000000				
15733						
15734	072302	103043	077113		EMT170: .WORD	EMS337, EMS35
15735	072306	105055	104436	000000	.WORD	EMS511, EMS501, 0
15736						

15737	072314	077764	077055	075046	EMT171:	.WORD	EM550,EMS34,EMS3
15738	072322	105055	104436	000000		.WORD	EM5511,EM5501,0
15739							
15740	072330	102067	102157	100033	EMT172:	.WORD	EMS301,EMS306,EM551
15741	072336	105055	104436	104556		.WORD	EM5511,EM5501,EM5504,0
15742	072344	000000					
15743							
15744	072346	103646	103320	103616	EMT173:	.WORD	EMS367,EMS353,EMS365,EMS47,EMS354,EMS3
15745	072354	077724	103344	075046			
15746	072362	105055	104436	000000		.WORD	EM5511,EM5501,0
15747							
15748	072370	102051	101337	102662	EMT174:	.WORD	EMS300,EMS250,EMS327,EMS255,EMS327,EMS256
15749	072376	101570	102662	101632			
15750	072404	103070	105154			.WORD	EMS341,EMS600
15751	072410	105055	104436	000000		.WORD	EM5511,EM5501,0
15752							
15753	072416	102051	101632	103070	EMT175:	.WORD	EMS300,EMS256,EMS341,EMS600
15754	072424	105154					
15755	072426	105055	104436	000000		.WORD	EM5511,EM5501,0
15756							
15757	072434	102051	101337	103070	EMT176:	.WORD	EMS300,EMS250,EMS341,EMS600
15758	072442	105154					
15759	072444	105055	104556	000000		.WORD	EM5511,EM5504,0
15760							
15761	072452	102051	101570	103070	EMT177:	.WORD	EMS300,EMS255,EMS341,EMS600
15762	072460	105154					
15763	072462	105055	104556	000000		.WORD	EM5511,EM5504,0
15764							
15765	072470	103043	100102	103070	EMT200:	.WORD	EMS337,EM552,EMS341,EMS601
15766	072476	105204					
15767	072500	105055	104436	000000		.WORD	EM5511,EM5501,0
15768							
15769	072506	103222	100102	103070	EMT201:	.WORD	EMS346,EM552,EMS341,EMS602
15770	072514	105224					
15771	072516	105055	104436	000000		.WORD	EM5511,EM5501,0
15772							
15773	072524	103222	100033	103070	EMT202:	.WORD	EMS346,EM551,EMS341,EMS602
15774	072532	105224					
15775	072534	105055	104436	000000		.WORD	EM5511,EM5501,0
15776							
15777	072542	103222	100102	103174	EMT203:	.WORD	EMS346,EM552,EMS345,EMS373,EMS255
15778	072550	104014	101570				
15779	072554	105055	104556	104436		.WORD	EM5511,EM5504,EM5501,0
15780	072562	000000					
15781							
15782	072564	103222	100102	103070	EMT204:	.WORD	EMS346,EM552,EMS341,EMS27
15783	072572	076541					
15784	072574	105055	104556	104436		.WORD	EM5511,EM5504,EM5501,0
15785	072602	000000					
15786							
15787	072604	100143	103344	075046	EMT205:	.WORD	EM553,EMS354,EMS3
15788	072612	105055	104532	104463		.WORD	EM5511,EM5503,EM5502,EM5510,0
15789	072620	105002	000000				
15790							
15791	072624	103043	100204	104020	EMT206:	.WORD	EMS337,EM554,EMS374,EMS250,EMS327,EMS255
15792	072632	101337	102662	101570			

15793	072640	102662	075046			.WORD	EMS327,EMS3
15794	072644	105055	104556	104436		.WORD	EMS511,EMS504,EMS501,0
15795	072652	000000					
15796							
15797	072654	100204	103344	075046	EMT207:	.WORD	EMS54,EMS354,EMS3
15798	072662	105055	104436	000000		.WORD	EMS511,EMS501,0
15799							
15800	072670	100204	103344	103575	EMT210:	.WORD	EMS54,EMS354,EMS364,EMS250
15801	072676	101337					
15802	072700	105055	104556	000000		.WORD	EMS511,EMS504,0
15803							
15804	072706	100204	103344	103575	EMT211:	.WORD	EMS54,EMS354,EMS364,EMS255
15805	072714	101570					
15806	072716	105055	104556	000000		.WORD	EMS511,EMS504,0
15807							
15808	07271	103043	100261		EMT212:	.WORD	EMS337,EMS55
15809	072730	105055	104556	000000		.WORD	EMS511,EMS504,0
15810							
15811	072736	100337	102771	103174	EMT213:	.WORD	EMS56,EMS334,EMS345,EMS373,EMS262,EMS327,EMS251
15812	072744	104014	102006	102662			
15813	072752	101375					
15814	072754	105055	104436			.WORD	EMS511,EMS501
15815	072760	100337	103001	000000		.WORD	EMS56,EMS335,0
15816							
15817	072766	103043	100337		EMT214:	.WORD	EMS337,EMS56
15818	072772	105055	104436	000000		.WORD	EMS511,EMS501,0
15819							
15820	073000	100432	102755	103261	EMT215:	.WORD	EMS57,EMS333,EMS350,EMS60,EMS334
15821	073006	100516	102771				
15822	073017	105055	104436	000000		.WORD	EMS511,EMS501,0
15823							
15824	073020	103560	102157	075144	EMT216:	.WORD	EMS363,EMS306,EMS5
15825	073026	105055	104436	000000		.WORD	EMS511,EMS501,0
15826							
15827	073034	103560	102157	100570	EMT217:	.WORD	EMS363,EMS306,EMS61
15828	073042	105055	104436	000000		.WORD	EMS511,EMS501,0
15829							
15830	073050	100643	102755	104051	EMT220:	.WORD	EMS62,EMS333,EMS375,EMS251
15831	073056	101375					
15832	073060	105055	104436	000000		.WORD	EMS511,EMS501,0
15833							
15834	073066	100643	102755	104065	EMT221:	.WORD	EMS62,EMS333,EMS376,EMS262
15835	073074	102006					
15836	073076	105055	104436	000000		.WORD	EMS511,EMS501,0
15837							
15838	073104	100643	102755	104065	EMT222:	.WORD	EMS62,EMS333,EMS376,EMS250
15839	073112	101337					
15840	073114	105055	104436	000000		.WORD	EMS511,EMS501,0
15841							
15842	073122	103222	100643	103070	EMT223:	.WORD	EMS346,EMS62,EMS341,EMS603
15843	073130	105265					
15844	073132	105055	104436	000000		.WORD	EMS511,EMS501,0
15845							
15846	073140	103222	100724	104065	EMT224:	.WORD	EMS346,EMS63,EMS376,EMS262
15847	073146	102006					
15848	073150	105055	104436	104532		.WORD	EMS511,EMS501,EMS503,0

15849	073156	000000					
15850							
15851	073160	100724	102755	103261	EMT225: .WORD	EMS63, EMS333, EMS350, EMS363, EMS310, EMS262	
15852	073166	103560	102234	102006			
15853	073174	105055	104436	000000	.WORD	EMS511, EMS501, 0	
15854							
15855	073202	100724	103441	077155	EMT226: .WORD	EMS63, EMS357, EMS36, EMS372	
15856	073210	103762					
15857	073212	105055	104436	000000	.WORD	EMS511, EMS501, 0	
15858							
15859	073220	100724	103421	103463	EMT227: .WORD	EMS63, EMS356, EMS360, EMS15	
15860	073226	075657					
15861	073230	105055	104436	000000	.WORD	EMS511, EMS501, 0	
15862							
15863	073236	100724	103421	103511	EMT230: .WORD	EMS63, EMS356, EMS361, EMS15	
15864	073244	075657					
15865	073246	105055	104436	000000	.WORD	EMS511, EMS501, 0	
15866							
15867	073254	100724	103421	077323	EMT231: .WORD	EMS63, EMS356, EMS41	
15868	073262	105055	104436	000000	.WORD	EMS511, EMS501, 0	
15869							
15870	073270	077323	104101	103077	EMT232: .WORD	EMS41, EMS377, EMS342, EMS365, EMS63, EMS332	
15871	073276	103616	100724	102737			
15872	073304	105055	104436	000000	.WORD	EMS511, EMS501, 0	
15873							
15874	073312	103646	103320	103616	EMT233: .WORD	EMS367, EMS353, EMS365, EMS63, EMS401	
15875	073320	100724	104127				
15876	073324	105055	104532	104436	.WORD	EMS511, EMS503, EMS501, 0	
15877	073332	000000					
15878							
15879	073334	075735	104101	103762	EMT234: .WORD	EMS16, EMS377, EMS372, EMS365, EMS64, EMS354, EMS3	
15880	073342	103616	100764	103344			
15881	073350	075046					
15882	073352	105055	104532	104436	.WORD	EMS511, EMS503, EMS501, 0	
15883	073360	000000					
15884							
15885	073362	102051	101034	102173	EMT235: .WORD	EMS300, EMS65, EMS307, EMS2	
15886	073370	074777					
15887	073372	105055	104463	104532	.WORD	EMS511, EMS502, EMS503, 0	
15888	073400	000000					
15889							
15890	073402	100337	104101	104065	EMT236: .WORD	EMS56, EMS377, EMS376, EMS252, EMS372, EMS350	
15891	073410	101441	103762	103261			
15892	073416	101034	104127		.WORD	EMS65, EMS401	
15893	073422	105055	104463	104532	.WORD	EMS511, EMS502, EMS503, EMS501, 0	
15894	073430	104436	000000				
15895							
15896	073434	102051	101075	102173	EMT237: .WORD	EMS300, EMS66, EMS307, EMS2	
15897	073442	074777					

15898	073444	105055	104436	104532		.WORD	EMSS11, EMS501, EMS503, 0
15899	073452	000000					
15900							
15901	073454	075657	104112	103762	EMT240:	.WORD	EMS15, EMS400, EMS372, EMS350, EMS66, EMS401
15902	073462	103261	101075	104127			
15903	073470	105055	104532	104436		.WORD	EMSS11, EMS503, EMS501, 0
15904	073476	000000					
15905							
15906	073500	101075	103024	103057	EMT241:	.WORD	EMS66, EMS336, EMS340, EMS15, EMS406, EMS405, EMS604
15907	073506	075657	104243	104233			
15908	073514	105306					
15909	073516	105055	104532	000000		.WORD	EMSS11, EMS503, 0
15910							
15911	073524	104153	105306	104144	EMT242:	.WORD	EMS403, EMS604, EMS402, EMS21, EMS377
15912	073532	076123	104101				
15913	073536	105055	104532			.WORD	EMSS11, EMS503
15914	073542	077155	103001	000000		.WORD	EMS36, EMS335, 0
15915							
15916	073550	101075	103024	103057	EMT243:	.WORD	EMS66, EMS336, EMS340, EMS26, EMS404, EMS405, EMS604
15917	073556	076462	104214	104233			
15918	073564	105306					
15919	073566	105055	104532	000000		.WORD	EMSS11, EMS503, 0
15920							
15921	073574	100724	104127	104233	EMT244:	.WORD	EMS63, EMS401, EMS405, EMS604
15922	073602	105306					
15923	073604	105055	104532	000000		.WORD	EMSS11, EMS503, 0
15924							
15925	073612	077155	103716	104233	EMT245:	.WORD	EMS36, EMS370, EMS405, EMS604
15926	073620	105306					
15927	073622	105055	104532	000000		.WORD	EMSS11, EMS503, 0
15928							
15929	073630	104153	105306	104144	EMT246:	.WORD	EMS403, EMS604, EMS402, EMS24, EMS377
15930	073636	076325	104101				
15931	073642	105055	104532			.WORD	EMSS11, EMS503
15932	073646	077155	103001	000000		.WORD	EMS36, EMS335, 0
15933							
15934	073654	101157	102737	104233	EMT247:	.WORD	EMS67, EMS332, EMS405, EMS604
15935	073662	105306					
15936	073664	105055	104532	000000		.WORD	EMSS11, EMS503, 0
15937							
15938	073672	101075	103024	103057	EMT250:	.WORD	EMS66, EMS336, EMS340, EMS15, EMS406, EMS405, EMS605
15939	073700	075657	104243	104233			
15940	073706	105333					
15941	073710	105055	104532	000000		.WORD	EMSS11, EMS503, 0
15942							
15943	073716	104153	105333	104144	EMT251:	.WORD	EMS403, EMS605, EMS402, EMS21, EMS377
15944	073724	076123	104101				
15945	073730	105055	104532			.WORD	EMSS11, EMS503
15946	073734	077155	103001	000000		.WORD	EMS36, EMS335, 0
15947							
15948	073742	101075	103024	103057	EMT252:	.WORD	EMS66, EMS336, EMS340, EMS26, EMS404, EMS405, EMS605
15949	073750	076462	104214	104233			
15950	073756	105333					
15951	073760	105055	104532	000000		.WORD	EMSS11, EMS503, 0
15952							
15953	073766	100724	104127	104233	EMT253:	.WORD	EMS63, EMS401, EMS405, EMS605

15954	073774	105333							
15955	073776	105055	104532	000000		.WORD	EMS511,EMS503,0		
15956									
15957	074004	077155	103716	104233	EMT254:	.WORD	EMS36,EMS370,EMS405,EMS605		
15958	074012	105333							
15959	074014	105055	104532	000000		.WORD	EMS511,EMS503,0		
15960									
15961	074022	104153	105333	104144	EMT255:	.WORD	EMS403,EMS605,EMS402,EMS24,EMS377		
15962	074030	076325	104101						
15963	074034	105055	104532			.WORD	EMS511,EMS503		
15964	074040	077155	103001	000000		.WORD	EMS36,EMS335,0		
15965									
15966	074046	101157	102737	104233	EMT256:	.WORD	EMS67,EMS332,EMS405,EMS605		
15967	074054	105333							
15968	074056	105055	104532	000000		.WORD	EMS511,EMS503,0		
15969									
15970	074064	101075	103024	103057	EMT257:	.WORD	EMS66,EMS336,EMS340,EMS15,EMS406,EMS405,EMS606		
15971	074072	075657	104243	104233					
15972	074100	105351							
15973	074102	105055	104532	000000		.WORD	EMS511,EMS503,0		
15974									
15975	074110	104153	105351	104144	EMT260:	.WORD	EMS403,EMS606,EMS402,EMS21,EMS377		
15976	074116	076123	104101						
15977	074122	105055	104532			.WORD	EMS511,EMS503		
15978	074126	077155	103001	000000		.WORD	EMS36,EMS335,0		
15979									
15980	074134	101075	103024	103057	EMT261:	.WORD	EMS66,EMS336,EMS340,EMS25,EMS404,EMS405,EMS606		
15981	074142	076462	104214	104233					
15982	074150	105351							
15983	074152	105055	104532	000000		.WORD	EMS511,EMS503,0		
15984									
15985	074160	100724	104127	104233	EMT262:	.WORD	EMS63,EMS401,EMS405,EMS606		
15986	074166	105351							
15987	074170	105055	104532	000000		.WORD	EMS511,EMS503,0		
15988									
15989	074176	077155	103716	104233	EMT263:	.WORD	EMS36,EMS370,EMS405,EMS606		
15990	074204	105351							
15991	074206	105055	104532	000000		.WORD	EMS511,EMS503,0		
15992									
15993	074214	104153	105351	104144	EMT264:	.WORD	EMS403,EMS606,EMS402,EMS24,EMS377		
15994	074222	076325	104101						
15995	074226	105055	104532			.WORD	EMS511,EMS503		
15996	074232	077155	103001	000000		.WORD	EMS36,EMS335,0		
15997									
15998	074240	101271	104127	104233	EMT265:	.WORD	EMS70,EMS401,EMS405,EMS606		
15999	074246	105351							
16000	074250	105055	104532	000000		.WORD	EMS511,EMS503,0		
16001									
16002	074256	101157	102737	104233	EMT266:	.WORD	EMS67,EMS332,EMS405,EMS606		
16003	074264	105351							
16004	074266	105055	104532	000000		.WORD	EMS511,EMS503,0		
16005									
16006	074274	101075	103421	104256	EMT267:	.WORD	EMS66,EMS356,EMS407		
16007	074302	105055	104532	000000		.WORD	EMS511,EMS503,0		
16008									
16009	074310	101075	103024	103057	EMT270:	.WORD	EMS66,EMS336,EMS340,EMS15,EMS406,EMS405,EMS607		

16010	074316	075657	104243	104233			
16011	074324	105371					
16012	074326	105055	104532	000000	.WORD	EMSS11,EMS503,0	
16013							
16014	074334	104153	105371	104144	EMT271:	.WORD	EMS403,EMS607,EMS402,EMS21,EMS377
16015	074342	076123	104101				
16016	074346	105055	104532		.WORD	EMSS11,EMS503	
16017	074352	076541	103024	000000	.WORD	EMS27,EMS336,0	
16018							
16019	074360	076541	103716	104233	EMT272:	.WORD	EMS27,EMS370,EMS405,EMS607
16020	074366	105371					
16021	074370	105055	104532	000000	.WORD	EMSS11,EMS503,0	
16022							
16023	074376	076541	103733	104233	EMT273:	.WORD	EMS27,EMS371,EMS405,EMS607
16024	074404	105371					
16025	074406	105055	104532	000000	.WORD	EMSS11,EMS503,0	
16026							
16027	074414	077155	104214	104233	EMT274:	.WORD	EMS36,EMS404,EMS405,EMS607
16028	074422	105371					
16029	074424	105055	104532	000000	.WORD	EMSS11,EMS503,0	
16030							
16031	074432	100143	104127	104233	EMT275:	.WORD	EMSS3,EMS401,EMS405,EMS607
16032	074440	105371					
16033	074442	105055	104532	104463	.WORD	EMSS11,EMS503,EMS502,0	
16034	074450	000000					
16035							
16036	074452	101157	102737	104233	EMT276:	.WORD	EMS67,EMS332,EMS405,EMS607
16037	074460	105371					
16038	074462	105055	104532	000000	.WORD	EMSS11,EMS503,0	
16039							
16040	074470	101075	103024	103057	EMT277:	.WORD	EMS66,EMS336,EMS340,EMS26,EMS404,EMS405,EMS607
16041	074476	076462	104214	104233			
16042	074504	105371					
16043	074506	105055	104532	000000	.WORD	EMSS11,EMS503,0	
16044							
16045	074514	104153	105371	104144	EMT300:	.WORD	EMS403,EMS607,EMS402,EMS24,EMS377
16046	074522	076325	104101				
16047	074526	105055	104532		.WORD	EMSS11,EMS503	
16048	074532	076541	103024	000000	.WORD	EMS27,EMS336,0	
16049							
16050	074540	101271	104127	104233	EMT301:	.WORD	EMS70,EMS401,EMS405,EMS607
16051	074546	105371					
16052	074550	105055	104532	000000	.WORD	EMSS11,EMS503,0	
16053							
16054	074556	105407	000000		EHT1:	.WORD	EH1,0
16055	074562	105466	000000		EHT2:	.WORD	EH2,0
16056	074566	105504	000000		EHT5:	.WORD	EH5,0
16057	074572	105543	000000		EHT7:	.WORD	EH7,0
16058	074576	105562	000000		EHT57:	.WORD	EH57,0
16059	074602	105661	000000		EHT65:	.WORD	EH65,0
16060	074606	105736	000000		EHT71:	.WORD	EH71,0
16061	074612	105475	000000		EHT74:	.WORD	EH74,0
16062	074616	106014	000000		EHT115:	.WORD	EH115,0
16063	074622	106112	000000		EHT130:	.WORD	EH130,0
16064	074626	106231	000000		EHT132:	.WORD	EH132,0
16065	074632	106330	000000		EHT145:	.WORD	EH145,0

16066	074676	106446	000000		EHT150:	.WORD	EH150,0
16067	074642	106545	000000		EHT213:	.WORD	EH213,0
16068	074646	106643	000000		EHT220:	.WORD	EH220,0
16069							
16070	074652	106702			EDT1:	.WORD	ED1
16071	074654	106712			EDT2:	.WORD	ED2
16072	074656	106716			EDT5:	.WORD	ED5
16073	074660	106724			EDT57:	.WORD	ED57
16074	074662	106736			EDT65:	.WORD	ED65
16075	074664	106746			EDT71:	.WORD	ED71
16076	074666	106712			EDT74:	.WORD	ED2
16077	074670	106756			EDT115:	.WORD	ED115
16078	074672	106770			EDT130:	.WORD	ED130
16079	074674	106756			EDT132:	.WORD	ED115
16080	074676	107004			EDT220:	.WORD	ED220
16081							
16082	074700	107010			EFT1:	.WORD	EF1
16083	074702	107013			EFT2:	.WORD	EF2
16084	074704	107014			EFT5:	.WORD	EF5
16085	074706	107016			EFT57:	.WORD	EF57
16086	074710	107010			EFT65:	.WORD	EF1
16087	074712	107010			EFT71:	.WORD	EF1
16088	074714	107013			EFT74:	.WORD	EF2
16089	074716	107016			EFT115:	.WORD	EF57
16090	074720	107022			EFT130:	.WORD	EF130
16091	074722	107016			EFT132:	.WORD	EF57
16092	074724	107014			EFT220:	.WORD	EF5
16093							
16094	074726	047516	042516	044530	EMS1:	.ASCIZ	NONEXISTENT DEVICE "NED" (RMCS2,BIT 12) @
16095	074734	052123	047105	020124			
16096	074742	042504	044526	042503			
16097	074750	021040	042516	021104			
16098	074756	024040	046522	051503			
16099	074764	026062	044502	020124			
16100	074772	031061	020051	000			
16101	074777	103	047117	051124	EMS2:	.ASCIZ	CONTROLLER CLEAR "CLR" (RMCS2,BIT 05) @
16102	075004	046117	042514	020122			
16103	075012	046103	040505	020122			
16104	075020	041442	051114	020042			
16105	075026	051050	041515	031123			
16106	075034	041054	052111	030040			
16107	075042	024465	000040				
16108	075046	052506	041516	044524	EMS3:	.ASCIZ	FUNCTION CODE (RMCS1, BITS 01 - 05) @
16109	075054	047117	041440	042117			
16110	075062	020105	051050	041515			
16111	075070	030523	020054	044502			
16112	075076	051524	030040	020061			
16113	075104	020055	032460	020051			
16114	075112	000					
16115	075113	125	052516	042523	EMS4:	.ASCIZ	UNUSED BIT POSITIONS OF @
16116	075120	020104	044502	020124			
16117	075126	047520	044523	044524			
16118	075134	047117	020123	043117			
16119	075142	000040					
16120	075144	042504	044526	042503	EMS5:	.ASCIZ	DEVICE AVAILABLE "DVA" (RMCS1, BIT 11) @
16121	075152	040440	040526	046111			

16122	075160	041101	042514	021040	
16123	075166	053104	021101	024040	
16124	075174	046522	051503	026061	
16125	075202	041040	052111	030440	
16126	075210	024461	000040		
16127	075214	040520	052122	054511	EMS6: .ASCIZ @PARTIY ERROR "PAR" (RMER1, BIT 03) @
16128	075222	042440	051122	051117	
16129	075230	021040	040520	021122	
16130	075236	024040	046522	051105	
16131	075244	026061	041040	052111	
16132	075252	030040	024463	000040	
16133	075260	040504	040524	050040	EMS7: .ASCIZ @DATA PARITY ERROR "DPE" (RMER2, BIT 03) @
16134	075266	051101	052111	020131	
16135	075274	051105	047522	020122	
16136	075302	042042	042520	020042	
16137	075310	051050	042515	031122	
16138	075316	020054	044502	020124	
16139	075324	031460	020051	000	
16140	075331	120	051101	052111	EMS10: .ASCIZ @PARITY TEST "PAT" (RMCS2, BIT 04) @
16141	075336	020131	042524	052123	
16142	075344	021040	040520	021124	
16143	075352	024040	046522	051503	
16144	075360	026062	041040	052111	
16145	075366	030040	024464	000040	
16146	075374	040515	051523	052502	EMS11: .ASCII @MASSBUS CONTROL BUS PARITY ERROR "MCPE" @
16147	075402	020123	047503	052116	
16148	075410	020114	047522	052502	
16149	075416	020123	040520	044522	
16150	075424	054524	042440	051122	
16151	075432	051117	021040	041515	
16152	075440	042520	020042		
16153	075444	051050	041515	030523	.ASCIZ @(RMCS1, BIT 13) @
16154	075452	020054	044502	020124	
16155	075460	031461	020051	000	
16156	075465	111	046114	043505	EMS12: .ASCIZ @ILLEGAL REGISTER ERROR "ILR" (RMER1, BIT 01) @
16157	075472	046101	051040	043505	
16158	075500	051511	042524	020122	
16159	075506	051105	047522	020122	
16160	075514	044442	051114	020042	
16161	075522	051050	042515	030522	
16162	075530	020054	044502	020124	
16163	075536	030460	020051	000	
16164	075543	104	040511	047107	EMS13: .ASCIZ @DIAGNOSTIC MODE "DMD" (RMMR1, BIT 00) @
16165	075550	051517	044524	020103	
16166	075556	047515	042504	021040	
16167	075564	046504	021104	024040	
16168	075572	046522	051115	026061	
16169	075600	041040	052111	030040	
16170	075606	024460	000040		
16171	075612	042515	044504	046525	EMS14: .ASCIZ @MEDIUM ON LINE "MOL" (RMDS, BIT 12) @
16172	075620	047440	020116	044514	
16173	075626	042516	021040	047515	
16174	075634	021114	024040	046522	
16175	075642	051504	020054	044502	
16176	075650	020124	031061	020051	
16177	075656	000			

16178	075657	115	044501	052116	EMS15: .ASCIZ @MAINTENANCE UNIT READY "MUR" (RMMR1, BIT 09) @
16179	075664	047105	047101	042503	
16180	075672	052440	044516	020124	
16181	075700	042522	042101	020131	
16182	075706	046442	051125	020042	
16183	075714	051050	046515	030522	
16184	075722	020054	044502	020124	
16185	075730	034460	020051	000	
16186	075735	115	044501	052116	EMS16: .ASCIZ @MAINTENANCE WRITE PROTECT "MWP" (RMMR1, BIT 03) @
16187	075742	047105	047101	042503	
16188	075750	053440	044522	042524	
16189	075756	050040	047522	042524	
16190	075764	052103	021040	053515	
16191	075772	021120	024040	046522	
16192	076000	051115	026061	041040	
16193	076006	052111	030040	024463	
16194	076014	000040			
16195	076016	051127	052111	020105	EMS17: .ASCIZ @WRITE LOCK "WRL" (RMD5, BIT 11) @
16196	076024	047514	045503	021040	
16197	076032	051127	021114	024040	
16198	076040	046522	051504	020054	
16199	076046	044502	020124	030461	
16200	076054	020051	000		
16201	076057	104	053105	041511	EMS20: .ASCIZ @DEVICE CHECK "DVC" (RMER2, BIT 07) @
16202	076064	020105	044103	041505	
16203	076072	020113	042042	041526	
16204	076100	020042	051050	042515	
16205	076106	031122	020054	044502	
16206	076114	020124	033460	020051	
16207	076122	000			
16208	076123	115	044501	052116	EMS21: .ASCIZ @MAINTENANCE DRIVE FAULT "MDF" (RMMR1, BIT 06) @
16209	076130	047105	047101	042503	
16210	076136	042040	044522	042526	
16211	076144	043040	052501	052114	
16212	076152	021040	042115	021106	
16213	076160	024040	046522	051115	
16214	076166	026061	041040	052111	
16215	076174	030040	024466	000040	
16216	076202	047125	040523	042506	EMS22: .ASCIZ @UNSAFE STATUS "UNS" (RMER1, BIT 14) @
16217	076210	051440	040524	052524	
16218	076216	020123	052442	051516	
16219	076224	020042	051050	042515	
16220	076232	030522	020054	044502	
16221	076240	020124	032061	020051	
16222	076246	000			
16223	076247	123	042505	020113	EMS23: .ASCIZ @SEEK INCOMPLETE STATUS "SKI" (RMER2, BIT 14) @
16224	076254	047111	047503	050115	
16225	076262	042514	042524	051440	
16226	076270	040524	052524	020123	
16227	076276	051442	044513	020042	
16228	076304	051050	042515	031122	
16229	076312	020054	044502	020124	
16230	076320	032061	020051	000	
16231	076325	115	044501	052116	EMS24: .ASCIZ @MAINTENANCE SEEK ERROR "MSER" (RMMR1, BIT 07) @
16232	076332	047105	047101	042503	
16233	076340	051440	042505	020113	

16234	076346	051105	047522	020122	
16235	076354	046442	042523	021122	
16236	076362	024040	045523	051115	
16237	076370	026061	041040	052111	
16238	076376	030040	024467	000040	
16239	076404	047520	044523	044524	EMS25: .ASCIZ 2POSITIONING IN PROGRESS "PIP" (RMDS, BIT 13) 2
16240	076412	047117	047111	020107	
16241	076420	047111	050040	047522	
16242	076428	051107	021505	020123	
16243	076434	050042	050111	020042	
16244	076442	051050	042115	026123	
16245	076450	041040	052111	030440	
16246	076456	024463	000040		
16247	076462	040515	047111	042524	EMS26: .ASCIZ 2MAINTENANCE ON CYLINDER "MOC" (RMMR1, BIT 08) 2
16248	076470	040516	041516	020105	
16249	076476	047117	041440	046131	
16250	076504	047111	042504	020122	
16251	076512	046442	041517	020042	
16252	076520	051050	046515	030522	
16253	076526	020054	044502	020124	
16254	076534	034060	020051	000	
16255	076541	105	042116	047440	EMS27: .ASCIZ 2END OF BLOCK "EBL" (RMMR1, BIT 13) 2
16256	076546	020106	046102	041517	
16257	076554	020113	042442	046102	
16258	076562	020042	051050	046515	
16259	076570	030522	020054	044502	
16260	076576	020124	031461	020051	
16261	076604	000			
16262	076605	104	040511	047107	EMS30: .ASCIZ 2DIAGNOSTIC END OF BLOCK "DEBL" (RMMR1, BIT 13) 2
16263	076612	051517	044524	020103	
16264	076620	047105	020104	043117	
16265	076627	041040	047514	045503	
16266	076634	021040	042504	046102	
16267	076642	020042	051050	046515	
16268	076650	030522	020054	044502	
16269	076656	020124	031461	020051	
16270	076664	000			
16271	076665	114	051501	020124	EMS31: .ASCIZ 2LAST SECTOR STATUS "LS" (RMMR1, BIT 02) 2
16272	076672	042523	052103	051117	
16273	076700	051440	040524	052524	
16274	076706	020123	046042	021123	
16275	076714	024040	046522	051115	
16276	076722	026061	041040	052111	
16277	076730	030040	024462	000040	
16278	076736	040514	052123	051440	EMS32: .ASCIZ 2LAST SECTOR TRACK STATUS "LST" (RMMR1, BIT 01) 2
16279	076744	041505	047524	027522	
16280	076752	051124	041501	020113	
16281	076760	052123	052101	051525	
16282	076766	021040	051514	021124	
16283	076774	024040	046522	051115	
16284	077002	026061	041040	052111	
16285	077010	030040	024461	000040	
16286	077016	042523	052103	051117	EMS33: .ASCIZ 2SECTOR ADDRESS, BITS 00-04 OF 2
16287	077024	040440	042104	043522	
16288	077032	051523	020054	044502	
16289	077040	051524	030040	026460	

16290	077046	032060	047440	020106	
16291	077054	000			
16292	077055	124	040522	045503	EMS34: .ASCIZ @TRACK ADDRESS, BITS 08-10 OF @
16293	077062	040440	042104	042522	
16294	077070	051523	020054	044502	
16295	077076	051524	030040	026470	
16296	077104	030061	047440	020106	
16297	077112	000			
16298	077113	126	046117	046525	EMS35: .ASCIZ @VOLUME VALID "VV" (RMDS, BIT 06) @
16299	077120	020105	040526	044514	
16300	077126	020104	053042	021126	
16301	077134	024040	046522	051504	
16302	077142	020054	044502	020124	
16303	077150	033060	020051	000	
16304	077155	107	020117	044502	EMS36: .ASCIZ @GO BIT (RMCS1, BIT 00) @
16305	077162	020124	051050	041515	
16306	077170	030523	020054	044502	
16307	077175	020124	030060	020051	
16308	077204	000			
16309	077205	103	046131	047111	EMS37: .ASCIZ @CYLINDER ADDRESS, BIT 00-09 OF @
16310	077212	042504	020122	042101	
16311	077220	051104	051505	026123	
16312	077226	041040	052111	030040	
16313	077234	026460	034460	047440	
16314	077242	020106	000		
16315	077245	114	051501	020124	EMS40: .ASCIZ @LAST BLOCK TAKEN STATUS "LBT" (RMDS, BIT 10) @
16316	077252	046102	041517	020113	
16317	077260	040524	042513	020116	
16318	077266	052123	052101	051525	
16319	077274	021040	041114	021124	
16320	077302	024040	046522	051504	
16321	077310	020054	044502	020124	
16322	077316	030061	020051	000	
16323	077323	103	046517	047520	EMS41: .ASCIZ @COMPOSITE ERROR "ERR" (RMDS, BIT 14) @
16324	077330	044523	042524	042440	
16325	077336	051122	051117	021040	
16326	077344	051105	021122	024040	
16327	077352	046522	051504	020054	
16328	077360	044502	020124	032061	
16329	077366	020051	000		
16330	077371	103	046517	040515	EMS42: .ASCIZ @COMMAND SEQUENCER TEST BIT "TST" (RMMR2, BIT 12) @
16331	077376	042116	051440	050505	
16332	077404	042525	041516	051105	
16333	077412	052040	051505	020124	
16334	077420	044502	020124	052042	
16335	077426	052123	020042	051050	
16336	077434	046515	031122	020054	
16337	077442	044502	020124	031061	
16338	077450	020051	000		

16339	077453	104	044522	042526	EMS43: .ASCIZ @DRIVE READY STATUS "DRY" (RMDS, BIT 07) @
16340	077460	051040	040509	054504	
16341	077466	051440	040524	052524	
16342	077474	020123	042042	054522	
16343	077502	020042	051050	042115	
16344	077510	026123	041040	052111	
16345	077516	030040	024467	000040	
16346	077524	047503	052116	047111	EMS44: .ASCIZ @CONTINUE "CONT" (RMMR1, BIT 06) @
16347	077532	042525	021040	047503	
16348	077540	052116	020042	051050	
16349	077546	046515	030522	020054	
16350	077554	044502	020124	033060	
16351	077562	020051	000		
16352	077565	111	053116	046101	EMS45: .ASCIZ @INVALID COMMAND ERROR "IVC" (RMER2, BIT 12) @
16353	077572	042111	041440	046517	
16354	077600	040515	042116	042440	
16355	077606	051122	051117	021040	
16356	077614	053111	021103	024040	
16357	077622	046522	051105	026062	
16358	077630	041040	052111	030440	
16359	077636	024462	000040		
16360	077642	047514	051523	047440	EMS46: .ASCIZ @LOSS OF SYSTEM CLOCK ERROR "LSC" (RMER2, BIT 11) @
16361	077650	020106	054523	052123	
16362	077656	046505	041440	047514	
16363	077664	045503	042440	051122	
16364	077672	051117	021040	051514	
16365	077700	021103	024040	046522	
16366	077706	051105	026062	041040	
16367	077714	052111	030440	024461	
16368	077722	000040			
16369	077724	041517	052503	044520	EMS47: .ASCIZ @OCCUPIED "OCC" (RMMR1, BIT 15) @
16370	077732	042105	021040	041517	
16371	077740	021103	024040	046522	
16372	077746	051115	026061	041040	
16373	077754	052111	030440	024465	
16374	077762	000040			
16375	077764	046111	042514	040507	EMSS0: .ASCIZ @ILLEGAL FUNCTION "ILF" (RMER1, BIT 0) @
16376	077772	020114	052506	041516	
16377	100000	044524	047117	021040	
16378	100006	046111	021106	024040	
16379	100014	046522	051105	026061	
16380	100022	041040	052111	030040	
16381	100030	020051	000		
16382	100033	117	043106	042523	EMSS1: .ASCIZ @OFFSET DIRECTION "OFD" (RMOF, BIT 07) @
16383	100040	020124	044504	042522	
16384	100046	052103	047511	020116	
16385	100054	047442	042106	020042	
16386	100062	051050	047515	026106	
16387	100070	041040	052111	030040	
16388	100076	024467	000040		
16389	100102	043117	051506	052105	EMSS2: .ASCIZ @OFFSET MODE "OM" (RMDS, BIT 00) @
16390	100110	046440	042117	020105	
16391	100116	047442	021115	024040	
16392	100124	046522	051504	020054	
16393	100132	044502	020124	030060	
16394	100140	020051	000		

16395	100143	122	047125	040440	EMSS3: .ASCIZ 2RUN AND GO "RG" (RMMR1, BIT 14) 2
16396	100150	042116	043440	020117	
16397	100156	051042	021107	024040	
16398	100164	046522	051115	026061	
16399	100172	041040	052111	030440	
16400	100200	024464	000040		
16401	100204	047111	040526	044514	EMSS4: .ASCIZ 2INVALID ADDRESS ERROR "IAE" (RMER1, BIT 10) 2
16402	100212	020104	042101	051104	
16403	100220	051505	020123	051105	
16404	100226	047522	020122	044442	
16405	100234	042501	020042	051050	
16406	100242	042515	030522	020054	
16407	100250	044502	020124	030061	
16408	100256	020051	000		
16409	100261	101	042104	042522	EMSS5: .ASCIZ 2ADDRESS OVERFLOW ERROR "AOE" (RMER1, BIT 09) 2
16410	100266	051523	047440	042526	
16411	100274	043122	047514	020127	
16412	100302	051105	047522	020122	
16413	100310	040442	042517	020042	
16414	100316	051050	042515	030522	
16415	100324	020054	044502	020124	
16416	100332	034460	020051	000	
16417	100337	122	043505	051511	EMSS6: .ASCII 2REGISTER MODIFICATION REFUSED ERROR 2
16418	100344	042524	020122	047515	
16419	100352	044504	044506	040503	
16420	100360	044524	047117	051040	
16421	100366	043105	051525	042105	
16422	100374	042440	051122	051117	
16423	100402	040			
16424	100403	042	046522	021122	.ASCIZ 2"RMR" (RMER1, BIT 02) 2
16425	100410	024040	046522	051105	
16426	100416	026061	041040	052111	
16427	100424	030040	024462	000040	
16428	100432	051104	053111	020105	EMSS7: .ASCIZ 2DRIVE REQUEST REQUIRED STATUS "DRQ" (RMDT, BIT 11) 2
16429	100440	042522	052521	051505	
16430	100446	020124	042522	052521	
16431	100454	051111	042105	051440	
16432	100462	040524	052524	020123	
16433	100470	042042	050522	020042	
16434	100476	051050	042115	026124	
16435	100504	041040	052111	030440	
16436	100512	024461	000040		
16437	100516	051120	043517	040522	EMSS8: .ASCIZ 2PROGRAMMABLE STATUS "PGM" (RMDS, BIT 09) 2
16438	100524	046515	041101	042514	
16439	100532	051440	040524	052524	
16440	100540	020123	050042	046507	
16441	100546	020042	051050	042115	
16442	100554	026123	041040	052111	
16443	100562	030040	024471	000040	
16444	100570	051104	053111	020105	EMSS9: .ASCIZ 2DRIVE PRESENT STATUS "DPR" (RMDS, BIT 08) 2
16445	100576	051120	051505	047105	
16446	100604	020124	052123	052101	
16447	100612	051525	021040	050104	
16448	100620	021122	024040	046522	
16449	100626	051504	020054	044502	
16450	100634	020124	034060	020051	

16451	100642	000			
16452	100643	120	051117	020124	EMS62: .ASCIZ @PORT REQUEST FLOP "RQA,RGB" (RMMR2, BITS 15,14) @
16453	100650	042522	052521	051505	
16454	100656	020124	046106	050117	
16455	100664	021040	050522	026101	
16456	100672	050522	021102	024040	
16457	100700	046522	051115	026062	
16458	100706	041040	052111	020123	
16459	100714	032461	030454	024464	
16460	100722	000040			
16461	100724	052101	042524	052116	EMS63: .ASCIZ @ATTENTION "ATA" (RMDS, BIT 15) @
16462	100732	047511	020116	040442	
16463	100740	040524	020042	051050	
16464	100746	042115	026123	041040	
16465	100754	052111	030440	024465	
16466	100762	000040			
16467	100764	051127	052111	020105	EMS64: .ASCIZ @WRITE LOCK ERROR "WLE" (RMER1, BIT 11) @
16468	100772	047514	045503	042440	
16469	101000	051122	051117	021040	
16470	101006	046127	021105	024040	
16471	101014	046522	051105	026061	
16472	101022	041040	052111	030440	
16473	101030	024461	000040		
16474	101034	054105	042503	052120	EMS65: .ASCIZ @EXCEPTION "REX" (RMMR1, BIT 12) @
16475	101042	047511	020116	051042	
16476	101050	054105	020042	051050	
16477	101056	046515	030522	020054	
16478	101064	044502	020124	031061	
16479	101072	020051	000		
16480	101075	117	042520	040522	EMS66: .ASCIZ @OPERATION INCOMPLETE ERROR "OPI" (RMER1, BIT 13) @
16481	101102	044524	047117	044440	
16482	101110	041516	046517	046120	
16483	101116	052105	020105	051105	
16484	101124	047522	020122	047442	
16485	101132	044520	020042	051050	
16486	101140	042515	030522	020054	
16487	101146	044502	020124	031461	
16488	101154	020051	000		
16489	101157	124	043501	041040	EMS67: .ASCIZ @TAG BUS (RMMR2, BITS 00-09) OR TAG CONTROL @
16490	101164	051525	024040	046522	
16491	101172	051115	026062	041040	
16492	101200	052111	020123	030060	
16493	101206	030055	024471	047440	
16494	101214	020122	040524	020107	
16495	101222	047503	052115	047522	
16496	101230	020114	000		
16497	101233	114	047111	051505	.ASCIZ @LINES (RMMR2, BITS 10,11,13) @
16498	101240	024040	046522	051115	
16499	101246	026062	041040	052111	
16500	101254	020123	030061	030454	
16501	101262	026061	031461	020051	
16502	101270	000			
16503	101271	123	040505	041522	EMS70: .ASCIZ @SEARCH ENABLE "ESRC" (RMMR1, BIT 11) @
16504	101276	020110	047105	041101	
16505	101304	042514	021040	051505	
16506	101312	041522	020042	051050	

E10

CZRMJBO RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07MACY11 30(1046) 23-NOV-77 12:14 PAGE 328
CONSOLE MESSAGES

SEQ 0328

16507	101320	046515	030522	020054	
16508	101326	044502	020124	030461	
16509	101334	020051	000		
16510					
16511	101337	104	051511	020113	EMS250: .ASCIZ @DISK ADDRESS REGISTER (RMDA) @
16512	101344	042101	051104	051505	
16513	101352	020123	042522	044507	
16514	101360	052123	051105	024040	
16515	101366	046522	040504	020051	
16516	101374	000			
16517	101375	103	047117	051124	EMS251: .ASCIZ @CONTROL STATUS REGISTER #1 (RMCS1) @
16518	101402	046117	051440	040524	
16519	101410	052524	020123	042522	
16520	101416	044507	052123	051105	
16521	101424	021440	020061	051050	
16522	101432	041515	030523	020051	
16523	101440	000			
16524	101441	105	051122	051117	EMS252: .ASCIZ @ERROR REGISTER #1 (RMER1) @
16525	101446	051040	043505	051511	
16526	101454	042524	020122	030443	
16527	101462	024040	046522	051105	
16528	101470	024461	000040		
16529	101474	051105	047522	020122	EMS253: .ASCIZ @ERROR REGISTER #2 (RMER2) @
16530	101502	042522	044507	052123	
16531	101510	051105	021440	020062	
16532	101516	051050	042515	031122	
16533	101524	020051	000		
16534	101527	115	044501	052116	EMS254: .ASCIZ @MAINTENANCE REGISTER #1 (RMMR1) @
16535	101534	047105	047101	042503	
16536	101542	051040	043505	051511	
16537	101550	042524	020122	030443	
16538	101556	024040	046522	051115	
16539	101564	024461	000040		
16540	101570	042504	044523	042522	EMS255: .ASCIZ @DESIRED CYLINDER REGISTER (RMDC) @
16541	101576	020104	054503	044514	
16542	101604	042116	051105	051040	
16543	101612	043505	051511	042524	
16544	101620	020122	051050	042115	
16545	101626	024503	000040		
16546	101632	043117	051506	052105	EMS256: .ASCIZ @OFFSET REGISTER (RMOF) @
16547	101640	051040	043505	051511	
16548	101646	042524	020122	051050	
16549	101654	047515	024506	000040	
16550	101662	051104	053111	020105	EMS257: .ASCIZ @DRIVE TYPE REGISTER (RMDT) @
16551	101670	054524	042520	051040	
16552	101676	043505	051511	042524	
16553	101704	020122	051050	042115	
16554	101712	024524	000040		
16555	101716	047510	042114	047111	EMS260: .ASCIZ @HOLDING REGISTER (RMHR) @
16556	101724	020107	042522	044507	
16557	101732	052123	051105	024040	
16558	101740	046522	051110	020051	
16559	101746	000			
16560	101747	123	051105	040511	EMS261: .ASCIZ @SERIAL NUMBER REGISTER (RMSN) @
16561	101754	020114	052515	041115	
16562	101762	051105	051040	043505	

16563	101770	051511	042524	020122	
16564	101776	051050	051515	024516	
16565	102004	000040			
16566	102006	052101	042524	052116	EMS262: .ASCIZ @ATTENTION SUMMARY REGISTER (RMAS) @
16567	102014	047511	020116	052523	
16568	102022	046515	051101	020131	
16569	102030	042522	044507	052123	
16570	102036	051105	024040	046522	
16571	102044	051501	020051	000	
16572					
16573	102051	103	047101	047516	EMS300: .ASCIZ @CANNOT CLEAR @
16574	102056	020124	046103	040505	
16575	102064	020122	000		
16576	102067	103	047101	047516	EMS301: .ASCIZ @CANNOT WRITE/READ @
16577	102074	020124	051127	052111	
16578	102102	027505	042522	042101	
16579	102110	000040			
16580	102112	047101	020131	042504	EMS302: .ASCIZ @ANY DEVICE REGISTER @
16581	102120	044526	042503	051040	
16582	102126	043505	051511	042524	
16583	102134	020122	000		
16584	102137	127	052111	047510	EMS303: .ASCIZ @WITHOUT @
16585	102144	052125	000040		
16586	102150	051105	047522	020122	EMS304: .ASCIZ @ERROR @
16587	102156	000			
16588	102157	101	047440	042516	EMS306: .ASCIZ @A ONE FROM @
16589	102164	043040	047522	020115	
16590	102172	000			
16591	102173	125	044523	043516	EMS307: .ASCIZ @USING MASSBUS INITIALIZE, I.E., @
16592	102200	046440	051501	041123	
16593	102206	051525	044440	044516	
16594	102214	044524	046101	055111	
16595	102222	026105	044440	042456	
16596	102230	026056	000040		
16597	102234	020101	042532	047522	EMS310: .ASCIZ @A ZERO FROM @
16598	102242	043040	047522	020115	
16599	102250	000			
16600	102251	105	042526	054522	EMS311: .ASCIZ @EVERY DEVICE REGISTER BIT POSITION @
16601	102256	042040	053105	041511	
16602	102264	020105	042522	044507	
16603	102272	052123	051105	041040	
16604	102300	052111	050040	051517	
16605	102306	052111	047511	020116	
16606	102314	000			
16607	102315	124	042510	043040	EMS312: .ASCIZ @THE FOLLOWING BITS ARE STUCK @
16608	102322	046117	047514	044527	
16609	102330	043516	041040	052111	
16610	102336	020123	051101	020105	
16611	102344	052123	041525	020113	
16612	102352	000			
16613	102353	101	051440	044510	EMS313: .ASCIZ @A SHIFTING ONE BIT FROM @
16614	102360	052106	047111	020107	
16615	102366	047117	020105	044502	
16616	102374	020124	051106	046517	
16617	102402	000040			
16618	102404	050101	042520	051101	EMS314: .ASCIZ @APPEARS STUCK AT ZERO @

16619	102412	020123	052123	041525	
16620	102420	020113	052101	055040	
16621	102426	051105	020117	000	
16622	102433	101	050120	040505	EMS315: .ASCIZ @APPEARS STUCK AT ONE @
16623	102440	051522	051440	052524	
16624	102446	045503	040440	020124	
16625	102454	047117	020105	000	
16626	102461	122	043505	051511	EMS316: .ASCIZ @REGISTER SELECT @
16627	102466	042524	020122	042523	
16628	102474	042514	052103	000040	
16629	102502	020061	030450	031054	EMS317: .ASCIZ @1 (1,2,4,8,16) @
16630	102510	032054	034054	030454	
16631	102516	024466	000040		
16632	102522	020062	030450	031054	EMS320: .ASCIZ @2 (1,2,4,8,16) @
16633	102530	032054	034054	030454	
16634	102536	024466	000040		
16635	102542	020064	030450	031054	EMS321: .ASCIZ @4 (1,2,4,8,16) @
16636	102550	032054	034054	030454	
16637	102556	024466	000040		
16638	102562	020070	030450	031054	EMS322: .ASCIZ @8 (1,2,4,8,16) @
16639	102570	032054	034054	030454	
16640	102576	024466	000040		
16641	102602	046101	020114	047117	EMS323: .ASCIZ @ALL ONES FROM @
16642	102610	051505	043040	047522	
16643	102616	020115	000		
16644	102621	101	046114	055040	EMS324: .ASCIZ @ALL ZEROS FROM @
16645	102626	051105	051517	043040	
16646	102634	047522	020115	000	
16647	102641	101	020124	042532	EMS325: .ASCIZ @AT ZERO @
16648	102646	047522	000040		
16649	102652	052101	047440	042516	EMS326: .ASCIZ @AT ONE @
16650	102660	000040			
16651	102662	020054	051117	000040	EMS327: .ASCIZ @ OR @
16652	102670	005015	051503	046440	EMS330: .ASCIZ (CR)(LF)@CS MBA CLRL @
16653	102676	040502	041440	051114	
16654	102704	020114	000		
16655	102707	103	047101	047516	EMS331: .ASCIZ @CANNOT READ ZEROS FROM @
16656	102714	020124	042522	042101	
16657	102722	055040	051105	051517	
16658	102730	043040	047522	020115	
16659	102736	000			
16660	102737	111	020123	047111	EMS332: .ASCIZ @IS INCORRECT @
16661	102744	047503	051122	041505	
16662	102752	020124	000		
16663	102755	111	020123	047516	EMS333: .ASCIZ @IS NOT SET @
16664	102762	020124	042523	020124	
16665	102770	000			
16666	102771	111	020123	042523	EMS334: .ASCIZ @IS SET @
16667	102776	020124	000		
16668	103001	123	047510	046125	EMS335: .ASCIZ @SHOULD NOT BE SET @
16669	103006	020104	047516	020124	
16670	103014	042502	051440	052105	
16671	103022	000040			
16672	103024	044123	052517	042114	EMS336: .ASCIZ @SHOULD BE SET @
16673	103032	041040	020105	042523	
16674	103040	020124	000		

H10

CZRMJBO RMO3-2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 331
CONSOLE MESSAGES

SEQ 0331

16675	103043	103	047101	047516	EMS337: .ASCIZ @CANNOT SET @
16676	103050	020124	042523	020124	
16677	103056	000			
16678	103057	102	041505	052501	EMS340: .ASCIZ @BECAUSE @
16679	103064	042523	000040		
16680	103070	051525	047111	020107	EMS341: .ASCIZ @USING @
16681	103076	000			
16682	103077	104	051125	047111	EMS342: .ASCIZ @DURING REGISTER TRANSFER @
16683	103104	020107	042522	044507	
16684	103112	052123	051105	052040	
16685	103120	040522	051516	042506	
16686	103126	020122	000		
16687	103131	125	042516	050130	EMS343: .ASCIZ @UNEXPECTED @
16688	103136	041505	042524	020104	
16689	103144	000			
16690	103145	102	051525	052040	EMS344: .ASCIZ @BUS TIMEOUT (04 TRAP) @
16691	103152	046511	047505	052125	
16692	103160	024040	032060	052040	
16693	103166	040522	024520	000040	
16694	103174	054502	051040	043505	EMS345: .ASCIZ @BY REGISTER TRANSFER @
16695	103202	051511	042524	020122	
16696	103210	051124	047101	043123	
16697	103216	051105	000040		
16698	103222	040503	047116	052117	EMS346: .ASCIZ @CANNOT RESET @
16699	103230	051040	051505	052105	
16700	103236	000040			
16701	103240	044527	044124	052517	EMS347: .ASCIZ @WITHOUT SETTING @
16702	103246	020124	042523	052124	
16703	103254	047111	020107	000	
16704	103261	102	052125	000040	EMS350: .ASCIZ @BUT @
16705	103266	040527	020123	042522	EMS351: .ASCIZ @WAS RESET BY @
16706	103274	042523	020124	054502	
16707	103302	000040			
16708	103304	040527	020123	042523	EMS352: .ASCIZ @WAS SET BY @
16709	103312	020124	054502	030040	
16710	103320	047111	042040	040511	EMS353: .ASCIZ @IN DIAGNOSTIC MODE @
16711	103326	047107	051517	044524	
16712	103334	020103	047515	042504	
16713	103342	000040			
16714	103344	051511	044440	041516	EMS354: .ASCIZ @IS INCORRECT ACCORDING TO @
16715	103352	051117	042522	052103	
16716	103360	040440	041503	051117	
16717	103366	044504	043516	052040	
16718	103374	020117	000		
16719	103377	103	047101	047516	EMS355: .ASCIZ @CANNOT INCREMENT @
16720	103404	020124	047111	051103	
16721	103412	046505	047105	020124	
16722	103420	000			
16723	103421	127	051501	047040	EMS356: .ASCIZ @WAS NOT SET BY @
16724	103426	052117	051440	052105	
16725	103434	041040	020131	000	
16726	103441	127	051501	047040	EMS357: .ASCIZ @WAS NOT RESET BY @
16727	103446	052117	051040	051505	
16728	103454	052105	041040	020131	
16729	103462	000			
16730	103463	060	052040	020117	EMS360: .ASCIZ @0 TO 1 TRANSITION OF @

16731	103470	020061	051124	047101	
16732	103476	044523	044524	047117	
16733	103504	047440	020106	000	
16734	103511	061	052040	020117	EMS361: .ASCIZ @1 TO 0 TRANSITION OF @
16735	103516	020060	051124	047101	
16736	103524	044523	044524	047117	
16737	103532	047440	020106	000	
16738	103537	111	020123	047111	EMS362: .ASCIZ @IS INCONSISTENT @
16739	103544	047503	051516	051511	
16740	103552	042524	052116	000040	
16741	103560	040503	047116	052117	EMS363: .ASCIZ @CANNOT READ @
16742	103566	051040	040505	020104	
16743	103574	000			
16744	103575	124	051505	020124	EMS364: .ASCIZ @TEST PATTERN IN @
16745	103602	040520	052124	051105	
16746	103610	020116	047111	000040	
16747	103616	047101	020104	000	EMS365: .ASCIZ @AND @
16748	103623	103	047101	047516	EMS366: .ASCIZ @CANNOT INITIALIZE @
16749	103630	020124	047111	052111	
16750	103636	040511	044514	042532	
16751	103644	000040			
16752	103646	044124	020105	047503	EMS367: .ASCIZ @THE COMMAND SEQUENCER HAS BEEN CLOKED @
16753	103654	046515	047101	020104	
16754	103662	042523	052521	047105	
16755	103670	042503	020122	040510	
16756	103676	020123	042502	047105	
16757	103704	041440	047514	045503	
16758	103712	042105	000040		

16759	103716	042522	042523	020124	EMS370:	.ASCIZ	ARESET EARLY	Q
16760	103724	040505	046122	020131				
16761	103732	000						
16762	103733	104	042111	047040	EMS371:	.ASCIZ	ADID NOT RESET ON TIME	Q
16763	103740	052117	051040	051505				
16764	103746	052105	047440	020116				
16765	103754	044524	042515	000040				
16766	103762	052504	044522	043516	EMS372:	.ASCIZ	ADURING COMMAND EXECUTION	Q
16767	103770	041440	046517	040515				
16768	103776	042116	042440	042530				
16769	104004	052503	044524	047117				
16770	104012	000040						
16771	104014	047524	000040		EMS373:	.ASCIZ	ATO	Q
16772	104020	044527	044124	040440	EMS374:	.ASCIZ	ADWITH ANY COMBINATION OF	Q
16773	104026	054516	041440	046517				
16774	104034	044502	040516	044524				
16775	104042	047117	047440	020106				
16776	104050	000						
16777	104051	102	020131	042522	EMS375:	.ASCIZ	ADBY READING	Q
16778	104056	042101	047111	020107				
16779	104064	000						
16780	104065	102	020131	051127	EMS376:	.ASCIZ	ADBY WRITING	Q
16781	104072	052111	047111	020107				
16782	104100	000						
16783	104101	127	051501	051440	EMS377:	.ASCIZ	ADWAS SET	Q
16784	104106	052105	000040					
16785	104112	040527	020123	047516	EMS400:	.ASCIZ	ADWAS NOT SET	Q
16786	104120	020124	042523	020124				
16787	104126	000						
16788	104127	104	042111	047040	EMS401:	.ASCIZ	ADID NOT SET	Q
16789	104134	052117	051440	052105				
16790	104142	000040						
16791	104144	044127	046111	020105	EMS402:	.ASCIZ	ADWHILE	Q
16792	104152	000						
16793	104153	103	046517	040515	EMS403:	.ASCIZ	ADCOMMAND SEQUENCER DID NOT ABORT	Q
16794	104160	042116	051440	050505				
16795	104166	042525	041516	051105				
16796	104174	042040	042111	047040				
16797	104202	052117	040440	047502				
16798	104210	052122	000040					
16799	104214	040527	020123	047516	EMS404:	.ASCIZ	ADWAS NOT RESET	Q
16800	104222	020124	042523	042523				
16801	104230	020124	000					
16802	104233	104	051125	047111	EMS405:	.ASCIZ	ADURING	Q
16803	104240	020107	000					
16804	104243	127	051501	051040	EMS406:	.ASCIZ	ADWAS RESET	Q
16805	104250	051505	052105	000040				
16806	104256	042523	051101	044103	EMS407:	.ASCIZ	ADSEARCH TIMEOUT	Q
16807	104264	052040	046511	047505				
16808	104272	052125	000040					
16809								
16810	104276	042011	053105	041511	EMSS00:	.ASCII	Q	DEVICE IS NONEXISTENT, Q<CR><LF>
16811	104304	020105	051511	047040				
16812	104312	047117	054105	051511				
16813	104320	042524	052116	006454				
16814	104326	012						

K10

CZRMJ80 RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 334
CONSOLE MESSAGES

SEQ 0334

16815	104327	011	042504	044526	.ASCII	a	DEVICE IS SWITCHED TO OTHER PORTa<CR><LF>
16816	104334	042503	044440	020123			
16817	104342	053523	052111	044103			
16818	104350	042105	052040	020117			
16819	104356	052117	042510	020122			
16820	104364	047520	052122	035015			
16821	104372	052011	040522	051516	.ASCIZ	a	TRANSCEIVER ENABLE SWITCH IS OFFa<CR><LF>
16822	104400	042503	053111	051105			
16823	104406	042440	040516	046102			
16824	104414	020105	053523	052111			
16825	104422	044103	044440	020123			
16826	104430	043117	006506	000012			
16827	104436	044411	020106	047515	EMSS01: .ASCIZ	a	IF MODULE, M7686,a<CR><LF>
16828	104444	052504	042514	020054			
16829	104452	033515	034066	026066			
16830	104460	005015	000				
16831	104463	011	040515	051523	EMSS02: .ASCIZ	a	MASSBUS TRANSCEIVER,M5922 OR M5923 a<CR><LF>
16832	104470	052502	020123	051124			
16833	104476	047101	041523	044505			
16834	104504	042526	026122	032515			
16835	104512	031071	020062	051117			
16836	104520	046440	034465	031462			
16837	104526	006411	000012				
16838	104532	041411	020123	047515	EMSS03: .ASCIZ	a	CS MODULE,M7684,a<CR><LF>
16839	104540	052504	042514	046454			
16840	104546	033067	032070	006454			
16841	104554	000012					
16842	104556	042011	020123	047515	EMSS04: .ASCIZ	a	DS MODULE,M7685,a<CR><LF>
16843	104564	052504	042514	046454			
16844	104572	033067	032470	006454			
16845	104600	000012					
16846	104602	042011	053105	041511	EMSS05: .ASCIZ	a	DEVICE IS SWITCHED TO A/B PORT POSITIONa<CR><LF>
16847	104610	020105	051511	051440			
16848	104616	044527	041524	042510			
16849	104624	020104	047524	040440			
16850	104632	041057	050040	051117			
16851	104640	020124	047520	044523			
16852	104646	044524	047117	005015			
16853	104654	000					
16854	104655	011	042504	044526	EMSS06: .ASCIZ	a	DEVICE IS NOT AN RMO3, ORa<CR><LF>
16855	104662	042503	044440	020123			
16856	104670	047516	020124	047101			
16857	104676	051040	030115	026063			
16858	104704	047440	006522	000012			
16859	104712	042011	053105	041511	EMSS07: .ASCIZ	a	DEVICE IS SWITCHED TO PROGRAMMABLE PORT POSITION, ORa<CR><LF>
16860	104720	020105	051511	051440			
16861	104726	044527	041524	042510			
16862	104734	020104	047524	050040			
16863	104742	047522	051107	046501			
16864	104750	040515	046102	020105			
16865	104756	047520	052122	050040			
16866	104764	051517	052111	047511			
16867	104772	026116	047440	006522			
16868	105000	000012					
16869	105002	040411	051523	046525	EMSS10: .ASCIZ	a	ASSUMING THE RM CONTROLLER HAS NO FAULTa<CR><LF>
16870	105010	047111	020107	044124			

16871	105016	020105	044122	041440	
16872	105024	047117	051124	046117	
16873	105032	042514	020122	040510	
16874	105040	020123	047516	043040	
16875	105046	052501	052114	005015	
16876	105054	000			
16877	105055	015	004412	051120	EMS511: .ASCII <CR><LF> PROBABLE FAULT(S):<CR><LF>
16878	105062	041117	041101	042514	
16879	105070	043040	052501	052114	
16880	105076	051450	035051	005015	
16881	105104	024011	047516	020124	.ASCIZ (NOT INCLUDING CABLES OR CONNECTORS)<CR><LF>
16882	105112	047111	046103	042125	
16883	105120	047111	020107	040503	
16884	105126	046102	051505	047440	
16885	105134	020122	047503	047116	
16886	105142	041505	047524	051522	
16887	105150	006451	000012		
16888					
16889	105154	042522	042101	044440	EMS600: .ASCIZ &READ IN PRESET COMMAND &
16890	105162	020116	051120	051505	
16891	105170	052105	041440	046517	
16892	105176	040515	042116	000040	
16893	105204	043117	051506	052105	EMS601: .ASCIZ &OFFSET COMMAND &
16894	105212	041440	046517	040515	
16895	105220	042116	000040		
16896	105224	042522	052524	047122	EMS602: .ASCIZ &RETURN TO CENTER CENTER COMMAND &
16897	105232	052040	020117	042503	
16898	105240	052116	051105	041440	
16899	105246	047105	042524	020122	
16900	105254	047503	046515	047101	
16901	105262	020104	000		
16902	105265	122	046105	040505	EMS603: .ASCIZ &RELEASE COMMAND &
16903	105272	042523	041440	046517	
16904	105300	040515	042116	000040	
16905	105306	042522	040503	044514	EMS604: .ASCIZ &RECALIBRATE COMMAND &
16906	105314	051102	052101	020105	
16907	105322	047503	046515	047101	
16908	105330	020104	000		
16909	105333	123	042505	020113	EMS605: .ASCIZ &SEEK COMMAND &
16910	105340	047503	046515	047101	
16911	105346	020104	000		
16912	105351	123	040505	041522	EMS606: .ASCIZ &SEARCH COMMAND &
16913	105356	020110	047503	046515	
16914	105364	047101	020104	000	
16915	105371	104	052101	020101	EMS607: .ASCIZ &DATA COMMAND &
16916	105376	047503	046515	047101	
16917	105404	020104	000		
16918					
16919	105407	105	050130	052103	EHI: .ASCII &EXPCD RECEVD REGSTR<CR><LF>
16920	105414	020104	051040	041505	
16921	105422	053105	020104	051040	
16922	105430	043505	052123	006522	
16923	105436	012			
16924	105437	123	040524	052524	.ASCIZ &STATUS STATUS ADDRESS&
16925	105444	020123	051440	040524	
16926	105452	052524	020123	040440	

16983	106126	042126	020040	042522						
16984	106134	051507	051124	020040						
16985	106142	052040	051505	020124						
16986	106150	020040	043117	051506						
16987	106156	052105	005015							
16988	106162	052123	052101	051525	.ASCIZ	STATUS	STATUS	ADRESS	PATTRN	REGSTR
16989	106170	020040	052123	052101						
16990	106176	051525	020040	042101						
16991	106204	042522	051523	020040						
16992	106212	040520	052124	047122						
16993	106220	020040	042522	051507						
16994	106226	051124	000							
16995	106231	105	050130	052103	EH132:	.ASCII	QEXPCTD	ACTUAL	REGSTR	OFFSETO<CR><LF>
16996	106236	020104	040440	052103						
16997	106244	040525	020114	051040						
16998	106252	043505	052123	020122						
16999	106260	047440	043106	042523						
17000	106266	006524	012							
17001	106271	103	052517	052116	.ASCIZ	QCOUNT	COUNT	ADRESS	REGSTR	
17002	106276	020040	041440	052517						
17003	106304	052116	020040	040440						
17004	106312	051104	051505	020123						
17005	106320	051040	043505	052123						
17006	106326	000122								
17007	106330	054105	041520	042124	EH145:	.ASCII	QEXPCTD	ACTUAL	REGSTR	RMER1 RMER2<CR><LF>
17008	106336	020040	041501	052524						
17009	106344	046101	020040	042522						
17010	106352	051507	051124	020040						
17011	106360	046522	051105	020061						
17012	106366	020040	046522	051105						
17013	106374	006462	012							
17014	106377	103	050115	051105	.ASCIZ	QCOMPERR	COMPERR	ADRESS	PATTRN	PATTRNO
17015	106404	020122	041440	050115						
17016	106412	051105	020122	040440						
17017	106420	051104	051505	020123						
17018	106426	050040	052101	051124						
17019	106434	020116	050040	052101						
17020	106442	051124	000116							
17021										
17022	106446	054105	041520	042124	EH150:	.ASCII	QEXPCTD	ACTUAL	REGSTR	FUNCTIONO<CR><LF>
17023	106454	020040	041501	052524						
17024	106462	046101	020040	042522						
17025	106470	051507	051124	043040						
17026	106476	047125	052103	047511						
17027	106504	006516	012							
17028	106507	122	051505	046125	.ASCIZ	QRESULT	RESULT	ADRESS	CODEO	
17029	106514	020124	051040	051505						
17030	106522	046125	020124	040440						
17031	106530	051104	051505	020123						
17032	106536	020040	047503	042504						
17033	106544	000								
17034	106545	105	050130	052103	EH213:	.ASCII	QEXPCTD	ACTUAL	STATUS	TESTO<CR><LF>
17035	106552	020104	040440	052103						
17036	106560	040525	020114	051440						
17037	106566	040524	052524	020123						
17038	106574	020040	042524	052123						

```

17039 106602 005015
17040 106604 042522 052523 052114 .ASCIZ @RESULT RESULT ADRESS REGSTR@
17041 106612 020040 042522 052523
17042 106620 052114 020040 042101
17043 106626 042522 051523 020040
17044 106634 042522 051507 051124
17045 106642 000
17046 106643 101 052103 040525 EH220: .ASCII @ACTUAL REGSTR@<CR><LF>
17047 106650 020114 051040 043505
17048 106656 052123 006522 012
17049 106663 132 051505 046125 .ASCIZ @RESULT ADRESS@
17050 106670 020124 040440 051104
17051 106676 051505 000123
17052 .EVEN
17053 106702 001140 001142 001136 ED1: .WORD $GDDAT,$BDDAT,$BDADR,0
17054 106710 000000
17055 106712 001136 000000 ED2: .WORD $BDADR,0
17056 106716 001140 001142 000000 ED5: .WORD $GDDAT,$BDDAT,0
17057 106724 001174 001176 001142 ED57: .WORD $TMPO,$TMP1,$BDDAT,$BDADR,0
17058 106732 001136 000000
17059 106736 001140 001142 001174 ED65: .WORD $GDDAT,$BDDAT,$TMPO,0
17060 106744 000000
17061 106746 001140 001142 001440 ED71: .WORD $GDDAT,$BDDAT,$RMHR0,0
17062 106754 000000
17063 106756 001140 001142 001136 ED115: .WORD $GDDAT,$BDDAT,$BDADR,$TMPO,0
17064 106764 001174 000000
17065 106770 001140 001142 001136 ED130: .WORD $GDDAT,$BDDAT,$BDADR,$TMPO,$TMP1,0
17066 106776 001174 001176 000000
17067 107004 001142 001136 ED220: .WORD $BDDAT,$BDADR
17068
17069 107010 000 000 000 EF1: .BYTE 0,0,0
17070 107013 000 EF2: .BYTE 0
17071 107014 000 000 EF5: .BYTE 0,0
17072 107016 000 000 000 EF57: .BYTE 0,0,0,0
17073 107021 000
17074 107022 000 000 000 EF130: .BYTE 0,0,0,0,0
17075 107025 000 000
17076
17077 107030 .EVEN
17078 107030 BUFFER:
17079 107030 000402 BUFONE: .BLKW 258.
17080 110034 000402 BUFTWO: .BLKW 258.
17081 107030 = BUFFER
17082 107030 .HELP:
17083 107030 .ASCII <CR><LF>
17084 107032 044514 052123 047440 .ASCII @LIST OF TESTS@<CR><LF>
17085 107040 020106 042524 052123
17086 107046 006523 012
17087 107051 055 026455 026455 .ASCII @-----@<CR><LF>
17088 107056 026455 026455 026455
17089 107064 026455 005015
17090 107070 005015 .ASCII <CR><LF>
17091 107072 030524 052011 040522 .PSCII @T1 TRANSFER TEST@<CR><LF>
17092 107100 051516 042506 020122
17093 107106 042524 052123 005015
17094 107114 031124 041411 047524 .ASCII @T2 CTOD TEST@<CR><LF>

```

17095	107122	020104	042524	052123			
17096	107130	005015					
17097	107132	031524	046411	051501	.ASCII	BT3	MASSBUS INITIALIZE TEST@<CR><LF>
17098	107140	041123	051525	044440			
17099	107146	044516	044524	046101			
17100	107154	055111	020105	042524			
17101	107162	052123	005015				
17102	107166	032123	051411	042514	.ASCII	BT4	CLEAR STUCK ACTIVE TEST@<CR><LF>
17103	107174	051101	051440	052524			
17104	107202	045503	040440	052103			
17105	107210	053111	020105	042524			
17106	107216	052123	005015				
17107	107222	032524	052011	044522	.ASCII	BT5	TRISTATE TRANSFER TEST@<CR><LF>
17108	107230	052123	052101	020105			
17109	107236	051124	047101	043123			
17110	107244	051105	052040	051505			
17111	107252	006524	012				
17112	107255	124	004466	042522	.ASCII	BT6	REGISTER SELECT TEST@<CR><LF>
17113	107262	044507	052123	051105			
17114	107270	051440	046105	041505			
17115	107276	020124	042524	052123			
17116	107304	005015					
17117	107306	033524	042011	044522	.ASCII	BT7	DRIVE TYPE TEST@<CR><LF>
17118	107314	042526	052040	050131			
17119	107322	020105	042524	052123			
17120	107330	005015					
17121	107332	030524	004460	042504	.ASCII	BT10	DEVICE AVAILABLE TEST@<CR><LF>
17122	107340	044526	042503	040440			
17123	107346	040526	046111	041101			
17124	107354	042514	052040	051505			
17125	107362	006524	012				
17126	107365	124	030461	044011	.ASCII	BT11	HOLDING REGISTER TRANSFER TEST@<CR><LF>
17127	107372	046117	044504	043516			
17128	107400	051040	043505	051511			
17129	107406	042524	020122	051124			
17130	107414	047101	043123	051105			
17131	107422	052040	051505	006524			
17132	107430	012					
17133	107431	124	031061	041411	.ASCII	BT12	CONTROL STATUS #1 TRANSFER TEST@<CR><LF>
17134	107436	047117	051124	046117			
17135	107444	051440	040524	052524			
17136	107452	020123	030443	052040			
17137	107450	040522	051516	042506			
17138	107466	020122	042524	052123			
17139	107474	005015					
17140	107476	030524	004463	051105	.ASCII	BT13	ERROR REGISTER 1 TRANSFER TEST@<CR><LF>
17141	107504	047522	020122	042522			
17142	107512	044507	052123	051105			
17143	107520	030440	052040	040522			
17144	107526	051516	042506	020122			
17145	107534	042524	052123	005015			
17146	107542	030524	004464	051105	.ASCII	BT14	ERROR REGISTER 2 TRANSFER TEST@<CR><LF>
17147	107550	047522	020122	042522			
17148	107556	044507	052123	051105			
17149	107564	031040	052040	040522			
17150	107572	051516	042506	020122			

17151	107600	042524	052123	005015			
17152	107606	030524	004465	046103	.ASCII	BT15	CLEAR OFFSET STUCK ACTIVE TEST@<CR><LF>
17153	107614	040505	020122	043117			
17154	107622	051506	052105	051440			
17155	107630	052524	045503	040440			
17156	107636	052103	053111	020105			
17157	107644	042524	052123	005015			
17158	107652	030524	004466	043117	.ASCII	BT16	OFFSET REGISTER TRANSFER TEST@<CR><LF>
17159	107660	051506	052105	051040			
17160	107666	043505	051511	042524			
17161	107674	020122	051124	047101			
17162	107702	043123	051105	052040			
17163	107710	051505	006524	012			
17164	107715	124	033461	051411	.ASCII	BT17	SERIAL NUMBER TEST@<CR><LF>
17165	107722	051105	040511	020114			
17166	107730	052516	041115	051105			
17167	107736	052040	051505	006524			
17168	107744	012					
17169	107745	124	030062	041411	.ASCII	BT20	CONTROL BUS PARITY DETECTION TEST@<CR><LF>
17170	107752	047117	051124	046117			
17171	107760	041040	051525	050040			
17172	107766	051101	052111	020131			
17173	107774	042504	042524	052103			
17174	110002	047511	020116	042524			
17175	110010	052123	005015				
17176	110014	031124	004461	047503	.ASCII	BT21	CONTROL BUS PARITY GENERATION TEST@<CR><LF>
17177	110022	052116	047522	020114			
17178	110030	052502	020123	040520			
17179	110036	044522	054524	043440			
17180	110044	047105	051105	052101			
17181	110052	047511	020116	042524			
17182	110060	052123	005015				
17183	110064	031124	004462	046522	.ASCII	BT22	RMDA,RMDC FAULT TEST@<CR><LF>
17184	110072	040504	051054	042115			
17185	110100	020103	040506	046125			
17186	110106	020124	042524	052123			
17187	110114	005015					
17188	110116	031124	004463	044504	.ASCII	BT23	DISK ADDRESS TRANSFER TEST@<CR><LF>
17189	110124	045523	040440	042104			
17190	110132	042522	051523	052040			
17191	110140	040522	051516	042506			
17192	110146	020122	042524	052123			
17193	110154	005015					
17194	110156	031124	004464	042504	.ASCII	BT24	DESIRED CYLINDER TRANSFER TEST@<CR><LF>
17195	110164	044523	042522	020104			
17196	110172	054503	044514	042116			
17197	110200	051105	052040	040522			
17198	110206	051516	042506	020122			
17199	110214	042524	052123	005015			
17200	110222	031124	004465	046111	.ASCII	BT25	ILLEGAL REGISTER TEST@<CR><LF>
17201	110230	042514	040507	020114			
17202	110236	042522	044507	052123			
17203	110244	051105	052040	051505			
17204	110252	006524	012				
17205	110255	124	033062	051011	.ASCII	BT26	RESET GO BY INIT TEST@<CR><LF>
17206	110262	051505	052105	043440			

17207	110270	020117	054502	044440			
17208	110276	044516	020124	042524			
17209	110304	052123	005015				
17210	110310	031124	004467	044504	.ASCII	@T27	DIAGNOSTIC MODE TEST@<CR><LF>
17211	110316	043501	047516	052123			
17212	110324	041511	046440	042117			
17213	110332	020105	042524	052123			
17214	110340	005015					
17215	110342	031524	004460	047515	.ASCII	@T30	MOL TEST@<CR><LF>
17216	110350	020114	042524	052123			
17217	110356	005015					
17218	110360	031524	004461	051127	.ASCII	@T31	WRITE LOCK TEST@<CR><LF>
17219	110366	052111	020105	047514			
17220	110374	045503	052040	051505			
17221	110402	006524	012				
17222	110405	124	031063	042011	.ASCII	@T32	DRIVE FAULT TEST@<CR><LF>
17223	110412	044522	042526	043040			
17224	110420	052501	052114	052040			
17225	110426	051505	006524	012			
17226	110433	124	031463	051411	.ASCII	@T33	SEEK ERROR TEST@<CR><LF>
17227	110440	042505	020113	051105			
17228	110446	047522	020122	042524			
17229	110454	052123	005015				
17230	110460	031524	004464	044520	.ASCII	@T34	PIP TEST@<CR><LF>
17231	110466	020120	042524	052123			
17232	110474	005015					
17233	110476	031524	004465	041105	.ASCII	@T35	EBL TEST@<CR><LF>
17234	110504	020114	042524	052123			
17235	110512	005015					
17236	110514	031524	004466	040514	.ASCII	@T36	LAST SECTOR, LAST TRACK TEST@<CR><LF>
17237	110522	052123	051440	041505			
17238	110530	047524	026122	046040			
17239	110536	051501	020124	051124			
17240	110544	041501	020113	042524			
17241	110552	052123	005015				
17242	110556	031524	004467	046522	.ASCII	@T37	RMDA COUNT TEST@<CR><LF>
17243	110564	040504	041440	052517			
17244	110572	052116	052040	051505			
17245	110600	006524	012				
17246	110603	124	030064	051011	.ASCII	@T40	RMDC COUNT TEST@<CR><LF>
17247	110610	042115	020103	047503			
17248	110616	047125	020124	042524			
17249	110624	052123	005015				
17250	110630	032124	004461	041114	.ASCII	@T41	LBT TEST@<CR><LF>
17251	110636	020124	042524	052123			
17252	110644	005015					
17253	110646	032124	004462	047503	.ASCII	@T42	COMPOSITE ERROR TEST@<CR><LF>
17254	110654	050115	051517	052111			
17255	110662	020105	051105	047522			
17256	110670	020122	042524	052123			
17257	110676	005015					
17258	110700	032124	004463	051127	.ASCII	@T43	WRITE GO TEST@<CR><LF>
17259	110706	052111	020105	047507			
17260	110714	052040	051505	006524			
17261	110722	012					
17262	110723	124	032064	041011	.ASCII	@T44	BRANCH MULTIPLEXOR TEST@<CR><LF>

17263	110730	040522	041516	020110			
17264	110736	052515	052114	050111			
17265	110744	042514	047530	020122			
17266	110752	042524	052123	005015			
17267	110760	032124	004465	042523	.ASCII	BT45	SET/RESET GO TEST@<CR><LF>
17268	110766	027524	042522	042523			
17269	110774	020124	047507	052040			
17270	111002	051505	006524	012			
17271	111007	124	033064	042411	.ASCII	BT46	END 1 RESET GO TEST@<CR><LF>
17272	111014	042116	030440	051040			
17273	111022	051505	052105	043440			
17274	111030	020117	042524	052123			
17275	111036	005015					
17276	111040	032124	004467	042523	.ASCII	BT47	SET PULSE TEST@<CR><LF>
17277	111046	020124	052520	051514			
17278	111054	020105	042524	052123			
17279	111062	005015					
17280	111064	032524	004460	042523	.ASCII	BT50	SET/RESET IVC TEST@<CR><LF>
17281	111072	027524	042522	042523			
17282	111100	020124	053111	020103			
17283	111106	042524	052123	005015			
17284	111114	032524	004461	042523	.ASCII	BT51	SET LSC TEST@<CR><LF>
17285	111122	020124	051514	020103			
17286	111130	042524	052123	005015			
17287	111136	032524	004462	042504	.ASCII	BT52	DECODE TEST@<CR><LF>
17288	111144	047503	042504	052040			
17289	111152	051505	006524	012			
17290	111157	124	031465	051411	.ASCII	BT53	SET/RESET VOLUME VALID TEST@<CR><LF>
17291	111164	052105	051057	051505			
17292	111172	052105	053040	046117			
17293	111200	046525	020105	040526			
17294	111206	044514	020104	042524			
17295	111214	052123	005015				
17296	111220	032524	004464	046111	.ASCII	BT54	ILLEGAL FUNCTION TEST@<CR><LF>
17297	111226	042514	040507	020114			
17298	111234	052506	041516	044524			
17299	111242	047117	052040	051505			
17300	111250	006524	012				
17301	111253	124	032465	047411	.ASCII	BT55	OCCUPIED TEST@<CR><LF>
17302	111260	041503	050125	042511			
17303	111266	020104	042524	052123			
17304	111274	005015					
17305	111276	032524	004466	042522	.ASCII	BT56	READ IN PRESET TEST@<CR><LF>
17306	111304	042101	044440	020116			
17307	111312	051120	051505	052105			
17308	111320	052040	051505	006524			
17309	111326	012					
17310	111327	124	033465	051011	.ASCII	BT57	RIP/RMOF TEST@<CR><LF>
17311	111334	050111	051057	047515			
17312	111342	020106	042524	052123			
17313	111350	005015					
17314	111352	033124	004460	046522	.ASCII	BT60	RMDA/RMDC/RIP TEST@<CR><LF>
17315	111360	040504	051057	042115			
17316	111366	027503	044522	020120			
17317	111374	042524	052123	005015			
17318	111402	033124	004461	043117	.ASCII	BT61	OFFSET COMMAND TEST@<CR><LF>

17319	111410	051506	052105	041440			
17320	111416	046517	040515	042116			
17321	111424	052040	051505	006524			
17322	111432	012					
17323	111433	124	031066	051011	.ASCII	BT62	RETURN TO CENTER TEST@<CR><LF>
17324	111440	052105	051125	020116			
17325	111446	047524	041440	047105			
17326	111454	042524	020122	042524			
17327	111462	052123	005015				
17328	111466	033124	004463	046522	.ASCII	BT63	RMDC CLEAR OFFSET TEST@<CR><LF>
17329	111474	041504	041440	042514			
17330	111502	051101	047440	043106			
17331	111510	042523	020124	042524			
17332	111516	052123	005015				
17333	111522	033124	004464	041105	.ASCII	BT64	EBL CLEAR OFFSET TEST@<CR><LF>
17334	111530	020114	046103	04J505			
17335	111536	020122	043117	051506			
17336	111544	052105	052040	051505			
17337	111552	006524	012				
17338	111555	124	032466	051011	.ASCII	BT65	RUN AND GO TEST@<CR><LF>
17339	111562	047125	040440	042116			
17340	111570	043440	020117	042524			
17341	111576	052123	005015				
17342	111602	033124	004466	042523	.ASCII	BT66	SET IAE TEST@<CR><LF>
17343	111610	020124	040511	020105			
17344	111616	042524	052123	005015			
17345	111624	033124	004467	042523	.ASCII	BT67	SEARCH, SEEK, READ WRITE TEST@<CR><LF>
17346	111632	051101	044103	020054			
17347	111640	042523	045505	020054			
17348	111646	042522	042101	053440			
17349	111654	044522	042524	052040			
17350	111662	051505	006524	012			
17351	111667	124	030067	044411	.ASCII	BT70	INVALID SECTOR/TRACK TEST@<CR><LF>
17352	111674	053116	046101	042111			
17353	111702	051440	041505	047524			
17354	111710	027522	051124	041501			
17355	111716	020113	042524	052123			
17356	111724	005015					
17357	111726	033524	004461	047111	.ASCII	BT71	INVALID CYLINDER TEST@<CR><LF>
17358	111734	040526	044514	020104			
17359	111742	054503	044514	042116			
17360	111750	051105	052040	051505			
17361	111756	006524	012				
17362	111761	124	031067	051411	.ASCII	BT72	SET AOE TEST@<CR><LF>
17363	111766	052105	040440	042517			
17364	111774	052040	051505	006524			
17365	112002	012					
17366	112033	124	031467	051411	.ASCII	BT73	SET RMR TEST@<CR><LF>
17367	112010	052105	051040	051115			
17368	112016	052040	051505	006524			
17369	112024	012					
17370	112025	124	032067	050011	.ASCII	BT74	PGM STATUS CHECK@<CR><LF>
17371	112032	046507	051440	040524			
17372	112040	052524	020123	044103			
17373	112046	041505	006513	012			
17374	112053	124	032467	042011	.ASCII	BT75	DPR STATUS CHECK@<CR><LF>

H11

CZRMJ80 RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 344
CONSOLE MESSAGES

SEQ 0344

17375	112060	051120	051440	040524			
17376	112066	052524	020123	044103			
17377	112074	041505	006513	012			
17378	112101	051124	033067	050011	.ASCII	QT76	PORT REQUEST TEST, PART 10<CR><LF>
17379	112106	051117	020124	042522			
17380	112114	052521	051505	020124			
17381	112122	042524	052123	020054			
17382	112130	040520	052122	030440			
17383	112136	005015					
17384	112140	033524	004467	047520	.ASCII	QT77	PORT REQUEST TEST, PART 20<CR><LF>
17385	112146	052122	051040	050505			
17386	112154	042523	052123	052040			
17387	112162	051505	026124	050040			
17388	112170	051101	020124	006462			
17389	112176	012					
17390	112177	124	030061	004460	.ASCII	QT100	PORT REQUEST TEST, PART 30<CR><LF>
17391	112204	047520	052122	051040			
17392	112212	050505	042525	052123			
17393	112220	052040	051505	026124			
17394	112226	050040	051101	020124			
17395	112234	006463	012				
17396	112237	124	030061	004461	.ASCII	QT101	RELEASE TEST0<CR><LF>
17397	112244	042522	042514	051501			
17398	112252	020105	042524	052123			
17399	112260	005015					
17400	112262	030524	031060	053411	.ASCII	QT102	WRITE ATA TEST0<CR><LF>
17401	112270	044522	042524	040440			
17402	112276	040524	052040	051505			
17403	112304	006524	012				
17404	112307	124	030061	004463	.ASCII	QT103	RESET ATA BY GO TEST0<CR><LF>
17405	112314	042522	042523	020124			
17406	112322	052101	020101	054502			
17407	112330	043440	020117	042524			
17408	112336	052123	005015				
17409	112342	030524	032060	052411	.ASCII	QT104	UNIT READY ATA TEST0<CR><LF>
17410	112350	044516	020124	042522			
17411	112356	042101	020131	052101			
17412	112364	020101	042524	052123			
17413	112372	005015					
17414	112374	030524	032460	042411	.ASCII	QT105	ERROR ATA TEST0<CR><LF>
17415	112402	051122	051117	040440			
17416	112410	040524	052040	051505			
17417	112416	006524	012				
17418	112421	124	030061	004466	.ASCII	QT106	REGISTER TRANSFER ATA TEST0<CR><LF>
17419	112426	042522	044507	052123			
17420	112434	051105	052040	040522			
17421	112442	051516	042506	020122			
17422	112450	052101	020101	042524			
17423	112456	052123	005015				
17424	112462	030524	033460	050011	.ASCII	QT107	P SET ATA TEST0<CR><LF>
17425	112470	051440	052105	040440			
17426	112476	040524	052040	051505			
17427	112504	006524	012				
17428	112507	124	030461	004460	.ASCII	QT110	SET WLE TEST0<CR><LF>
17429	112514	042523	020124	046127			
17430	112522	020105	042524	052123			

```

17431 112530 005015
17432 112532 030524 030461 042411 .ASCII @T111 EXCEPTION TEST@<CR><LF>
17433 112540 041530 050105 044524
17434 112546 047117 052040 051505
17435 112554 006524 012
17436 112557 124 030461 004462 .ASCII @T112 SET OPI TEST@<CR><LF>
17437 112564 042523 020124 050117
17438 112572 020111 042524 052123
17439 112600 005015
17440 112602 030524 031461 051011 .ASCII @T113 RECALIBRATE TEST@<CR><LF>
17441 112610 041505 046101 041111
17442 112616 040522 042524 052040
17443 112624 051505 006524 012
17444 112631 124 030461 004464 .ASCII @T114 SEEK TEST@<CR><LF>
17445 112636 042523 045505 052040
17446 112644 051505 006524 012
17447 112651 124 030461 004465 .ASCII @T115 SEARCH TEST@<CR><LF>
17448 112656 042523 051101 044103
17449 112664 052040 051505 006524
17450 112672 012
17451 112673 124 030461 004466 .ASCII @T116 SEARCH TIMEOUT TEST@<CR><LF>
17452 112700 042523 051101 044103
17453 112706 052040 046511 047505
17454 112714 052125 052040 051505
17455 112722 006524 012
17456 112725 124 030461 004467 .ASCII @T117 DATA COMMAND TESTS (1)@<CR><LF>
17457 112732 040504 040524 041440
17458 112740 046517 040515 042116
17459 112746 052040 051505 051524
17460 112754 024040 024461 005015
17461 112762 030524 030062 042011 .ASCII @T120 DATA COMMAND TESTS (2)@<CR><LF>
17462 112770 052101 020101 047503
17463 112776 046515 047101 020104
17464 113004 042524 052123 020123
17465 113012 031050 006451 012
17466 113017 124 031061 004461 .ASCII @T12. DATA COMMAND TESTS (3)@<CR><LF>
17467 113024 040504 040524 041440
17468 113032 046517 040515 042116
17469 113040 052040 051505 051524
17470 113046 031450 006451 012
17471 113053 015 012 .ASCII <CR><LF>
17472 113055 015 012 .ASCII <CR><LF>
17473 113057 123 044527 041524 .ASCII @SWITCH USE@<CR><LF>
17474 113064 004510 004411 051525
17475 113072 006505 012
17476 113075 055 026455 026455 .ASCII @----- -----@<CR><LF>
17477 113102 004455 026411 026455
17478 113110 026455 026455 026455
17479 113116 026455 026455 026455
17480 113124 026455 026455 006455
17481 113132 012
17482 113133 040 030440 004465 .ASCII @ 15 HALT ON ERROR@<CR><LF>
17483 113140 044011 046101 020124
17484 113146 047117 042440 051122
17485 113154 051117 005015
17486 113160 020040 032061 04411 .ASCII @ 14 LOOP ON TEST@<CR><LF>

```

17487	113166	047514	050117	047440					
17488	113174	020116	042524	052123					
17489	113202	005015							
17490	113204	020040	031461	004411	.ASCII	a	13		INHIBIT ERROR TYPEOUTS<CR><LF>
17491	113212	047111	044510	044502					
17492	113220	020124	051105	047522					
17493	113226	020122	054524	042520					
17494	113234	052517	051524	005015					
17495	113242	020040	030461	004411	.ASCII	a	11		INHIBIT ITERATIONS<CR><LF>
17496	113250	047111	044510	044502					
17497	113256	020124	052111	051105					
17498	113264	052101	047511	051516					
17499	113272	005015							
17500	113274	020040	030061	004411	.ASCII	a	10		BELL ON ERROR<CR><LF>
17501	113302	042502	046114	047440					
17502	113310	020116	051105	047522					
17503	113316	006522	012						
17504	113321	040	020040	004471	.ASCII	a	9		LOOP ON ERROR<CR><LF>
17505	113326	046011	047517	020120					
17506	113334	047117	042440	051122					
17507	113342	051117	005015						
17508	113346	020040	034040	004411	.ASCII	a	8		LOOP ON TEST IN SWR<7:0><CR><LF>
17509	113354	047514	050117	047440					
17510	113362	020116	042524	052123					
17511	113370	044440	020116	053523					
17512	113376	036122	035067	037060					
17513	113404	005015							
17514	113406	020040	033440	004411	.ASCII	a	7		TN128<CR><LF>
17515	113414	047124	031061	006470					
17516	113422	012							
17517	113423	040	020040	004466	.ASCII	a	6		TN64<CR><LF>
17518	113430	052011	033116	006464					
17519	113436	012							
17520	113437	040	020040	004465	.ASCII	a	5		TN32<CR><LF>
17521	113444	052011	031516	006462					
17522	113452	012							
17523	113453	040	020040	004464	.ASCII	a	4		TN16<CR><LF>
17524	113460	052011	030516	006466					
17525	113466	012							
17526	113467	040	020040	004463	.ASCII	a	3		TN8<CR><LF>
17527	113474	052011	034116	005015					
17528	113502	020040	031040	004411	.ASCII	a	2		TN4<CR><LF>
17529	113510	047124	006464	012					
17530	113515	040	020040	004461	.ASCII	a	1		TN2<CR><LF>
17531	113522	052011	031116	005015					
17532	113530	020040	030040	004411	.ASCII	a	0		TN1<CR><LF>
17533	113536	047124	006461	012					
17534	113543	015	000012		.ASCIZ				<CR><LF>
17535		000001			.END				

AUNIT = 000000	3227	3234												
AUSWR = 000000	3227	3241												
AVECT1 = 100254	3120*	3227	3266											
AVECT2 = 000000	3227	3267												
A16 = 000400	3069*	7026												
A17 = 001000	3068*	7026												
BAI = 000010	3088*													
BB00 = 000001	3005*													
BB01 = 000002	3004*													
BB02 = 000004	3003*	13566												
BB03 = 000010	3002*	13562	13565											
BB04 = 000020	3001*													
BB05 = 000040	3000*													
BB06 = 000100	2999*	11434	11435	11436	11437	11438	11439	11440	11441	11442	11443			
BB07 = 000200	2998*													
BB08 = 000400	2997*													
BB09 = 001000	2996*													
BIT0 = 000001	2792*	5029	5567	5996	5998	6070	6160	6264	6272	6362	6369	6376	6393	
	6400	6416	6521	6528	6544	6809	6877	6889	7117	7185	7271	7356	7483	
	7569	7643	7657	8704										
BIT00 = 000001	2782*	2792	2818	2867	2886	2905	2943	2962	3005	3092	3108			
BIT01 = 000002	2781*	2791	2817	2866	2904	2942	2961	3004	3091	3107				
BIT02 = 000004	2780*	2790	2816	2865	2903	2941	2960	3003	3090	3106				
BIT03 = 000010	2779*	2789	2815	2864	2902	2940	2959	3002	3017	3088	3105			
BIT04 = 000020	2778*	2788	2814	2863	2901	2958	3001	3087						
BIT05 = 000040	2777*	2787	2813	2900	2939	2957	3000	3086						
BIT06 = 000100	2776*	2786	2885	2899	2921	2938	2956	2999	3071	3085	3104			
BIT07 = 000200	2775*	2785	2884	2898	2920	2937	2955	2979	2998	3016	3070	3084		
BIT08 = 000400	2774*	2784	2862	2883	2897	2919	2936	2954	2997	3069	3082	14388		
BIT09 = 001000	2773*	2783	2861	2882	2896	2918	2935	2953	2996	3068	3081	14406	14556	
BIT1 = 000002	2791*	5624	6085	6176	6296	6425	6553	6824	6903	7130	7136	7198	7284	
	7389	7498	7580	7670										
BIT10 = 002000	2772*	2860	2881	2895	2917	2934	2952	2978	2995	3015	3067	3080	3103	
	14533													
BIT11 = 004000	2771*	2812	2880	2894	2933	2951	2969	2977	2994	3014	3079	3102	14413	
BIT12 = 010000	2770*	2879	2893	2932	2950	2976	2993	3013	3078	3101				
BIT13 = 020000	2769*	2878	2892	2931	2949	2968	2992	3012	3065	3077	3100	14540		
BIT14 = 040000	2768*	2877	2891	2930	2948	2967	2991	3011	3023	3064	3076	3099	14374	
BIT15 = 100000	2767*	2876	2890	2929	2947	2966	2990	3010	3022	3063	3075	3098		
BIT2 = 000004	2790*													
BIT3 = 000010	2789*													
BIT4 = 000020	2788*													
BIT5 = 000040	2787*													
BIT6 = 000100	2786*													
BIT7 = 000200	2785*	5024												
BIT8 = 000400	2784*													
BIT9 = 001000	2783*													
BOAOR 060202	13806*	13824*	13827	13842	13887*									
BOFLG 060204	13792*	13834*	13837	13840*	13888*									
BPTVEC = 000014	2799*													
BSE = 100000	3010*													
BUFFER 107030	10136	13450	17078*	17081										
BUFONE 107030	17079*													
BUFTWO 110034	17080*													
CC = 004000	2994*	11434	11435	11436	11437	11438	11439	11440	11441	11442	11443	11823	11824	
	11825	11826	11827	11828	11829	11830	11831	11832	12330	12331	12332	12333	12334	

M11

CZRMJBO RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 350
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0349

	12335	12336	12337	12338	12339	12692	13559	13627	13628	13629	13630	13631	13632
CM = 002000	13633	13634	13635	13636	13670	13671	13672	13673	13674	13675	13676	12886	12887
	2995#	11434	11435	11436	11437	11438	11439	11440	11441	11442	11443	13639	13641
	12888	12889	12890	12891	12892	12893	12894	12895	13557	13638	13639	13640	13641
CHRCNT = 060205	13642	13643	13644	13645	13646	13647	13670	13671	13672	13673	13674	13675	13676
CKSWR = 104410	13793*	13811*	13817*	13818	13821*	13825	13830*	13841*	13889#				
CLOCK = 001526	14373	14529	14555	14906#									
	3361#	8906	9727	11001	12432	12726	12810	12961	13019	13131	13214	13326	13504
CLR = 000040	13913*	13926*	13937*										
	3086#	5031	5279	5300	5362	5415	5425	5463	5505	5570	5631	5753	5776
	5808	5840	5864	5896	5919	5941	6024	6051	6129	6135	6222	6304	6352
	6434	6473	6595	6641	6714	6751	6794	6863	6950	7060	7066	7097	7171
	7261	7337	7466	7552	7636	7719	7819	7876	7943	7964	8030	8088	8122
	8148	8198	8251	8394	8568	8663	8819	8885	9042	9116	9317	9384	9435
	9712	10188	10261	10295	10533	10589	10630	10685	10727	10963	11057	14009	
CLSPRN = 065260	5102	5118	5139	15015#									
CMNSTA = 006162	5054	5189#											
CNSL00 = 065424	5064	15037#											
CNSL01 = 065463	5099	15043#											
CNSL02 = 065510	5108	15047#											
CNSL03 = 065571	5113	15056#											
CNSL04 = 065621	5124	15061#											
CNSL05 = 065675	5134	15070#											
CNSL06 = 065731	5145	15075#											
CNSL07 = 065756	5158	15079#											
CONT = 000100	2956#	8676	8702	8706									
CP = 000015	2707#	5172	13809	14348	14358	15017	15020	15026	15037	15043	15047	15051	15056
	15061	15065	15070	15075	15080	15090	15097	16652	16810	16815	16821	16827	16831
	16838	16842	16846	16854	16859	16869	16877	16881	16919	16928	16932	16941	16953
	16961	16969	16981	16995	17007	17022	17034	17046	17083	17084	17087	17090	17091
	17094	17097	17102	17107	17112	17117	17121	17126	17133	17140	17146	17152	17158
	17164	17169	17176	17183	17188	17194	17200	17205	17210	17215	17218	17222	17226
	17230	17233	17236	17242	17246	17250	17253	17258	17262	17267	17271	17276	17280
	17284	17287	17290	17296	17301	17305	17310	17314	17318	17323	17328	17333	17338
	17342	17345	17351	17357	17362	17366	17370	17374	17378	17384	17390	17396	17400
	17404	17409	17414	17418	17424	17428	17432	17436	17440	17444	17447	17451	17456
	17461	17466	17471	17472	17473	17476	17482	17486	17490	17495	17500	17504	17508
	17514	17517	17520	17523	17526	17528	17530	17532	17534				
CRLF = 000200	2708#	5013	14319	14358									
CYLMASK = 001777	2984#	5370											
DBCK = 100000	2929#	8286	8441	8599	8697	8844	8950	8977	9004	9143	9210	9271	9340
	9402	9455	9516	9575	9798	9894	9961	10063	10146	10490	10809	10877	10880
	10992	11078	11139	11150	11191	11203	11241	11281	11295	11335	11347	11368	11422
	11494	11505	11551	11563	11606	11651	11665	11710	11722	11743	11804	11883	11894
	11940	11952	11998	12044	12061	12078	12123	12135	12156	12199	12216	12229	12247
	12308	12388	12402	12412	12492	12504	12547	12559	12590	12622	12664	12684	12749
	12761	12838	12862	12981	13039	13051	13060	13072	13151	13166	13234	13249	13261
	13273	13346	13359	13369	13392	13533	13580	13607					
DBEM = 040000	2930#	7833	7840	7842	7885	7890	7892	7973	7978	7980	8060	8062	8203
	8256	8286	8288	8399	8441	8443	8573	8599	8601	8668	8697	8699	8824
	8844	8846	8890	8950	8952	8977	8979	9004	9006	9047	9121	9143	9145
	9203	9210	9212	9262	9271	9273	9322	9340	9342	9389	9402	9404	9440
	9455	9457	9510	9516	9518	9569	9575	9577	9676	9680	9682	9717	9787
	9798	9800	9885	9894	9896	9955	9961	9963	10057	10063	10065	10128	10146
	10148	10153	10155	10193	10203	10484	10490	10492	10598	10801	10809	10811	10868
	10986	10992	10994	11070	11078	11080	11128	11139	11141	11146	11150	11152	11180

N11

CZRMJBC RM03/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 351
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0350

11191	11193	11198	11203	11205	11230	11241	11243	11270	11281	11283	11288	11290
11295	11297	11324	11335	11337	11342	11347	11349	11363	11368	11370	11395	11422
11424	11479	11494	11436	11501	11505	11507	11536	11551	11553	11558	11563	11565
11591	11606	11608	11635	11651	11653	11658	11660	11665	11667	11695	11710	11712
11717	11722	11724	11738	11743	11745	11772	11804	11806	11868	11883	11885	11890
11894	11896	11925	11940	11942	11947	11952	11954	11983	11998	12000	12029	12044
12046	12053	12056	12061	12063	12069	12071	12073	12078	12080	12108	12123	12125
12130	12135	12137	12151	12156	12158	12184	12199	12201	12208	12211	12216	12218
12229	12231	12242	12247	12249	12276	12308	12310	12373	12388	12390	12395	12397
12402	12404	12412	12414	12430	12477	12492	12494	12499	12504	12506	12532	12547
12549	12554	12559	12561	12570	12572	12590	12592	12601	12603	12622	12624	12649
12664	12666	12684	12686	12713	12749	12751	12756	12761	12763	12772	12774	12805
12838	12840	12845	12847	12862	12864	12948	12981	12983	13006	13039	13041	13046
13051	13053	13060	13062	13067	13072	13074	13083	13085	13122	13151	13153	13159
13161	13166	13168	13205	13234	13236	13242	13244	13249	13251	13256	13261	13263
13268	13273	13275	13284	13286	13313	13346	13348	13354	13359	13361	13369	13371
13387	13392	13394	13403	13405	13495	13533	13535	13547	13549	13580	13582	13595
13597	13607	13609										
3103#												
2890#	2907	6235	6266									
2714#	3203	4973										
2931#	7649	7663	7840	7890	7978	8060	9680	10153				
3203#	4973*	4981*	14428*	14532*								
3130#	4981											
3075#												
2943#	2962#	5471	5483	7101	7108	7113	7123	7126	7128	7144	7178	7189
7191	7214	7231	7233	7236	7265	7275	7277	7294	7311	7313	7316	7346
7379	7381	7420	7440	7442	7445	7474	7483	7492	7508	7526	7528	7531
7560	7562	7573	7589	7605	7607	7610	7647	7649	7651	7661	7663	7679
7694	7696	7699	7822	7833	7840	7842	7883	7885	7890	7892	7971	7973
7978	7980	8052	8060	8062	8095	8125	8151	8201	8203	8254	8256	8286
8288	8397	8399	8441	8443	8571	8573	8599	8601	8666	8666	8697	8699
8822	8824	8844	8846	8886	8890	8912	8950	8952	8977	8979	9004	9006
9045	9047	9119	9121	9143	9145	9203	9210	9212	9262	9271	9273	9320
9322	9340	9342	9387	9389	9402	9404	9438	9440	9455	9457	9510	9516
9518	9569	9575	9577	9676	9680	9682	9715	9717	9787	9798	9800	9885
9894	9896	9955	9961	9963	10057	10063	10065	10128	10146	10148	10153	10155
10191	10193	10203	10484	10490	10492	10536	10542	10592	10598	10633	10642	10656
10688	10730	10801	10809	10811	10868	10986	10992	10994	11068	11070	11078	11080
11128	11139	11141	11146	11150	11152	11180	11191	11193	11198	11203	11205	11230
11241	11243	11270	11281	11283	11288	11290	11295	11297	11324	11335	11337	11342
11347	11349	11363	11368	11370	11395	11422	11424	11479	11494	11496	11501	11505
11507	11536	11551	11553	11558	11563	11565	11591	11606	11608	11636	11651	11653
11658	11660	11665	11667	11695	11710	11712	11717	11722	11724	11738	11743	11745
11772	11804	11806	11868	11883	11885	11890	11894	11896	11925	11940	11942	11947
11952	11954	11983	11998	12000	12029	12044	12046	12053	12056	12061	12063	12069
12071	12073	12078	12080	12108	12123	12125	12130	12135	12137	12151	12156	12158
12184	12199	12201	12208	12211	12216	12218	12229	12231	12242	12247	12249	12276
12308	12310	12373	12388	12390	12395	12397	12402	12404	12412	12414	12430	12477
12492	12494	12499	12504	12506	12532	12547	12549	12554	12559	12561	12570	12572
12590	12592	12601	12603	12622	12624	12649	12664	12666	12684	12686	12713	12749
12751	12756	12761	12763	12772	12774	12805	12838	12840	12845	12847	12862	12864
12948	12981	12983	13006	13039	13041	13046	13051	13053	13060	13062	13067	13072
13074	13083	13085	13122	13151	13153	13159	13161	13166	13168	13205	13234	13236
13242	13244	13249	13251	13256	13261	13263	13268	13273	13275	13284	13286	13313
13346	13348	13354	13359	13361	13369	13371	13387	13392	13394	13403	13405	13495

DBL = 002000
DCK = 100000
DDISP = 177570
DEBL = 020000
DISPLA = 001156
DISPRE = 000174
DLT = 100000
DMD = 000001

		13533	13535	13547	13549	13580	13582	13595	13597	13607	13609	14012	14014	14047
DPE	= 000010	3017#	6373	6397	6651									
DPEH	= 040000	3099#												
DPELO	= 020000	3100#												
DPR	= 000400	2883#	10308	10310										
DRQ	= 004000	2969#	10268	10502										
DRVCLR	= 000010	2827#	8321	8744										
DRY	= 000200	2884#	8425	8429	8464	8471								
DSWR	= 177570	2713#	3202	4972										
DTE	= 010000	2893#	2907	6243	6275									
DTO	= 010000	2932#	12056	12061	12063	12069	12071	12073	12078	12080	12211	12216	12218	12229
		12231	12242	12247	12249	12402	12404	12412	12414					
DULPRT	= 024024	2972#	5994	5998	6001									
DVA	= 004000	2812#	5037	6028	6030	10299	10301							
DVC	= 000200	3016#	5481	6443	6445	7353	7377	7384	7427	7431				
EBL	= 020000	2949#	7640	7654	7666	7668	7684	7686	12575	12580	12610	12777	12782	13088
		13093	13289	13294	13409	13414								
ECH	= 000100	2899#	2907	6235	6266	15126	15127	15134	15135					
ECL	= 004000	2977#	5372	6549										
EJRC	= 001000	2953#												
EDT1	074652	3388	3509	3517	3525	3533	3541	3549	3565	3573	3581	3589	3597	3605
		3613	3621	3629	3637	3645	3661	3669	3677	3685	3693	3701	3709	3717
		3725	3733	3741	3749	3765	3773	3781	3789	3797	3821	3829	3869	3877
		3885	3901	3909	3925	3933	3949	3957	3965	3973	3981	3989	4005	4013
		4029	4037	4053	4061	4069	4117	4125	4133	4141	4157	4165	4173	4181
		4197	4205	4221	4229	4237	4245	4253	4269	4285	4293	4301	4309	4325
		4333	4357	4381	4389	4397	4405	4413	4421	4429	4437	4485	4509	4517
		4525	4565	4573	4581	4589	4597	4605	4638	4654	4670	4678	4687	4695
		4703	4711	4719	4728	4736	4745	4753	4761	4769	4777	4785	4793	4802
		4810	4818	4826	4834	4842	4850	4858	4866	4874	4882	4890	4898	4906
		4915	4923	4931	15070#									
EDT115	074670	3997	4021	4045	4077	4213	4261	4277	4341	4349	4365	4445	4461	4469
		4477	4493	4501	4614	4622	4630	4646	4662	16077#				
EDT130	074672	4085	4093	4189	16078#									
EDT132	074674	4101	4109	4149	16079#									
EDT2	074654	3396	16071#											
EDT220	074676	4533	4541	4549	4557	16080#								
EDT5	074656	3420	3428	3437	16072#									
EDT57	074660	3757	16073#											
EDT65	074662	3805	3813	3893	3917	3941	16074#							
EDT71	074664	3837	3845	3853	16075#									
EDT74	074666	3861	16076#											
ED1	106702	16070	17053#											
ED115	106756	16077	16079	17063#										
ED130	106770	16078	17065#											
ED2	106712	16071	17076	17055#										
ED220	107004	16080	17067#											
ED5	106716	16072	17056#											
ED57	106724	16073	17057#											
ED65	106736	16074	17059#											
ED71	106746	16075	17061#											
EECC	= 000020	2958#												
EFT1	074700	3389	3510	3518	3526	3534	3542	3550	3566	3574	3582	3590	3598	3606
		3614	3622	3630	3638	3646	3662	3670	3678	3686	3694	3702	3710	3718
		3726	3734	3742	3750	3766	3774	3782	3790	3798	3822	3830	3870	3878
		3886	3902	3910	3926	3934	3950	3958	3966	3974	3982	3990	4006	4014

EM220	106643	16068	17046#																
EM3	105475	16061	16930#																
EM5	105504	16056	16932#																
EM57	105562	16058	16941#																
EM65	105661	16059	16953#																
EM7	105543	16057	16938#																
EM71	105736	16060	16961#																
EM51	074726	15291	15292	16094#															
EM510	075331	15468	15473	16140#															
EM511	075374	15478	16146#																
EM512	075465	15450	15453	15456	16156#														
EM513	075543	15486	15489	15495	15495	16164#													
EM514	075612	15499	15504	15512	16171#														
EM515	075657	15499	15504	15509	15527	15859	15863	15901	15906	15938	15970	16009	16178#						
EM516	075735	15514	15519	15524	15879	16186#													
EM517	076016	15514	15519	15527	16195#														
EM52	074777	15299	15591	15634	15685	15693	15702	15724	15885	15896	16101#								
EM520	076057	15529	15534	15538	15547	15547	15552	15559	16201#										
EM521	076123	15529	15534	15542	15547	15552	15556	15911	15943	15975	16014	16208#							
EM522	076202	15534	15538	15547	15552	16216#													
EM523	076247	15561	15566	15574	16223#														
EM524	076325	15561	15566	15571	15929	15961	15993	16045	16231#										
EM525	076404	15576	15581	15589	16239#														
EM5250	101337	15337	15340	15343	15385	15608	15612	15616	15620	15748	15757	15791	15800	15838					
		16511#																	
EM5251	101375	15345	15348	15351	15811	15830	16517#												
EM5252	101441	15353	15358	15360	15363	15366	15368	15371	15374	15376	15652	15656	15890	16524#					
EM5253	101474	15353	15391	15394	15398	15402	15406	15410	15414	15652	15656	16529#							
EM5254	101527	15353	15495	15509	15524	15556	15571	15586	15603	16534#									
EM5255	101570	15378	15380	15383	15385	15624	15638	15748	15761	15777	15791	15804	16540#						
EM5256	101632	15417	15420	15424	15428	15748	15753	16546#											
EM5257	101662	15431	16550#																
EM526	076462	15576	15581	15586	15916	15948	15980	16040	16247#										
EM5260	101716	15439	15443	15447	15478	16555#													
EM5261	101747	15631	16560#																
EM5262	102006	15811	15834	15846	15851	16566#													
EM527	076541	15591	15595	15599	15606	15782	16017	16019	16023	16048	16255#								
EM53	075046	15389	15667	15689	15697	15737	15744	15787	15793	15797	15879	16108#							
EM530	076605	15597	15599	15603	15616	15620	15638	15645	16262#										
EM5300	102051	15291	15299	15358	15360	15363	15389	15394	15398	15450	15459	15463	15486	15591					
		15634	15685	15693	15702	15724	15748	15753	15757	15761	15885	15896	16573#						
EM5301	102067	15292	15296	15303	15307	15310	15337	15340	15343	15345	15348	15351	15353	15366					
		15368	15371	15374	15376	15378	15380	15383	15385	15402	15406	15410	15414	15420					
		15424	15428	15439	15443	15447	15478	15495	15509	15524	15556	15571	15586	15603					
		15660	15740	16576#															
EM5302	102112	15292	15296	15310	16580#														
EM5303	102137	15292	15478	16584#															
EM5304	102150	15292	16586#																
EM5306	102157	15296	15307	15353	15660	15740	15824	15827	16588#										
EM5307	102173	15299	15591	15634	15685	15693	15702	15724	15885	15896	16591#								
EM531	076665	15608	16271#																
EM5310	102234	15303	15649	15851	16597#														
EM5311	102251	15303	15307	16600#															
EM5312	102315	15306	15339	15342	15347	15350	15379	15382	15404	15408	15412	15422	15426	15441					
		15445	16607#																
EM5313	102352	15310	15343	15351	15376	15383	15414	15428	15447	15495	15509	15524	15556	15571					

EAT112	070754	3971	15542#
EAT113	070774	3979	15547#
EAT114	071020	3987	15552#
EAT115	071042	3995	15556#
EAT116	071066	4003	15561#
EAT117	071106	4011	15566#
EAT12	067122	3459	15319#
EAT120	071122	4019	15571#
EAT121	071152	4027	15576#
EAT122	071172	4035	15581#
EAT123	071212	4043	15586#
EAT124	071236	4051	15591#
EAT125	071254	4059	15595#
EAT126	071274	4067	15599#
EAT127	071312	4075	15603#
EAT13	067140	3467	15322#
EAT130	071336	4083	15608#
EAT131	071354	4091	15612#
EAT132	071372	4099	15616#
EAT133	071412	4107	15620#
EAT134	071432	4115	15624#
EAT135	071446	4123	15627#
EAT136	071464	4131	15631#
EAT137	071476	4139	15634#
EAT14	067156	3475	15325#
EAT140	071514	4147	15638#
EAT141	071534	4155	15642#
EAT142	071550	4163	15645#
EAT143	071566	4171	15649#
EAT144	071602	4179	15652#
EAT145	071624	4187	15656#
EAT146	071646	4195	15660#
EAT147	071664	4203	15664#
EAT15	067174	3483	15328#
EAT150	071676	4211	15667#
EAT151	071720	4219	15671#
EAT152	071732	4227	15674#
EAT153	071746	4235	15677#
EAT154	071766	4243	15681#
EAT155	072006	4251	15685#
EAT156	072024	4259	15689#
EAT157	072046	4267	15693#
EAT16	067212	3491	15331#
EAT160	072064	4275	15697#
EAT161	072114	4283	15702#
EAT162	072132	4291	15706#
EAT163	072150	4299	15710#
EAT164	072172	4307	15714#
EAT165	072214	4315	15718#
EAT166	072240	4323	15724#
EAT167	072260	4331	15729#
EAT17	067230	3499	15334#
EAT170	072302	4339	15734#
EAT171	072314	4347	15737#
EAT172	072330	4355	15740#
EAT173	072346	4363	15744#

EMT174	072370	4371	15748#
EMT175	072416	4379	15753#
EMT176	072434	4387	15757#
EMT177	072452	4395	15761#
EMT2	066706	3394	15292#
EMT20	067246	3507	15337#
EMT200	072470	4403	15765#
EMT201	072506	4411	15769#
EMT202	072524	4419	15773#
EMT203	072542	4427	15777#
EMT204	072564	4435	15782#
EMT205	072604	4443	15787#
EMT206	072624	4451	15791#
EMT207	072654	4459	15797#
EMT21	067266	3515	15340#
EMT210	072670	4467	15800#
EMT211	072706	4475	15804#
EMT212	072724	4483	15808#
EMT213	072736	4491	15811#
EMT214	072766	4499	15817#
EMT215	073000	4507	15820#
EMT216	073020	4515	15824#
EMT217	073034	4523	15827#
EMT22	067306	3523	15343#
EMT220	073050	4531	15830#
EMT221	073066	4539	15834#
EMT222	073104	4547	15838#
EMT223	073122	4555	15842#
EMT224	073140	4563	15846#
EMT225	073160	4571	15851#
EMT226	073202	4579	15855#
EMT227	073220	4587	15859#
EMT23	067322	3531	15345#
EMT230	073236	4595	15863#
EMT231	073254	4603	15867#
EMT232	073270	4612	15870#
EMT233	073312	4620	15874#
EMT234	073334	4628	15879#
EMT235	073362	4636	15885#
EMT236	073402	4644	15890#
EMT237	073434	4652	4660
EMT24	067342	3539	15348#
EMT240	073454	15901#	
EMT241	073500	4668	15906#
EMT242	073524	4676	15911#
EMT243	073550	4685	15916#
EMT244	073574	4693	15921#
EMT245	073612	4701	15925#
EMT246	073630	4709	15929#
EMT247	073654	4717	15934#
EMT25	067362	3547	15351#
EMT250	073672	4726	15938#
EMT251	073716	4734	15943#
EMT252	073742	4743	15948#
EMT253	073766	4751	15953#
EMT254	074004	4759	15957#

15896#

EMT255	074022	4767	15961#
EMT256	074046	4775	15966#
EMT257	074064	4783	15970#
EMT26	067376	3555	15353#
EMT260	074110	4791	15975#
EMT261	074134	4800	15980#
EMT262	074160	4808	15985#
EMT263	074176	4816	15989#
EMT264	074214	4824	15993#
EMT265	074240	4832	15998#
EMT266	074256	4840	16002#
EMT267	074274	4848	16006#
EMT27	067430	3563	15358#
EMT270	074310	4856	16009#
EMT271	074334	4864	16014#
EMT272	074360	4872	16019#
EMT273	074376	4880	16023#
EMT274	074414	4888	16027#
EMT275	074432	4896	16031#
EMT276	074452	4904	16036#
EMT277	074470	4913	16040#
EMT3	066734	3402	15296#
EMT30	067442	3571	15360#
EMT300	074514	4921	16045#
EMT301	074540	4929	16050#
EMT31	067456	3579	15363#
EMT32	067472	3587	15366#
EMT33	067506	3595	15368#
EMT34	067524	3603	15371#
EMT35	067542	3611	15374#
EMT36	067556	3619	15376#
EMT37	067572	3627	15378#
EMT4	066754	3410	15299#
EMT40	067606	3635	15380#
EMT41	067626	3643	15383#
EMT42	067642	3651	15385#
EMT43	067666	3659	15389#
EMT44	067700	3667	15391#
EMT45	067714	3675	15394#
EMT46	067730	3683	15398#
EMT47	067744	3691	15402#
EMT5	066776	3418	15303#
EMT50	067766	3699	15406#
EMT51	070010	3707	15410#
EMT52	070030	3715	15414#
EMT53	070044	3723	15417#
EMT54	070060	3731	15420#
EMT55	070100	3739	15424#
EMT56	070120	3747	15428#
EMT57	070134	3755	15431#
EMT5	067022	3426	15307#
EMT60	070150	3763	15435#
EMT61	070164	3771	15439#
EMT62	070204	3779	15443#
EMT63	070224	3787	15447#
EMT64	070240	3795	15450#

ILF = 000001	2905#	6227	6258	9084	9217	9219	15107	15116	15117	15119	15120	15121	15122
	15123	15124	15125	15128	15129	15132	15133	15136	15137				
ILF02 = 000002	2824#	8488											
ILF24 = 000024	2834#	8503	8732	9081									
ILF26 = 000026	2835#	8506	8768										
ILF30 = 000030	2837#		8772										
ILF32 = 000032	2838#	8361	8635	8780									
ILF34 = 000034	2839#	8509	8784										
ILF36 = 000036	2840#	8512	8788										
ILF40 = 000040	2841#												
ILF42 = 000042	2842#	8515											
ILF44 = 000044	2843#	8518											
ILF46 = 000046	2844#	8521											
ILF54 = 000054	2847#	8524											
ILF56 = 000056	2848#	8527											
ILF64 = 000064	2851#	8530											
ILF66 = 000066	2852#	8533											
ILF74 = 000074	2855#	8536											
ILF76 = 000076	2856#	5366	5387	5419	5435	5509	5546	5582	5607	5847	5854	6134	6139
	6149	6154	6166	6169	6170	6184	6191	8225	8406	8539	8580	8859	9135
	9229	9294	9746	9789	9915	11094							
	2904#	6227	6258	6954	6992	6996	7024						
ILR = 000002	3043#												
ILRG50 = 000050	3044#												
ILRG52 = 000052	3045#												
ILRG54 = 000054	3046#												
ILRG56 = 000056	3047#												
ILRG60 = 000060	3048#												
ILRG62 = 000062	3049#												
ILRG64 = 000064	3050#												
ILRG66 = 000066	3051#												
ILRG70 = 000070	3052#												
ILRG72 = 000072	3053#												
ILRG74 = 000074	3054#												
ILRG76 = 000076	2800#	4954*	4955*										
IOTVEC = 000020	3108#												
IPCK0 = 000001	3107#												
IPCK1 = 000002	3106#												
IPCK2 = 000004	3105#												
IPCK3 = 000010	3085#	5287	5308										
IP = 000100	3013#	6366	6390	8831	8849	8851	15107	15108	15109	15111	15112	15113	15116
IVC = 010000	15117	15118	15119	15120	15121	15122	15123	15124	15125	15126	15127	15128	15129
	15130	15131	15132	15133	15134	15135	15136	15137					
	3015#	6373	6397										
LBC = 002000	2881#	8040	8065	8067									
_BT = 002000	13926	13955#											
LC_LOCK = 060522	13928	13973#											
LCJUNT = 060572	2706#	13813	14352	14358	15017	15020	15026	15037	15043	15047	15051	15056	15061
LF = 000012	15065	15070	15075	15080	15090	15097	16652	16810	16815	16821	16827	16831	16838
	16842	16846	16854	16859	16869	16877	16881	16919	16928	16932	16941	16953	16961
	16969	16981	16995	17007	17022	17034	17046	17083	17084	17087	17090	17091	17094
	17097	17102	17107	17112	17117	17121	17126	17133	17140	17146	17152	17158	17164
	17169	17176	17183	17188	17194	17200	17205	17210	17215	17218	17222	17226	17230
	17233	17236	17242	17246	17250	17253	17258	17262	17267	17271	17276	17280	17284
	17287	17290	17296	17301	17305	17310	17314	17318	17323	17328	17333	17338	17342
	17345	17351	17357	17362	17366	17370	17374	17378	17384	17390	17396	17400	17404

M12

CZRMJBO RMO3/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 363
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0362

		17409	17414	17418	17424	17428	17432	17436	17440	17444	17447	17451	17456	17461
		17466	17471	17472	17473	17476	17482	17486	17490	17495	17500	17504	17508	17514
		17517	17520	17523	17526	17528	17530	17532	17534					
LS	= 000004	2960	7742	7748	7752									
LSC	= 004000	3014	6366	6390	8897	8915	8917							
LST	= 000002	2961	7760	7766	7770									
LSTOP	= 060634	13927	13987											
MCLK	= 004000	2933	12570	12601	12772	13083	13284	13403						
MCPE	= 020000	3065	6723											
MOF	= 000100	2938	7381	7429	11198	11203	11205	11558	11563	11565	11947	11952	11954	12242
		12247	12249	12554	12559	12561	12570	12572	12590	12592	12601	12603	12622	12624
		12756	12761	12763	12772	12774	13083	13085	13268	13273	13275	13284	13286	13387
		13392	13394	13403	13405									
MDPE	= 000400	3082												
MI	= 000004	2941	12053	12208										
MOC	= 000400	2936	7562	7596	11270	11281	11283	11290	11295	11297	11324	11335	11337	11395
		11424	11636	11651	11653	11660	11665	11667	11695	11710	11712	11772	11806	12029
		12044	12046	12056	12061	12063	12069	12071	12073	12078	12080	12108	12123	12125
		12184	12199	12201	12211	12216	12218	12229	12231	12242	12247	12249	12276	12308
		12310	12373	12388	12390	12397	12402	12404	12412	12414	12430	12477	12492	12494
		12499	12504	12506	12532	12547	12549	12554	12559	12561	12570	12572	12590	12592
		12601	12603	12622	12624	12649	12664	12666	12684	12686	12713	12749	12751	12756
		12761	12763	12772	12774	12805	12838	12840	12847	12862	12864	12948	12981	12983
		13006	13039	13041	13083	13085	13122	13151	13153	13161	13166	13168	13205	13234
		13236	13244	13249	13251	13284	13286	13313	13346	13348	13359	13361	13369	13371
		13387	13392	13394	13403	13405	13495	13533	13535	13549	13580	13582	13597	13607
		13609												
MOH	= 020600	2968												
MOL	= 010000	2879	7181	7194	7196	7219	7223							
MRO	= 002000	2934												
MS	= 000040	2939	12071											
MSC	= 000002	2942	12069	12071	12073									
MSE	= 100000	3022												
MSER	= 000200	2937	7492	7517	11363	11368	11370	11738	11743	11745	12151	12156	12158	13067
		13072	13074											
MUR	= 001000	2935	7191	7221	8203	8256	8286	8288	8399	8441	8443	8573	8599	8601
		8668	8697	8699	8824	8844	8846	8950	8952	8977	8979	9004	9006	9047
		9121	9143	9145	9203	9210	9212	9262	9271	9273	9322	9340	9342	9389
		9402	9404	9440	9455	9457	9510	9516	9518	9569	9575	9577	9676	9680
		9682	9787	9798	9800	9885	9894	9896	9955	9961	9963	10057	10063	10065
		10128	10146	10148	10153	10155	10193	10203	10484	10490	10492	10542	10598	10642
		10801	10809	10811	10868	10986	10992	10994	11128	11139	11141	11180	11191	11193
		11198	11203	11205	11230	11241	11243	11270	11281	11283	11288	11290	11295	11297
		11324	11335	11337	11342	11347	11349	11363	11368	11370	11395	11422	11424	11479
		11494	11496	11536	11551	11553	11558	11563	11565	11591	11606	11608	11636	11651
		11653	11658	11660	11665	11667	11695	11710	11712	11717	11722	11724	11738	11743
		11745	11772	11804	11806	11868	11883	11885	11925	11940	11942	11947	11952	11954
		11983	11998	12000	12029	12044	12046	12053	12056	12061	12053	12069	12071	12073
		12078	12080	12108	12123	12125	12130	12135	12137	12151	12156	12158	12184	12199
		12201	12208	12211	12216	12218	12229	12231	12242	12247	12249	12276	12308	12310
		12373	12388	12390	12395	12397	12402	12404	12412	12414	12430	12477	12492	12494
		12532	12547	12549	12554	12559	12561	12570	12572	12590	12592	12601	12603	12622
		12624	12649	12664	12666	12684	12686	12713	12749	12751	12756	12761	12763	12772
		12774	12805	12838	12840	12845	12847	12862	12864	12948	12981	12983	13006	13039
		13041	13046	13051	13053	13060	13062	13067	13072	13074	13083	13085	13122	13151
		13153	13159	13161	13166	13168	13205	13234	13236	13242	13244	13249	13251	13256

		5811*	5823	5899*	5907	5924*	5932	5954*	6061*	6063*	6075*	6077*	6095*	6097*
		6756*	6760*	6764	6798*	6800*	6802	6807	6813*	6815*	6817*	6822	6831*	6833*
		6835	6840	7738*	7806	7828*	7844	7881*	7895	7969*	8035*	9329*	9347	9446*
		9462	9466	9673*	9780*	9878*	9952*	10052*	10132*	10236	10443*	10762	10982*	11481*
		11538*	11593*	11638*	11697*	11774*	11870*	11927*	11985*	12031*	12110*	12186*	12278*	12381*
		12483*	12538*	12655*	12719*	12811*	12954*	13014*	13111*	13194*	13319*	13477*		
RMDAI	001334	3293*	5379*	5389	5535*	5551*	5557	5597*	5613	5646*	5658	5791*	5796	5823*
		5828	5907*	5910	5932*	5935	6764*	6767						
RMDAO	001410	3323*	7865*	7881	7906*	7908*	7949*	7969	8001*	13444*	13445*	13477	13571	
RMDB	= 000022	3115*	5333											
RMDBI	001352	3299*												
RMDBJ	001424	3329*												
RMDC	= 000034	3037*	5370*	5381	5528*	5541	5579*	5588*	5601	5639*	5650	5758*	5766	5845*
		5853	5871*	5883	5926*	5946*	5958	6758*	6762*	6766	6867	6871*	6873	6881*
		6883*	6885	6893*	6895*	6897	6912*	6914*	6918	7947	7952*	7982	8037*	9331*
		9351	9448*	9471	9475	9507*	9630*	9782*	9880*	9948*	10054*	10134*	10984*	11483*
		11540*	11595*	11640*	11699*	11776*	11872*	11929*	11987*	12033*	12112*	12188*	12280*	12379*
		12485*	12540*	12657*	12721*	12813*	12956*	13012*	13113*	13196*	13321*	13479*		
RMDCI	001362	3304*	5381*	5391*	5541*	5548*	5554*	5560	5601*	5609*	5615	5650*	5668	5766*
		5769*	5770	5853*	5857*	5858	5883*	5889*	5890	5958*	5963	5964	6766*	6769*
		6770												
RMDCO	001436	3334*	13446*	13479	13529									
RMDS	= 000012	3029*	5903*	5950*	7174	7180	7193	7205	7218	7259	7267	7279	7298	7556
		7564	7575	7593	8023	8039	8064	8091	8101	8113	8133	8159	8428	8433
		8470	8473	8956	8963	8983	8990	9054	9056	9063	9112	9127	9149	9497
		9523	9546	9582	9616	9632	9657	9684	10266	10274	10307	10312	10352	10399
		10446	10552	10556	10603	10608	10644	10648	10658	10662	10699	10704	10740	10747
		10816	10820	11302	11307	11672	11677	12085	12090	14022	14026	14055	14060	
		3295*												
RMDSI	001340	3325*												
RMDSO	001414	3034*	5873*	5991	6003	10264	10497							
RMOT	= 000026	3301*												
RMOTI	001354	3331*												
RMOTO	001430	3041*	5877*											
RMEC1	= 000044	3308*												
RMEC1I	001372	3338*												
RMEC1O	001446	3042*	5335	5821*	6983									
RMEC2	= 000046	3309*												
RMEC2I	001374	3339*												
RMEC2O	001450	3030*	5421*	5431	5467*	5474	5513*	5524*	5537	5590*	5603	5641*	5652	5756*
RMER1	= 000014	5764	5843*	5851	5867*	5879	5922*	5930	5952*	6221*	6225	6231	6239	6247
		6252*	6254*	6256	6262	6270	6279	6285*	6287*	6289	6294	6307*	6309	6320
		6644	6650	6665	6953	6958	6991	7350*	7362	7367	7399	7404	7824*	8054*
		8097*	8109*	8127*	8131*	8153*	8205*	8258*	8401*	8575*	8670*	8826*	8892*	8944*
		8971*	8998*	9049*	9083	9087	9123*	9216	9223	9324*	9391*	9442*	9444*	9719*
		9805	9901	9908	9968	9973	10070	10075	10158	10162	10184	10195*	10206	10538*
		10594*	10635*	10690*	10697*	10732*	10887	10893	10999*	11054	11060	11084	11130*	11157
		11162	11182*	11232*	11248	11252	11272*	11326*	11397*	11485*	11512	11517	11542*	11597*
		11613	11617	11642*	11701*	11778*	11874*	11901	11906	11931*	11989*	12005	12009	12035*
		12114*	12190*	12282*	12375*	12435	12443	12479*	12510	12515	12534*	12651*	12715*	12807*
		12950*	12987	12992	13008*	13124*	13173	13178	13207*	13315*	13497*	14016*	14049*	
RMER1I	001342	3296*	5431*	5439	5474*	5479*	5537*	5552*	5558	5603*	5616	5652*	5674	5764*
		5767	5851*	5854*	5855	5879*	5884	5930*	5933	6225*	6226	6234	6242	6256*
		6257	6265	6274										
RMER1O	001416	3326*	8938*	8944	8959	8971	8986	8998	9013	9022	9031*			
RMER2	= 000042	3040*	5423*	5433	5469*	5476	5517*	5530*	5543	5592*	5605	5643*	5654	5783*

E13

CZRMJ80 RM03/2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 368
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0367

RMMR2 = 000040	3039#	5789*	8247	8264	8294	10344	10356	10360	10391	10403	10407	10438	10450
	10454	10499	10512	11406	11412	11787	11794	12291	12298	12690	12696	12852	12879
	13526	13540	13554	13588	13602	13614							
RMMR2I 001366	3306#												
RMMR20 001442	3336#												
RMOF = 000032	3036#	5372*	5383	5515*	5526*	5539	5577*	5586*	5599	5637*	5648	5781*	5793
	5813#	5825	5869*	5881	5905*	5944*	5956	6476*	6478	6483	6506	6511*	6513*
	6515	6534*	6536*	6538	6563*	6565*	6567	7736*	7830*	7879*	7954*	7967*	8033*
	9333#	9355	9380	9395*	9409	9555*	9590	9595	9671*	9784*	9882*	9950*	10130*
	13481*												
RMOFI 001360	3303#	5383*	5393*	5539*	5547*	5553*	5559	5599*	5608*	5614	5648*	5662	5793*
	5798#	5799	5825*	5830*	5831	5881*	5886*	5887	5956*	5961*	6515*	6517	6524
	6538#	6540	6547										
RMOFO 001434	3333#	7724*	7736	7744	7756	7762	7782	7784*	7807*	7830	7831	7879	7913
	7915#	7948*	7954	7955	7967	7998	8000*	13447*	13458*	13481	13563		
	2903#	6227	6458	10207	10212	10237	11031						
RMR = 000004	3035#	5787*	5849*	6598	6602	6606							
RMSN = 000030	3302#												
RMSNI 001356	3332#												
RMSNO 001432	3112#	5325	10138*	13490*									
RMWC = 000002	3291#												
RMWCI 001330	3321#	13449*	13490										
RMWCO 001404	2990#	10345	10357	10392	10404	10439	10451	10500	10504	11407	11788	12292	12691
RGA = 100000	12853	13527	13555	13603									
RQB = 040000	2991#	10345	10357	10392	10404	10439	10451	10500	10506	11407	11788	12292	12691
	12853	13527	13555	13603									
	2830#	8500	8756	9571	10837								
RTC = 000016	2872#												
SADMSK = 000037	13753	14910#											
SAVREG = 104414	2867#												
SA1 = 000001	2863#												
SA16 = 000020	2866#												
SA2 = 000002	2865#												
SA4 = 000004	2864#												
SA8 = 000010	3063#												
SC = 100000	2923#												
SC TMSK = 003700	2921#												
SC0 = 000100	2920#												
SC1 = 000200	2919#												
SC2 = 000400	2918#												
SC3 = 001000	2917#												
SC4 = 002000	2836#	8353	8632	8776	9822	9957	10059	11878	11935	11993	12039	12118	12194
SEARCH = 000030	12286	12383											
	2825#	8491	8736	9825	11489	11546	11601	11646	11705	11782			
SEEK = 000004	9557	9624	9665	13115	13198	13483	14045#						
SETOM 061004	9196	9255	9501	9549	9563	9618	9659	9773	9870	9940	10045	10120	10334
SETVV 060654	10381	10428	10477	10794	10861	10975	11121	11172	11223	11263	11317	11388	11471
	11527	11583	11628	11687	11764	11860	11916	11975	12021	12100	12176	12268	12366
	12470	12525	12642	12706	12798	12941	12999	13104	13187	13306	13470	14008#	
	5210	13905#											
SIZCLK 060244	3011#	5481	6440	6442	7479	7495	7497	7515	7519				
SFI = 040000	2971#	5992	5996	6000									
SNGPRT = 020024	2700#	4952	5273	5359	5412	5460	5501	5715	5987	6021	6048	6125	6215
STACK = 001100	6341	6470	6501	6592	6629	6701	6748	6790	6859	6942	7057	7090	7168
	7254	7334	7463	7549	7628	7716	7802	7940	8019	8085	8190	8243	8387
	8560	8655	8811	8878	8935	9108	9190	9247	9313	9376	9429	9493	9542

T111	042560	10954	10955	10956#
T112	043122	11046	11047	11048#
T113	043374	11110	11111	11112#
T114	045016	11460	11461	11462#
T115	046574	11849	11850	11851#
T116	050760	12359	12360	12361#
T117	051370	12461	12462	12463#
T12	012714	6122	6123	6124#
T120	053542	12932	12933	12934#
T121	056060	13431	13432	13433#
T13	013326	6212	6213	6214#
T14	014154	6338	6339	6340#
T15	014710	6467	6468	6469#
T16	015046	6498	6499	6500#
T17	015422	6589	6590	6591#
T2	006754	5356	5357	5358#
T20	015562	6626	6627	6628#
T21	016104	6698	6699	6700#
T22	016260	6745	6746	6747#
T23	016444	6787	6788	6789#
T24	016754	6856	6857	6858#
T25	017312	6939	6940	6941#
T26	017772	7054	7055	7056#
T27	020132	7087	7088	7089#
T3	007174	5409	5410	5411#
T30	020456	7165	7166	7167#
T31	021030	7251	7252	7253#
T32	021356	7331	7332	7333#
T33	022132	7460	7461	7462#
T34	022502	7546	7547	7548#
T35	023032	7625	7626	7627#
T36	023412	7713	7714	7715#
T37	024000	7799	7800	7801#
T4	007410	5457	5458	5459#
T40	024520	7937	7938	7939#
T41	025050	8016	8017	8018#
T42	025312	8082	8083	8084#
T43	025752	8187	8188	8189#
T44	026160	8240	8241	8242#
T45	026576	8384	8385	8386#
T46	027344	8557	8558	8559#
T47	027676	8652	8653	8654#
T5	007576	5498	5499	5500#
T50	030336	8808	8809	8810#
T51	030630	8875	8876	8877#
T52	031074	8932	8933	8934#
T53	031764	9105	9106	9107#
T54	032274	9187	9188	9189#
T55	032526	9244	9245	9246#
T56	032770	9310	9311	9312#
T57	033222	9373	9374	9375#
T6	010734	5712	5713	5714#
T60	033440	9426	9427	9428#
T61	033714	9490	9491	9492#
T62	034112	9539	9540	9541#
T63	034366	9609	9610	9611#

T64	034532	9650	9651	9652#														
T65	034734	9700	9701	9702#														
T66	035222	9764	9765	9766#														
T67	035470	9861	9862	9863#														
T7	012114	5984	5985	5986#														
T70	035746	9930	9931	9932#														
T71	036234	10036	10037	10038#														
T72	036476	10113	10114	10115#														
T73	036740	10177	10178	10179#														
T74	037224	10254	10255	10256#														
T75	037370	10288	10289	10290#														
T76	037552	10327	10328	10329#														
T77	037732	10374	10375	10376#														
UBUSQS	065325	5088	15025#															
UNS	= 040000	2891#	5479	6313	6315	7357	7363	7368	7396	7400	7405	9031						
UNTMSK	= 000007	3095#	5286	5307	6635	10545												
UPE	= 020000	3077#																
USE	= 040000	3023#																
UO	= 000001	3092#																
U1	= 000002	3091#																
U2	= 000004	3090#																
VV	= 000100	2885#	7206	8957	8984	9064	9128	9150	9153	14023	14027							
WATCH	001524	3360#	8905*	8907	9726*	9736	11000*	11007	12431*	12438	12725*	12732	12817*	12824				
		12960*	12967	13018*	13025	13130*	13137	13213*	13220	13325*	13332	13503*	13510	13977*				
		13979*																
		2957#																
WC	= 000040	2845#	8325	9000	9748	9840												
WCD	= 000050	3076#	15126	15127														
WCE	= 040000	3101#																
WCEHI	= 010000	3102#																
WCELO	= 004000	2900#																
WCF	= 000040	2846#	2907	6235	6266													
WCH	= 000052	2849#	8333	9831														
WD	= 000060	2850#	8341	9837	10140	10911	10915	10988										
WH	= 000062	2894#	8349	9828														
WLE	= 004000	2880#	2907	10888	10913	15130	15131											
WRL	= 004000	2985#	7268	7280	7282	7299	7303											
XNUDC	= 176000	6886	5391	5548	5588	5609	5669	5672	5769	5857	5889	5963	6769	6874				
		3018#	6898	6916	9352	9472												
XNUER2	= 001567	6420	5436	5549	5592	5610	5679	5682	5801	5833	5912	5966	6359	6413				
		2980#	6421	6447	8162													
XNUOF	= 161577	6541	5393	5547	5586	5608	5663	5666	5798	5830	5886	5961	6518	6525				
		5024#	6548	6569	9356	9410												
		15182#	5165															
XSIZ	005212	3161	3167#															
ZEROS	066354	14989	15004															
\$APTHD	001100	14960	14962#															
\$ASTAT	***** U	14958#																
\$ATYC	065036	14307	14959#															
\$ATY1	065012	14548	14961#															
\$ATY3	065020	3199#	5009*	5225*	14666	14823												
\$ATY4	065030	3268#	5030	5100	5110*	5274	5341	5360	5413	5461	5502	5716	5988	6022				
\$AUTOB	001150	6049	6126	6216	6342	6471	6502	6593	6630	6702	6749	6791	6860	6943				
\$BASE	001276	7058	7091	7169	7255	7335	7464	7550	7629	7717	7803	7941	8020	8086				
		8191	8244	8388	8561	8656	8812	8879	8936	9109	9191	9248	9314	9377				
		9430	9494	9543	9613	9654	9704	9768	9865	9934	10040	10117	10181	10258				

K13

\$BDADR 001136

10292	10331	10378	10425	10473	10530	10587	10627	10682	10722	10788	10856	10958
11050	11114	11464	11853	12363	12465	12936	13435					
3194*	5288*	5289*	5309*	5310*	5341*	6002*	6003*	6031*	6054*	6055*	6142*	6143*
6157*	6158*	6173*	6174*	6195*	6196*	6230*	6231*	6238*	6239*	6246*	6247*	6261*
6262*	6269*	6270*	6278*	6279*	6293*	6294*	6319*	6320*	6345*	6346*	6482*	6483*
6505*	6506*	6597*	6598*	6649*	6650*	6806*	6807*	6821*	6822*	6839*	6840*	6866*
6867*	6957*	6958*	6985*	7062*	7093*	7094*	7173*	7174*	7258*	7259*	7340*	7341*
7366*	7367*	7386*	7387*	7403*	7404*	7434*	7435*	7469*	7470*	7555*	7556*	7632*
7633*	7722*	7723*	7805*	7806*	7946*	7947*	8022*	8023*	8090*	8091*	8193*	8246*
8247*	8416*	8432*	8433*	8447*	8472*	8473*	8564*	8658*	8659*	8814*	8815*	8881*
8882*	8962*	8963*	8989*	8990*	9016*	9017*	9053*	9054*	9073*	9074*	9086*	9087*
9111*	9112*	9222*	9223*	9287*	9288*	9379*	9380*	9465*	9466*	9474*	9475*	9490*
9497*	9545*	9546*	9594*	9595*	9615*	9616*	9656*	9657*	9708*	9709*	9907*	9908*
9972*	9973*	10074*	10075*	10161*	10162*	10183*	10184*	10273*	10302*	10311*	10312*	10359*
10360*	10406*	10407*	10453*	10454*	10511*	10512*	10555*	10556*	10568*	10569*	10607*	10608*
10647*	10648*	10661*	10662*	10703*	10704*	10746*	10747*	10819*	10820*	10892*	10893*	10969*
10970*	11011*	11012*	11053*	11054*	11161*	11162*	11212*	11251*	11252*	11306*	11307*	11358*
11377*	11411*	11412*	11516*	11517*	11572*	11616*	11617*	11676*	11677*	11733*	11752*	11793*
11794*	11905*	11906*	11961*	12008*	12009*	12089*	12090*	12146*	12165*	12235*	12237*	12257*
12297*	12298*	12423*	12424*	12442*	12443*	12514*	12515*	12581*	12582*	12613*	12614*	12632*
12675*	12676*	12695*	12696*	12736*	12737*	12783*	12784*	12828*	12829*	12878*	12879*	12971*
12972*	12991*	12992*	13029*	13030*	13094*	13095*	13142*	13177*	13178*	13224*	13225*	13295*
13296*	13336*	13337*	13380*	13381*	13415*	13416*	13514*	13515*	13539*	13540*	13587*	13588*
13613*	13614*	14025*	14026*	14059*	14060*	17053	17055	17057	17063	17065	17067	
3196*	5282*	5283*	5303*	5304*	5564*	5621*	5692*	5991*	5992	5996	5998	
6027*	6028*	6065*	6066*	6079*	6080*	6082	6099*	6100*	6101	6138*	6139*	6153*
6154*	6168*	6169*	6171	6190*	6191*	6192	6226*	6227*	6234*	6235*	6242*	6243*
6257*	6258*	6265*	6266*	6274*	6275*	6289*	6291	6309*	6312*	6315*	6316	6358*
6359*	6365*	6366*	6372*	6373*	6389*	6390*	6396*	6397*	6412*	6413*	6419*	6420*
6422	6439*	6442*	6445*	6448*	6478*	6479*	6517*	6518*	6524*	6525*	6540*	6541*
6547*	6548*	6550	6567*	6570	6606*	6607	6644*	6647*	6665*	6666*	6667	6722*
6723*	6802*	6803	6817*	6818	6835*	6836	6873*	6874*	6885*	6886*	6897*	6898*
6900	6918*	6919	6953*	6954*	6991*	6992*	6993	7069*	7070*	7100*	7101*	7112*
7113*	7125*	7126*	7143*	7144*	7180*	7181*	7193*	7194*	7205*	7206*	7218*	7219*
7224	7267*	7268*	7279*	7280*	7298*	7299*	7304	7352*	7353*	7362*	7363*	7364*
7383*	7384*	7399*	7400*	7401	7426*	7427*	7432	7478*	7479*	7494*	7495*	7514*
7515*	7520	7564*	7565*	7575*	7576*	7593*	7594*	7599	7639*	7640*	7653*	7654*
7665*	7666*	7683*	7684*	7687	7741*	7742*	7753	7759*	7760*	7771	7844*	7845
7853	7857	7895*	7896	7903	7906	7982*	7983	7993	8039*	8040*	8064*	8065*
8101*	8102*	8113*	8114*	8133*	8134*	8159*	8160*	8165	8214*	8215	8218*	8264*
8265	8267*	8294*	8295*	8297	8411*	8412	8414*	8423	8428*	8429*	8430	8445*
8446*	8452	8458	8465	8470*	8471*	8474	8586*	8587	8589*	8603*	8604*	8607
8613	8675*	8676*	8701*	8702*	8707	8830*	8831*	8848*	8849*	8852	8896*	8897*
8914*	8915*	8956*	8957*	8983*	8984*	9010*	9011*	9056*	9057*	9063*	9064*	9070*
9071*	9083*	9084*	9127*	9128*	9149*	9150*	9216*	9217*	9220	9278*	9279*	9284
9409*	9410*	9462*	9463	9471*	9472*	9523*	9524*	9582*	9583*	9590*	9591*	9632*
9633*	9684*	9685*	9732*	9733*	9734	9901*	9902*	9905	9968*	9969*	10070*	10071*
10158*	10159*	10206*	10207*	10209	10266*	10270*	10298*	10299*	10307*	10308*	10356*	10357*
10403*	10404*	10450*	10451*	10499*	10500*	10504	10506	10552*	10553*	10563*	10565*	10603*
10604*	10644*	10645*	10658*	10659*	10699*	10700*	10740*	10741*	10743	10816*	10817*	10887*
10888*	10889	10966*	10967*	11004*	11005*	11060*	11061*	11084*	11085*	11157*	11158*	11209*
11210*	11248*	11249*	11302*	11303*	11354*	11355*	11374*	11375*	11406*	11407*	11408	11512*
11513*	11569*	11570*	11613*	11614*	11672*	11673*	11729*	11730*	11749*	11750*	11787*	11788*
11791	11901*	11902*	11958*	11959*	12005*	12006*	12085*	12086*	12142*	12143*	12162*	12163*
12225*	12226*	12253*	12254*	12291*	12292*	12295	12416*	12417*	12435*	12436*	12510*	12511*
12574*	12575*	12609*	12610*	12628*	12629*	12671*	12672*	12690*	12691*	12693	12729*	12730*

\$BDDAT 001142

	5133*	5135	5141*	5143	5148*	5149*	5150*	5151*	5152*	5153	5161*	5162	5163
	5170*	5171	5172	5174	5176	5180	6001*	7756*	8137*	8168*	10895*	17057	17065
\$TMP2	001200												
\$TMP3	001202												
\$TMP4	001204												
\$TM =	000122												
	2677*	5260	5265*	5270	5347	5352*	5356	5399	5404*	5409	5447	5452*	5457
	5489	5494*	5498	5702	5707*	5712	5974	5979*	5984	6008	6013*	6018	6035
	6040*	6045	6112	6117*	6122	6202	6207*	6212	6328	6333*	6338	6457	6462*
	6467	6488	6493*	6498	6579	6584*	6589	6616	6621*	6626	6688	6693*	6698
	6735	6740*	6745	6778	6783*	6787	6846	6851*	6856	6930	6935*	6939	7044
	7049*	7054	7077	7082*	7087	7155	7160*	7165	7241	7246*	7251	7321	7326*
	7331	7450	7455*	7460	7536	7541*	7546	7615	7620*	7625	7704	7709*	7713
	7789	7794*	7799	7927	7932*	7937	8006	8011*	8016	8072	8077*	8082	8177
	8182*	8187	8230	8235*	8240	8374	8379*	8384	8547	8552*	8557	8642	8647*
	8652*	8798	8803*	8808	8865	8870*	8875	8922	8927*	8932	9095	9100*	9105
	9177	9182*	9187	9234	9239*	9244	9300	9305*	9310	9363	9368*	9373	9416
	9421*	9426	9480	9485*	9490	9529	9534*	9539	9599	9604*	9609	9640	9645*
	9650	9631	9696*	9700	9754	9759*	9764	9851	9856*	9861	9920	9925*	9930
	10026	10031*	10036	10103	10108*	10113	10167	10172*	10177	10244	10249*	10254	10278
	10283*	10288	10317	10322*	10327	10364	10369*	10374	10411	10416*	10421	10459	10464*
	10469	10516	10521*	10526	10573	10578*	10583	10613	10618*	10623	10668	10673*	10678
	10709	10714*	10718	10774	10779*	10784	10842	10847*	10852	10932	10937*	10954	11036
	11041*	11046	11100	11105*	11110	11450	11455*	11460	11839	11844*	11849	12349	12354*
	12359	12451	12456*	12461	12922	12927*	12932	13421	13426*	13431	14393	14433	
	3207*	14347*	14358										
\$TPB	001166												
\$TPFLG	001173												
\$TPS	001164												
\$TRAP	064526												
\$TRAP2	064566												
\$TRF =	000016												
	14885*	14896											
	14889*	14898*	14899*	14900*	14901*	14902*	14903*	14904	14905*	14906	14907*	14908*	14909*
	14910*	14911*	14912*										
	14879	14896*											
\$TRPAD	064600												
\$STM	001104												
\$STNM	001116												
\$TTYIN	064352												
\$TYPBN	061202												
\$TYPDS	061256												
\$TYPE	061730												
\$TYPEC	062142												
\$TYPEX	062210												
\$TYPOC	061526												
\$TYPON	061542												
\$TYPOS	061502												
\$UNIT	001234												
\$UNITM	001110												
\$USWR	001246												
\$VECT1	001272												
\$VECT2	001274												
\$XTSTR	062224												
\$SGT4=	000000												
\$SSW08=	000122												
	14377*												
	13727*												
	14432*	14433	14434*	14435*	14436*	14437*	14438*	14439*	14440*	14441*	14442*	14443*	14444*
	14445*	14446*	14447*	14448*	14449*	14450*	14451*	14452*	14453*	14454*	14455*	14456*	14457*
	14458*	14459*	14460*	14461*	14462*	14463*	14464*	14465*	14466*	14467*	14468*	14469*	14470*
	14471*	14472*	14473*	14474*	14475*	14476*	14477*	14478*	14479*	14480*	14481*	14482*	14483*
	14484*	14485*	14486*	14487*	14488*	14489*	14490*	14491*	14492*	14493*	14494*	14495*	14496*
	14497*	14498*	14499*	14500*	14501*	14502*	14503*	14504*	14505*	14506*	14507*	14508*	14509*

CZRMJBO RM03.2 DSKLS DIAG
CZRMJB.P11 23-NOV-77 12:07

MACY1: 30(1046) 23-NOV-77 12:14 PAGE 380
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0379

\$OFILL 061725
\$4OCAT= ***** U
= 113546

14510#	14511#	14512#	14513#	14514#																
14228#	14232#	14242	14277#																	
14374	14542																			
3125#	3129#	3138	3139#	3141#	3143#	3148#	3151#	3157	3158#	3160#	3162#	3174								
3181#	3223	3349#	4951	4966	4967	13712#	13735	13736#	14201#	14358	14431	14432								
14568	14571	14575#	14576	14577#	14815#	14816	14823#	14928	14950	15007#	15102#	17077#								
17079#	17080#	17081#																		
14959	14962																			
3157#	3162																			

.\$ASTA= ***** U
.\$X = 001100

CLEAR	2666#	5279	5300	5362	5415	5425	5463	5505	5570	5630	5753	5776	5808	5840	5864
	5896	5919	5941	6024	6051	6129	6135	6222	6303	6352	6434	6473	6595	6641	6714
	6751	6794	6863	6950	7060	7066	7097	7171	7261	7337	7466	7552	7636	7719	7819
	7876	7943	7964	8030	8088	8122	8148	8198	8250	8393	8568	8663	8819	8885	9042
	9116	9317	9384	9435	9712	10188	10261	10295	10533	10589	10630	10685	10726	10963	11057
	14009														
CLKOFF	2666#	8909	9738	9743	11009	11018	12440	12446	12734	12740	12826	12832	12969	12975	13027
	13033	13139	13145	13222	13228	13334	13340	13512	13518						
CLKON	2666#	8905	9726	11000	12431	12725	12817	12960	13018	13130	13213	13325	13503		
COMMEN	2807#														
ENDCOM	2807#														
ERR	2666#	3385	3393	3401	3409	3417	3425	3434	3442	3450	3458	3466	3474	3482	3490
	3498	3506	3514	3522	3530	3538	3546	3554	3562	3570	3578	3586	3594	3602	3610
	3618	3626	3634	3642	3650	3658	3666	3674	3682	3690	3698	3706	3714	3722	3730
	3738	3746	3754	3762	3770	3778	3786	3794	3802	3810	3818	3826	3834	3842	3850
	3858	3866	3874	3882	3890	3898	3906	3914	3922	3930	3938	3946	3954	3962	3970
	3978	3986	3994	4002	4010	4018	4026	4034	4042	4050	4058	4066	4074	4082	4090
	4098	4106	4114	4122	4130	4138	4146	4154	4162	4170	4178	4186	4194	4202	4210
	4218	4226	4234	4242	4250	4258	4266	4274	4282	4290	4298	4306	4314	4322	4330
	4338	4346	4354	4362	4370	4378	4386	4394	4402	4410	4418	4426	4434	4442	4450
	4458	4466	4474	4482	4490	4498	4506	4514	4522	4530	4538	4546	4554	4562	4570
	4578	4586	4594	4602	4611	4619	4627	4635	4643	4651	4659	4667	4675	4684	4692
	4700	4708	4716	4725	4733	4742	4750	4758	4766	4774	4782	4790	4799	4807	4815
	4823	4831	4839	4847	4855	4863	4871	4879	4887	4895	4903	4912	4920	4928	4936
ERROR	2701#	5212	5290	5311	5342	5396	5444	5485	5566	5623	5693	5772	5804	5836	5860
	5892	5915	5937	5969	6004	6032	6069	6084	6104	6144	6159	6175	6197	6232	6240
	6248	6263	6271	6280	6295	6321	6361	6368	6375	6392	6399	6415	6424	6450	6484
	6520	6527	6543	6552	6573	6609	6653	6655	6671	6673	6725	6772	6808	6823	6841
	6876	6888	6902	6922	6959	6986	6998	7000	7073	7104	7116	7129	7149	7184	7197
	7209	7228	7270	7283	7308	7355	7370	7373	7388	7407	7410	7437	7482	7523	7568
	7579	7602	7642	7656	7669	7691	7757	7774	7847	7898	7985	8043	8068	8105	8116
	8138	8169	8220	8269	8299	8417	8434	8454	8461	8476	8590	8609	8616	8679	8709
	8834	8855	8900	8918	8964	8991	9018	9025	9059	9066	9075	9088	9131	9154	9198
	9225	9257	9289	9359	9412	9467	9476	9503	9526	9551	9559	9565	9586	9596	9620
	9626	9636	9661	9667	9688	9740	9775	9815	9872	9910	9942	9975	10047	10077	10122
	10164	10214	10216	10275	10303	10313	10336	10361	10383	10408	10430	10455	10479	10513	10558
	10570	10609	10650	10664	10705	10748	10796	10823	10863	10896	10972	10977	11014	11064	11089
	11123	11163	11174	11214	11225	11254	11265	11308	11319	11359	11379	11390	11413	11473	11518
	11529	11574	11585	11619	11630	11678	11689	11734	11754	11766	11795	11862	11907	11918	11963
	11977	12011	12023	12091	12102	12147	12167	12178	12238	12258	12270	12299	12368	12425	12444
	12472	12516	12527	12583	12615	12633	12644	12677	12697	12708	12738	12785	12800	12830	12980
	12943	12973	12993	13001	13031	13096	13106	13117	13143	13179	13189	13200	13226	13297	13308
	13338	13382	13417	13472	13485	13516	13541	13589	13615						
ESCAPE	2807#														
GETAS	2666#	10562													
GETBA	2666#														
GETBAE	2666#														
GETCS1	2666#	5375	5428	5532	5594	6026	6137	6152	6167	6189	6721	7068	8213	8410	8444
	8585	8602	10297	10348	11208	11353	11373	11568	11728	11748	11957	12141	12161	12252	12627
GETCS2	2666#	5281	5302												
GETDA	2666#	5378	5534	5596	5645	5790	5822	5906	5931	6763	6801	6816	6834	7843	7894
	9346	9461													
GETDB	2666#														
GETDC	2666#	5380	5540	5600	5649	5765	5852	5892	5957	6765	6872	6884	6896	6917	7981
	9350	9470													
GETDS	2666#	7179	7192	7204	7217	7266	7278	7297	7563	7574	7592	8038	8063	8100	8112

H14

CZRMJBG RM03/2 DSKLS DIAG
CZRMJB.F11 23-NOV-77 12:07

MACY11 30(1046) 23-NOV-77 12:14 PAGE 385
CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0383

PUTX	2666#														
REPORT	2807#														
RGBFMC	2666#	3283	3313												
SCOPE	2702#	5264	5351	5403	5451	5493	5706	5978	6012	6039	6116	6206	6332	6461	6492
	2583#	5520	5682	6739	6782	6850	6934	7048	7087	7159	7245	7325	7454	7540	7676
	7708#	7793	7931	8010	8076	8181	8234	8378	8551	8646	8802	8869	8926	9099	9181
	9238#	9304	9367	9420	9484	9533	9603	9644	9695	9758	9855	9924	10030	10107	10171
	10248#	10282	10321	10368	10415	10463	10520	10577	10617	10672	10713	10778	10846	10936	11040
	11104#	11454	11843	12353	12455	12926	13425								
SETOM	2666#	9556	9623	9664	13114	13197	13482								
SETPRI	2807#	4989	13964	13990	14744										
SETTRA	14889#	14898	14899	14900	14901	14902	14904	14906	14907	14908	14909	14910	14911		
SETUP	2807#	4946													
SETVV	2666#	9195	9254	9500	9548	9562	9617	9658	9772	9869	9939	10044	10119	10333	10380
	10427#	10476	10793	10860	10974	11120	11171	11222	11262	11316	11387	11470	11526	11582	11627
	11686#	11763	11859	11915	11974	12020	12099	12175	12267	12365	12469	12523	12640	12704	12796
	12940#	12998	13103	13185	13304	13468									
SKIP	2807#														
SLASH	2807#														
SPACE	2807#														
STARCS	2807#														
	5450#	3136	3154	3156	3163	3177	3223	3226	5260	5263	5347	5350	5399	5402	5447
	5489#	5489	5492	5702	5705	5974	5977	6008	6011	6035	6038	6072	6087	6112	6115
	6202#	6205	6328	6331	6348	6379	6403	6428	6457	6460	6488	6491	6508	6531	6556
	6579#	6582	6616	6619	6688	6691	6735	6758	6778	6781	6846	6849	6930	6933	6948
	6962#	7044	7047	7077	7080	7155	7158	7241	7244	7321	7324	7450	7453	7536	7539
	7615#	7618	7704	7707	7789	7792	7810	7862	7911	7927	7930	8006	8009	8072	8075
	8177#	8180	8230	8233	8374	8377	8486	8547	8550	8642	8645	8798	8801	8865	8868
	8922#	8925	9095	9098	9177	9180	9234	9237	9300	9303	9363	9366	9416	9419	9480
	9483#	9529	9532	9599	9602	9640	9643	9691	9694	9754	9757	9851	9854	9920	9923
	10026#	10029	10103	10106	10167	10170	10244	10247	10278	10281	10317	10320	10364	10367	10411
	10414#	10459	10462	10516	10519	10573	10576	10613	10616	10668	10671	10709	10712	10774	10777
	10842#	10845	10932	10935	11036	11039	11100	11103	11117	11167	11218	11258	11312	11383	11450
	11453#	11467	11522	11578	11623	11682	11759	11839	11842	11856	11911	11970	12015	12095	12171
	12263#	12349	12352	12451	12454	12467	12520	12637	12701	12793	12922	12925	12938	12995	13099
	13182#	13301	13421	13424	13437	13440	13455	13465	13690	13739	13904	13949	14068	14113	14137
	14204#	14281	14360	14516	14570	14647	14662	14733	14757	14826	14864	14914	14930	14957	
SWRSL	2807#	4968#													
TACS	2666#	3282													
TRMTRP	14889#														
TESTEY	2666#	5267	5353	5406	5454	5495	5709	5981	6015	6042	6119	6209	6335	6464	6495
	6586#	6623	6695	6742	6784	6853	6936	7051	7084	7162	7248	7328	7457	7543	7622
	7710#	7796	7934	8013	8079	8184	8237	8381	8554	8649	8805	8872	8929	9102	9184
	9241#	9307	9370	9423	9487	9536	9606	9647	9697	9761	9858	9927	10033	10110	10174
	10251#	10285	10324	10371	10418	10466	10523	10580	10620	10675	10715	10781	10849	10951	11043
	11107#	11457	11846	12356	12458	12929	13428								
TYPBIN	2807#														
TYPDEC	2807#	13713	13720												
TYPNAM	2666#	2807#	4993												
TYPNUM	2807#														
TYPOCS	2807#	5114	5135	13760	13768	13777	13783								
TYPOCT	2807#	5100	14675												
TYPTXT	2807#	13709	13716												
SSCMRE	3175#														
SSCMTH	3175#	3212	3213	3214	3215	3216									
SSESCH	2807#														
SSNEW	2807#	5260	5347	5399	5447	5489	5702	5974	6008	6035	6112	6202	6328	6457	6488

	6579	6616	6688	6735	6778	6846	6930	7044	7077	7155	7241	7321	7450	7536	7615
	7704	7789	7927	8006	8072	8177	8230	8374	8547	8642	8798	8865	8922	9095	9177
	9234	9300	9363	9416	9480	9529	9599	9640	9691	9754	9851	9920	10026	10103	10167
	10244	10278	10317	10364	10411	10459	10516	10573	10613	10668	10709	10774	10842	10932	11036
	11100	11450	11839	12349	12451	12922	13421								
SSSET	14889#	14898	14899	14900	14901	14902	14904	14906	14907	14908	14909	14910	14911		
SSSETH	4984#														
SSSKIP	2807#														
.EQUAT	2666#	2697													
.GETPR	2666#														
.HEADE	2666#	2667													
.SETUP	2666#	3122													
.SWRHI	2666#	2678													
.SWRLO	2666#	2689#													
.SACTI	2666#	3134													
.SAPT8	2666#	3224#													
.SAPTH	2666#	3152													
.SAPTY	2666#	14955													
.SCATC	2666#	3123													
.SCMTA	2666#	3175													
.SDIV	2666#														
.SEOP	2666#	13688													
.SERRO	2666#	14514													
.SERRT	2666#														
.SMULT	2666#														
.SPOWE	2666#	14912													
.SRAND	2666#														
.SRDOE	2666#														
.SRDOC	2666#	14824													
.SREAD	2666#	14568													
.SSAVE	2666#	14066													
.SSCOP	2666#	14358													
.SSIZE	2666#														
.STRAP	2666#	14862													
.STYP8	2666#	14111													
.STYPD	2666#	14135													
.STYPE	2666#	14279													
.STYPO	2666#	14202													

. ABS. 113546 000

ERRORS DETECTED: 0

RMO3:CZRMJB.BIN,RMO3:CZRMJB.SEG·DOC/SOL/NL:TOC/CRF=RMO3:CZRMJB.P11

RUN-TIME: 48 47 5 SECONDS

RUN-TIME RATIO: 602/101=5.9

CORE USED: 33K (65 PAGES)

DOCUMENT PAGES: 384