

RM03,02

FUNCTIONAL TEST 2
CZRMDBO

AH-B001B-MC

COPYRIGHT © 1977

FICHE 1 OF 2

JAN 1978

digital

MADE IN USA

digital
MADE IN USA

B01

00010000 PAGE 1
CZRMDB.P11 23-NOV-77 12:19

780105

PDPI0010001

CZRMDB0 RM03/2 FCTNL TST 2

MACY11 30(1046) 23-N

.REM \

IDENTIFICATION

PRODUCT CODE: AC-B0008-MC
PRODUCT NAME: CZRMDB0 RM03/RM02 FUNCTIONAL TEST,
PART 2
DATE CREATED: 1 AUGUST 77
MAINTAINER: DIAGNOSTIC GROUP
AUTHOR: DOUG RIIKONEN

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

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED TO THE PURSUAER
UNDER A LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED
(WITH INCLUSION OF DIGITAL'S COPYRIGHT NOTICE) ONLY FOR USE IN SUCH
SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING BY DIGITAL.

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

COPYRIGHT (C) 1977, DIGITAL EQUIPMENT CORPORATION

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

.PAGE

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. HALTING
 5. RESTARTING
4. OPERATOR INTERFACE
 1. PROGRAM I.D.
 2. CONSOLE DIALOGUE
 3. PROGRESS REPORTS
 4. PERFORMANCE REPORTS
 5. PROGRAM HALTS
 6. ERROR REPORTS
5. ENVIRONMENTAL SUPPORT
 1. PROCESSOR COMPATIBILITY

53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108

CZRMDB0 RM03/2 FCTNL TST 2
CZRMDB.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 3

DO1

SEQ 0003

109

2. DUAL PORT CONFIGURATIONS

CZRMO80 RMO3/2 FCTNL TST 2
CZRMO8.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 4

EO1

SEQ 0004

PAGE 2

110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125

- 3. MEMORY PARITY HARDWARE
- 4. MEMORY MANAGEMENT HARDWARE
- 5. ACT,APT COMPATIBILITY
- 6. XXDP COMPATIBILITY
- 7. OPERATING SYSTEM COMPATIBILITY
- 6. TEST DESCRIPTION

1.0 INTRODUCTION

1.1 ABSTRACT

THE RM03 SUBSYSTEM FUNCTIONAL TEST IS A STAND ALONE PROGRAM WHICH USES FUNCTIONAL MEANS TO VERIFY THE OPERABILITY OF THE RM03 DISK SUBSYSTEM. IN PARTICULAR, THE PROGRAM SERVES THE FOLLOWING PURPOSES:

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

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

TO VERIFY THE FUNCTIONALITY OF THE RM03 SUBSYSTEM, INCLUDING THE MASSBUS CONTROLLER, MASSBUS ADAPTER AND THE DISK DRIVE.

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

1.2 UNIT UNDER TEST

THE UNIT UNDER TEST (UUT) IS THE RM03 DISK SUBSYSTEM WHICH CONSISTS OF THE RHXX MASSBUS CONTROLLER, THE RM03 MASSBUS ADAPTER, AND THE STORAGE MODULE DISK DRIVE. NOTE THAT A DISK PACK IS REQUIRED FOR TESTING AND IS CONSIDERED AN INTEGRAL OF THE STORAGE MODULE DISK DRIVE.

2.0 OPERATING REQUIREMENTS

2.1 HARDWARE REQUIREMENTS

THE FOLLOWING MINIMUM HARDWARE CONFIGURATION, ASSUMED TO BE OPERATIONAL, IS REQUIRED TO LOAD AND EXECUTE THE RM03 SUBSYSTEM FUNCTIONAL TEST:

PDP-11 PROCESSOR
16K MEMORY
KW11-L OR KW11-P CLOCK
PROGRAM LOADING DEVICE
TERMINAL
RH11 OR RH70 CONTROLLER
UNIT UNDER TEST,

GO1

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 6

SEQ 0006

182
183

WHERE THE UNIT UNDER TEST CONSISTS OF ONE TO EIGHT RM03 ADAPTERS, DISK
DRIVES AND DISK PACKS.

2.2 MEDIA REQUIREMENTS

EACH UNIT BEING TESTED MUST BE LOADED WITH A SCRATCH DISK PACK BEFORE TESTING BEGINS ON THAT UNIT. THE PACK MAY BE FORMATTED OR UNFORMATTED, BUT MUST NOT CONTAIN NEEDED INFORMATION BECAUSE THE PACK WILL BE WRITTEN DURING THE TEST.

NOTE

WHEN THE PROGRAM IS LOADED VIA THE RM03, DRIVE 0 IS NOT TESTED UNLESS THE OPERATOR SPECIFICALLY ENTERS THE DRIVE NUMBER DURING CONSOLE DIALOGUE.

2.3 PREREQUISITE DIAGNOSTIC PROGRAMS

RM03 DISKLESS DIAGNOSTIC, CZRMJ-B

3.0 OPERATING PROCEDURE

3.1 LOADING

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

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

NOTE

WHEN THE PROGRAM IS LOADED VIA THE RM03, DRIVE 0 IS NOT TESTED UNLESS THE OPERATOR SPECIFICALLY ENTERS THE DRIVE NUMBER DURING CONSOLE DIALOGUE.

3.2 SWITCH OPTIONS

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

101

CZRMDBO RMO3/2 FCTNL TST 2
CZRMDB.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 8

SEQ 0008

240
241

SW15 HALT ON ERROR
SW14 LOOP ON TEST (CURRENTLY BEING EXECUTED)

SW13 INHIBIT ERROR TYPEOUTS
SW12 UNUSED
SW11 INHIBIT TEST ITERATIONS
SW10 BELL ON ERROR
SW09 LOOP ON ERROR
SW08 LOOP ON TEST IN SW07-00

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

3.3 STARTING

THE PROGRAM STARTS AT LOCATION 200, WHICH PROVIDES WORST CASE TEST CONDITIONS IF RUNNING IN AN AUTOMATIC ENVIRONMENT. IF RUNNING IN A STAND-ALONE ENVIRONMENT, THE PROGRAM USES CONSOLE DIALOGUE TO ALLOW THE OPERATOR TO CONTROL TEST CONDITIONS.

3.4 HALTING

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

3.5 RESTARTING

THE PROGRAM CAN BE RESTARTED AT ADDRESS 200.

4.0 OPERATOR INTERFACE

4.1 PROGRAM ID

THE PROGRAM TYPES ITS TITLE AND MAINDEC NUMBER THE FIRST TIME IT IS STARTED AFTER BEING LOADED. PROGRAM IDENTIFICATION DOES NOT OCCUR IF THE PROGRAM IS RESTARTED.

4.2 CONSOLE DIALOGUE

WHEN THE PROGRAM IS RUNNING IN STAND ALONE MODE, IT ENTERS A CONSOLE DIALOGUE SEQUENCE AFTER TYPING THE PROGRAM I.D..

THE FIRST QUESTION TYPED OUT IS: "TYPE HELP TEXT (Y OR N)??". IF THE OPERATOR RESPONDS WITH A Y, THE PROGRAM WILL TYPE A BRIEF HELP MESSAGE WHICH WILL LIST SWITCH OPTIONS, ETC.

THE SECOND QUESTION TYPED OUT IS, "CHANGE RM03 UNIBUS ADDRESS OR VECTOR ADDRESS (Y OR N)??". IF THE UNIBUS ADDRESS OF THE RM03 IS NON STANDARD, THE OPERATOR SHOULD RESPOND Y, THEN ANSWER SUBSEQUENT QUESTIONS TO SPECIFY THE UNIBUS ADDRESS, VECTOR ADDRESS AND INTERRUPT PRIORITY. IF THE OPERATOR DOES NOT RESPOND WITH A Y, THE PROGRAM SKIPS TO THE THIRD QUESTION.

THE THIRD QUESTION TYPED OUT IS, "TYPE (A) TO TEST ALL DEVICES, OR TYPE DEVICE NUMBER(S). TERMINATE INPUT WITH CARRIAGE RETURN". IF THE OPERATOR TYPES A, ALL POSSIBLE DEVICES ARE TESTED. OTHERWISE, THE OPERATOR CAN TYPE THE NUMBER(S) OF THE DEVICE(S) HE WANTS TESTED, AND TERMINATE HIS INPUT WITH A CARRIAGE RETURN.

IF THE PROGRAM IS RESTARTED, THE FIRST QUESTION TYPED IS, "USE SAME DEVICES (Y OR N)??". IF THE OPERATOR TYPES Y, THE TEST IS RESTARTED USING THE SAME DEVICES AS THE LAST TIME. OTHERWISE, THE TEST RESTARTS THE DIALOGUE AS IF THE PROGRAM WERE STARTED FOR THE FIRST TIME.

4.3 PROGRESS REPORTS

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

4.4 PERFORMANCE REPORT

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

4.5 PROGRAM HALTS

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

4.6 ERROR REPORTS

THE FIRST LINE OF THE ERROR REPORT CONTAINS THE NUMBER OF THE UNIT 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: ONE OR MORE LINES OF TEXT WHICH GIVE A BRIEF YET COMPREHENSIVE DESCRIPTION OF THE ERROR. THE ERROR MESSAGE IS NORMALLY FOLLOWED BY ONE OR MORE PAIRS OF LINES CONTAINING DATA HEADERS AND DATA PERTINENT TO THE ERROR, INCLUDING EXPECTED AND ACTUAL TEST RESULTS.

5.0 ENVIRONMENTAL SUPPORT

5.1 PROCESSOR COMPATIBILITY

THE RMO3 SUBSYSTEM FUNCTIONAL TEST IS EXECUTABLE ON ANY PDP-11 PROCESSOR, PROVIDING PREVIOUSLY MENTIONED HARDWARE REQUIREMENTS ARE MET, AND PROVIDING THAT DATA THROUGHPUT ON THE SYSTEM IS SUFFICIENT TO SUSTAIN DATA TRANSFER OPERATIONS.

5.2 DUAL PORT CONFIGURATIONS

THE RMO3 SUBSYSTEM FUNCTIONAL TEST DOES NOT SPECIFICALLY TEST DUAL PORT LOGIC IN THE RMO3 ADAPTER BUT IS EXECUTABLE ON RMO3 SUBSYSTEMS HAVING THE DUAL PORT OPTION PROVIDING THE DUAL PORT SWITCH IS SET TO THE APPROPRIATE PORT (A OR B).

5.3 MEMORY PARITY HARDWARE

MEMORY PARITY HARDWARE IS NOT USED DURING THE EXECUTION OF THE RMO3 SUBSYSTEM FUNCTIONAL TEST.

5.4 MEMORY MANAGEMENT HARDWARE

MEMORY MANAGEMENT HARDWARE IS NOT USED DURING THE RMO3 SUBSYSTEM FUNCTIONAL TEST. CAPABILITIES OF THE MASSBUS CONTROLLER.

5.5 ACT11, APT11 COMPATIBILITY

THE RMO3 SUBSYSTEM FUNCTIONAL TEST IS COMPATIBLE WITH ACT11 AND APT11 IN BOTH DUMP AND AUTOMATIC MODES. FURTHER, THE PROGRAM WILL EXECUTE A QUICK PASS DURING THE FIRST PASS IN SUPPORT OF QUICK VERIFY MODE.

5.6 XXDP COMPATIBILITY

THE RMO3 SUBSYSTEM FUNCTIONAL TEST IS COMPATIBLE WITH XXDP IN DUMP AND CHAIN MODES, AND PROVIDES MEDIA PROTECTION IN THE CASE WHERE THE RMO3 IS THE XXDP LOADING DEVICE.

MO1

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 12

SEQ 0012

PAGE 8

405
406
407
408
409
410

5.7 OPERATING SYSTEM COMPATIBILITY

THE PROGRAM IS NOT COMPATIBLE WITH ANY SOFTWARE OPERATING SYSTEM.

6.0 TEST DESCRIPTION

CONTROLLER ACCESS TEST

PURPOSE:

TO VERIFY THAT THE UNIBUS ADDRESS OF THE RM03 SUBSYSTEM IS CORRECT, AS DEFINED AT LOCATION \$BASE.

PROCEDURE:

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

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

DEVICE AVAILABLE TEST

PURPOSE:

TO VERIFY THAT THE UNIT UNDER TEST IS AVAILABLE FOR TESTING, AND IS NOT LOCKED OR PROGRAMMED TO THE ALTERNATE PORT.

PROCEDURE:

THIS TEST SELECTS THE DEVICE AND READS CONTROL STATUS REGISTERS 1 AND 2 TO VERIFY THAT THE SELECTED DEVICE IS AVAILABLE FOR TESTING, AS INDICATED BY DVA STATUS, BIT 11 OF RMCS1 AND NED STATUS, BIT 12 OF RMCS2. THE RESULTS OF THE TEST CAN VARY AS FOLLOWS:

.NONEXISTANT DEVICE - THE DEVICE IS NONEXISTENT OR IS LOCKED ON THE ALTERNATE PORT AND IS THEREFORE NOT AVAILABLE FOR TESTING;

.DEVICE NOT AVAILABLE - THE DEVICE EXISTS BUT IS SEIZED BY THE ALTERNATE PORT AND IS NOT AVAILABLE FOR TESTING;

.BUS TIMEOUT - THE MASSBUS CONTROLLER FAILED TO DETECT A

B02

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 14

SEQ 0014

467

NONEXISTENT DEVICE;

.DEVICE AVAILABLE - THE DEVICE IS AVAILABLE FOR TESTING.

THE PROGRAM JUMPS TO THE SUBPASS HANDLER WHICH SELECTS THE NEXT DRIVE TO BE TESTED IF THE DEVICE IS NOT AVAILABLE.

DRIVE TYPE TEST

PURPOSE:

TO VERIFY THAT THE UNIT UNDER TEST IS AN RM03 SINGLE PORT OR DUAL PORT SUBSYSTEM.

PROCEDURE:

THIS TEST READS THE DRIVE TYPE REGISTER, RMDT, OF THE SELECTED DEVICE AND VERIFIES THAT THE DEVICE IS A SINGLE PORT OR DUAL PORT RM03 SUBSYSTEM. IF THE SELECTED DEVICE IS NOT AN RM03, THE PROGRAM JUMPS TO THE SUBPASS HANDLER WHICH WILL SELECT THE NEXT DEVICE FOR TESTING.

WRITE/READ HEADER AND DATA (FORMAT) TESTS

PURPOSE:

TO TEST WRITE HEADER AND DATA AND READ HEADER AND DATA FUNCTIONALITY OF THE RM03 SUBSYSTEM USING A SET OF VARIABLES WHICH INCLUDE WORD COUNT, HEAD MOTION, HEAD SWITCHING AND ERROR CONDITIONS.

PROCEDURE:

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

524
525
526
527
528
529
530
531
532
533
534
535
536
537
538

THE SAME TYPE OF ERRORS AS IN THE WRITE COMMAND.

NOTE THAT THE SECTOR USED DURING A TEST MAY DIFFER FROM THE PROGRAM LISTING BECAUSE THE PROGRAM SUBSTITUTES A GOOD SECTOR IF THE ONE SELECTED IS LISTED IN THE BAD BLOCK TABLE OF THE LAST TRACK. UNLESS SPECIFIED OTHERWISE, ALL TESTS ARE IN 16 BIT (32 SECTOR) FORMAT.

FORMAT ZEROS - 18

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

FORMAT ZEROS - 16

THIS TEST IS THE SAME AS THE PREVIOUS TEST, EXCEPT THAT DATA IS WRITTEN IN 16 BIT FORMAT.

ZERO FILL TEST

THE TEST EXECUTES A SEEK TO CYLINDER 0 TO INSURE THAT THERE IS NO HEAD MOTION DURING DATA TRANSFER. THIS IS FOLLOWED BY A WRITE HEADER AND DATA COMMAND WITH THE WORD COUNT EQUAL TO THE SIZE OF THE HEADER WHICH CAUSES THE RH70 TO ZERO FILL THE DATA FIELD. THE READ HEADER AND DATA COMMAND THAT FOLLOWS READS A FULL SECTOR AND VERIFIES THAT DATA WAS ZERO FILLED.

FORMAT CHECK ZEROS

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

FORMAT CHECK ZEROS W/ WCE ERROR

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

539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593

FORMAT ONES

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

FORMAT CHECK ONES

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

FORMAT CHECK ONES W/ WCE ERROR

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

FORMAT MULTIPLE SECTORS

THE TEST SEEKS TO CYLINDER 0 TO INSURE THERE IS NO HEAD MOTION DURING DATA TRANSFER. THE WRITE HEADER AND DATA COMMAND FOLLOWS, WITH THE WORD COUNT EQUAL TO MULTIPLE SECTORS. THE WORD COUNT DURING THE READ HEADER AND DATA PORTION OF THE TEST IS SET FOR ALL OF THE FIRST SECTOR AND THE HEADER OF THE SECOND SECTOR.

FORMAT WITH HEAD SWITCHING

THE TEST SEEKS TO CYLINDER 0 TO INSURE THERE IS NO HEAD MOTION DURING DATA TRANSFER. THE WRITE HEADER AND DATA COMMAND STARTS WITH CYLINDER 0, TRACK 0, SECTOR 31. THE WORD COUNT IS EQUAL TO MULTIPLE SECTORS WHICH CAUSES THE SUBSYSTEM TO SWITCH

G02

CZRMDB0 RM03/2 FCTNL TST 2
CZRMDB.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 19

SEQ 0019

650
651

FROM TRACK 0 TO TRACK 1 AFTER THE FIRST SECTOR IS WRITTEN. THE
READ HEADER AND DATA COMMAND USES THE SAME WORD COUNT AND

652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705

STARTING SECTOR.

FORMAT WITH IMPLIED SEEK

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

FORMAT WITH MIDTRANSFER SEEK

THIS TEST WRITES MULTIPLE SECTORS STARTING WITH CYLINDER 0, TRACK 4, SECTOR 31, CAUSING A MIDTRANSFER SEEK AFTER THE FIRST SECTOR IS WRITTEN. THE SAME SECTORS ARE READ WITH READ HEADER AND DATA COMMAND.

FORMAT EACH SECTOR ADDRESS

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

FORMAT EACH TRACK ADDRESS

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

FORMAT PRIME CYLINDERS

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

PAGE 32

FORMAT LAST SECTOR

THIS TEST READS HEADER AND DATA ON THE LAST SECTOR, I.E.
CYLINDER 822, TRACK 4, SECTOR 31, AND VERIFIES THAT LBT STATUS
SETS.

FOR W/ AOE ERROR

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

FORMAT INVALID SECTOR ADDRESS

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

FORMAT INVALID TRACK ADDRESS

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

FORMAT INVALID CYLINDER ADDRESS

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

FORMAT AT OFFSET

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

IVC FORMAT TEST

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

FORMAT ERROR TEST(18 AND 16)

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

FORMAT HCE (FIRST AND SECOND HEADER WORDS)

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

760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798

CZRMDB0 RM03/2 FCTNL TST 2
CZRMDB.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 23

K02

SEQ 0023

799

```
800      ;PROGRAM REVISION #001
801
802      .TITLE CZRM080 RM03/2 FCTNL TST 2
803      ;*COPYRIGHT (C) 1977
804      ;*DIGITAL EQUIPMENT CORP.
805      ;*MAYNARD, MASS. 01754
806      ;*
807      ;*PROGRAM BY DOUG RIIKONEN
808      ;*
809      ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
810      ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
811
812      000001      $TN=1
813      .SBTTL OPERATIONAL SWITCH SETTINGS
814      ;*
815      ;*      SWITCH      USE
816      ;*      -----
817      ;*      15      HALT ON ERROR
818      ;*      14      LOOP ON TEST
819      ;*      13      INHIBIT ERROR TYPEOUTS
820      ;*      12      ENABLE EXTENDED STATUS
821      ;*      11      INHIBIT ITERATIONS
822      ;*      10      BELL ON ERROR
823      ;*      9      LOOP ON ERROR
824      ;*      8      LOOP ON TEST IN SWR<7:0>
825      ;*      7      TN128
826      ;*      6      TN64
827      ;*      5      TN32
828      ;*      4      TN16
829      ;*      3      TN8
830      ;*      2      TN4
831      ;*      1      TN2
832      ;*      0      TN1
833      .SP*TL BASIC DEFINITIONS
834
835      ;*1 .TIAL ADDRESS OF THE STACK POINTER *** 1100 ***
836      001100      STACK= 1100
837      .EQUIV EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
838      .EQUIV IOT,SCOPE      ;;BASIC DEFINITION OF SCOPE CALL
839
840      ;*MISCELLANEOUS DEFINITIONS
841      000011      HT= 11      ;;CODE FOR HORIZONTAL TAB
842      000012      LF= 12      ;;CODE FOR LINE FEED
843      000015      CR= 15      ;;CODE FOR CARRIAGE RETURN
844      000200      CRLF= 200    ;;CODE FOR CARRIAGE RETURN-LINE FEED
845      177776      PS= 177776   ;;PROCESSOR STATUS WORD
846      .EQUIV PS,PSW
847      177774      STKLMT= 177774 ;;STACK LIMIT REGISTER
848      177772      PIRQ= 177772 ;;PROGRAM INTERRUPT REQUEST REGISTER
849      177570      DSWR= 177570 ;;HARDWARE SWITCH REGISTER
850      177570      DDISP= 177570 ;;HARDWARE DISPLAY REGISTER
851
852      ;*GENERAL PURPOSE REGISTER DEFINITIONS
853      000000      R0= %0      ;;GENERAL REGISTER
854      000001      R1= %1      ;;GENERAL REGISTER
855      000002      R2= %2      ;;GENERAL REGISTER
```

856	000003	R3=	%3	:: GENERAL REGISTER
857	000004	R4=	%4	:: GENERAL REGISTER
858	000005	R5=	%5	:: GENERAL REGISTER
859	000006	R6=	%6	:: GENERAL REGISTER
860	000007	R7=	%7	:: GENERAL REGISTER
861	000006	SP=	%6	:: STACK POINTER
862	000007	PC=	%7	:: PROGRAM COUNTER
863				
864		: *PRIORITY LEVEL DEFINITIONS		
865	000000	PR0=	0	:: PRIORITY LEVEL 0
866	000040	PR1=	40	:: PRIORITY LEVEL 1
867	000100	PR2=	100	:: PRIORITY LEVEL 2
868	000140	PR3=	140	:: PRIORITY LEVEL 3
869	000200	PR4=	200	:: PRIORITY LEVEL 4
870	000240	PR5=	240	:: PRIORITY LEVEL 5
871	000300	PR6=	300	:: PRIORITY LEVEL 6
872	000340	PR7=	340	:: PRIORITY LEVEL 7
873				
874		: *"SWITCH REGISTER" SWITCH DEFINITIONS		
875	100000	SW15=	100000	
876	040000	SW14=	40000	
877	020000	SW13=	20000	
878	010000	SW12=	10000	
879	004000	SW11=	4000	
880	002000	SW10=	2000	
881	001000	SW09=	1000	
882	000400	SW08=	400	
883	000200	SW07=	200	
884	000100	SW06=	100	
885	000040	SW05=	40	
886	000020	SW04=	20	
887	000010	SW03=	10	
888	000004	SW02=	4	
889	000002	SW01=	2	
890	000001	SW00=	1	
891		.EQUIV	SW09, SW9	
892		.EQUIV	SW08, SW8	
893		.EQUIV	SW07, SW7	
894		.EQUIV	SW06, SW6	
895		.EQUIV	SW05, SW5	
896		.EQUIV	SW04, SW4	
897		.EQUIV	SW03, SW3	
898		.EQUIV	SW02, SW2	
899		.EQUIV	SW01, SW1	
900		.EQUIV	SW00, SW0	
901				
902		: *DATA BIT DEFINITIONS (BIT00 TO BIT15)		
903	100000	BIT15=	100000	
904	040000	BIT14=	40000	
905	020000	BIT13=	20000	
906	010000	BIT12=	10000	
907	004000	BIT11=	4000	
908	002000	BIT10=	2000	
909	001000	BIT09=	1000	
910	000400	BIT08=	400	
911	000200	BIT07=	200	

912	000100	BIT06=	100	
913	000040	BIT05=	40	
914	000020	BIT04=	20	
915	000010	BIT03=	10	
916	000004	BIT02=	4	
917	000002	BIT01=	2	
918	000001	BIT00=	1	
919		.EQUIV	BIT09, BIT9	
920		.EQUIV	BIT08, BIT8	
921		.EQUIV	BIT07, BIT7	
922		.EQUIV	BIT06, BIT6	
923		.EQUIV	BIT05, BIT5	
924		.EQUIV	BIT04, BIT4	
925		.EQUIV	BIT03, BIT3	
926		.EQUIV	BIT02, BIT2	
927		.EQUIV	BIT01, BIT1	
928		.EQUIV	BIT00, BIT0	
929				
930		;BASIC "CPU" TRAP VECTOR ADDRESSES		
931	000004	ERRVEC=	4	TIME OUT AND OTHER ERRORS
932	000010	RESVEC=	10	RESERVED AND ILLEGAL INSTRUCTIONS
933	000014	TBITVEC=	14	"T" BIT
934	000014	TRIVEC=	14	TRACE TRAP
935	000014	BPTVEC=	14	BREAKPOINT TRAP (BPT)
936	000020	IOTVEC=	20	INPUT/OUTPUT TRAP (IOT) **SCOPE**
937	000024	PWRVEC=	24	POWER FAIL
938	000030	EMTVEC=	30	EMULATOR TRAP (EMT) **ERROR**
939	000034	TRAPVEC=	34	"TRAP" TRAP
940	000060	TKVEC=	60	TTY KEYBOARD VECTOR
941	000064	TPVEC=	64	TTY PRINTER VECTOR
942	000240	PIRQVEC=	240	PROGRAM INTERRUPT REQUEST VECTOR
943				
944		.SBTTL RM03 REGISTER BIT DEFINITIONS		
945				
946		;RMCS1 CONTROL STATUS REGISTER		
947				
948	004000	DVA	=	BIT11 ;DEVICE AVAILABLE-READ ONLY
949	000040	F4	=	BIT05 ;FUNCTION CODE
950	000020	F3	=	BIT04 ;FUNCTION CODE
951	000010	F2	=	BIT03 ;FUNCTION CODE
952	000004	F1	=	BIT02 ;FUNCTION CODE
953	000002	F0	=	BIT01 ;FUNCTION CODE
954	000001	GO	=	BIT00 ;GO BIT
955	000077	FNCMSK	=	000077 ;FUNCTION CODE MASK
956				
957		;FUNCTION CODES (BITS 01-05 OF RMCS1)		
958				
959	000000	NOP	=	000000 ;NOP COMMAND
960	000002	ILF02	=	000002 ;ILLEGAL COMMAND
961	000004	SEEK	=	000004 ;SEEK COMMAND
962	000006	RECAL	=	000006 ;RECALIBRATE COMMAND
963	000010	DRVCLR	=	000010 ;DRIVE CLEAR COMMAND
964	000012	RELEASE	=	000012 ;RELEASE COMMAND
965	000014	OFFSET	=	000014 ;OFFSET COMMAND
966	000016	RTC	=	000016 ;RETURN TO CENTERLINE COMMAND
967	000020	RIP	=	000020 ;READ IN PRESET COMMAND

968	000022	PAKACK	=	000022	;PACK ACKNOWLEDGE COMMAND
969	000022	PACACK	=	PAKACK	
970	000024	ILF24	=	000024	;ILLEGAL COMMAND
971	000026	ILF26	=	000026	;ILLEGAL COMMAND
972	000030	SEARCH	=	000030	;SEARCH COMMAND
973	000030	ILF30	=	000030	;ILLEGAL COMMAND
974	000032	ILF32	=	000032	;ILLEGAL COMMAND
975	000034	ILF34	=	000034	;ILLEGAL COMMAND
976	000036	ILF36	=	000036	;ILLEGAL COMMAND
977	000040	ILF40	=	000040	;ILLEGAL COMMAND
978	000042	ILF42	=	000042	;ILLEGAL COMMAND
979	000044	ILF44	=	000044	;ILLEGAL COMMAND
980	000046	ILF46	=	000046	;ILLEGAL COMMAND
981	000050	WCD	=	000050	;WRITE CHECK DATA COMMAND
982	000052	WCH	=	000052	;WRITE CHECK HEADER AND DATA
983	000054	ILF54	=	000054	;ILLEGAL COMMAND
984	000056	ILF56	=	000056	;ILLEGAL COMMAND
985	000060	WD	=	000060	;WRITE DATA COMMAND
986	000062	WH	=	000062	;WRITE HEADER AND DATA COMMAND
987	000064	ILF64	=	000064	;ILLEGAL COMMAND
988	000066	ILF66	=	000066	;ILLEGAL COMMAND
989	000070	RD	=	000070	;READ DATA COMMAND
990	000072	RH	=	000072	;READ HEADER AND DATA COMMAND
991	000074	ILF74	=	000074	;ILLEGAL COMMAND
992	000076	ILF76	=	000076	;ILLEGAL COMMAND
993					
994					
995					
996	002000	TA4	=	BIT10	;TRACK ADDRESS 4
997	001000	TA2	=	BIT09	;TRACK ADDRESS 2
998	000400	TA1	=	BIT08	;TRACK ADDRESS 1
999	000020	SA16	=	BIT04	;SECTOR ADDRESS 16
1000	000010	SA8	=	BIT03	;SECTOR ADDRESS 8
1001	000004	SA4	=	BIT02	;SECTOR ADDRESS 4
1002	000002	SA2	=	BIT01	;SECTOR ADDRESS 2
1003	000001	SA1	=	BIT00	;SECTOR ADDRESS 1
1004					
1005					
1006					
1007	003400	TADMSK	=	003400	;TRACK ADDRESS MASK
1008	000037	SADMSK	=	000037	;SECTOR ADDRESS MASK
1009					
1010					
1011					
1012	100000	ATA	=	BIT15	;ATTENTION ACTIVE
1013	040000	ERR	=	BIT14	;COMPOSITE ERROR
1014	020000	PIP	=	BIT13	;POSITIONING IN PROGRESS
1015	010000	MOL	=	BIT12	;MEDIUM ON LINE
1016	004000	WRL	=	BIT11	;WRITE LOCK
1017	002000	LBT	=	BIT10	;LAST BLOCK TRANSFERRED
1018	001000	PGM	=	BIT09	;PROGRAMMABLE
1019	000400	DPR	=	BIT08	;DRIVE PRESENT
1020	000200	DRY	=	BIT07	;DRIVE READY
1021	000100	VV	=	BIT06	;VOLUME VALID
1022	000001	OM	=	BIT00	;OFFSET MODE ACTIVE
1023					


```

1024
1025
1026      100000      ;RMR1 ERROR REGISTER #1
1027      040000      DCK      =      BIT15      ; DATA CHECK ERROR
1028      020000      UNS      =      BIT14      ; DRIVE UNSAFE
1029      010000      OPI      =      BIT13      ; OPERATION INCOMPLETE
1030      004000      DTE      =      BIT12      ; DRIVE TIMING ERROR
1031      002000      WLE      =      BIT11      ; WRITE LOCK ERROR
1032      001000      YRE      =      BIT10      ; INVALID ADDRESS ERROR
1033      000400      AOE      =      BIT09      ; ADDRESS OVERFLOW ERROR
1034      000200      HCRC     =      BIT08      ; HEADER CRC ERROR
1035      000100      HCE      =      BIT07      ; HEADER COMPARE ERROR
1036      000040      ECH      =      BIT06      ; ECC "HARD" ERROR
1037      000020      WCF      =      BIT05      ; WRITE CLOCK FAILURE
1038      000010      FER      =      BIT04      ; FORMAT ERROR
1039      000004      PAR      =      BIT03      ; PARITY ERROR
1040      000002      RMR      =      BIT02      ; REGISTER MODIFICATION REFUSED
1041      000001      ILR      =      BIT01      ; ILLEGAL REGISTER
1042      000001      ILF      =      BIT00      ; ILLEGAL FUNCTION
1043
1044      115760      NDTMSK   =      DCK!DTE!WLE!AOE!HCRC!HCE!ECH!WCF!FER
1045      ;"NDTMSK" IS USED TO MASK ERROR REGISTER 1 DURING NON - DATA
1046      ;COMMANDS, I.E., HOUSEKEEPING AND POSITIONING COMMANDS
1047
1048      ;RMS ATTENTION SUMMARY REGISTER
1049
1050      000377      ATNMSK   =      377      ; MASK FOR ATTENTION BITS
1051
1052      ;RMLA LOOK AHEAD REGISTER
1053
1054      002000      SC4      =      BIT10      ; SECTOR COUNT = 16
1055      001000      SC3      =      BIT09      ; SECTOR COUNT = 8
1056      000400      SC2      =      BIT08      ; SECTOR COUNT = 4
1057      000200      SC1      =      BIT07      ; SECTOR COUNT = 2
1058      000100      SC0      =      BIT06      ; SECTOR COUNT = 1
1059
1060      003700      SCTMSK   =      003700      ; SECTOR COUNT MASK
1061
1062      ;RMMR MAINTENANCE REGISTER
1063
1064      ; WRITE ONLY BITS
1065
1066      100000      DBCK      =      BIT15      ; DEBUG CLOCK
1067      040000      DBEN      =      BIT14      ; DEBUG CLOCK ENABLE
1068      020000      DEBL      =      BIT13      ; DIAGNOSTIC END OF BLOCK
1069      010000      DTO      =      BIT12      ; DIAGNOSTIC TIMEOUT
1070      004000      MCLK      =      BIT11      ; MAINTENANCE CLOCK
1071      002000      MRD      =      BIT10      ; READ DATA
1072      001000      MUR      =      BIT09      ; UNIT READY
1073      000400      MOC      =      BIT08      ; ON CYLINDER
1074      000200      MSER      =      BIT07      ; SEEK ERROR
1075      000100      MDF      =      BIT06      ; DRIVE FAULT
1076      000040      MS      =      BIT05      ; SECTOR PULSE
1077      000010      MWP      =      BIT03      ; WRITE PROTECT
1078      000004      MI      =      BIT02      ; INDEX PULSE
1079      000002      MSC      =      BIT01      ; SECTOR COMPARE
1080      000001      DMO      =      BIT00      ; DIAGNOSTIC MODE

```

```

1080
1081
1082
1083      100000      OCC      =      BIT15      ;OCCUPIED
1084      040000      RG       =      BIT14      ;RUN AND GO
1085      020000      EBL      =      BIT13      ;END OF BLOCK
1086      010000      REX      =      BIT12      ;EXCEPTION
1087      004000      ESRC     =      BIT11      ;ENABLE SEARCH
1088      002000      PLFS     =      BIT10      ;LOOKING FOR SYNC
1089      001000      ECRC     =      BIT09      ;ENABLE CRC OUT
1090      000400      PDA      =      BIT08      ;DATA AREA
1091      000200      PHA      =      BIT07      ;HEADER AREA
1092      000100      CONT     =      BIT06      ;CONTINUE
1093      000040      WC       =      BIT05      ;WORD CLOCK
1094      000020      EECC     =      BIT04      ;ENABLE ECC OUT
1095      000010      MWD      =      BIT03      ;WRITE DATA BIT
1096      000004      LS       =      BIT02      ;LAST SECTOR
1097      000002      LST      =      BIT01      ;LAST SECTOR AND TRACK
1098      000001      DMD      =      BIT00      ;DIAGNOSTIC MODE
1099
1100      ;RMDT  DRIVE TYPE REGISTER
1101
1102      100000      NSA       =      BIT15      ;NOT SECTOR ADDRESSED=0
1103      040000      TAP      =      BIT14      ;TAPE DRIVE = 0
1104      020000      MOH      =      BIT13      ;MOVING HEAD = 1
1105      004000      DRQ      =      BIT11      ;DRIVE REQUEST REQUIRED
1106
1107      020024      SNGPRT    =      020024      ;SINGLE PORT DRIVE TYPE
1108      024024      DULPRT    =      024024      ;DUAL PORT DRIVE TYPE
1109
1110      ;RMOF  OFFSET REGISTER
1111
1112      010000      FMT16     =      BIT12      ;16 BIT WORD FORMAT
1113      004000      ECI       =      BIT11      ;ECC INHIBIT
1114      002000      HCI       =      BIT10      ;HEADER COMPARE INHIBIT
1115      000200      OFD       =      BIT07      ;OFFSET FORWARD
1116
1117
1118      ;RMDC  DESIRED CYLINDER ADDRESS REGISTER
1119      001777      CYLMSK    =      1777      ;MASK FOR CYLINDER ADDRESS
1120
1121      ;RMMR2  MAINTENANCE REGISTER #2
1122
1123      ;      READ ONLY BITS
1124      100000      RQA       =      BIT15      ;PORT A REQUEST
1125      040000      RQB       =      BIT14      ;PORT B REQUEST
1126      020000      TAG      =      BIT13      ;TAG CONTROL
1127      010000      TST      =      BIT12      ;COMMAND SEQUENCE TEST BIT
1128      004000      CC        =      BIT11      ;CONTROL OR CYLINDER TAG
1129      002000      CH        =      BIT10      ;CONTROL OR HEAD TAG
1130      001000      BB09      =      BIT09      ;TAG BUS
1131      000400      BB08      =      BIT08      ;TAG BUS
1132      000200      BB07      =      BIT07      ;TAG BUS
1133      000100      BB06      =      BIT06      ;TAG BUS
1134      000040      BB05      =      BIT05      ;TAG BUS
1135      000020      BB04      =      BIT04      ;TAG BUS

```

1136	000010	8803	=	BIT03	;TAG BUS
1137	000004	8802	=	BIT02	;TAG BUS
1138	000002	8801	=	BIT01	;TAG BUS
1139	000001	8800	=	BIT00	;TAG BUS
1140					
1141					
1142		;RMER2 ERROR REGISTER 2			
1143					
1144	100000	BSE	=	BIT15	;BAD SECTOR ERROR
1145	040000	SKI	=	BIT14	;SEEK INCOMPLETE
1146	020000	OPE	=	BIT13	;OPERATOR PLUG ERROR
1147	010000	IVC	=	BIT12	;INVALID COMMAND ERROR
1148	004000	LSC	=	BIT11	;LOSS OF SYSTEM CLOCK
1149	002000	LBC	=	BIT10	;LOSS OF BIT CLOCK
1150	000200	DVC	=	BIT07	;DEVICE CHECK
1151	000010	DPE	=	BIT03	;DATA PARITY ERROR
1152					
1153		.SBTTL PROGRAM MNEMONICS			
1154					
1155	100000	MSE	=	BIT15	;MANUFACTURING DETECTED SECTOR ERROR
1156	040000	USE	=	BIT14	;USER DETECTED SECTOR ERROR
1157					
1158		.SBTTL RM03 REGISTER INDEX VALUES			
1159					
1160	000000	RMCS1	=	00	;CONTROL STATUS REGISTER
1161	000006	RMDA	=	06	;DISK ADDRESS REGISTER
1162	000012	RMOS	=	12	;DRIVE STATUS REGISTER
1163	000014	RMER1	=	14	;ERROR REGISTER 1
1164	000016	RMA5	=	16	;ATTENTION SUMMARY REGISTER
1165	000020	RMLA	=	20	;LOOK AHEAD REGISTER
1166	000024	RMMR1	=	24	;MAINTENANCE REGISTER
1167	000026	RMDT	=	26	;DRIVE TYPE REGISTER
1169	000030	RMSN	=	30	;SERIAL NUMBER REGISTER
1169	000032	RMOF	=	32	;OFFSET REGISTER
1170	000034	RMOC	=	34	;DESIRED CYLINDER REGISTER
1171	000036	RMCC	=	36	;CURRENT CYLINDER REGISTER
1172	000040	RMMR2	=	40	;MAINTENANCE REGISTER 2
1173	000042	RMER2	=	42	;ERROR REGISTER 2
1174	000044	RMEC1	=	44	;ECC POSITION REGISTER
1175	000046	RMEC2	=	46	;ECC PATTERN REGISTER
1176	000050	ILRG50	=	50	;ILLEGAL REGISTER 50
1177	000052	ILRG52	=	52	;ILLEGAL REGISTER 52
1178	000054	ILRG54	=	54	;ILLEGAL REGISTER 54
1179	000056	ILRG56	=	56	;ILLEGAL REGISTER 56
1180	000060	ILRG60	=	60	;ILLEGAL REGISTER 60
1181	000062	ILRG62	=	62	;ILLEGAL REGISTER 62
1182	000064	ILRG64	=	64	;ILLEGAL REGISTER 64
1183	000066	ILRG66	=	66	;ILLEGAL REGISTER 66
1184	000070	ILRG70	=	70	;ILLEGAL REGISTER 70
1185	000072	ILRG72	=	72	;ILLEGAL REGISTER 72
1186	000074	ILRG74	=	74	;ILLEGAL REGISTER 74
1187	000076	ILRG76	=	76	;ILLEGAL REGISTER 76
1188					
1189					
1190	000077	IDXMSK	=	77	;MASK FOR REGISTER INDEX NUMBER
1191					

```

1192 .SBTTL RH CONTROLLER REGISTER BIT DEFINITIONS
1193 ;RMCS1 CONTROL STATUS REGISTER #1
1194
1195 SC = BIT15 ;SPECIAL CONDITION-READ ONLY
1196 TRE = BIT14 ;TRANSFER ERROR
1197 MCPE = BIT13 ;MASSBUS CONTROL BUS PARITY
1198 ;ERROR-READ ONLY
1199 PSEL = BIT10 ;PORT B SELECT
1200 A17 = BIT09 ;ADDRESS EXTENSION
1201 A16 = BIT08 ;ADDRESS EXTENSION
1202 RDY = BIT07 ;READY-READ ONLY
1203 IE = BIT06 ;INTERRUPT ENABLE
1204
1205 ;RMCS2 RH CONTROL STATUS REGISTER #2
1206
1207 DLT = BIT15 ;DATA LATE-READ ONLY
1208 WCE = BIT14 ;WRITE CHECK ERROR-READ ONLY
1209 UPE = BIT13 ;UNIBUS PARITY ERROR
1210 NED = BIT12 ;NONEXISTANT DRIVE-READ ONLY
1211 NEM = BIT11 ;NONEXISTANT MEMORY-READ ONLY
1212 PGE = BIT10 ;PROGRAM ERROR-READ ONLY
1213 MXF = BIT09 ;MISC TRANSFER
1214 MDPE = BIT08 ;MASSBUS DATA BUS PARITY
1215 ;ERROR-READ ONLY
1216 OR = BIT07 ;OUTPUT READY-READ ONLY
1217 IR = BIT06 ;INPUT READY-READ ONLY
1218 CLR = BIT05 ;CONTROLLER CLEAR
1219 PAT = BIT04 ;PARITY TEST
1220 BAI = BIT03 ;UNIBUS ADDRESS INCREMENT
1221 ;INHIBIT
1222 U2 = BIT02 ;UNIT SELECT
1223 U1 = BIT01 ;UNIT SELECT
1224 U0 = BIT00 ;UNIT SELECT
1225
1226 ;UNIT SELECT MASK
1227 UNTMSK = 7 ;UNIT SELECT MASK
1228
1229 ;RMCS3 RH70 CONTROL STATUS REGISTER #3
1230 APE = BIT15 ;ADDRESS PARITY ERROR
1231 DPEHI = BIT14 ;DATA PARITY ERROR HIGH WORD
1232 DPELO = BIT13 ;DATA PARITY ERROR LOW WORD
1233 WCEHI = BIT12 ;WRITE CHECK ERROR HIGH WORD
1234 WCELO = BIT11 ;WRITE CHECK ERROR LOW WORD
1235 DBL = BIT10 ;DOUBLE WORD TRANSFER
1236 IE = BIT06 ;INTERRUPT ENABLE
1237 IPCK3 = BIT03 ;INVERT PARITY CHECK
1238 IPCK2 = BIT02 ;INVERT PARITY CHECK
1239 IPCK1 = BIT01 ;INVERT PARITY CHECK
1240 IPCK0 = BIT00 ;INVERT PARITY CHECK
1241
1242 .SBTTL RH CONTROLLER REGISTER INDEX VALUES
1243
1244 RMCS1 = 00 ;CONTROL STATUS REGISTER
1245 RMWC = 02 ;WORD COUNT REGISTER
1246 RMBA = 04 ;BUS ADDRESS REGISTER
1247 RMCS2 = 10 ;CONTROLLER STATUS REGISTER

```

```

1248      000022      RMOB      =      22      ;DATA BUFFER
1249      000050      RMBAR      =      50      ;BUS ADDRESS EXTENSION
1250      000052      RMC53      =      52      ;CONTROL STATUS REGISTER #3
1251
1252      176700      ABASE      =      176700      ;UNIBUS ADDRESS
1253      120254      AVECT1      =      120254      ;UNIBUS VECTOR ADDRESS AND PRIORITY
1254
1255
1256      .SBTTL  TRAP CATCHER
1257
1258      000000      .=0
1259      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
1260      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
1261      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
1262
1263      000174      000174      .=174
1264      000176      000000      DISPREG: .WORD 0      ;;SOFTWARE DISPLAY REGISTER
1265      000000      000000      SWREG: .WORD 0      ;;SOFTWARE SWITCH REGISTER
1266
1267      .SBTTL  ACT11 HOOKS
1268
1269      ;*****
1270      ;HOOKS REQUIRED BY ACT11
1271      000200      $SVPC=.      ;SAVE PC
1272      000046      .=46
1273      000046      033156      $ENDAD      ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
1274      000052      000052      .=52
1275      000052      000000      .WORD 0      ;;2)SET LOC.52 TO ZERO
1276      000200      000200      .=$SVPC      ;; RESTORE PC
1277
1278      .SBTTL  STARTING ADDRESS
1279
1280      ;THE PROGRAM STARTS AT LOCATION 200
1281      000200      000200      =
1282      000200      000137      005324      JMP      START      ;JUMP TO START OF PROGRAM
1283
1284      001100      .=1100
1285      .SBTTL  APT PARAMETER BLOCK
1286
1287      ;*****
1288      ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
1289      ;*****
1290      001100      001100      .SX=.      ;SAVE CURRENT LOCATION
1291      000024      000024      .=24      ;SET POWER FAIL TO POINT TO START OF PROGRAM
1292      000024      000200      200      ;FOR APT START UP
1293      000044      000044      .=44      ;POINT TO APT INDIRECT ADDRESS PNTR.
1294      000044      001100      $APTHDR      ;POINT TO APT HEADER BLOCK
1295      001100      001100      .=$X      ;RESET LOCATION COUNTER
1296
1297      ;*****
1298      ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-POPI1 DIAGNOSTIC
1299      ;INTERFACE SPEC.
1300
1301      001100      000000      $APTHD:
1302      001102      001222      $HIBTS: .WORD 0      ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
1303      001104      000001      $MBADR: .WORD $MAIL      ;;ADDRESS OF APT MAILBOX (BITS 0-15)
1304      $STMT: .WORD 1      ;;RUN TIM OF LONGEST TEST

```

H03

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 33
APT PARAMETER BLOCK

SEQ 0033

1304 001106 000002
1305 001110 000002
1306 001112 000042
1307 001114 001114

\$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
TAGADR = .WORD \$ETEND-\$MAIL/2 ;;LENGTH MAILBOX-ETABLE (WORDS)


```

1308 .SBTTL COMMON TAGS
1309
1310 ;*****
1311 ;THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
1312 ;USED IN THE PROGRAM.
1313
1314 001114 .=TAGADR
1315 SCMTAG: .WORD 0 ;;START OF COMMON TAGS
1316 001114 000000 ;;CONTAINS THE TEST NUMBER
1317 001116 000 ;;CONTAINS ERROR FLAG
1318 001117 000 ;;CONTAINS SUBTEST ITERATION COUNT
1319 001120 000000 ;;CONTAINS SCOPE LOOP ADDRESS
1320 001122 000000 ;;CONTAINS SCOPE RETURN FOR ERRORS
1321 001124 000000 ;;CONTAINS TOTAL ERRORS DETECTED
1322 001126 000000 ;;CONTAINS ITEM CONTROL BYTE
1323 001130 000 ;;CONTAINS MAX. ERRORS PER TEST
1324 001131 001 ;;CONTAINS PC OF LAST ERROR INSTRUCTION
1325 001132 000000 ;;CONTAINS ADDRESS OF 'GOOD' DATA
1326 001134 000000 ;;CONTAINS ADDRESS OF 'BAD' DATA
1327 001136 000000 ;;CONTAINS 'GOOD' DATA
1328 001140 000000 ;;CONTAINS 'BAD' DATA
1329 001142 000000 ;;RESERVED--NOT TO BE USED
1330 001144 000000
1331 001146 000000
1332 001150 000 ;;AUTOMATIC MODE INDICATOR
1333 001151 000 ;;INTERRUPT MODE INDICATOR
1334 001152 000000
1335 001154 177570 SWR: .WORD DSWR ;;ADDRESS OF SWITCH REGISTER
1336 001156 177570 DISPLAY: .WORD DOISP ;;ADDRESS OF DISPLAY REGISTER
1337 001160 177550 STKS: 177560 ;;TTY KBO STATUS
1338 001162 177562 STKB: 177562 ;;TTY KBO BUFFER
1339 001164 177564 STPS: 177564 ;;TTY PRINTER STATUS REG. ADDRESS
1340 001166 177566 STPB: 177566 ;;TTY PRINTER BUFFER REG. ADDRESS
1341 001170 000 $NULL: .BYTE 0 ;;CONTAINS NULL CHARACTER FOR FILLS
1342 001171 002 $FILLS: .BYTE 2 ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
1343 001172 012 $FILLC: .BYTE 12 ;;INSERT FILL CHARS. AFTER A "LINE FEED"
1344 001173 000 STPFLG: .BYTE 0 ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
1345 001174 000000 STMP0: .WORD 0 ;;USER DEFINED
1346 001176 000000 STMP1: .WORD 0 ;;USER DEFINED
1347 001200 000000 STMP2: .WORD 0 ;;USER DEFINED
1348 001202 000000 STMP3: .WORD 0 ;;USER DEFINED
1349 001204 000000 STMP4: .WORD 0 ;;USER DEFINED
1350 001206 000000 STIMES: 0 ;;MAX. NUMBER OF ITERATIONS
1351 001210 000000 $ESCAPE: 0 ;;ESCAPE ON ERROR ADDRESS
1352 001212 177607 000377 $BELL: .ASCIZ <207><377><377> ;;CODE FOR BELL
1353 001216 077 $QJES: .ASCII /?/ ;;QUESTION MARK
1354 001217 015 $CRLF: .ASCII <15> ;;CARRIAGE RETURN
1355 001220 000012 $LF: .ASCIZ <12> ;;LINE FEED
1356 ;*****
1357 .SBTTL APT MAILBOX-ETABLE
1358
1359 ;*****
1360 .EVEN
1361 001222 $MAIL: ;;APT MAILBOX
1362 001222 000000 $MSGTY: .WORD AMSGTY ;;MESSAGE TYPE CODE
1363 001224 000000 $FATAL: .WORD AFATAL ;;FATAL ERROR NUMBER

```

1364	001226	000000	\$TESTN:	.WORD	ATESTN	:: TEST NUMBER
1365	001230	000000	\$PASS:	.WORD	APASS	:: PASS COUNT
1366	001232	000000	\$DEVCT:	.WORD	ADEVCT	:: DEVICE COUNT
1367	001234	000000	\$UNIT:	.WORD	AUNIT	:: I/O UNIT NUMBER
1368	001236	000000	\$MSGAD:	.WORD	AMSGAD	:: MESSAGE ADDRESS
1369	001240	000000	\$MSGLG:	.WORD	AMSGLG	:: MESSAGE LENGTH
1370	001242		\$ETABLE:			:: APT ENVIRONMENT TABLE
1371	001242	000	\$ENV:	.BYTE	AENV	:: ENVIRONMENT BYTE
1372	001243	000	\$ENVM:	.BYTE	AENVM	:: ENVIRONMENT MODE BITS
1373	001244	000000	\$SWREG:	.WORD	ASWREG	:: APT SWITCH REGISTER
1374	001246	000000	\$USWR:	.WORD	AUSWR	:: USER SWITCHES
1375	001250	000000	\$CPUOP:	.WORD	ACPUOP	:: CPU TYPE, OPTIONS
1376			::			BITS 15-11=CPU TYPE
1377			::			11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
1378			::			11/70=06, PDQ=07, Q=10
1379			::			BIT 10=REAL TIME CLOCK
1380			::			BIT 9=FLOATING POINT PROCESSOR
1381			::			BIT 8=MEMORY MANAGEMENT
1382	001252	000	\$MAMS1:	.BYTE	AMAMS1	:: HIGH ADDRESS, M.S. BYTE
1383	001253	000	\$MTYP1:	.BYTE	AMTYP1	:: MEM. TYPE, BLK#1
1384			::			MEM. TYPE BYTE -- (HIGH BYTE)
1385			::			900 NSEC CORE=001
1386			::			300 NSEC BIPOLAR=002
1387			::			500 NSEC MOS=003
1388	001254	000000	\$MAOR1:	.WORD	AMAOR1	:: HIGH ADDRESS, BLK#1
1389			::			MEM. LAST ADDRESS, 3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
1390	001256	000	\$MAMS2:	.BYTE	AMAMS2	:: HIGH ADDRESS, M.S. BYTE
1391	001257	000	\$MTYP2:	.BYTE	AMTYP2	:: MEM. TYPE, BLK#2
1392	001260	000000	\$MAOR2:	.WORD	AMAOR2	:: MEM. LAST ADDRESS, BLK#2
1393	001262	000	\$MAMS3:	.BYTE	AMAMS3	:: HIGH ADDRESS, M.S. BYTE
1394	001263	000	\$MTYP3:	.BYTE	AMTYP3	:: MEM. TYPE, BLK#3
1395	001264	000000	\$MAOR3:	.WORD	AMAOR3	:: MEM. LAST ADDRESS, BLK#3
1396	001266	000	\$MAMS4:	.BYTE	AMAMS4	:: HIGH ADDRESS, M.S. BYTE
1397	001267	000	\$MTYP4:	.BYTE	AMTYP4	:: MEM. TYPE, BLK#4
1398	001270	000000	\$MAOR4:	.WORD	AMAOR4	:: MEM. LAST ADDRESS, BLK#4
1399	001272	120254	\$VECT1:	.WORD	AVECT1	:: INTERRUPT VECTOR#1, BUS PRIORITY#1
1400	001274	000000	\$VECT2:	.WORD	AVECT2	:: INTERRUPT VECTOR#2, BUS PRIORITY#2
1401	001276	176700	\$BASE:	.WORD	ABASE	:: BASE ADDRESS OF EQUIPMENT UNDER TEST
1402	001300	000000	\$DEVN:	.WORD	ADEVN	:: DEVICE MAP
1403	001302	000000	\$CDW1:	.WORD	ACDW1	:: CONTROLLER DESCRIPTION WORD#1
1404	001304	000000	\$CDW2:	.WORD	ACDW2	:: CONTROLLER DESCRIPTION WORD#2
1405	001306	000000	\$DDW0:	.WORD	ADDW0	:: DEVICE DESCRIPTOR WORD#0
1406	001310	000000	\$DDW1:	.WORD	ADDW1	:: DEVICE DESCRIPTOR WORD#1
1407	001312	000000	\$DDW2:	.WORD	ADDW2	:: DEVICE DESCRIPTOR WORD#2
1408	001314	000000	\$DDW3:	.WORD	ADDW3	:: DEVICE DESCRIPTOR WORD#3
1409	001316	000000	\$DDW4:	.WORD	ADDW4	:: DEVICE DESCRIPTOR WORD#4
1410	001320	000000	\$DDW5:	.WORD	ADDW5	:: DEVICE DESCRIPTOR WORD#5
1411	001322	000000	\$DDW6:	.WORD	ADDW6	:: DEVICE DESCRIPTOR WORD#6
1412	001324	000000	\$DDW7:	.WORD	ADDW7	:: DEVICE DESCRIPTOR WORD#7
1413	001326		\$ETEND:			
1414			.MEXIT			
1415	001326	000000	CTLFG:	.WORD	0	
1416						
1417						
1418						
1419						

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

1420 001330

GETBUF:

1421
1422
1423 001330 000000
1424 001332 000000
1425 001334 000000
1426 001336 000000
1427 001340 000000
1428 001342 000000
1429 001344 000000
1430 001346 000000
1431 001350 000000
1432 001352 000000
1433 001354 000000
1434 001356 000000
1435 001360 000000
1436 001362 000000
1437 001364 000000
1438 001366 000000
1439 001370 000000
1440 001372 000000
1441 001374 000000
1442 001376 000000
1443

;REGISTER INPUT BUFFER

RMCS11: .WORD
RMWC1: .WORD
RMB1: .WORD
RMD1: .WORD
RMCS21: .WORD
RMC51: .WORD
RMR11: .WORD
RMA1: .WORD
RMLA1: .WORD
RMD1: .WORD
RMR11: .WORD
RMT1: .WORD
RMSN1: .WORD
RMOF1: .WORD
RMO1: .WORD
RMCC1: .WORD
RMR21: .WORD
RMR21: .WORD
RMEC11: .WORD
RMEC21: .WORD

;CONTROL STATUS REGISTER
;WORD COUNT REGISTER
;BUS ADDRESS REGISTER
;DISK ADDRESS REGISTER
;CONTROLLER STATUS REGISTER
;DRIVE STATUS REGISTER
;ERROR REGISTER 1
;ATTENTION SUMMARY REGISTER
;LOOK AHEAD REGISTER
;DATA BUFFER
;MAINTENANCE REGISTER #1
;DRIVE TYPE REGISTER
;SERIAL NUMBER REGISTER
;OFFSET REGISTER
;DESIRED CYLINDER REGISTER
;CURRENT CYLINDER REGISTER
;MAINTENANCE REGISTER #2
;ERROR REGISTER 2
;ECC POSITION REGISTER
;ECC PATTERN REGISTER

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

1446 001400

PUTBUF:

1447
1448
1449
1450 001400 000000
1451 001402 000000
1452 001404 000000
1453 001406 000000
1454 001410 000000
1455 001412 000000
1456 001414 000000
1457 001416 000000
1458 001420 000000
1459 001422 000000
1460 001424 000000
1461 001426 000000
1462 001430 000000
1463 001432 000000
1464 001434 000000
1465 001436 000000
1466 001440 000000
1467 001442 000000
1468 001444 000000
1469 001446 000000
1470

;REGISTER OUTPUT BUFFER

RMCS10: .WORD
RMWC0: .WORD
RMA0: .WORD
RMA0: .WORD
RMC520: .WORD
RMD50: .WORD
RMR10: .WORD
RMA50: .WORD
RMLA0: .WORD
RMDA0: .WORD
RMR10: .WORD
RMDT0: .WORD
RMSN0: .WORD
RMOF0: .WORD
RMO0: .WORD
RMCC0: .WORD
RMR20: .WORD
RMR20: .WORD
RMEC10: .WORD
RMEC20: .WORD

;CONTROL STATUS REGISTER
;WORD COUNT REGISTER
;BUS ADDRESS REGISTER
;DISK ADDRESS REGISTER
;CONTROLLER STATUS REGISTER
;DRIVE STATUS REGISTER
;ERROR REGISTER 1
;ATTENTION SUMMARY REGISTER
;LOOK AHEAD REGISTER
;DATA BUFFER
;MAINTENANCE REGISTER #1
;DRIVE TYPE REGISTER
;SERIAL NUMBER REGISTER
;OFFSET REGISTER
;DESIRED CYLINDER REGISTER
;CURRENT CYLINDER REGISTER
;MAINTENANCE REGISTER #2
;ERROR REGISTER 2
;ECC POSITION REGISTER
;ECC PATTERN REGISTER

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

1475 001450 000012

TSTQUE: .BLKW 10. ;TEST QUE

```

1476
1477
1478
1479 001474 000000
1480
1481
1482
1483 001476 000000
1484 001500 000000
1485 001502 000000
1486 001504 000000
1487
1488
1489
1490
1491 001506 000027
1492
1493
1494
1495
1496 001535 000027
1497
1498
1499

```

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

;LOCATIONS "ASNOC" AND "ASNOC" CONTAIN THE CYLINDER, TRACK AND SECTOR
 ;ADDRESS ASSIGNED BY THE BAD SECTOR MODULE.
 ASNOC: .WORD ;ASSIGNED DESIRED CYLINDER
 ASNOA: .WORD ;ASSIGNED TRACK, AND SECTOR
 CLKADR: .WORD ;UNIBUS ADDRESS OF KW11 CLOCK
 CLKVCT: .WORD ;VECTOR ADDRESS OF KW11 CLOCK

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

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

;PUT TAGS HERE

.SBTTL ERROR POINTER TABLE

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

001564

1517			;ERROR	1	WRONG UNIT SELECTED
1518	001564	065432		EMT1	
1519	001566	071524		EHT1	
1520	001570	071650		EDT1	
1521	001572	071740		EFT1	
1522					
1523					
1524			;ERROR	2	DEVICE WENT UNAVAILABLE
1525	001574	065436		EMT2	
1526	001576	071524		EHT1	
1527	001600	071650		EDT1	
1528	001602	071740		EFT1	
1529					
1530					
1531			;ERROR	3	DEVICE WENT NONEXISTENT
1532	001604	065444		EMT3	
1533	001606	071524		EHT1	
1534	001610	071650		EDT1	
1535	001612	071740		EFT1	
1536					
1537					
1538			;ERROR	4	CONTROLLER NOT READY
1539	001614	065452		EMT4	
1540	001616	071524		EHT1	
1541	001620	071650		EDT1	
1542	001622	071740		EFT1	
1543					
1544					
1545			;ERROR	5	DRIVE NOT READY AND GO NOT RESET
1546	001624	065460		EMT5	
1547	001626	071524		EHT1	
1548	001630	071650		EDT1	
1549	001632	071740		EFT1	
1550					
1551					
1552			;ERROR	6	UNEXPECTED VALUE FOR "ATA" STATUS
1553	001634	065466		EMT6	
1554	001636	071524		EHT1	
1555	001640	071650		EDT1	
1556	001642	071740		EFT1	
1557					
1558					
1559			;ERROR	7	BUS TIMEOUT TRYING TO READ OR WRITE REGISTER
1560	001644	065474		EMT7	
1561	001646	000000		0	
1562	001650	000000		0	
1563	001652	000000		0	
1564					
1565					
1566			;ERROR	10	DRIVE NOT READY BUT GO IS RESET
1567	001654	065502		EMT10	
1568	001656	071524		EHT1	
1569	001660	071650		EDT1	
1570	001662	071740		EFT1	
1571					
1572					

1573			;ERROR	11	GO NOT RESET BUT DRIVE IS READY
1574	001664	065506		EMT11	
1575	001666	071524		EHT1	
1576	001670	071650		EDT1	
1577	001672	071740		EFT1	
1578					
1579					
1580			;ERROR	12	INCORRECT FUNCTION CODE
1581	001674	065512		EMT12	
1582	001676	071524		EHT1	
1583	001700	071650		EDT1	
1584	001702	071740		EFT1	
1585					
1586					
1587			;ERROR	13	PARITY ERROR READING REMOTE REGISTERS
1588	001704	065520		EMT13	
1589	001706	071524		EHT1	
1590	001710	071650		EDT1	
1591	001712	071740		EFT1	
1592					
1593					
1594			;ERROR	14	TRANSFER ERROR IS INCORRECT
1595	001714	065532		EMT14	
1596	001716	071524		EHT1	
1597	001720	071650		EDT1	
1598	001722	071740		EFT1	
1599					
1600					
1601			;ERROR	15	INCORRECT WORD COUNT
1602	001724	065540		EMT15	
1603	001726	071524		EHT1	
1604	001730	071650		EDT1	
1605	001732	071740		EFT1	
1606					
1607					
1608			;ERROR	16	INCORRECT BUS ADDRESS
1609	001734	065546		EMT16	
1610	001736	071524		EHT1	
1611	001740	071650		EDT1	
1612	001742	071740		EFT1	
1613					
1614					
1615			;ERROR	17	INCORRECT LBT STATUS
1616	001744	065556		EMT17	
1617	001746	071524		EHT1	
1618	001750	071650		EDT1	
1619	001752	071740		EFT1	
1620					
1621					
1622			;ERROR	20	INCORRECT AOE
1623	001754	065566		EMT20	
1624	001756	071524		EHT1	
1625	001760	071650		EDT1	
1626	001762	071740		EFT1	
1627					
1628					

1629			;ERROR	21	INCORRECT DISK ADDRESS
1630	001764	065576		EMT21	
1631	001766	071524		EHT1	
1632	001770	071650		EDT1	
1633	001772	071740		EFT1	
1634					
1635					
1636			;ERROR	22	INCORRECT CYLINDER ADDRESS
1637	001774	065606		EMT22	
1638	001776	071524		EHT1	
1639	002000	071650		EDT1	
1640	002002	071740		EFT1	
1641					
1642					
1643			;ERROR	23	INCORRECT WLE STATUS
1644	002004	065616		EMT23	
1645	002006	071524		EHT1	
1646	002010	071650		EDT1	
1647	002012	071740		EFT1	
1648					
1649					
1650			;ERROR	24	INCORRECT LPE STATUS
1651	002014	065626		EMT24	
1652	002016	071524		EHT1	
1653	002020	071650		EDT1	
1654	002022	071740		EFT1	
1655					
1656					
1657			;ERROR	25	INCORRECT WCF STATUS
1658	002024	065636		EMT25	
1659	002026	071524		EHT1	
1660	002030	071650		EDT1	
1661	002032	071740		EFT1	
1662					
1663					
1664			;ERROR	26	INCORRECT WCE STATUS
1665	002034	065646		EMT26	
1666	002036	071524		EHT1	
1667	002040	071650		EDT1	
1668	002042	071740		EFT1	
1669					
1670					
1671			;ERROR	27	INCORRECT MOPE STATUS
1672	002044	065656		EMT27	
1673	002046	071524		EHT1	
1674	002050	071650		EDT1	
1675	002052	071740		EFT1	
1676					
1677					
1678			;ERROR	30	INCORRECT DCK STATUS
1679	002054	065666		EMT30	
1680	002056	071524		EHT1	
1681	002060	071650		EDT1	
1682	002062	071740		EFT1	
1683					
1684					

1685			;ERROR	31	INCORRECT ECH STATUS
1686	002064	065676		EMT31	
1687	002066	071524		EHT1	
1688	002070	071650		EDT1	
1689	002072	071740		EFT1	
1690					
1691					
1692			;ERROR	32	DLT SHOULD NOT BE SET
1693	002074	065706		EMT32	
1694	002076	071524		EHT1	
1695	002100	071650		EDT1	
1696	002102	071740		EFT1	
1697					
1698					
1699			;ERROR	33	MXF SHOULD NOT BE SET
1700	002104	065716		EMT33	
1701	002106	071524		EHT1	
1702	002110	071650		EDT1	
1703	002112	071740		EFT1	
1704					
1705					
1706			;ERROR	34	DTE SHOULD NOT BE SET
1707	002114	065726		EMT34	
1708	002116	071524		EHT1	
1709	002120	071650		EDT1	
1710	002122	071740		EFT1	
1711					
1712					
1713			;ERROR	35	INCORRECT MCRC STATUS
1714	002124	065736		EMT35	
1715	002126	071524		EHT1	
1716	002130	071650		EDT1	
1717	002132	071740		EFT1	
1718					
1719					
1720			;ERROR	36	INCORRECT MCE STATUS
1721	002134	065746		EMT36	
1722	002136	071524		EHT1	
1723	002140	071650		EDT1	
1724	002142	071740		EFT1	
1725					
1726					
1727			;ERROR	37	INCORRECT FER STATUS
1728	002144	065756		EMT37	
1729	002146	071524		EHT1	
1730	002150	071650		EDT1	
1731	002152	071740		EFT1	
1732					
1733					
1734			;ERROR	40	DPE SHOULD NOT BE SET (NOT A DATA COMMAND)
1735	002154	065766		EMT40	
1736	002156	071524		EHT1	
1737	002160	071650		EDT1	
1738	002162	071740		EFT1	
1739					
1740					

1741			;ERROR 41	LOST "MOL" DURING PACK ACKNOWLEDGE
1742	002164	065774	EMT41	
1743	002166	071524	EHT1	
1744	002170	071650	EDT1	
1745	002172	071740	EFT1	
1746				
1747				
1748			;ERROR 42	UNSAFE ERROR DURING PACK ACKNOWLEDGE
1749	002174	066004	EMT42	
1750	002176	071524	EHT1	
1751	002200	071650	EDT1	
1752	002202	071740	EFT1	
1753				
1754				
1755			;ERROR 43	"OPI" ERROR DURING PACK ACKNOWLEDGE
1756	002204	066016	EMT43	
1757	002206	071524	EHT1	
1758	002210	071650	EDT1	
1759	002212	071740	EFT1	
1760				
1761				
1762			;ERROR 44	"RMR" ERROR DURING PACK ACKNOWLEDGE
1763	002214	066026	EMT44	
1764	002216	071524	EHT1	
1765	002220	071650	EDT1	
1766	002222	071740	EFT1	
1767				
1768				
1769			;ERROR 45	"ILR" ERROR DURING PACK ACKNOWLEDGE
1770	002224	066036	EMT45	
1771	002226	071524	EHT1	
1772	002230	071650	EDT1	
1773	002232	071740	EFT1	
1774				
1775				
1776			;ERROR 46	"ILF" ERROR DURING PACK ACKNOWLEDGE
1777	002234	066046	EMT46	
1778	002236	071524	EHT1	
1779	002240	071650	EDT1	
1780	002242	071740	EFT1	
1781				
1782				
1783			;ERROR 47	COMPOSITE ERROR STATUS IS INCORRECT
1784	002244	066056	EMT47	
1785	002246	071524	EHT1	
1786	002250	071650	EDT1	
1787	002252	071740	EFT1	
1788				
1789				
1790			;ERROR 50	PARITY ERROR WRITING REMOTE REGISTERS
1791	002254	066064	EMT50	
1792	002256	071524	EHT1	
1793	002260	071650	EDT1	
1794	002262	071740	EFT1	
1795				
1796				

1797			;ERROR	51	INCORRECT IAE STATUS DURING SEEK COMMAND
1798	002264	066074		EMT51	
1799	002266	071524		EHT1	
1800	002270	071650		EDT1	
1801	002272	071740		EFT1	
1802					
1803					
1804			;ERROR	52	OPI ERROR DURING SEEK - MEDIUM IS NOT ON LINE
1805	002274	066106		EMT52	
1806	002276	071524		EHT1	
1807	002300	071650		EDT1	
1808	002302	071740		EFT1	
1809					
1810					
1811			;ERROR	53	OPI ERROR DURING SEEK - MEDIUM IS ON LINE, ASSUME
1812			;		ON CYLINDER LATCH DIDN'T RESET
1813	002304	066124		EMT53	
1814	002306	071524		EHT1	
1815	002310	071650		EDT1	
1816	002312	071740		EFT1	
1817					
1818					
1819			;ERROR	54	SEEK INCOMPLETE ERROR DURING SEEK COMMAND
1820	002314	066142		EMT54	
1821	002316	071524		EHT1	
1822	002320	071650		EDT1	
1823	002322	071740		EFT1	
1824					
1825					
1826			;ERROR	55	DEVICE CHECK DURING SEEK COMMAND
1827	002324	066152		EMT55	
1828	002326	071524		EHT1	
1829	002330	071650		EDT1	
1830	002332	071740		EFT1	
1831					
1832					
1833			;ERROR	56	PIP IS STILL SET AFTER SEEK - SKI IS RESET
1834	002334	066164		EMT56	
1835	002336	071524		EHT1	
1836	002340	071650		EDT1	
1837	002342	071740		EFT1	
1838					
1839					
1840			;ERROR	57	ATA DID NOT SET DURING SEEK COMMAND
1841	002344	066202		EMT57	
1842	002346	071524		EHT1	
1843	002350	071650		EDT1	
1844	002352	071740		EFT1	
1845					
1846					
1847			;ERROR	60	IVC ERROR DURING SEEK COMMAND - LOST
1848			;		VOLUME VALID
1849	002354	066212		EMT60	
1850	002356	071524		EHT1	
1851	002360	071650		EDT1	
1852	002362	071740		EFT1	

1853				
1854				
1855			;ERROR	61
1856				
1857	002364	066230		EMT61
1858	002366	071524		EHT1
1859	002370	071650		EDT1
1860	002372	071740		EFT1
1861				
1862				
1863			;ERROR	62
1864				
1865	002374	066250		EMT62
1866	002376	071524		EHT1
1867	002400	071650		EDT1
1868	002402	071740		EFT1
1869				
1870				
1871			;ERROR	63
1872	002404	000000		0
1873	002406	000000		0
1874	002410	000000		0
1875	002412	000000		0
1876				
1877				
1878			;ERROR	64
1879	002414	066266		EMT64
1880	002416	071524		EHT1
1881	002420	071650		EDT1
1882	002422	071740		EFT1
1883				
1884				
1885			;ERROR	65
1886	002424	066306		EMT65
1887	002426	071524		EHT1
1888	002430	071650		EDT1
1889	002432	071740		EFT1
1890				
1891				
1892			;ERROR	66
1893	002434	066326		EMT66
1894	002436	071524		EHT1
1895	002440	071650		EDT1
1896	002442	071740		EFT1
1897				
1898				
1899			;ERROR	67
1900	002444	066340		EMT67
1901	002446	071524		EHT1
1902	002450	071650		EDT1
1903	002452	071740		EFT1
1904				
1905				
1906			;ERROR	70
1907	002454	066352		EMT70
1908	002456	071524		EHT1

ERRONEOUS IVC ERROR DURING SEEK COMMAND -
VOLUME VALID IS STILL SET

MOL IS ZERO, BUT OPI WAS NOT
REPORTED DURING SEEK COMMAND

UNUSED

DRIVE DID NOT DETECT "IVC" ERROR DURING SEEK

DRIVE EXECUTED A SEEK WITH ERROR SET

UNEXPECTED ERROR SET IN RMER1

UNEXPECTED ERROR SET IN RMER2

ERRONEOUS "IAE" ERROR DURING RECALIBRATE

1909	002460	071650	EDT1	
1910	002462	071740	EFT1	
1911				
1912				
1913			;ERROR	71 "ILF" ERROR DURING RECALIBRATE
1914	002464	066362	EMT71	
1915	002466	071524	EHT1	
1916	002470	071650	EDT1	
1917	002472	071740	EFT1	
1918				
1919				
1920			;ERROR	72 "OPI" ERROR DURING RECALIBRATE DUE TO "MOL" = 0
1921	002474	066372	EMT72	
1922	002476	071524	EHT1	
1923	002500	071650	EDT1	
1924	002502	071740	EFT1	
1925				
1926				
1927			;ERROR	73 "OPI" ERROR DURING RECALIBRATE BECAUSE ON
1928			:	CYLINDER DIDNT DROP
1929	002504	066410	EMT73	
1930	002506	071524	EHT1	
1931	002510	071650	EDT1	
1932	002512	071740	EFT1	
1933				
1934				
1935			;ERROR	74 "IVC" ERROR DURING RECALIBRATE - "VV" = 0
1936	002514	066426	EMT74	
1937	002516	071524	EHT1	
1938	002520	071650	EDT1	
1939	002522	071740	EFT1	
1940				
1941				
1942			;ERROR	75 ERRONEOUS "IVC" ERROR DURING RECALIBRATE - "VV" = 1
1943	002524	066436	EMT75	
1944	002526	071524	EHT1	
1945	002530	071650	EDT1	
1946	002532	071740	EFT1	
1947				
1948				
1949			;ERROR	76 "SKI" ERROR DURING RECALIBRATE
1950	002534	066456	EMT76	
1951	002536	071524	EHT1	
1952	002540	071650	EDT1	
1953	002542	071740	EFT1	
1954				
1955				
1956			;ERROR	77 "DVC" OCCURRED DURING RECALIBRATE
1957	002544	066466	EMT77	
1958	002546	071524	EHT1	
1959	002550	071650	EDT1	
1960	002552	071740	EFT1	
1961				
1962				
1963			;ERROR	100 LOST "MOL" DURING RECALIBRATE - "OPI" = 0
1964	002554	066500	EMT100	

1965	002556	071524		EHT1	
1966	002560	071650		EDT1	
1967	002562	071740		EFT1	
1968					
1969					
1970			;ERROR	101	LOST "VV" DURING RECALIBRATE - "IVC" = 0
1971	002564	066516		EMT101	
1972	002566	071524		EHT1	
1973	002570	071650		EDT1	
1974	002572	071740		EFT1	
1975					
1976					
1977			;ERROR	102	"ATA" DID NOT SET DURING RECALIBRATE
1978	002574	066534		EMT102	
1979	002576	071524		EHT1	
1980	002600	071650		EDT1	
1981	002602	071740		EFT1	
1982					
1983					
1984			;ERROR	103	"OM" DID NOT RESET DURING RECALIBRATE
1985	002604	066544		EMT103	
1986	002606	071524		EHT1	
1987	002610	071650		EDT1	
1988	002612	071740		EFT1	
1989					
1990					
1991			;ERROR	104	"PIP" IS STIL SET AFTER RECALIBRATE
1992	002614	066556		EMT104	
1993	002616	071524		EHT1	
1994	002620	071650		EDT1	
1995	002622	071740		EFT1	
1996					
1997					
1998			;ERROR	105	UNEXPECTED "ILR" ERROR DURING RECALIBRATE
1999	002624	066574		EMT105	
2000	002626	071524		EHT1	
2001	002630	071650		EDT1	
2002	002632	071740		EFT1	
2003					
2004					
2005			;ERROR	106	UNEXPECTED "RMR" ERROR DURING RECALIBRATE
2006	002634	066604		EMT106	
2007	002636	071524		EHT1	
2008	002640	071650		EDT1	
2009	002642	071740		EFT1	
2010					
2011					
2012			;ERROR	107	"UNS" ERROR DURING RECALIBRATE - AC POWER IS LOW
2013	002644	066614		EMT107	
2014	002646	071524		EHT1	
2015	002650	071650		EDT1	
2016	002652	071740		EFT1	
2017					
2018					
2019			;ERROR	110	CANNOT ACCESS MASSBUS CONTROLLER VIA UNIBUS
2020	002654	066634		EMT110	

2021	002656	071546		EHT110	
2022	002660	071666		EDT110	
2023	002662	071756		EFT110	
2024					
2025					
2026			;ERROR	111	NONEXISTENT DEVICE
2027	002664	066646		EMT111	
2028	002666	071552		EHT111	
2029	002670	071670		EDT111	
2030	002672	071760		EFT111	
2031					
2032					
2033			;ERROR	112	DEVICE NOT AVAILABLE
2034	002674	066654		EMT112	
2035	002676	071552		EHT111	
2036	002700	071670		EDT111	
2037	002702	071760		EFT111	
2038					
2039					
2040			;ERROR	113	BUS TIMEOUT-NED STATUS FAILURE
2041	002704	066662		EMT113	
2042	002706	000000		0	
2043	002710	000000		0	
2044	002712	000000		0	
2045					
2046					
2047			;ERROR	114	DEVICE NOT AN RM03
2048	002714	066676		EMT114	
2049	002716	071556		EHT114	
2050	002720	071672		EDT114	
2051	002722	071762		EFT114	
2052					
2053					
2054			;ERROR	115	RMCS1 NOT INITIALIZED BY UNIBUS
2055	002724	066704		EMT115	
2056	002726	071524		EHT1	
2057	002730	071650		EDT1	
2058	002732	071740		EFT1	
2059					
2060					
2061			;ERROR	116	RMBA NOT INITIALIZED BY UNIBUS
2062	002734	066714		EMT116	
2063	002736	071524		EHT1	
2064	002740	071650		EDT1	
2065	002742	071740		EFT1	
2066					
2067					
2068			;ERROR	117	RMCS2 NOT INITIALIZED BY UNIBUS
2069	002744	066724		EMT117	
2070	002746	071524		EHT1	
2071	002750	071650		EDT1	
2072	002752	071740		EFT1	
2073					
2074					
2075			;ERROR	120	RMER1 NOT INITIALIZED BY UNIBUS
2076	002754	066734		EMT120	

K04

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19MACY11 30(1046) 23-NOV-77 12:38 PAGE 49
ERROR POINTER TABLE

SEQ 0049

2077	002756	071524		EHT1	
2078	002760	071650		EDT1	
2079	002762	071740		EFT1	
2080					
2081					
2082			;ERROR	121	RMA5 NOT INITIALIZED BY UNIBUS
2083	002764	066744		EMT121	
2084	002766	071524		EHT1	
2085	002770	071650		EDT1	
2086	002772	071740		EFT1	
2087					
2088					
2089			;ERROR	122	RMMR1 NOT INITIALIZED BY UNIBUS
2090	002774	066754		EMT122	
2091	002776	071524		EHT1	
2092	003000	071650		EDT1	
2093	003002	071740		EFT1	
2094					
2095					
2096			;ERROR	123	RMDS NOT INITIALIZED BY UNIBUS
2097	003004	066764		EMT123	
2098	003006	071524		EHT1	
2099	003010	071650		EDT1	
2100	003012	071740		EFT1	
2101					
2102					
2103			;ERROR	124	RMEC2 NOT INITIALIZED BY UNIBUS
2104	003014	066774		EMT124	
2105	003016	071524		EHT1	
2106	003020	071650		EDT1	
2107	003022	071740		EFT1	
2108					
2109					
2110			;ERROR	125	RMMR2 NOT INITIALIZED BY UNIBUS
2111	003024	067004		EMT125	
2112	003026	071524		EHT1	
2113	003030	071650		EDT1	
2114	003032	071740		EFT1	
2115					
2116					
2117			;ERROR	126	RMCS1 NOT CLEARED BY CONTROLLER CLEAR
2118	003034	067014		EMT126	
2119	003036	071524		EHT1	
2120	003040	071650		EDT1	
2121	003042	071740		EFT1	
2122					
2123					
2124			;ERROR	127	RMBA NOT CLEARED BY CONTROLLER CLEAR
2125	003044	067026		EMT127	
2126	003046	071524		EHT1	
2127	003050	071650		EDT1	
2128	003052	071740		EFT1	
2129					
2130					
2131			;ERROR	130	RMCS2 NOT CLEARED BY CONTROLLER CLEAR
2132	003054	067040		EMT130	

2133	003056	071524		EHT1	
2134	003060	071650		EDT1	
2135	003062	071740		EFT1	
2136					
2137					
2138			;ERROR	131	RMER1 NOT CLEARED BY CONTROLLER CLEAR
2139	003064	067052		EMT131	
2140	003066	071524		EHT1	
2141	003070	071650		EDT1	
2142	003072	071740		EFT1	
2143					
2144					
2145			;ERROR	132	RNAS NOT CLEARED BY CONTROLLER CLEAR
2146	003074	067064		EMT132	
2147	003076	071524		EHT1	
2148	003100	071650		EDT1	
2149	003102	071740		EFT1	
2150					
2151					
2152			;ERROR	133	RMMR1 NOT CLEARED BY CONTROLLER CLEAR
2153	003104	067076		EMT133	
2154	003106	071524		EHT1	
2155	003110	071650		EDT1	
2156	003112	071740		EFT1	
2157					
2158					
2159			;ERROR	134	RMDS NOT CLEARED BY CONTROLLER CLEAR
2160	003114	067110		EMT134	
2161	003116	071524		EHT1	
2162	003120	071650		EDT1	
2163	003122	071740		EFT1	
2164					
2165					
2166			;ERROR	135	RMEC2 NOT CLEARED BY CONTROLLER CLEAR
2167	003124	067122		EMT135	
2168	003126	071524		EHT1	
2169	003130	071650		EDT1	
2170	003132	071740		EFT1	
2171					
2172					
2173			;ERROR	136	RMMR2 NOT CLEARED BY CONTROLLER CLEAR
2174	003134	067134		EMT136	
2175	003136	071524		EHT1	
2176	003140	071650		EDT1	
2177	003142	071740		EFT1	
2178					
2179					
2180			;ERROR	137	RMCS1 NOT CLEARED BY ERROR CLEAR
2181	003144	067146		EMT137	
2182	003146	071524		EHT1	
2183	003150	071650		EDT1	
2184	003152	071740		EFT1	
2185					
2186					
2187			;ERROR	140	RMCS2 NOT CLEARED BY ERROR CLEAR
2188	003154	067156		EMT140	

2189	003156	071524	EHT1	
2190	003160	071650	EDT1	
2191	003162	071740	EFT1	
2192				
2193				
2194				;ERROR 141 RMCS1 NOT CLEARED BY DRIVE CLEAR
2195	003164	067166	EMT141	
2196	003166	071524	EHT1	
2197	003170	071650	EDT1	
2198	003172	071740	EFT1	
2199				
2200				
2201				;ERROR 142 RMD5 NOT CLEARED BY DRIVE CLEAR
2202	003174	067176	EMT142	
2203	003176	071524	EHT1	
2204	003200	071650	EDT1	
2205	003202	071740	EFT1	
2206				
2207				
2208				;ERROR 143 RMER1 NOT CLEARED BY DRIVE CLEAR
2209	003204	067206	EMT143	
2210	003206	071524	EHT1	
2211	003210	071650	EDT1	
2212	003212	071740	EFT1	
2213				
2214				
2215				;ERROR 144 RMA5 NOT CLEARED BY DRIVE CLEAR
2216	003214	067216	EMT144	
2217	003216	071524	EHT1	
2218	003220	071650	EDT1	
2219	003222	071740	EFT1	
2220				
2221				
2222				;ERROR 145 RMMR1 NOT CLEARED BY DRIVE CLEAR
2223	003224	067226	EMT145	
2224	003226	071524	EHT1	
2225	003230	071650	EDT1	
2226	003232	071740	EFT1	
2227				
2228				
2229				;ERROR 146 RMMR2 NOT CLEARED BY DRIVE CLEAR
2230	003234	067236	EMT146	
2231	003236	071524	EHT1	
2232	003240	071650	EDT1	
2233	003242	071740	EFT1	
2234				
2235				
2236				;ERROR 147 RMER2 NOT CLEARED BY DRIVE CLEAR
2237	003244	067246	EMT147	
2238	003246	071524	EHT1	
2239	003250	071650	EDT1	
2240	003252	071740	EFT1	
2241				
2242				
2243				;ERROR 150 RMEC2 NOT CLEARED BY DRIVE CLEAR
2244	003254	067256	EMT150	

N04

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19MACY11 30(1046) 23-NOV-77 12:38 PAGE 52
ERROR POINTER TABLE

SEQ 0052

2245	003256	071524	EH11	
2246	003260	071650	ED11	
2247	003262	071740	EFT1	
2248				
2249				
2250				
2251	003264	067266	EH11	MEDIUM NOT ON LINE
2252	003266	071524	ED11	
2253	003270	071650	EFT1	
2254	003272	071740		
2255				
2256				
2257				
2258	003274	067300	EH11	DRIVE FAULT
2259	003276	071524	ED11	
2260	003300	071650	EFT1	
2261	003302	071740		
2262				
2263				
2264				
2265	003304	067312	EH11	UNSAFE SHOULD BE SET BECAUSE DVC IS SET
2266	003306	071524	ED11	
2267	003310	071650	EFT1	
2268	003312	071740		
2269				
2270				
2271				
2272	003314	067330	EH11	UNSAFE SHOULD NOT BE SET, AC IS LOW
2273	003316	071524	ED11	
2274	003320	071650	EFT1	
2275	003322	071740		
2276				
2277				
2278				
2279	003324	067346	EH11	VOLUME VALID NOT SET BY PACK ACK
2280	003326	071524	ED11	
2281	003330	071650	EFT1	
2282	003332	071740		
2283				
2284				
2285				
2286	003334	067360	EH11	OFFSET MODE NOT SET BY OFFSET COMMAND
2287	003336	071524	ED11	
2288	003340	071650	EFT1	
2289	003342	071740		
2290				
2291				
2292				
2293	003344	067372	EH11	OFFSET MODE NOT RESET BY RTC COMMAND
2294	003346	071524	ED11	
2295	003350	071650	EFT1	
2296	003352	071740		
2297				
2298				
2299				
2300	003354	067404	EH11	RMOF NOT RESET BY RIP COMMAND

2301	003356	071524	EHT1	
2302	003360	071650	EDT1	
2303	003362	071740	EFT1	
2304				
2305				
2306			;ERROR 161	RMDA NOT RESET BY RIP COMMAND
2307	003364	067414	EMT161	
2308	003366	071524	EHT1	
2309	003370	071650	EDT1	
2310	003372	071740	EFT1	
2311				
2312				
2313			;ERROR 162	RMDC NOT RESET BY RIP COMMAND
2314	003374	067426	EMT162	
2315	003376	071524	EHT1	
2316	003400	071650	EDT1	
2317	003402	071740	EFT1	
2318				
2319				
2320			;ERROR 163	DATA WAS ECC CORRECTED BUT DOES NOT COMPARE WITH
2321			:	WRITE BUFFER
2322	003404	071320	EMT336	
2323	003406	071606	EHT336	
2324	003410	071704	EDT336	
2325	003412	071774	EFT336	
2326				
2327				
2328			;ERROR 164	OPI SHOULD NOT BE SET
2329	003414	067450	EMT164	
2330	003416	071524	EHT1	
2331	003420	071650	EDT1	
2332	003422	071740	EFT1	
2333				
2334				
2335			;ERROR 165	IVC SHOULD NOT BE SET
2336	003424	067456	EMT165	
2337	003426	071524	EHT1	
2338	003430	071650	EDT1	
2339	003432	071740	EFT1	
2340				
2341				
2342			;ERROR 166	IAE SHOULD NOT BE SET
2343	003434	067464	EMT166	
2344	003436	071524	EHT1	
2345	003440	071650	EDT1	
2346	003442	071740	EFT1	
2347				
2348				
2349			;ERROR 167	NEM SHOULD NOT BE SET
2350	003444	067472	EMT167	
2351	003446	071524	EHT1	
2352	003450	071650	EDT1	
2353	003452	071740	EFT1	
2354				
2355				
2356			;ERROR 170	UNUSED

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19MACY11 30(1046) 23-NOV-77 12:38 PAGE 54
ERROR POINTER TABLE

SEQ 0054

2357	003454	000000	0	
2358	003456	000000	0	
2359	003460	000000	0	
2360	003462	000000	0	
2361				
2362				
2363				
2364	003464	067510	;ERROR 171	"ATA" NOT SET DURING RETURN TO CENTERLINE
2365	003466	071524	EMT171	
2366	003470	071650	EHT1	
2367	003472	071740	EDT1	
2368			EFT1	
2369				
2370				
2371	003474	067520	;ERROR 172	"ATA" NOT SET BY OFFSET COMMAND
2372	003476	071524	EMT172	
2373	003500	071650	EHT1	
2374	003502	071740	EDT1	
2375			EFT1	
2376				
2377				
2378	003504	067530	;ERROR 173	RMER2 NOT INITIALIZED BY UNIBUS INIT
2379	003506	071524	EMT173	
2380	003510	071650	EHT1	
2381	003512	071740	EDT1	
2382			EFT1	
2383				
2384				
2385	003514	067540	;ERROR 174	RMER2 NOT INITIALIZED BY CONTROLLER CLEAR
2386	003516	071524	EMT174	
2387	003520	071650	EHT1	
2388	003522	071740	EDT1	
2389			EFT1	
2390				
2391				
2392	003524	067552	;ERROR 175	SELECTED DEVICE IS IN WRITE PROTECT
2393	003526	071524	EMT175	
2394	003530	071650	EHT1	
2395	003532	071740	EDT1	
2396			EFT1	
2397				
2398				
2399	003534	067560	;ERROR 176	CANNOT SET DIAGNOSTIC MODE
2400	003536	071524	EMT176	
2401	003540	071650	EHT1	
2402	003542	071740	EDT1	
2403			EFT1	
2404				
2405				
2406	003544	067566	;ERROR 177	INCORRECT "MOL" STATUS DURING DIAGNOSTIC MODE
2407	003546	071524	EMT177	
2408	003550	071650	EHT1	
2409	003552	071740	EDT1	
2410			EFT1	
2411				
2412			;ERROR 200	INCORRECT "PIP" STATUS DURING DIAGNOSTIC MODE

2413	003554	067600	EMT200	
2414	003556	071524	EHT1	
2415	003560	071650	EDT1	
2416	003562	071740	EFT1	
2417				
2418				
2419				
2420	003564	067612	EMT201	;ERROR 201 INCORRECT "WRL" STATUS DURING DIAGNOSTIC MODE
2421	003566	071524	EHT1	
2422	003570	071650	EDT1	
2423	003572	071740	EFT1	
2424				
2425				
2426				
2427	003574	067624	EMT202	;ERROR 202 INCORRECT "SKI" STATUS DURING DIAGNOSTIC MODE
2428	003576	071524	EHT1	
2429	003600	071650	EDT1	
2430	003602	071740	EFT1	
2431				
2432				
2433				
2434	003604	067636	EMT203	;ERROR 203 INCORRECT "DVC" STATUS DURING DIAGNOSTIC MODE
2435	003606	071524	EHT1	
2436	003610	071650	EDT1	
2437	003612	071740	EFT1	
2438				
2439				
2440				
2441	003614	067650	EMT204	;ERROR 204 "VV" WAS NOT RESET BY MAINTENANCE UNIT READY
2442	003616	071524	EHT1	
2443	003620	071650	EDT1	
2444	003622	071740	EFT1	
2445				
2446				
2447				
2448	003624	067666	EMT205	;ERROR 205 SELECTED DEVICE HAS A PERSISTENT "SKI" ERROR
2449	003626	071524	EHT1	
2450	003630	071650	EDT1	
2451	003632	071740	EFT1	
2452				
2453				
2454				
2455	003634	067676	EMT206	;ERROR 206 "LBC" DID NOT SET DURING DIAGNOSTIC MODE
2456	003636	071524	EHT1	
2457	003640	071650	EDT1	
2458	003642	071740	EFT1	
2459				
2460				
2461				
2462	003644	067706	EMT207	;ERROR 207 UNEXPECTED LOSS OF "MOL" - MEDIUM IS OFF LINE
2463	003646	071524	EHT1	
2464	003650	071650	EDT1	
2465	003652	071740	EFT1	
2466				
2467				
2468				;ERROR 210 UNEXPECTED LOSS OF VOLUME VALID - "VV" = 0

2469	003654	067720	EMT210	
2470	003656	071524	EHT1	
2471	003660	071650	EDT1	
2472	003662	071740	EFT1	
2473				
2474				
2475			;ERROR 211	UNEXPECTED MECHANICAL MOTION - "PIP" = 1
2476	003664	067726	EMT211	
2477	003666	071524	EHT1	
2478	003670	071650	EDT1	
2479	003672	071740	EFT1	
2480				
2481				
2482			;ERROR 212	UNEXPECTED DEVICE FAULT - "DVC" = 1
2483	003674	067742	EMT212	
2484	003676	071524	EHT1	
2485	003700	071650	EDT1	
2486	003702	071740	EFT1	
2487				
2488				
2489			;ERROR 213	UNEXPECTED SEEK INCOMPLETE ERROR - "SKI" = 1
2490	003704	067756	EMT213	
2491	003706	071524	EHT1	
2492	003710	071650	EDT1	
2493	003712	071740	EFT1	
2494				
2495				
2496			;ERROR 214	DRIVE EXECUTED A RECALIBRATE WITH ERROR SET
2497	003714	067766	EMT214	
2498	003716	071536	EHT2	
2499	003720	071660	EDT2	
2500	003722	071750	EFT2	
2501				
2502				
2503			;ERROR 215	DRIVE DID NOT DETECT "IVC" ERROR DURING RECALIBRATE
2504	003724	070006	EMT215	
2505	003726	071536	EHT2	
2506	003730	071660	EDT2	
2507	003732	071750	EFT2	
2508				
2509				
2510			;ERROR 216	INCORRECT "IVC" STATUS
2511	003734	070020	EMT216	
2512	003736	071524	EHT1	
2513	003740	071650	EDT1	
2514	003742	071740	EFT1	
2515				
2516				
2517			;ERROR 217	INCORRECT "IAE" STATUS
2518	003744	070030	EMT217	
2519	003746	071524	EHT1	
2520	003750	071650	EDT1	
2521	003752	071740	EFT1	
2522				
2523				
2524			;ERROR 220	INCORRECT "WLE" STATUS

2525	003754	070040	EMT220	
2526	003756	071524	EHT1	
2527	003760	071650	EDT1	
2528	003762	071740	EFT1	
2529				
2530				
2531			; ERROR	221 INCORRECT "OPI" STATUS
2532	003764	070050	EMT221	
2533	003766	071524	EHT1	
2534	003770	071650	EDT1	
2535	003772	071740	EFT1	
2536				
2537				
2538			; ERROR	222 RM03 DID NOT DETECT RMR ERROR
2539	003774	070060	EMT222	
2540	003776	071524	EHT1	
2541	004000	071650	EDT1	
2542	004002	071740	EFT1	
2543				
2544				
2545			; ERROR	223 RM03 DID NOT DETECT PARITY ERROR ON MASSBUS CONTROL BUS
2546	004004	070070	EMT223	
2547	004006	071562	EHT223	
2548	004010	071674	EDT223	
2549	004012	071764	EFT223	
2550				
2551				
2552			; ERROR	224 UNUSED
2553	004014	000000	0	
2554	004016	000000	0	
2555	004020	000000	0	
2556	004022	000000	0	
2557				
2558				
2559			; ERROR	225 UNUSED
2560	004024	000000	0	
2561	004026	000000	0	
2562	004030	000000	0	
2563	004032	000000	0	
2564				
2565				
2566			; ERROR	226 UNUSED
2567	004034	000000	0	
2568	004036	000000	0	
2569	004040	000000	0	
2570	004042	000000	0	
2571				
2572				
2573			; ERROR	227 UNUSED
2574	004044	000000	0	
2575	004046	000000	0	
2576	004050	000000	0	
2577	004052	000000	0	
2578				
2579				
2580			; ERROR	230 UNUSED

2581	004054	000000	0	
2582	004056	000000	0	
2583	004060	000000	0	
2584	004062	000000	0	
2585				
2586				
2587			; ERROR	231 UNUSED
2588	004064	000000	0	
2589	004066	000000	0	
2590	004070	000000	0	
2591	004072	000000	0	
2592				
2593				
2594			; ERROR	232 UNUSED
2595	004074	000000	0	
2596	004076	000000	0	
2597	004100	000000	0	
2598	004102	000000	0	
2599				
2600				
2601			; ERROR	233 UNUSED
2602	004104	000000	0	
2603	004106	000000	0	
2604	004110	000000	0	
2605	004112	000000	0	
2606				
2607				
2608			; ERROR	234 UNUSED
2609	004114	000000	0	
2610	004116	000000	0	
2611	004120	000000	0	
2612	004122	000000	0	
2613				
2614				
2615			; ERROR	235 UNUSED
2616	004124	000000	0	
2617	004126	000000	0	
2618	004130	000000	0	
2619	004132	000000	0	
2620				
2621				
2622			; ERROR	236 UNUSED
2623	004134	000000	0	
2624	004136	000000	0	
2625	004140	000000	0	
2626	004142	000000	0	
2627				
2628				
2629			; ERROR	237 UNUSED
2630	004144	000000	0	
2631	004146	000000	0	
2632	004150	000000	0	
2633	004152	000000	0	
2634				
2635				
2636			; ERROR	240 UNUSED

2637	004154	000000	0	
2638	004156	000000	0	
2639	004160	000000	0	
2640	004162	000000	0	
2641				
2642				
2643			; ERROR	241 UNUSED
2644	004164	000000	0	
2645	004166	000000	0	
2646	004170	000000	0	
2647	004172	000000	0	
2648				
2649				
2650			; ERROR	242 UNUSED
2651	004174	000000	0	
2652	004176	000000	0	
2653	004200	000000	0	
2654	004202	000000	0	
2655				
2656				
2657			; ERROR	243 UNUSED
2658	004204	000000	0	
2659	004206	000000	0	
2660	004210	000000	0	
2661	004212	000000	0	
2662				
2663				
2664			; ERROR	244 UNUSED
2665	004214	000000	0	
2666	004216	000000	0	
2667	004220	000000	0	
2668	004222	000000	0	
2669				
2670				
2671			; ERROR	245 UNUSED
2672	004224	000000	0	
2673	004226	000000	0	
2674	004230	000000	0	
2675	004232	000000	0	
2676				
2677				
2678			; ERROR	246 "ATA" NOT RESET BY GO WHEN "ERR" = 0
2679	004234	070144	EMT246	
2680	004236	071524	EHT1	
2681	004240	071650	EDT1	
2682	004242	071740	EFT1	
2683				
2684				
2685			; ERROR	247 "ATA" NOT RESET BY WRITING RMAS
2686	004244	070154	EMT247	
2687	004246	071524	EHT1	
2688	004250	071650	EDT1	
2689	004252	071740	EFT1	
2690				
2691				
2692			; ERROR	250 "ATA" WAS RESET BY GO WHEN "ERR" = 1

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 60
ERROR POINTER TABLE

SEQ 0060

2693	004254	070166	EMT250	
2694	004256	071524	EHT1	
2695	004260	071650	EDT1	
2696	004262	071740	EFT1	
2697				
2698				
2699				
2700	004264	070202	;ERROR 251	PROGRAM INTERRUPT WAS NOT GENERATED
2701	004266	071536	EMT251	
2702	004270	071660	EHT2	
2703	004272	071750	EDT2	
2704			EFT2	
2705				
2706				
2707	004274	070210	;ERROR 252	PROGRAM INTERRUPT SHOULD NOT HAVE BEEN GENERATED
2708	004276	071536	EMT252	
2709	004300	071660	EHT2	
2710	004302	071750	EDT2	
2711			EFT2	
2712				
2713				
2714	004304	070216	;ERROR 253	OFFSET MODE WAS NOT RESET BY WRITING RMDC
2715	004306	071524	EMT253	
2716	004310	071650	EHT1	
2717	004312	071740	EDT1	
2718			EFT1	
2719				
2720				
2721	004314	070234	;ERROR 254	INCORRECT "ILF" STATUS
2722	004316	071524	EMT254	
2723	004320	071650	EHT1	
2724	004322	071740	EDT1	
2725			EFT1	
2726				
2727				
2728	004324	070244	;ERROR 255	INCORRECT "ATA" STATUS
2729	004326	071524	EMT255	
2730	004330	071650	EHT1	
2731	004332	071740	EDT1	
2732			EFT1	
2733				
2734				
2735	004334	070254	;ERROR 256	INCORRECT "ILR" STATUS
2736	004336	071574	EMT256	
2737	004340	071674	EHT256	
2738	004342	071764	EDT223	
2739			EFT223	
2740				
2741				
2742	004344	070264	;ERROR 257	INVALID IAE STATUS DURING SEARCH COMMAND
2743	004346	071524	EMT257	
2744	004350	071650	EHT1	
2745	004352	071740	EDT1	
2746			EFT1	
2747				
2748			;ERROR 260	"IVC" WAS NOT DETECTED DURING SEARCH COMMAND

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 61
ERROR POINTER TABLE

SEQ 0061

2749	004354	070276	EMT260	
2750	004356	071524	EHT1	
2751	004360	071650	EDT1	
2752	004362	071740	EFT1	
2753				
2754				
2755			;ERROR 261	DRIVE EXECUTED SEARCH WITH ERROR SET
2756	004364	070310	EMT261	
2757	004366	071524	EHT1	
2758	004370	071650	EDT1	
2759	004372	071740	EFT1	
2760				
2761				
2762			;ERROR 262	"LBC" ERROR NOT SET DURING DIAGNOSTIC MODE
2763	004374	070330	EMT262	
2764	004376	071524	EHT1	
2765	004400	071650	EDT1	
2766	004402	071740	EFT1	
2767				
2768				
2769			;ERROR 263	"SKI" ERROR DURING SEARCH COMMAND
2770	004404	070340	EMT263	
2771	004406	071524	EHT1	
2772	004410	071650	EDT1	
2773	004412	071740	EFT1	
2774				
2775				
2776			;ERROR 264	"IVC" ERROR DURING SEARCH - LOST VOLUME VALID
2777	004414	070350	EMT264	
2778	004416	071524	EHT1	
2779	004420	071650	EDT1	
2780	004422	071740	EFT1	
2781				
2782				
2783			;ERROR 265	ERRONEOUS IVC ERROR DURING SEARCH - VOLUME IS VALID
2784	004424	070370	EMT265	
2785	004426	071524	EHT1	
2786	004430	071650	EDT1	
2787	004432	071740	EFT1	
2788				
2789				
2790			;ERROR 266	DEVICE FAULT (DVC) DURING SEARCH
2791	004434	070410	EMT266	
2792	004436	071524	EHT1	
2793	004440	071650	EDT1	
2794	004442	071740	EFT1	
2795				
2796				
2797			;ERROR 267	SKI SHOULD HAVE BEEN SET BECAUSE CYLINDER ADDRESS IS TOO LARGE
2798			;	
2799	004444	070422	EMT267	
2800	004446	071524	EHT1	
2801	004450	071650	EDT1	
2802	004452	071740	EFT1	
2803				
2804				

K05

CZRMD80 RM03/2 FCTNL TST 2
CZRMD8.P11 23-NOV-77 12:19MACY11 30(1046) 23-NOV-77 12:38 PAGE 62
ERROR POINTER TABLE

SEQ 0062

2805			;ERROR 270	OPI ERROR DURING SEARCH BECAUSE MOL = 0
2806	004454	070440	EMT270	
2807	004456	071524	EHT1	
2808	004460	071650	EDT1	
2809	004462	071740	EFT1	
2810				
2811			;ERROR 271	OPI ERROR DURING SEARCH BECAUSE ON CYLINDER
2812			:	DIDN'T DROP
2813			EMT271	
2814	004464	070454	EHT1	
2815	004466	071524	EDT1	
2816	004470	071650	EFT1	
2817	004472	071740		
2818				
2819			;ERROR 272	LOST MOL DURING SEARCH, OPI IS NOT SET
2820			EMT272	
2821	004474	070472	EHT1	
2822	004476	071524	EDT1	
2823	004500	071650	EFT1	
2824	004502	071740		
2825				
2826			;ERROR 273	PIP STIL SET AFTER SEARCH
2827			EMT273	
2828	004504	070510	EHT1	
2829	004506	071524	EDT1	
2830	004510	071650	EFT1	
2831	004512	071740		
2832				
2833			;ERROR 274	PARITY ERROR OCCURRED WHILE WRITING REMOTE
2834			:	REGISTERS BUT MXF DID NOT SET
2835			EMT274	
2836	004514	070526	EHT1	
2837	004516	071524	EDT1	
2838	004520	071650	EFT1	
2839	004522	071740		
2840				
2841			;ERROR 275	MXF ERROR - COMPOSITE ERROR OCCURRED BEFORE DATA
2842			:	COMMAND STARTED
2843			EMT275	
2844	004524	070544	EHT1	
2845	004526	071524	EDT1	
2846	004530	071650	EFT1	
2847	004532	071740		
2848				
2849			;ERROR 276	"OPI" ERROR DURING DATA TRANSFER BECAUSE "MOL" WAS
2850			:	ZERO
2851			EMT276	
2852	004534	070556	EHT1	
2853	004536	071524	EDT1	
2854	004540	071650	EFT1	
2855	004542	071740		
2856				
2857			;ERROR 277	"OPI" ERROR DURING DATA TRANSFER BECAUSE 1) ON
2858			:	CYLINDER DIDN'T DROP OR 2) SEARCH TIMED OUT OR
2859			:	3) RUN TIMED OUT
2860			:	

2861	004544	070572	EMT277	
2862	004546	071524	EHT1	
2863	004550	071650	EDT1	
2864	004552	071740	EFT1	
2865				
2866				
2867			;ERROR 300	"IVC" ERROR DURING DATA TRANSFER BECAUSE VOLUME
2868				WAS NOT VALID
2869	004554	070610	EMT300	
2870	004556	071524	EHT1	
2871	004560	071650	EDT1	
2872	004562	071740	EFT1	
2873				
2874				
2875			;ERROR 301	ERRONEOUS "IVC" ERROR DURING DATA TRANSFER - VOLUME
2876				IS VALID
2877	004564	070630	EMT301	
2878	004566	071524	EHT1	
2879	004570	071650	EDT1	
2880	004572	071740	EFT1	
2881				
2882				
2883				
2884	004574	070652	;ERROR 302	"ILR" ERROR DURING DATA TRANSFER
2885	004576	071524	EMT302	
2886	004600	071650	EHT1	
2887	004602	071740	EDT1	
2888			EFT1	
2889				
2890				
2891	004604	070664	;ERROR 303	"ILF" ERROR DURING DATA TRANSFER
2892	004606	071524	EMT303	
2893	004610	071650	EHT1	
2894	004612	071740	EDT1	
2895			EFT1	
2896				
2897				
2898	004614	070676	;ERROR 304	"RMR" ERROR DURING DATA TRANSFER
2899	004616	071524	EMT304	
2900	004620	071650	EHT1	
2901	004622	071740	EDT1	
2902			EFT1	
2903				
2904				
2905	004624	070710	;ERROR 305	INCORRECT "IAE" STATUS DURING DATA TRANSFER
2906	004626	071524	EMT305	
2907	004630	071650	EHT1	
2908	004632	071740	EDT1	
2909			EFT1	
2910				
2911				
2912	004634	070722	;ERROR 306	"SKI" ERROR DURING DATA TRANSFER
2913	004636	071524	EMT306	
2914	004640	071650	EHT1	
2915	004642	071740	EDT1	
2916			EFT1	

2917				
2918			;ERROR	307 DRIVE DID NOT DETECT SKI ERROR DUE TO CYLINDER
2919	004644	070732		EMT307
2920	004646	071524		EHT1
2921	004650	071650		EDT1
2922	004652	071740		EFT1
2923				
2924				
2925			;ERROR	310 DEVICE FAULT DURING DATA TRANSFER
2926	004654	070752		EMT310
2927	004656	071524		EHT1
2928	004660	071650		EDT1
2929	004662	071740		EFT1
2930				
2931				
2932			;ERROR	311 LOSS OF BIT CLOCK DURING DATA TRANSFER
2933	004664	070764		EMT311
2934	004666	071524		EHT1
2935	004670	071650		EDT1
2936	004672	071740		EFT1
2937				
2938				
2939			;ERROR	312 LOSS OF SYSTEM CLOCK DURING DATA TRANSFER
2940	004674	070776		EMT312
2941	004676	071524		EHT1
2942	004700	071650		EDT1
2943	004702	071740		EFT1
2944				
2945				
2946			;ERROR	313 UNSAFE ERROR DURING DATA TRANSFER (DVC = 0)
2947	004704	071010		EMT313
2948	004706	071524		EHT1
2949	004710	071650		EDT1
2950	004712	071740		EFT1
2951				
2952				
2953			;ERROR	314 DRIVE TIMING ERROR DURING DATA TRANSFER
2954	004714	071030		EMT314
2955	004716	071524		EHT1
2956	004720	071650		EDT1
2957	004722	071740		EFT1
2958				
2959				
2960			;ERROR	315 WRITE LOCK ERROR
2961	004724	071042		EMT315
2962	004726	071524		EHT1
2963	004730	071650		EDT1
2964	004732	071740		EFT1
2965				
2966				
2967			;ERROR	316 ERRONEOUS WRITE LOCK ERROR
2968	004734	071054		EMT316
2969	004736	071524		EHT1
2970	004740	071650		EDT1
2971	004742	071740		EFT1
2972				

2973				
2974			;ERROR	317 HEADER CRC ERROR DURING DATA TRANSFER
2975	004744	071066		EMT317
2976	004746	071524		EHT1
2977	004750	071650		EDT1
2978	004752	071740		EFT1
2979				
2980				
2981			;ERROR	320 FORMAT ERROR DURING DATA TRANSFER
2982	004754	071076		EMT320
2983	004756	071524		EHT1
2984	004760	071650		EDT1
2985	004762	071740		EFT1
2986				
2987				
2988			;ERROR	321 HEADER COMPARE ERROR DURING DATA TRANSFER
2989	004764	071106		EMT321
2990	004766	071524		EHT1
2991	004770	071650		EDT1
2992	004772	071740		EFT1
2993				
2994				
2995			;ERROR	322 HEADER ERRORS SHOULD NOT BE SET
2996	004774	071116		EMT322
2997	004776	071524		EHT1
2998	005000	071650		EDT1
2999	005002	071740		EFT1
3000				
3001				
3002			;ERROR	323 DATA CHECK ERROR DURING DATA TRANSFER
3003	005004	071124		EMT323
3004	005006	071524		EHT1
3005	005010	071650		EDT1
3006	005012	071740		EFT1
3007				
3008				
3009			;ERROR	324 CORRECTABLE DATA CHECK ERROR DURING DATA TRANSFER
3010	005014	071134		EMT324
3011	005016	071524		EHT1
3012	005020	071650		EDT1
3013	005022	071740		EFT1
3014				
3015				
3016			;ERROR	325 UNCORRECTABLE DATA CHECK ERROR DURING DATA TRANSFER
3017	005024	071146		EMT325
3018	005026	071524		EHT1
3019	005030	071650		EDT1
3020	005032	071740		EFT1
3021				
3022				
3023			;ERROR	326 DATA PARITY ERROR DURING READ COMMAND
3024	005034	071160		EMT326
3025	005036	071524		EHT1
3026	005040	071650		EDT1
3027	005042	071740		EFT1
3028				

3029				
3030			;ERROR	327 OFFSET MODE NOT RESET BY WRITE COMMAND
3031	005044	071176		EMT327
3032	005046	071524		EHT1
3033	005050	071650		EDT1
3034	005052	071740		EFT1
3035				
3036				
3037			;ERROR	330 DATA PARITY ERROR DURING WRITE COMMAND
3038	005054	071210		EMT330
3039	005056	071524		EHT1
3040	005060	071650		EDT1
3041	005062	071740		EFT1
3042				
3043				
3044			;ERROR	331 WRITE CLOCK FAILURE DURING WRITE COMMAND
3045	005064	071220		EMT331
3046	005066	071524		EHT1
3047	005070	071650		EDT1
3048	005072	071740		EFT1
3049				
3050				
3051			;ERROR	332 DATA LATE ERROR DURING DATA TRANSFER
3052	005074	071232		EMT332
3053	005076	071524		EHT1
3054	005100	071650		EDT1
3055	005102	071740		EFT1
3056				
3057				
3058			;ERROR	333 PIP STIL SET AFTER DATA TRANSFER - SKI = 0
3059	005104	071244		EMT333
3060	005106	071524		EHT1
3061	005110	071650		EDT1
3062	005112	071740		EFT1
3063				
3064				
3065			;ERROR	334 LOST MOL DURING DATA TRANSFER - OPI = 0
3066	005114	071262		EMT334
3067	005116	071524		EHT1
3068	005120	071650		EDT1
3069	005122	071740		EFT1
3070				
3071				
3072			;ERROR	335 LOST VOLUME VALID DURING DATA TRANSFER - IVC = 0
3073	005124	071300		EMT335
3074	005126	071524		EHT1
3075	005130	071650		EDT1
3076	005132	071740		EFT1
3077				
3078				
3079			;ERROR	336 DATA READ DOES NOT COMPARE WITH DATA WRITTEN
3080	005134	071320		EMT336
3081	005136	071606		EHT336
3082	005140	071704		EDT336
3083	005142	071774		EFT336
3084				

3085				
3086			;ERROR	337 WRITE CHECK ERROR NOT DETECTED
3087	005144	071330		EMT337
3088	005146	071620		EHT337
3089	005150	071714		EDT337
3090	005152	072004		EFT337
3091				
3092				
3093			;ERROR	340 WRITE CHECK ERROR AT UNEXPECTED ADDRESS
3094	005154	071340		EMT340
3095	005156	071606		EHT336
3096	005160	071704		EDT336
3097	005162	071774		EFT336
3098				
3099				
3100			;ERROR	341 INCORRECT DATA DURING WRITE CHECK ERROR
3101	005164	071352		EMT341
3102	005166	071606		EHT336
3103	005170	071704		EDT336
3104	005172	071774		EFT336
3105				
3106				
3107			;ERROR	342 "IVC" ERROR NOT DETECTED DURING DATA TRANSFER
3108	005174	071360		EMT342
3109	005176	071524		EHT1
3110	005200	071650		EDT1
3111	005202	071740		EFT1
3112				
3113				
3114			;ERROR	343 "FER" NOT DETECTED DURING DATA TRANSFER
3115	005204	071372		EMT343
3116	005206	071524		EHT1
3117	005210	071650		EDT1
3118	005212	071740		EFT1
3119				
3120				
3121			;ERROR	344 "HCE" NOT DETECTED DURING DATA TRANSFER
3122	005214	071404		EMT344
3123	005216	071632		EHT344
3124	005220	071724		EDT344
3125	005222	072014		EFT344
3126				
3127				
3128			;ERROR	345 "BSE" NOT DETECTED DURING DATA TRANSFER
3129	005224	071416		EMT345
3130	005226	071524		EHT1
3131	005230	071650		EDT1
3132	005232	071740		EFT1
3133				
3134				
3135			;ERROR	346 HEADER ERROR WAS DETECTED W/ HCI SET
3136	005234	071426		EMT346
3137	005236	071524		EHT1
3138	005240	071650		EDT1
3139	005242	071740		EFT1
3140				

```

3141
3142 ;ERROR 347 DATA TRANSFER NOT ABORTED W/ COMP ERROR SET
3143 EMT347
3144 005244 071442 EHT1
3145 005246 071524 EDT1
3146 005250 071650 EFT1
3147 005252 071740
3148
3149 ;ERROR 350 LOST VOLUME VALID DURING SEARCH - "IVC" = 0
3150 EMT350
3151 005254 071454 EHT1
3152 005256 071524 EDT1
3153 005260 071650 EFT1
3154 005262 071740
3155
3156 ;ERROR 351 "ATA" DID NOT SET DURING SEARCH
3157 EMT351
3158 005264 071472 EHT1
3159 005266 071524 EDT1
3160 005270 071650 EFT1
3161 005272 071740
3162
3163 ;ERROR 352 PROGRAM TIMEOUT WHILE TESTING RMLA
3164 EMT352
3165 005274 071502 0
3166 005276 000000 0
3167 005300 000000 0
3168 005302 000000
3169
3170 ;ERROR 353 LOOK AHEAD TEST FAILS
3171 EMT353
3172 005304 071506 EHT353
3173 005306 071644 EHT353
3174 005310 071736 EDT353
3175 005312 072024 EFT353
3176
3177 ;ERROR 354 BSE SHOULD NOT BE SET
3178 EMT354
3179 005314 071516 EHT1
3180 005316 071524 EDT1
3181 005320 071650 EFT1
3182 005322 071740
3183
3184 ;PUT ERROR TABLE HERE
3185 ;SBTTL ERROR TABLE USAGE
3186
3187 ;THE ERROR TABLE ABOVE CONSISTS OF FOUR WORD ENTRIES FOR EACH ERROR
3188 ;NUMBER, I.E.,
3189 :
3190 : EMT - ERROR MESSAGE TABLE ADDRESS
3191 : EHT - ERROR HEADER TABLE ADDRESS
3192 : EDT - ERROR DATA TABLE ADDRESS
3193 : EFT - ERROR FORMAT TABLE ADDRESS
3194 :
3195 ;THE EMT ENTRY IS THE ADDRESS OF THE TABLE OF ERROR MESSAGE STRINGS
3196 ;FOR THE PARTICULAR ERROR. EACH ERROR MESSAGE TABLE LISTS THE ADDRESS

```

3197
3198
3199
3200
3201
3202
3203
3204
3205
3206
3207
3208
3209
3210
3211
3212
3213
3214
3215

: OF ONE OR MORE ERROR MESSAGE STRINGS WHICH ARE TO BE FORMATTED AND
: TYPED BY THE ERROR TYPE SUBROUTINE. IF THE EMT ENTRY IS ZERO, THERE IS
: NO MESSAGE TO BE TYPED FOR THE ERROR.

: SIMILARLY, THE EHT, EDT, AND EFT ENTRIES ARE ADDRESSES OF TABLES
: OF HEADER, DATA AND FORMAT INFORMATION FOR A GIVEN ERROR. EACH ENTRY
: IN THE ERROR HEADER TABLE MAY OR MAY NOT HAVE AN ASSOCIATED LINE OF
: DATA. HOWEVER, EACH DATA LINE MUST HAVE AN ASSOCIATED FORMAT AND
: HEADER. THAT IS, A HEADER LINE MAY BE PRINTED WITHOUT ANY DATA,
: BUT A DATA LINE IS NOT PRINTED WITHOUT A HEADER, AND EACH DATA LINE
: MUST ALSO HAVE A FORMAT.

: IN SUMMARY,

: EACH NONZERO ENTRY IS THE ADDRESS OF A TABLE.
: EACH TABLE IS A LIST OF ADDRESSES WHICH DEFINES THE LOCATIONS
: OF MESSAGE STRINGS, HEADERS, DATA OR FORMAT.

```

3216      .SBTTL  START OF PROGRAM
3217
3218
3219      005324
3220
3221      START:
3222      ;CLEAR AND SETUP FOR TEST EXECUTION
3223      NOP
3224      RESET
3225      MOV     PR6, -(SP)
3226      MOV     #64$, -(SP)
3227      RTI
3228      64$:
3229      ; INITIALIZE THE SYSTEM
3230      ; PUT NEW PS ON STACK
3231      ; PUT NEW PC ON STACK
3232      ; POP NEW PC AND PS
3233
3234      .SBTTL  INITIALIZE THE COMMON TAGS
3235      ; CLEAR THE COMMON TAGS ($CHTAG) AREA
3236      MOV     #CHTAG, R6
3237      CLR     (R6)+
3238      CMP     #SWR, R6 ;; DONE?
3239      BNE     .-6
3240      MOV     #STACK, SP
3241      ; LOOP BACK IF NO
3242      ; SETUP THE STACK POINTER
3243      ; INITIALIZE A FEW VECTORS
3244      MOV     #SCOPE, #IOTVEC
3245      MOV     #340, #IOTVEC+2
3246      MOV     #ERROR, #EMTVEC
3247      MOV     #340, #EMTVEC+2
3248      MOV     #TRAP, #TRAPVEC
3249      MOV     #340, #TRAPVEC+2
3250      MOV     #SPWRON, #PWAVEC
3251      MOV     #340, #PWAVEC+2
3252      MOV     #SENDOCT, #SEOPCT
3253      CLR     $TIMES
3254      CLR     $ESCAPE
3255      MOV     #1, $ERMAX
3256      MOV     #., $LPADR
3257      MOV     #., $LPERR
3258      ; SETUP END-OF-PROGRAM COUNTER
3259      ; INITIALIZE NUMBER OF ITERATIONS
3260      ; CLEAR THE ESCAPE ON ERROR ADDRESS
3261      ; ALLOW ONE ERROR PER TEST
3262      ; INITIALIZE THE LOOP ADDRESS FOR SCOPE
3263      ; SETUP THE ERROR LOOP ADDRESS
3264      ; SIZE FOR A HARDWARE SWITCH REGISTER, IF NOT FOUND OR IT IS
3265      ; EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
3266      MOV     #ERRVEC, -(SP)
3267      MOV     #6, #ERRVEC
3268      MOV     #0SWR, SWR
3269      MOV     #0DISP, DISPLAY
3270      CMP     #-1, #SWR
3271      BNE     67$
3272      BR      66$
3273      65$: MOV     #66$, (SP)
3274      RTI
3275      66$: MOV     #SWREG, SWR
3276      MOV     #DISPREG, DISPLAY
3277      MOV     (SP)+, #ERRVEC
3278      ; POINT TO SOFTWARE SWR
3279      ; RESTORE ERROR VECTOR
3280      CLR     $PASS
3281      BITB    #APTSIZE, $ENVN
3282      BEQ     68$
3283      MOV     #SSWREG, SWR
3284      ; CLEAR PASS COUNT
3285      ; TEST USER SIZE UNDER APT
3286      ; YES, USE NON-APT SWITCH
3287      ; NO, USE APT SWITCH REGISTER

```



```

3272 .SBTTL TYPE PROGRAM NAME
3273 ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
3274 005612 005227 177777 INC 0-1 ;;FIRST TIME?
3275 005616 001061 BNE 69$ ;;BRANCH IF NO
3276 005620 022737 033156 000042 CMP 0$ENDAD,2#42 ;;ACT-11?
3277 005626 001455 BEQ 69$ ;;BRANCH IF YES
3278 005630 104401 005675 TYPE 70$ ;;TYPE ASCIZ STRING
3279 .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
3280 005634 005737 000042 TST 2#42 ;;ARE WE RUNNING UNDER XXDP/ACT?
3281 005640 001012 BNE 71$ ;;BRANCH IF YES
3282 005642 123727 001242 000001 CMPB $ENV,01 ;;ARE WE RUNNING UNDER APT?
3283 005650 001406 BEQ 71$ ;;BRANCH IF YES
3284 005652 023727 001154 000176 CMP SWR,#SWREG ;;SOFTWARE SWITCH REG SELECTED?
3285 005660 001005 BNE 72$ ;;BRANCH IF NO
3286 005662 104407 GTSWR ;;GET SOFT-SWR SETTINGS
3287 005664 000403 BR 72$
3288 005666 112737 000001 001150 71$: MOVB 01,$AUTOB ;;SET AUTO-MODE INDICATOR
3289 005674 000432 72$: BR 69$
3290 005674 000432 ;;GET OVER THE ASCIZ
3291 .:70$: .ASCIZ <CRLF>0CZRM08 - RM03/RM02 SUBSYS FUNCTIONAL TEST,PART 2 0<CRLF>
3292 69$:
3293
3294
3295 ;FIND OUT IF PROGRAM IS RUNNING IN STANDALONE MODE
3296 005762 122737 000077 000041 CMPB 077,2#41 ;;LOAD FROM XXDP MEDIM ?
3297 005770 001003 BNE 5$ ;;BRANCH IF NOT
3298 005772 104401 064562 TYPE ,XDPMG ;;TYPE THE MESSAGE
3299 005776 000000 HALT
3300 5$:
3301 006000 005737 000042 TST 42 ;;IS LOC 42 ZERO ??
3302 006004 001003 BNE 10$ ;;NO - NOT IN STANDALONE
3303 006006 105737 001242 TSTB $ENV ;;IS APT ENVIRONMENT ZERO ??
3304 006012 001454 BEQ STANDALONE ;;YES - PROGRAM IN STANDALONE
3305 006014 10$:
3306
3307 ;PROGRAM NOT RUNNING IN STANDALONE - SEE IF SIZING IS ALLOWED
3308 006014 132737 000200 001243 XSIZ: BITB 0BIT7,$ENVM ;;SIZING ALLOWED ??
3309 006022 001046 BNE 20$ ;;NO
3310 006024 012737 000377 001300 MOV 0377,$DEVN ;;YES - SET DEVICE MAP FOR ALL DEVICES
3311 006032 005037 001300 CLR $DEVN ;;CLEAR THE BIT MAP
3312 006036 005001 R1 ;;START FROM THE DRIVE 0
3313 006040 012704 000001 MOV 0BIT0,R4 ;;BIT MAP
3314 006044 013700 001276 MOV $BASE,R0 ;;LOAD BASE ADDRESS
3315 006050 012760 000040 000010 15$: MOV 0CLR,RMCS2(R0) ;;MASS BUS CLEAR
3316 006056 010160 000010 MOV R1,RMCS2(R0) ;;LOAD THE DRIVE ADDRESS
3317 006062 016003 000010 MOV RMCS2(R0),R3 ;;READ DRIVE STATUS TO SEE NED BIT
3318 006066 032703 010000 BIT 0NED,R3 ;;NED BIT SET ?
3319 006072 001010 BNE 16$ ;;BRANCH IF 50
3320 006074 016003 000000 MOV RMCS1(R0),R3 ;;CHECK THE DVA BIT
3321 006100 032703 004000 BIT 0DVA,R3 ;;DRIVE AVAILABLE ?
3322 006104 001403 BEQ 16$ ;;BRANCH IF NOT
3323 006106 050437 001300 BIS R4,$DEVN ;;SET THE BIT MAP
3324 006112 000405 BR 17$ ;;NOT TYPE THE NONE-EXIST MSG
3325 006114
3326 006114 104401 064647 16$: TYPE ,NOTEX ;;NOT EXIST DRIVE
3327 006120 010146 MOV R1,-(SP) ;;DRIVE NUMBER

```

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 72
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0072

3328 006122 104403
3329 006124 006
3330 006125 000
3331 006126 005201
3332 006130 006304
3333 006132 022701 000007
3334 006136 103344
3335 006140
3336
3337
3338 006140 000137 006772

17\$: TYPOS
 .BYTE 6
 .BYTE 0
 INC R1 ; INCREMENT THE DRIVE ADDRESS
 ASL R4 ; SET UP BIT MAP
 CMP #7,R1 ; ALL DRIVE CHECKED ?
 BHS 15\$; BRANCH IF NOT

20\$:
 ; GO TO COMMON START CODE
 JMP CMNSTART

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 73
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0073

```

3339 006144          STANDALONE:
3340 006144 004737 06.400      JSR      PC,$TKINT      ;INITIALIZE CONSOLE
3341
3342          ;SEE IF THIS IS THE FIRST START AFTER PROGRAM WAS LOADED
3343 006150 005327 000001      DEC      #1          ;FIRST START ??
3344 006154 100024          BPL      10$          ;YES !!
3345
3346          ;SEE IF THE USER WANTS TO KEEP SAME DEVICES FOR TESTING
3347          5$:      TYPE      ,CNSLOO          ;MAINTAIN PREVIOUS PARAMETERS??
3348 006156 104401 064071      RDCHR          ;GET RESPONSE
3349 006162 104411          MOV      (SP)+,$TMP1      ;ECHO RESPONSE
3350 006164 012637 001176      TYPE      $TMP1
3351 006170 104401 001176      CMPB      $TMP1,#'Y      ;YES RESPONSE??
3352 006174 123727 001176 000131      BNE      6$          ;NO!!
3353 006202 001002          JMP      READY          ;KEEP PREVIOUS PARAMETERS
3354 006204 000137 007124      CMPB      $TMP1,#'N      ;NO RESPONSE??
3355 006210 123727 001176 000116 6$:      BEQ      20$          ;GET NEW PARAMETERS
3356 006216 001420          TYPE      ,QSTMRK          ;NOT YES OR NO, TYPE ""
3357 006220 104401 063601      BR      5$          ;RETRY
3358 006224 000754
3359 006226          10$:
3360
3361          ;SEE IF OPERATOR WANTS HELP FILE
3362 006226 104401 063603      TYPE      ,HELPGST          ;WANT HELP ??
3363 006232 104411          RDCHR          ;GET RESPONSE
3364 006234 012637 001176      MOV      (SP)+,$TMP1      ;SAVE AND ECHO RESPONSE
3365 006240 104401 001176      TYPE      $TMP1
3366 006244 123727 001176 000131      CMPB      $TMP1,#'Y      ;WAS IT A YES RESPONSE ??
3367 006252 001002          BNE      20$          ;NO - DONT TYPE HELP
3368 006254 104401 103566      TYPE      ,HELP          ;YES - TYPE HELP TEXT
3369 006260
3370          20$:
3371
3372          ;SEE IF USER WANTS TO CHANGE RM03 UNIBUS ADDRESS
3373 006260 104401 063772      TYPE      ,UBUSQST          ;WANT TO CHANGE ADDRESS ??
3374 006264 104411          RDCHR          ;GET RESPONSE
3375 006266 012637 001176      MOV      (SP)+,$TMP1      ;SAVE AND ECHO RESPONSE
3376 006272 104401 001176      TYPE      , $TMP1
3377 006276 104411          RDCHR          ;READ ONE MORE CHARACTER
3378 006300 005726          TST      (SP)+          ;CLEAR OFF THE STACK
3379 006302 123727 001176 000131      CMPB      $TMP1,#'Y      ;WAS IT A YES RESPONSE ??
3380 006310 001137          BNE      30$          ;NO !!
3381
3382          30$:
3383          ;DIALOGUE TO CHANGE THE UNIBUS ADDRESS, AND INTERRUPT VECTOR
3384 006312 104401 064130      TYPE      ,CNSLO1          ;TYPE CURRENT BUS ADDRESS
3385 006316 013746 001276      MOV      $BASE,-(SP)      ;SAVE $BASE FOR TYPEOUT
3386 006322 104402          TYPOC          ;GO TYPE--OCTAL ASCII(ALL DIGITS)
3387 006324 104401 063572      TYPE      ,CLSPRN
3388 006330 104413          RDOCT          ;GET NEW BUS ADDRESS
3389 006332 012637 001176      MOV      (SP)+,$TMP1      ;CARRIAGE RETURN??
3390 006336 001412          BEQ      50$          ;YES-SKIP TO NEXT ENTRY
3391 006340 022737 160000 001176      CMP      #160000,$TMP1      ;BASE ADDRESS IN I/O PAGE??
3392 006346 101403          BLOS      40$          ;YES
3393 006350 104401 064155      TYPE      ,CNSLO2          ;TYPE WARNING MESSAGE
3394 006354 000756          BR      30$          ;RETRY
3395 006356 013737 001176 001276 40$:      MOV      $TMP1,$BASE      ;STORE NEW BUS ADDRESS

```

```

3395 006364 113737 001272 001176 50$: MOV $VECT1,$TMP1 ;TYPE CURRENT VECTOR ADDRESS
3396 006372 105037 001177 CLRB $TMP1+1
3397 006376 104401 064236 TYPE ,CNSLO3
3398 006402 013746 001176 MOV $TMP1,-(SP) ;SAVE $TMP1 FOR TYPEOUT
3399 006406 104403 TYPEPOS ;GO TYPE--OCTAL ASCII
3400 006410 003 .BYTE 3 ;TYPE 3 DIGIT(S)
3401 006411 000 .BYTE 0 ;SUPPRESS LEADING ZEROS
3402 006412 104401 063572 TYPE ,CLSPRN
3403 006416 104413 RDOCT ;GET NEW VECTOR ADDRESS
3404 006420 012637 001176 MOV (SP)+,$TMP1 ;CARRIAGE RETURN?
3405 006424 001412 BEQ 70$ ;YES-SKIP TO NEXT ENTRY
3406 006426 022737 001000 001176 CMP #1000,$TMP1 ;VECTOR ADDRESS < 1000??
3407 006434 101003 BHI 60$ ;YES!!
3408 006436 104401 064266 TYPE ,CNSLO4 ;TYPE WARNING MESSAGE
3409 006442 000750 BR 50$ ;RETRY
3410 006444 113737 001176 001272 60$: MOV $TMP1,$VECT1 ;STORE NEW VECTOR ADDRESS
3411 006452 113737 001273 001176 70$: MOV $VECT1+1,$TMP1 ;TYPE CURRENT PRIORITY
3412 006460 006237 001176 ASR $TMP1
3413 006464 006237 001176 ASR $TMP1
3414 006470 006237 001176 ASR $TMP1
3415 006474 006237 001176 ASR $TMP1
3416 006500 006237 001176 ASR $TMP1
3417 006504 105037 001177 CLRB $TMP1+1
3418 006510 104401 064342 TYPE ,CNSLO5
3419 006514 013746 001176 MOV $TMP1,-(SP) ;SAVE $TMP1 FOR TYPEOUT
3420 006520 104403 TYPEPOS ;GO TYPE--OCTAL ASCII
3421 006522 001 .BYTE 1 ;TYPE 1 DIGIT(S)
3422 006523 000 .BYTE 0 ;SUPPRESS LEADING ZEROS
3423 006524 104401 063572 TYPE ,CLSPRN
3424 006530 104413 RDOCT ;GET NEW PRIORITY
3425 006532 012637 001176 MOV (SP)+,$TMP1 ;CARRIAGE RETURN?
3426 006536 001424 BEQ 90$ ;YES-SKIP TO NEXT ENTRY
3427 006540 023727 001176 000007 CMP $TMP1,#7 ;LEGAL PRIORITY??
3428 006546 002403 BLT 80$ ;YES!!
3429 006550 104401 064376 TYPE ,CNSLO6 ;TYPE WARNING MESSAGE
3430 006554 000736 BR 70$ ;RETRY
3431 006556 006337 001176 80$: ASL $TMP1 ;STORE NEW PRIORITY
3432 006562 006337 001176 ASL $TMP1
3433 006566 006337 001176 ASL $TMP1
3434 006572 006337 001176 ASL $TMP1
3435 006576 006337 001176 ASL $TMP1
3436 006602 113737 001176 001273 90$: MOV $TMP1,$VECT1+1
3437 006610 ;DIALOGUE TO INPUT DEVICE NUMBERS
3438
3439
3440
3441 006610 005037 001300 CLR $DEVN ;CLEAR DEVICE MAP
3442 006614 104401 064423 TYPE ,CNSLO7 ;TYPE INPUT INSTRUCTIONS
3443 006620 104401 063575 TYPE ,PROMPT ;TYPE PROMPTING CHARACTER
3444 006624 104411 RDCHR ;GET RESPONSE
3445 006626 012637 001176 MOV (SP)+,$TMP1 ;ECHO RESPONSE
3446 006632 104401 001176 TYPE $TMP1
3447 006636 023727 001176 000101 CMP $TMP1,#'A ;TEST ALL DRIVES??
3448 006644 001021 BNE 110$ ;NO
3449 006646 000137 006014 JMP XSIZ ;AUTO SIZE
3450 006652 012737 000377 001300 MOV #377,$DEVN ;TEST ALL DEVICES

```

K06

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19MACY11 30(1046) 23-NOV-77 12:38 PAGE 75
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0075

3451	006660	000444			BR	140\$; SKIP TO NEXT ENTRY
3452	006662	104401	063575	100\$:	TYPE	, PROMPT	; TYPE PROMPTING CHARACTER
3453	006666	104411			RDCHR		; GET RESPONSE
3454	006670	012637	001176		MOV	(SP)+, STMP1	; ECHO RESPONSE
3455	006674	104401	001176		TYPE	STMP1	
3456	006700	023727	001176	000015	CMP	STMP1, #CR	; CARRIAGE RETURN??
3457	006706	001431			BEQ	140\$	
3458	006710	023727	001176	000060	110\$:	CMP	STMP1, #'0
3459	006716	002404			BLT	120\$; YES!!
3460	006720	023727	001176	000067	CMP	STMP1, #'7	; NUMBER > 7??
3461	006726	003403			BLE	130\$; NO!!
3462	006730	104401	063601	120\$:	TYPE	, QSTMRK	; TYPE "??"
3463	006734	000752			BR	100\$; RETRY
3464	006736	013701	001176	130\$:	MOV	STMP1, R1	; R1=DRIVE NUMBER
3465	006742	042701	177770		BIC	#1C7, R1	
3466	006746	116102	064774		MOVB	ATNTBL(R1), R2	; DECODE DEVICE NUMBER
3467	006752	042702	177400		BIC	#1C377, R2	; CLEAR UNUSED BITS
3468	006756	050237	001300		BIS	R2, \$DEVN	; SET DEVICE # IN MAP
3469	006762	122737	000377	001300	CMPB	#377, \$DEVN	; DONE ??
3470	006770	101334			BHI	100\$; NO
3471	006772			140\$:			
3472							

```

3473 006772
3474
3475
3476 006772 013700 001300
3477 006776 012701 001452
3478 007002 010137 001450
3479 007006 012702 000001
3480 007012 005003
3481 007014 030200
3482 007016 001406
3483 007020 010311
3484 007022 116361 064774 000001
3485 007030 062701 000002
3486 007034 006302
3487 007036 105702
3488 007040 001402
3489 007042 005203
3490 007044 000763
3491 007046 005011
3492
3493
3494 007050 004737 040204
3495 007054 000403
3496 007056 104000
3497 007060 000000
3498 007062 000775
3499 007064
3500
3501 007064 005737 000042
3502 007070 001012
3503 007072 123727 001242 000001
3504 007100 001406
3505 007102 023727 001154 000176
3506 007110 001005
3507 007112 104407
3508 007114 000403
3509 007116 112737 000001 001150
3510 007124
3511 007124 000240
3512 007126 005037 001326
3513 007132 004737 061400
3514 007136 013746 000300
3515 007142 012746 007150
3516 007146 000002
3517 007150
3518 007150 117737 172274 001234
3519 007156 005037 001474

```

CMNSTART:

; ASSEMBLE TEST QUE FROM DEVICE MAP

```

      MOV      $DEVN,R0      ; R0 = DEVICE MAP
      MOV      @TSTQUE+2,R1  ; R1 = ADDRESS OF FIRST ENTRY IN QUE
      MOV      R1,TSTQUE     ; INITIALIZE ENTRY POINTER
      MOV      R1,R2         ; R2 = DEVICE POINTER
      CLR      R3            ; R3 = DEVICE NUMBER
10$:   BIT      R2,R0         ; IS THIS DEVICE IN MAP ??
      BEQ      20$           ; NO !!
      MOV      R3,(R1)       ; YES - ENTER DEVICE NUMBER IN QUE
      MOV      @TBL(R3),1(R1) ; ENTER ATTENTION BIT IN QUE
      ADD      R2,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$:   ERROR
      HALT
      BR       30$           ; NO CLOCK
40$:
.SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
      TST      0$42          ; ARE WE RUNNING UNDER XXDP/ACT?
      BNE      64$           ; 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
      BR       65$           ; GET SOFT-SWR SETTINGS
64$:   MOV      R1,$AUTOB    ; SET AUTO-MODE INDICATOR
65$:
READY: NOP
      CLR      CTLFG         ; READY TO START TEST
      JSR      PC,$TKINT     ; CLEAR THE CONTROL-C FLAG
      MOV      PR6,-(SP)     ; INITIALIZE TTY
      MOV      R6,-(SP)      ; PUT NEW PS ON STACK
      RTI                  ; PUT NEW PC ON STACK
                        ; POP NEW PC AND PS
64$:   MOV      @TSTQUE,$UNIT ; LOAD UNIT NUMBER
      CLR      MEDENB        ; CLEAR MEDIA ENABLE

```

```

3520 .....
3521 ;*TEST 1 CONTROLLER ACCESS TEST
3522 .....
3523 ;*ST1:
3524 NOP ;START OF TEST
3525 MOV #15,$LPADR
3526 MOV #15,$LPERR
3527 15:
3528 MOV #STACK,SP ;INITIALIZE STACK POINTER
3529 MOV $BASE,R0 ;R0=UNIBUS ADDRESS
3530 MOV TSTQUE,R1 ;(R1) = DEVICE BEING TESTED
3531 MOV #1,$TESTN ;SET TEST NUMBER IN APT MAIL BOX
3532 CLR R1
3533 MOV ERRVEC,-(SP) ;PUSH ERRVEC ON STACK
3534 MOV ERRVEC+2,-(SP) ;PUSH ERRVEC+2 ON STACK
3535 MOV #3,$ERRVEC
3536 MOV #PR6,ERRVEC+2
3537
3538 MOV R1,RMCS1+1(R0) ;MOVE HI BYTE TO RMCS1
3539 MOV R1,RMWC(R0) ;MOVE WORD COUNT REGISTER
3540 MOV RMWC(R0),R2
3541 MOV R1,RMBA(R0) ;MOVE BUS ADDRESS REGISTER
3542 MOV RMBA(R0),R2
3543 MOV R1,RMCS2(R0) ;MOVE CONTROL STATUS REGISTER
3544 MOV RMCS2(R0),R2
3545 MOV R1,RM08(R0) ;MOVE DATA BUFFER
3546 MOV RM08(R0),R2
3547 MOV (SP)+,ERRVEC+2 ;POP STACK INTO ERRVEC+2
3548 MOV (SP)+,ERRVEC ;POP STACK INTO ERRVEC
3549 BR 75 ;NO BUS TIMEOUT OCCURRED
3550
3551 35: CMP (SP)+,(SP)+ ;ADJUST STACK
3552 MOV (SP)+,ERRVEC+2 ;POP STACK INTO ERRVEC+2
3553 MOV (SP)+,ERRVEC ;POP STACK INTO ERRVEC
3554 ERROR 110 ;CANNOT ACCESS MASSBUS CONTROLLER
3555 TST 42 ;STAND ALONE MODE??
3556 BNE 55 ;NO!!
3557 JMP START ;YES-GO GET $BASE
3558 55: JMP SEOP ;GO TO END OF PASS HANDLER
3559 75:
3560 .....
3561 ;*TEST 2 DEVICE AVAILABLE TEST
3562 .....
3563 ;*ST2:
3564 SCOPE ;SCOPE CALL
3565 NOP ;START OF TEST
3566 MOV #STACK,SP ;INITIALIZE STACK POINTER
3567 MOV $BASE,R0 ;R0=UNIBUS ADDRESS
3568 MOV TSTQUE,R1 ;(R1) = DEVICE BEING TESTED
3569 MOV #2,$TESTN ;SET TEST NUMBER IN APT MAIL BOX
3570
3571 JSR PC,CNTCLR
3572 BR 25 ;GO TO 25 IF NO ERROR
3573 NOP ;RETURN HERE IF ERROR
3574
3575

```

```

3576 007416 104000          ERROR          ;ERROR NUMBER DEFINED BY SUB
3577 007420 000137 007540    2$:          JMP      7$          ;GO TO 7$ IF ERROR WAS FOUND
3578 007424                  ;
3579 007424 013746 000004    MOV      ERRVEC, -(SP)      ;; PUSH ERRVEC ON STACK
3580 007430 013746 000004    MOV      ERRVEC+2, -(SP)    ;; PUSH ERRVEC+2 ON STACK
3581 007434 012737 000004    MOV      #5$, ERRVEC
3582 007442 013737 000006    MOV      PR6, ERRVEC+2
3583
3584 007450 016037 000000 001176    MOV      RMCS1(R0), $TMP1    ;GET DVA STATUS
3585 007456 016037 000010 001174    MOV      RMCS2(R0), $TMP0    ;GET NED STATUS
3586 007464 012637 000006    MOV      (SP)+, ERRVEC+2    ;; POP STACK INTO ERRVEC+2
3587 007470 012637 000004    MOV      (SP)+, ERRVEC      ;; POP STACK INTO ERRVEC
3588 007474 032737 010000 001174    BIT      #NED, $TMP0      ;; NONEXISTENT DEVICE??
3589 007502 001402          BEQ      3$          ;NO!!
3590 007504 104111          ERROR    111          ;NONEXISTENT DEVICE
3591 007506 000414          BR       7$
3592 007510 032737 004000 001176    3$:      BIT      #DVA, $TMP1      ;DEVICE AVAILABLE??
3593 007516 001012          BNE      9$          ;YES!!
3594 007520 104112          ERROR    112          ;DEVICE NOT AVAILABLE
3595 007522 000406          BR       7$
3596
3597 007524 022626          5$:      CMP      (SP)+, (SP)+    ;ADJUST STACK
3598 007526 012637 000006    MOV      (SP)+, ERRVEC+2    ;; POP STACK INTO ERRVEC+2
3599 007532 012637 000004    MOV      (SP)+, ERRVEC      ;; POP STACK INTO ERRVEC
3600 007536 104113          ERROR    113          ;BUS TIMEOUT (04 TRAP)
3601 007540 000137 032732    7$:      JMP      $E0SP
3602
3603 007544          9$:
3604
3605          ;*****
3606          ;*TEST 3          DRIVE TYPE TEST
3607
3608          ;*****
3609          ;*TST3:
3610          SCOPE          ;SCOPE CALL
3611          NOP          ;START OF TEST
3612 007546 000240          MOV      #STACK, SP      ;INITIALIZE STACK POINTER
3613 007550 012706 001100    MOV      $BASE, R0      ;R0=UNIBUS ADDRESS
3614 007554 013700 001276    MOV      TSTQUE, R1     ;(R1) = DEVICE BEING TESTED
3615 007560 013701 001450    MOV      #3, $TESTN    ;SET TEST NUMBER IN APT MAIL BOX
3616 007564 012737 000003 001226
3617 007572 004737 046660    JSR      PC, CNTCLR
3618 007576 000404          BR       2$          ;GO TO 2$ IF NO ERROR
3619 007600 000240          NOP          ;RETURN HERE IF ERROR
3620 007602 104000          ERROR          ;ERROR NUMBER DEFINED BY SUB
3621 007604 000137 007722    JMP      4$          ;GO TO 4$ IF ERROR WAS FOUND
3622
3623 007610          2$:
3624 007610 112737 000026 001506    MOVB     #RMDT, GETINX    ;SETUP GET INDEX TABLE
3625 007616 112737 000200 001507    MOVB     #200, GETINX+1
3626 007624 012737 007726 001356    MOV      #5$, RMDTI      ;RMDT INPUT BUFFER = 5$
3627 007632 004737 037516    JSR      PC, GET          ;GO GET DRIVE TYPE
3628 007636 000402          BR       3$          ;GO TO 3$ IF NO ERROR
3629 007640 000240          NOP          ;RETURN HERE IF ERROR
3630 007642 104000          ERROR          ;ERROR NUMBER SPECIFIED BY GET
3631 007644 022737 020024 001356    3$:      CMP      #SNGPRT, RMDTI    ;SINGLE PORT RM03??
3632 007652 001425          BEQ      5$          ;YES!!

```



```

3632 007654 022737 024024 001356      CMP      #DULPRT,RMDTI ;DUAL PORT RMD3??
3633 007662 001421                BEQ      5$ ;YES!!
3634 007664 022737 020025 001356      CMP      #SNGPRT!BITO,RMDTI
3635 007672 001415                BEQ      5$
3636 007674 022737 024025 001356      CMP      #DULPRT!BITO,RMDTI
3637 007702 001411                BEQ      5$
3638 007704 012737 020024 001176      MOV      #SNGPRT,$TMP1
3639 007712 012737 024024 001200      MOV      #DULPRT,$TMP2
3640 007720 104114                ERROR    114 ;NOT AN RMD3
3641 007722 000137 032732      4$:      JMP      $E0SP ;GO TO SUBPASS HANDLER.
3642
3643 007726
3644
3645
3646
3647 007726
3648 007726 000004
3649 007730 000240
3650 007732 012706 001100
3651 007736 013700 001276
3652 007742 013701 001450
3653 007746 012737 000004 001226
3654 007754
3655
3656
3657 007754 012737 000000 001434
3658 007762 012737 000000 001406
3659 007770 012737 000000 001432
3660 007776 012737 177376 001402
3661 010004 012737 103566 001404
3662 010012 012737 000062 001400
3663 010020 012737 065106 001174
3664 010026 012737 000001 001176
3665 010034 004737 036620
3666 010040
3667
3668
3669 010040 004737 034000
3670 010044 154130
3671 010046 000404
3672 010050 000240
3673 010052 104000
3674 010054 000137 010526
3675 010060
3676
3677
3678 010060 012737 000005 001400
3679 010066 012702 001535
3680 010072 112722 000006
3681 010076 112722 000034
3682 010102 112722 000032
3683 010106 112722 000000
3684 010112 112722 000200
3685 010116 004737 037766
3686 010122 000404
3687 010124 000240

      CMP      #DULPRT,RMDTI ;DUAL PORT RMD3??
      BEQ      5$ ;YES!!
      CMP      #SNGPRT!BITO,RMDTI
      BEQ      5$
      CMP      #DULPRT!BITO,RMDTI
      BEQ      5$
      MOV      #SNGPRT,$TMP1
      MOV      #DULPRT,$TMP2
      ERROR    114 ;NOT AN RMD3
      JMP      $E0SP ;GO TO SUBPASS HANDLER.

5$:
*****
;TEST 4          FORMAT ZEROS - 18
*****
TST4:
      SCOPE
      MOV      #STACK,SP ;SCOPE CALL
      MOV      $BASE,R0 ;START OF TEST
      MOV      TSTQUE,R1 ;INITIALIZE STACK POINTER
      MOV      #4,$TESTN ;R0=UNIBUS ADDRESS
                        ;(R1) = DEVICE BEING TESTED
                        ;;SET TEST NUMBER IN APT MAIL BOX

10$:
;SETUP PARAMETERS FOR GENERATING DATA BUFFER
      MOV      #0,RMDC0 ;CYLINDER = 0
      MOV      #0,RMDA0 ;TRACK = SECTOR = 0
      MOV      #0,RMFO0 ;18 BIT FORMAT
      MOV      #((C<2+256.>+1),RMWC0 ;2 + 256 WORDS
      MOV      #BUFONE,RMBAO ;DATA BUFFER ADDRESS
      MOV      #MH,RMCS10 ;WRITE HEADER AND DATA
      MOV      #ZEROS,$TMP0 ;USE ALL ZEROS DATA PATTERN
      MOV      #1,$TMP1
      JSR      PC,GENBUF ;GO GENERATE DATA BUFFER

20$:
;PREPARE DEVICE FOR DATA TRANSFER
      JSR      PC,TSTPRP ;PREPARE DEVICE FOR TEST
      WORD    154130 ;TASK DESCRIPTOR
      BR      30$ ;GO TO 30$ IF NO ERROR
      NOP
      ERROR    ;RETURN HERE IF ERROR
      JMP      180$ ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                        ;GO TO 180$ IF ERROR WAS FOUND

30$:
;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
      MOV      #SEEK!GO,RMCS10 ;CHANGE COMMAND TO SEEK
      MOV      #PUTINX,R2 ;WRITE REGISTER INDEX TABLE
      MOVB     #RMDA,(R2)+
      MOVB     #RMDC,(R2)+
      MOVB     #RMFO,(R2)+
      MOVB     #RMCS1,(R2)+
      MOVB     #200,(R2)+
      JSR      PC,PUT ;GO WRITE REGISTERS VIA PUT SUB
      BR      40$ ;GO TO 40$ IF NO ERROR
      NOP
      ;RETURN HERE IF ERROR

```

C07

3688 010126 104000
3689 010130 000137 010526
3690 010134
3691
3692
3693
3694 010134 004737 037432
3695 010140 004737 040326
3696 010144
3697
3698 010144 004737 037516
3699 010150 000404
3700 010152 000240
3701 010154 104000
3702 010156 000137 010526
3703 010162
3704
3705
3706 010162 004737 045420
3707 010166 000405
3708 010170 000240
3709 010172 104000
3710 010174 004737
3711 010176 000137 010526
3712 010202
3713
3714
3715 010202 012737 000063 001400
3716 010210 012702 001540
3717 010214 112722 000002
3718 010220 112722 000004
3719 010224 112722 000000
3720 010230 112722 000200
3721 010234 004737 037766
3722 010240 000404
3723 010242 000240
3724 010244 104000
3725 010246 000137 010526
3726 010252
3727
3728
3729 010252 004737 040326
3730 010256 004737 037516
3731 010262 000404
3732 010264 000240
3733 010266 104000
3734 010270 000137 010526
3735 010274
3736
3737
3738 010274 004737 040512
3739 010300 000405
3740 010302 000240
3741 010304 104000
3742 010306 004737
3743 010310 000137 010526

ERROR ;ERROR DEFINED BY PUT SUB
JMP 180\$;GO TO 180\$ IF ERROR WAS FOUND
40\$:
;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
JSR PC,GETSTS ;SETUP FOR STATUS
JSR PC,TIMOUT ;WAIT FOR SEEK TO COMPLETE
50\$:
;GO READ SEEK STATUS
JSR PC,GET ;GO READ REGISTERS VIA GET SUB
BR 60\$;GO TO 60\$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR DEFINED BY GET SUB
JMP 180\$;GO TO 180\$ IF ERROR WAS FOUND
60\$:
;VERIFY THE RESULTS OF THE SEEK COMMAND
JSR PC,SEKSTS ;GO VERIFY RESULTS OF SEEK OPERATION
BR 70\$;GO TO 70\$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR # DEFINED BY SEKSTS SUBROUTINE
JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
JMP 180\$;GO TO 180\$ IF ERROR WAS FOUND
70\$:
;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
MOV #10, RMC510 ;LOAD WRITE HEADER AND DATA
MOV #PUTINX+3, R2 ;EXTEND REGISTER INDEX TABLE
MOVB #A, (R2)+
MOVB #RMC51, (R2)+
MOVB #200, (R2)+
JSR PC,PUT ;GO WRITE REGISTERS VIA PUT SUB
BR 80\$;GO TO 80\$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR DEFINED BY PUT SUB
JMP 180\$;GO TO 180\$ IF ERROR WAS FOUND
80\$:
;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
JSR PC,TIMOUT ;WAIT FOR COMMAND TO COMPLETE
JSR PC,GET ;GO READ REGISTERS VIA GET SUB
BR 90\$;GO TO 90\$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR DEFINED BY GET SUB
JMP 180\$;GO TO 180\$ IF ERROR WAS FOUND
90\$:
;VERIFY RESULTS OF WRITE COMMAND
JSR PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
BR 100\$;GO TO 100\$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR # DEFINED BY PRIERR SUBROUTINE
JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
JMP 180\$;GO TO 180\$ IF ERROR WAS FOUND

```

3744 010314
3745 010314 004737 053016
3746 010320 000405
3747 010322 000240
3748 010324 104000
3749 010326 004736
3750 010330 000137 010526
3751 010334
3752 010334 004737 041344
3753 010340 000405
3754 010342 000240
3755 010344 104000
3756 010346 004736
3757 010350 000137 010526
3758 010354
3759
3760
3761
3762 010354 012737 000073 001400
3763 010362 012737 104572 001404
3764 010370 004737 037766
3765 010374 000404
3766 010376 000240
3767 010400 104000
3768 010402 000137 010526
3769 010406
3770
3771
3772 010406 004737 040326
3773 010412 004737 037516
3774 010416 000404
3775 010420 000240
3776 010422 104000
3777 010424 000137 010526
3778 010430
3779
3780
3781 010430 004737 040512
3782 010434 000405
3783 010436 000240
3784 010440 104000
3785 010442 004736
3786 010444 000137 010526
3787 010450
3788 010450 004737 053016
3789 010454 000405
3790 010456 000240
3791 010460 104000
3792 010462 004736
3793 010464 000137 010526
3794 010470
3795 010470 004737 041344
3796 010474 000405
3797 010476 000240
3798 010500 104000
3799 010502 004736

100$: JSR PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
      BR 110$ ;GO TO 110$ IF NO ERROR
      NOP ;RETURN HERE IF ERROR
      ERROR ;ERROR # DEFINED BY DTASTS SUBROUTINE
      JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
      JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND

110$: JSR PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
      BR 120$ ;GO TO 120$ IF NO ERROR
      NOP ;RETURN HERE IF ERROR
      ERROR ;ERROR # DEFINED BY SECERR SUBROUTINE
      JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
      JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND

120$:
;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
      MOV #RH:GO,RMC510 ;READ HEADER & DATA COMMAND
      MOV #BUFTWO,RMB40 ;CHANGE BUS ADDRESS
      JSR PC,PUT ;GO WRITE REGISTERS VIA PUT SUB
      BR 130$ ;GO TO 130$ IF NO ERROR
      NOP ;RETURN HERE IF ERROR
      ERROR ;ERROR DEFINED BY PUT SUB
      JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND

130$:
;WAIT FOR READ TO COMPLETE AND GET STATUS
      JSR PC,TIMOUT ;WAIT FOR READ TO COMPLETE
      JSR PC,GET ;GO READ REGISTERS VIA GET SUB
      BR 140$ ;GO TO 140$ IF NO ERROR
      NOP ;RETURN HERE IF ERROR
      ERROR ;ERROR DEFINED BY GET SUB
      JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND

140$:
;VERIFY THE RESULTS OF READ OPERATION
      JSR PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
      BR 150$ ;GO TO 150$ IF NO ERROR
      NOP ;RETURN HERE IF ERROR
      ERROR ;ERROR # DEFINED BY PRIERR SUBROUTINE
      JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
      JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND

150$: JSR PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
      BR 160$ ;GO TO 160$ IF NO ERROR
      NOP ;RETURN HERE IF ERROR
      ERROR ;ERROR # DEFINED BY DTASTS SUBROUTINE
      JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
      JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND

160$: JSR PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
      BR 170$ ;GO TO 170$ IF NO ERROR
      NOP ;RETURN HERE IF ERROR
      ERROR ;ERROR # DEFINED BY SECERR SUBROUTINE
      JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS

```

```

3800 010504 000137 010526      JMP      180$      ;GO TO 180$ IF ERROR WAS FOUND
3801 010510
3802
3803      ;VERIFY DATA
3804 010510 004737 037064      JSR      PC,CMPBUF      ;GO COMPARE WRITE, READ DATA BUFFERS
3805 010514 103566      .WORD    BUFO1      ;STARTING ADDRESS OF WRITE BUFFER
3806 010516 104572      .WORD    BUFTWO     ;STARTING ADDRESS OF READ BUFFER
3807 010520 000402      BR        180$      ;GO TO 180$ IF NO ERROR
3808 010522 000240      NOP
3809 010524 104000      ERROR      ;RETURN HERE IF ERROR
3810 010526      180$:      ;ERROR # DEFINED BY CMPBUF SUBROUTINE
3811
3812      ;*****
3813      ;*TEST 5      FORMAT ZEROS - 16
3814      ;*****
3815      TST5:
3816 010526 000004      SCOPE      ;SCOPE CALL
3817 010530 000240      NOP      ;START OF TEST
3818 010532 012706 001100      MOV      #STACK,SP      ;INITIALIZE STACK POINTER
3819 010536 013700 001276      MOV      $BASE,R0      ;R0=UNIBUS ADDRESS
3820 010542 013701 001450      MOV      TSTQUE,R1      ;(R1) = DEVICE BEING TESTED
3821 010546 012737 000005 001226      MOV      #5,$TESTN      ;SET TEST NUMBER IN APT MAIL BOX
3822 010554
3823
3824      ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
3825 010554 012737 000000 001434      MOV      #0,RMDC0      ;CYLINDER = 0
3826 010562 012737 000000 001406      MOV      #0,RMDA0      ;TRACK = SECTOR = 0
3827 010570 012737 010000 001432      MOV      #FMT16,RMOFO      ;16 BIT FORMAT
3828 010576 012737 177376 001402      MOV      #(<1C(2+256.)+1>),RMWCO      ;? + 256 WORDS
3829 010604 012737 103566 001404      MOV      #BUFO1,RMBAO      ;DATA BUFFER ADDRESS
3830 010612 012737 000062 001400      MOV      #WH,RMCS10      ;WRITE HEADER AND DATA
3831 010620 012737 065106 001174      MOV      #ZEROS,$TMPD      ;USE ALL ZEROS DATA PATTERN
3832 010626 012737 000001 001176      MOV      #1,$TMP1
3833 010634 004737 036620      JSR      PC,GENBUF      ;GO GENERATE DATA BUFFER
3834 010640
3835
3836      ;PREPARE DEVICE FOR DATA TRANSFER
3837 010640 004737 034000      JSR      PC,TSTPRP      ;PREPARE DEVICE FOR TEST
3838 010644 154130      .WORD    154130      ;TASK DESCRIPTOR
3839 010646 000404      BR        30$      ;GO TO 30$ IF NO ERROR
3840 010650 000240      NOP      ;RETURN HERE IF ERROR
3841 010652 104000      ERROR      ;ERROR # DEFINED BY TSTPRP SUBROUTINE
3842 010654 000137 011326      JMP      180$      ;GO TO 180$ IF ERROR WAS FOUND
3843 010660
3844
3845      30$:
3846      ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
3847 010660 012737 000005 001400      MOV      #SEEK!GO,RMCS10      ;CHANGE COMMAND TO SEEK
3848 010666 012702 001535      MOV      #PUTINX,R2      ;WRITE REGISTER INDEX TABLE
3849 010672 112722 000006      MOVB     #RMDA,(R2)+
3850 010676 112722 000034      MOVB     #RMDC,(R2)+
3851 010702 112722 000032      MOVB     #RMOF,(R2)+
3852 010706 112722 000000      MOVB     #RMCS1,(R2)+
3853 010712 112722 000200      MOVB     #200,(R2)+
3854 010716 004737 037766      JSR      PC,PUT      ;GO WRITE REGISTERS VIA PUT SUB
3855 010724 000404      BR        40$      ;GO TO 40$ IF NO ERROR
      NOP      ;RETURN HERE IF ERROR

```

3856	010726	104000		ERROR		;ERROR DEFINED BY PUT SUB
3857	010730	000137	011326	JMP	180\$;GO TO 180\$ IF ERROR WAS FOUND
3858	010734			40\$:		
3859						
3860						;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
3861	010734	004737	037432	JSR	PC,GETSTS	;SETUP FOR STATUS
3862	010740	004737	040326	JSR	PC,TIMOUT	;WAIT FOR SEEK TO COMPLETE
3863	010744			50\$:		
3864						
3865						;GO READ SEEK STATUS
3866	010744	004737	037516	JSR	PC,GET	;GO READ REGISTERS VIA GET SUB
3867	010750	000404		BR	60\$;GO TO 60\$ IF NO ERROR
3868	010752	000240		NOP		;RETURN HERE IF ERROR
3869	010754	104000		ERROR		;ERROR DEFINED BY GET SUB
3870	010756	000137	011326	JMP	180\$;GO TO 180\$ IF ERROR WAS FOUND
3871	010762			60\$:		
3872						
3873						;VERIFY THE RESULTS OF THE SEEK COMMAND
3874	010762	004737	045420	JSR	PC,SEKSTS	;GO VERIFY RESULTS OF SEEK OPERATION
3875	010766	000405		BR	70\$;GO TO 70\$ IF NO ERROR
3876	010770	000240		NOP		;RETURN HERE IF ERROR
3877	010772	104000		ERROR		;ERROR # DEFINED BY SEKSTS SUBROUTINE
3878	010774	004736		JSR	PC,(SP)+	;GO BACK FOR MORE ERROR CHECKS
3879	010776	000137	011326	JMP	180\$;GO TO 180\$ IF ERROR WAS FOUND
3880	011002			70\$:		
3881						
3882						;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
3883	011002	012737	000063	MOV	#WH!GO,RMCS10	;LOAD WRITE HEADER AND DATA
3884	011010	012702	001540	MOV	#PUTINX+3,R2	;EXTEND REGISTER INDEX TABLE
3885	011014	112722	000002	MOVB	#RMWC,(R2)+	
3886	011020	112722	000004	MOVB	#RMBR,(R2)+	
3887	011024	112722	000000	MOVB	#RMCS1,(R2)+	
3888	011030	112722	000200	MOVB	#200,(R2)+	
3889	011034	004737	037766	JSR	PC,PUT	;GO WRITE REGISTERS VIA PUT SUB
3890	011040	000404		BR	80\$;GO TO 80\$ IF NO ERROR
3891	011042	000240		NOP		;RETURN HERE IF ERROR
3892	011044	104000		ERROR		;ERROR DEFINED BY PUT SUB
3893	011046	000137	011326	JMP	180\$;GO TO 180\$ IF ERROR WAS FOUND
3894	011052			80\$:		
3895						
3896						;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
3897	011052	004737	040326	JSR	PC,TIMOUT	;WAIT FOR COMMAND TO COMPLETE
3898	011056	004737	037516	JSR	PC,GET	;GO READ REGISTERS VIA GET SUB
3899	011062	000404		BR	90\$;GO TO 90\$ IF NO ERROR
3900	011064	000240		NOP		;RETURN HERE IF ERROR
3901	011066	104000		ERROR		;ERROR DEFINED BY GET SUB
3902	011070	000137	011326	JMP	180\$;GO TO 180\$ IF ERROR WAS FOUND
3903	011074			90\$:		
3904						
3905						;VERIFY RESULTS OF WRITE COMMAND
3906	011074	004737	040512	JSR	PC,PRIERR	;GO CHECK FOR PRIMARY ERRORS
3907	011100	000405		BR	100\$;GO TO 100\$ IF NO ERROR
3908	011102	000240		NOP		;RETURN HERE IF ERROR
3909	011104	104000		ERROR		;ERROR # DEFINED BY PRIERR SUBROUTINE
3910	011106	004736		JSR	PC,(SP)+	;GO BACK FOR MORE ERROR CHECKS
3911	011110	000137	011326	JMP	180\$;GO TO 180\$ IF ERROR WAS FOUND

```

3912 011114 004737 053016 100$: JSR PC_DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
3913 011114 000405 BR 110$ ;GO TO 110$ IF NO ERROR
3914 011120 000240 NOP ;RETURN HERE IF ERROR
3915 011122 104000 ERROR ;ERROR # DEFINED BY DTASTS SUBROUTINE
3916 011124 004736 JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
3917 011126 000137 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
3918 011130 011326
3919 011134 110$: JSR PC_SECERR ;GO CHECK FOR SECONDARY ERRORS
3920 011134 004737 041344 BR 120$ ;GO TO 120$ IF NO ERROR
3921 011140 000405 NOP ;RETURN HERE IF ERROR
3922 011142 000240 ERROR ;ERROR # DEFINED BY SECERR SUBROUTINE
3923 011144 104000 JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
3924 011146 004736 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
3925 011150 000137 011326 120$:
3926 011154
3927
3928
3929 ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
3930 011154 012737 000073 001400 MOV #RH!GO,RMC510 ;READ HEADER & DATA COMMAND
3931 011162 012737 104572 001404 MOV #BUFTWO,RMBA0 ;CHANGE BUS ADDRESS
3932 011170 004737 037766 JSR PC_PUT ;GO WRITE REGISTERS VIA PUT SUB
3933 011174 000404 BR 130$ ;GO TO 130$ IF NO ERROR
3934 011176 000240 NOP ;RETURN HERE IF ERROR
3935 011200 104000 ERROR ;ERROR DEFINED BY PUT SUB
3936 011202 000137 011326 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
3937 011206 130$:
3938
3939 ;WAIT FOR READ TO COMPLETE AND GET STATUS
3940 011206 004737 040326 JSR PC_TIMEOUT ;WAIT FOR READ TO COMPLETE
3941 011212 004737 037516 JSR PC_GET ;GO READ REGISTERS VIA GET SUB
3942 011216 000404 BR 140$ ;GO TO 140$ IF NO ERROR
3943 011220 000240 NOP ;RETURN HERE IF ERROR
3944 011222 104000 ERROR ;ERROR DEFINED BY GET SUB
3945 011224 000137 011326 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
3946 011230 140$:
3947
3948 ;VERIFY THE RESULTS OF READ OPERATION
3949 011230 004737 040512 JSR PC_PRIERR ;GO CHECK FOR PRIMARY ERRORS
3950 011234 000405 BR 150$ ;GO TO 150$ IF NO ERROR
3951 011236 000240 NOP ;RETURN HERE IF ERROR
3952 011240 104000 ERROR ;ERROR # DEFINED BY PRIERR SUBROUTINE
3953 011242 004736 JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
3954 011244 000137 011326 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
3955 011250 150$:
3956 011250 004737 053016 JSR PC_DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
3957 011254 000405 BR 160$ ;GO TO 160$ IF NO ERROR
3958 011256 000240 NOP ;RETURN HERE IF ERROR
3959 011260 104000 ERROR ;ERROR # DEFINED BY DTASTS SUBROUTINE
3960 011262 004736 JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
3961 011264 000137 011326 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
3962 011270 160$:
3963 011270 004737 041344 JSR PC_SECERR ;GO CHECK FOR SECONDARY ERRORS
3964 011274 000405 BR 170$ ;GO TO 170$ IF NO ERROR
3965 011276 000240 NOP ;RETURN HERE IF ERROR
3966 011300 104000 ERROR ;ERROR # DEFINED BY SECERR SUBROUTINE
3967 011302 004736 JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS

```

```

170$: JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND

;VERIFY DATA
JSR PC,CMPBUF ;GO COMPARE WRITE, READ DATA BUFFERS
.WORD BUFONE ;STARTING ADDRESS OF WRITE BUFFER
.WORD BUFTWO ;STARTING ADDRESS OF READ BUFFER
BR 180$ ;GO TO 180$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR # DEFINED BY CMPBUF SUBROUTINE

180$:

```

```

: *****
: *TEST 6          ZERO FILL TEST
: *****
↑TST6:
      SCOPE
      NOP
      MOV      #STACK_SP
      MOV      $BASE, R0
      MOV      TSTQUE, R1
      MOV      #6, $TESTN
;SCOPE CALL
;START OF TEST
;INITIALIZE STACK POINTER
;R0=UNIBUS ADDRESS
;(R1) = DEVICE BEING TESTED
;;SET TEST NUMBER IN APT MAIL BOX
10$:
```

```

;SETUP PARAMETERS FOR GENERATING DATA BUFFER
MOV      #0,RMDCO                      ;CYLINDER = 0
MOV      #0,RMDAO                      ;TRACK = SECTOR = 0
MOV      #FMT16,RMOFO                  ;16 BIT FORMAT
MOV      #(<C(2+256.)+1>),RMWCO       ;2 + 256 WORDS
MOV      #BUFONE,RMBAO                 ;DATA BUFFER ADDRESS
MOV      #WH,RMC$10                   ;WRITE HEADER AND DATA
MOV      #ZEROS,$TMP0                 ;USE ALL ZEROS DATA PATTERN
MOV      #1,$TMP1
JSR      PC,GENBUF                     ;GO GENERATE DATA BUFFER
20$:

```

```

;PREPARE DEVICE FOR DATA TRANSFER
JSR      PC TSTPRP      ;PREPARE DEVICE FOR TEST
.WORD    154130 ;TASK DESCRIPTOR
BR       30$            ;GO TO 30$ IF NO ERROR
NOP      ;RETURN HERE IF ERROR
ERROR    ;ERROR # DEFINED BY TSTPRP SUBROUTINE
JMP      180$           ;GO TO 180$ IF ERROR WAS FOUND

30$:

```

```

;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
MOV      #SEEK!GO,RMCS10          ;CHANGE COMMAND TO SEEK
MOV      #PUTINX,R2               ;WRITE REGISTER INDEX TABLE
MOVB     #RMOA,(R2)+
MOVB     #RMDC,(R2)+
MOVB     #RMOF,(R2)+
MOVB     #RMCS1,(R2)+
MOVB     #200,(R2)+
JSR      PC,PUT                   ;GO WRITE REGISTERS VIA PUT SUB
BR       40$                      ;GO TO 40$ IF NO ERROR
NOP                                ;RETURN HERE IF ERROR

```

```

4024 011526 104000          ERROR          ;ERROR DEFINED BY PUT SUB
4025 011530 000137 012142  JMP          180$ ;GO TO 180$ IF ERROR WAS FOUND
4026 011534
4027
4028 ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
4029 011534 004737 037432  JSR          PC,GETSTS ;SETUP FOR STATUS
4030 011540 004737 040326  JSR          PC,TIMOUT ;WAIT FOR SEEK TO COMPLETE
4031 011544
4032
4033 ;GO READ SEEK STATUS
4034 011544 004737 037516  JSR          PC,GET ;GO READ REGISTERS VIA GET SUB
4035 011550 000404          BR          60$ ;GO TO 60$ IF NO ERROR
4036 011552 000240          NOP          ;RETURN HERE IF ERROR
4037 011554 104000          ERROR        ;ERROR DEFINED BY GET SUB
4038 011556 000137 012142  JMP          180$ ;GO TO 180$ IF ERROR WAS FOUND
4039 011562
4040
4041 ;VERIFY THE RESULTS OF THE SEEK COMMAND
4042 011562 004737 045420  JSR          PC,SEKSTS ;GO VERIFY RESULTS OF SEEK OPERATION
4043 011566 000405          BR          70$ ;GO TO 70$ IF NO ERROR
4044 011570 000240          NOP          ;RETURN HERE IF ERROR
4045 011572 104000          ERROR        ;ERROR * DEFINED BY SEKSTS SUBROUTINE
4046 011574 004736          JSR          PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
4047 011576 000137 012142  JMP          180$ ;GO TO 180$ IF ERROR WAS FOUND
4048 011602
4049
4050 ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
4051 011602 012737 177776 001402  MOV          2(C2+1),RMWCO ;FORMAT PARTIAL SECTOR
4052 011610 012737 000063 001400  MOV          4H,GO,RMCS10 ;LOAD WRITE HEADER AND DATA
4053 011616 012702 001540          MOV          8PUTINX+3,R2 ;EXTEND REGISTER INDEX TABLE
4054 011622 112722 000002          MOVB        8RMWC,(R2)+
4055 011626 112722 000004          MOVB        8RMB,(R2)+
4056 011632 112722 000000          MOVB        8RMCS1,(R2)+
4057 011636 112722 000200          MOVB        8200,(R2)+
4058 011642 004737 037766  JSR          PC,PUT ;GO WRITE REGISTERS VIA PUT SUB
4059 011646 000404          BR          80$ ;GO TO 80$ IF NO ERROR
4060 011650 000240          NOP          ;RETURN HERE IF ERROR
4061 011652 104000          ERROR        ;ERROR DEFINED BY PUT SUB
4062 011654 000137 012142  JMP          180$ ;GO TO 180$ IF ERROR WAS FOUND
4063 011660
4064
4065 ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
4066 011660 004737 040326  JSR          PC,TIMOUT ;WAIT FOR COMMAND TO COMPLETE
4067 011664 004737 037516  JSR          PC,GET ;GO READ REGISTERS VIA GET SUB
4068 011670 000404          BR          90$ ;GO TO 90$ IF NO ERROR
4069 011672 000240          NOP          ;RETURN HERE IF ERROR
4070 011674 104000          ERROR        ;ERROR DEFINED BY GET SUB
4071 011676 000137 012142  JMP          180$ ;GO TO 180$ IF ERROR WAS FOUND
4072 011702
4073
4074 ;VERIFY RESULTS OF WRITE COMMAND
4075 011702 004737 040512  JSR          PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
4076 011706 000405          BR          100$ ;GO TO 100$ IF NO ERROR
4077 011710 000240          NOP          ;RETURN HERE IF ERROR
4078 011712 104000          ERROR        ;ERROR * DEFINED BY PRIERR SUBROUTINE
4079 011714 004736          JSR          PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS

```



```

4080 011716 000137 012142      JMP      180$      ;GO TO 180$ IF ERROR WAS FOUND
4081 011722      100$:      JSR      PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
4082 011722 004737 053016      BR      110$      ;GO TO 110$ IF NO ERROR
4083 011726 000405      NOP      ;RETURN HERE IF ERROR
4084 011730 000240      ERROR     ;ERROR # DEFINED BY DTASTS SUBROUTINE
4085 011732 104000      JSR      PC,2(SP)+      ;GO BACK FOR MORE ERROR CHECKS
4086 011734 004736      JMP      180$      ;GO TO 180$ IF ERROR WAS FOUND
4087 011736 000137 012142      110$:      JSR      PC,SECERR      ;GO CHECK FOR SECONDARY ERRORS
4088 011742      BR      120$      ;GO TO 120$ IF NO ERROR
4089 011742 004737 041344      NOP      ;RETURN HERE IF ERROR
4090 011746 000405      ERROR     ;ERROR # DEFINED BY SECERR SUBROUTINE
4091 011750 000240      JSR      PC,2(SP)+      ;GO BACK FOR MORE ERROR CHECKS
4092 011752 104000      JMP      180$      ;GO TO 180$ IF ERROR WAS FOUND
4093 011754 004736      120$:      JSR      PC,SECERR      ;GO CHECK FOR SECONDARY ERRORS
4094 011756 000137 012142      BR      170$      ;GO TO 170$ IF NO ERROR
4095 011762      NOP      ;RETURN HERE IF ERROR
4096      ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
4097      MOV      2(2+256)+1,RMWC0      ;READ A FULL SECTOR
4098      MOV      2(RH),GO,RMCS10      ;READ HEADER & DATA COMMAND
4099 011762 012737 177376 001402      MOV      2(BUFTWO),RMBAO      ;CHANGE BUS ADDRESS
4100 011770 012737 000073 001400      JSR      PC,PUT      ;GO WRITE REGISTERS VIA PUT SUB
4101 011776 012737 104572 001404      BR      130$      ;GO TO 130$ IF NO ERROR
4102 012004 004737 037766      NOP      ;RETURN HERE IF ERROR
4103 012010 000404      ERROR     ;ERROR DEFINED BY PUT SUB
4104 012012 000240      JSR      180$      ;GO TO 180$ IF ERROR WAS FOUND
4105 012014 104000      130$:      ;WAIT FOR READ TO COMPLETE AND GET STATUS
4106 012016 000137 012142      JSR      PC,TIMOUT      ;WAIT FOR READ TO COMPLETE
4107 012022      JSR      PC,GET      ;GO READ REGISTERS VIA GET SUB
4108      BR      140$      ;GO TO 140$ IF NO ERROR
4109      NOP      ;RETURN HERE IF ERROR
4110      ERROR     ;ERROR DEFINED BY GET SUB
4111 012022 004737 040326      JMP      180$      ;GO TO 180$ IF ERROR WAS FOUND
4112 012026 004737 037516      140$:      ;VERIFY THE RESULTS OF READ OPERATION
4113 012032 000404      JSR      PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
4114 012034 000240      BR      150$      ;GO TO 150$ IF NO ERROR
4115 012036 104000      NOP      ;RETURN HERE IF ERROR
4116 012040 000137 012142      ERROR     ;ERROR # DEFINED BY PRIERR SUBROUTINE
4117      JSR      PC,2(SP)+      ;GO BACK FOR MORE ERROR CHECKS
4118      JMP      180$      ;GO TO 180$ IF ERROR WAS FOUND
4119 012044 004737 040512      150$:      JSR      PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
4120 012050 000405      BR      160$      ;GO TO 160$ IF NO ERROR
4121 012052 000240      NOP      ;RETURN HERE IF ERROR
4122 012054 104000      ERROR     ;ERROR # DEFINED BY DTASTS SUBROUTINE
4123 012056 004736      JSR      PC,2(SP)+      ;GO BACK FOR MORE ERROR CHECKS
4124 012060 000137 012142      JMP      180$      ;GO TO 180$ IF ERROR WAS FOUND
4125 012064      160$:      JSR      PC,SECERR      ;GO CHECK FOR SECONDARY ERRORS
4126 012064 004737 053016      BR      170$      ;GO TO 170$ IF NO ERROR
4127 012070 000405      NOP      ;RETURN HERE IF ERROR
4128 012072 000240      ERROR     ;ERROR # DEFINED BY SECERR SUBROUTINE
4129 012074 104000      JSR      PC,2(SP)+      ;GO BACK FOR MORE ERROR CHECKS
4130 012076 004736      JMP      180$      ;GO TO 180$ IF ERROR WAS FOUND
4131 012100 000137 012142      170$:      JSR      PC,SECERR      ;GO CHECK FOR SECONDARY ERRORS
4132 012104      BR      170$      ;GO TO 170$ IF NO ERROR
4133 012104 004737 041344      NOP      ;RETURN HERE IF ERROR
4134 012110 000405
4135 012112 000240

```

K07

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19MACY11 30(1046) 23-NOV-77 12:38 PAGE 88
T6 ZERO FILL TEST

SEQ 0088

```

4136 012114 104000          ERROR          ;ERROR # DEFINED BY SECERR SUBROUTINE
4137 012116 004736          JSR          PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
4138 012120 000137 012142    JMP          180$      ;GO TO 180$ IF ERROR WAS FOUND
4139 012124
4140
4141
4142 012124 004737 037064    ;VERIFY DATA
4143 012130 103566          JSR          PC,CMPBUF ;GO COMPARE WRITE, READ DATA BUFFERS
4144 012132 104572          .WORD        BUFO1     ;STARTING ADDRESS OF WRITE BUFFER
4145 012134 000402          .WORD        BUFTWO    ;STARTING ADDRESS OF READ BUFFER
4146 012136 000240          BR          180$      ;GO TO 180$ IF NO ERROR
4147 012140 104000          NOP
4148 012142          ERROR          ;RETURN HERE IF ERROR
4149
4150          180$:          ;ERROR # DEFINED BY CMPBUF SUBROUTINE
4151
4152          ;*****
4153          ;*TEST 7          FORMAT CHECK ZEROS - 16
4154          ;*****
4155          TST7:
4156          SCOPE          ;SCOPE CALL
4157          NOP            ;START OF TEST
4158          MOV          #STACK,SP ;INITIALIZE STACK POINTER
4159          MOV          $BASE,R0   ;R0=UNIBUS ADDRESS
4160          MOV          TSTQUE,R1  ;(R1) = DEVICE BEING TESTED
4161          MOV          #7,$TESTN  ;SET TEST NUMBER IN APT MAIL BOX
4162
4163          10$:
4164          ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
4165          MOV          #0,RMDC0   ;CYLINDER = 0
4166          MOV          #0,RMDA0   ;TRACK = SECTOR = 0
4167          MOV          #FMT16,RMOFO ;16 BIT FORMAT
4168          MOV          #(<1C(2+256)+1>),RMWCO ;2 + 256 WORDS
4169          MOV          #BUFO1,RMBAO ;DATA BUFFER ADDRESS
4170          MOV          #WH,RMCS10 ;WRITE HEADER AND DATA
4171          MOV          #ZEROS,$TMP0 ;USE ALL ZEROS DATA PATTERN
4172          MOV          #1,$TMP1
4173          JSR          PC,GENBUF   ;GO GENERATE DATA BUFFER
4174
4175          20$:
4176          ;PREPARE DEVICE FOR DATA TRANSFER
4177          JSR          PC,TSTPRP   ;PREPARE DEVICE FOR TEST
4178          .WORD        154130 ;TASK DESCRIPTOR
4179          BR          30$          ;GO TO 30$ IF NO ERROR
4180          NOP
4181          ERROR          ;RETURN HERE IF ERROR
4182          JMP          260$        ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4183
4184          30$:
4185          ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
4186          MOV          #SEEK!GO,RMCS10 ;CHANGE COMMAND TO SEEK
4187          MOV          #PUTINX,R2     ;WRITE REGISTER INDEX TABLE
4188          MOVB         #RMDA,(R2)+
4189          MOVB         #RMDC,(R2)+
4190          MOVB         #RMOF,(R2)+
4191          MOVB         #RMCS1,(R2)+
4192          MOVB         #200,(R2)+
4193          JSR          PC,PUT      ;GO WRITE REGISTERS VIA PUT SUB

```

4192	012336	000404		BR	40\$;GO TO 40\$ IF NO ERROR
4193	012340	000240		NOP		;RETURN HERE IF ERROR
4194	012342	104000		ERROR		;ERROR DEFINED BY PUT SUB
4195	012344	000137	012716	JMP	260\$;GO TO 260\$ IF ERROR WAS FOUND
4196	012350				40\$:	
4197						
4198						;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
4199	012350	004737	037432	JSR	PC,GETSTS	;SETUP FOR STATUS
4200	012354	004737	040326	JSR	PC,TIMOUT	;WAIT FOR SEEK TO COMPLETE
4201	012360				50\$:	
4202						
4203						;GO READ SEEK STATUS
4204	012360	004737	037516	JSR	PC,GET	;GO READ REGISTERS VIA GET SUB
4205	012364	000404		BR	60\$;GO TO 60\$ IF NO ERROR
4206	012366	000240		NOP		;RETURN HERE IF ERROR
4207	012370	104000		ERROR		;ERROR DEFINED BY GET SUB
4208	012372	000137	012716	JMP	260\$;GO TO 260\$ IF ERROR WAS FOUND
4209	012376				60\$:	
4210						
4211						;VERIFY THE RESULTS OF THE SEEK COMMAND
4212	012376	004737	045420	JSR	PC,SEKSTS	;GO VERIFY RESULTS OF SEEK OPERATION
4213	012402	000405		BR	70\$;GO TO 70\$ IF NO ERROR
4214	012404	000240		NOP		;RETURN HERE IF ERROR
4215	012406	104000		ERROR		;ERROR # DEFINED BY SEKSTS SUBROUTINE
4216	012410	004736		JSR	PC,2(SP)+	;GO BACK FOR MORE ERROR CHECKS
4217	012412	000137	012716	JMP	260\$;GO TO 260\$ IF ERROR WAS FOUND
4218	012416				70\$:	
4219						
4220						;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
4221	012416	012737	000063	MOV	#WH!GO, RMCS10	;LOAD WRITE HEADER AND DATA
4222	012424	012702	001540	MOV	#PUTINX+3, R2	;EXTEND REGISTER INDEX TABLE
4223	012430	112722	000002	MOVB	#RMHC, (R2)+	
4224	012434	112722	000004	MOVB	#RMBR, (R2)+	
4225	012440	112722	000000	MOVB	#RMCS1, (R2)+	
4226	012444	112722	000200	MOVB	#200, (R2)+	
4227	012450	004737	037766	JSR	PC,PUT	;GO WRITE REGISTERS VIA PUT SUB
4228	012454	000404		BR	80\$;GO TO 80\$ IF NO ERROR
4229	012456	000240		NOP		;RETURN HERE IF ERROR
4230	012460	104000		ERROR		;ERROR DEFINED BY PUT SUB
4231	012462	000137	012716	JMP	260\$;GO TO 260\$ IF ERROR WAS FOUND
4232	012466				80\$:	
4233						
4234						;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
4235	012466	004737	040326	JSR	PC,TIMOUT	;WAIT FOR COMMAND TO COMPLETE
4236	012472	004737	037516	JSR	PC,GET	;GO READ REGISTERS VIA GET SUB
4237	012476	000404		BR	90\$;GO TO 90\$ IF NO ERROR
4238	012500	000240		NOP		;RETURN HERE IF ERROR
4239	012502	104000		ERROR		;ERROR DEFINED BY GET SUB
4240	012504	000137	012716	JMP	260\$;GO TO 260\$ IF ERROR WAS FOUND
4241	012510				90\$:	
4242						
4243						;VERIFY RESULTS OF WRITE COMMAND
4244	012510	004737	040512	JSR	PC,PRIERR	;GO CHECK FOR PRIMARY ERRORS
4245	012514	000405		BR	100\$;GO TO 100\$ IF NO ERROR
4246	012516	000240		NOP		;RETURN HERE IF ERROR
4247	012520	104000		ERROR		;ERROR # DEFINED BY PRIERR SUBROUTINE

```

4248 012522 004736 JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
4249 012524 000137 012716 JMP 260$ ;GO TO 260$ IF ERROR WAS FOUND
4250 012530 100$: JSR PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
4251 012530 004737 053016 BR 110$ ;GO TO 110$ IF NO ERROR
4252 012531 000405 ;RETURN HERE IF ERROR
4253 012536 000240 ;ERROR # DEFINED BY DTASTS SUBROUTINE
4254 012540 104000 ERROR ;GO BACK FOR MORE ERROR CHECKS
4255 012542 004736 JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
4256 012544 000137 012716 JMP 260$ ;GO TO 260$ IF ERROR WAS FOUND
4257 012550 110$: JSR PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
4258 012550 004737 041344 BR 120$ ;GO TO 120$ IF NO ERROR
4259 012554 000405 ;RETURN HERE IF ERROR
4260 012556 000240 ;ERROR # DEFINED BY SECERR SUBROUTINE
4261 012560 104000 ERROR ;GO BACK FOR MORE ERROR CHECKS
4262 012562 004736 JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
4263 012564 000137 012716 JMP 260$ ;GO TO 260$ IF ERROR WAS FOUND
4264 012570 120$:
4265
4266
4267 ;WRITE CHECK HEADER AND DATA FOR SECTOR JUST WRITTEN
4268 012570 012737 000053 001400 MOV #WCH!GO,RMC$10 ;WRITE CHECK COMMAND
4269 012576 004737 037766 JSR PC,PUT ;GO WRITE REGISTERS VIA PUT SUB
4270 012602 000404 BR 130$ ;GO TO 130$ IF NO ERROR
4271 012604 000240 ;RETURN HERE IF ERROR
4272 012606 104000 ERROR ;ERROR DEFINED BY PUT SUB
4273 012610 000137 012716 JMP 260$ ;GO TO 260$ IF ERROR WAS FOUND
4274 012614 130$:
4275
4276 ;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS
4277 012614 004737 040326 JSR PC,TIMOUT ;WAIT FOR READ TO COMPLETE
4278 012620 004737 037516 JSR PC,GET ;GO READ REGISTERS VIA GET SUB
4279 012624 000404 BR 140$ ;GO TO 140$ IF NO ERROR
4280 012626 000240 ;RETURN HERE IF ERROR
4281 012630 104000 ERROR ;ERROR DEFINED BY GET SUB
4282 012632 000137 012716 JMP 260$ ;GO TO 260$ IF ERROR WAS FOUND
4283 012636 140$:
4284
4285 ;VERIFY THE RESULTS OF WRITE CHECK OPERATION
4286 012636 004737 040512 JSR PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
4287 012642 000405 BR 150$ ;GO TO 150$ IF NO ERROR
4288 012644 000240 ;RETURN HERE IF ERROR
4289 012646 104000 ERROR ;ERROR # DEFINED BY PRIERR SUBROUTINE
4290 012650 004736 JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
4291 012652 000137 012716 JMP 260$ ;GO TO 260$ IF ERROR WAS FOUND
4292 012656 150$:
4293 012656 004737 053016 JSR PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
4294 012662 000405 BR 160$ ;GO TO 160$ IF NO ERROR
4295 012664 000240 ;RETURN HERE IF ERROR
4296 012666 104000 ERROR ;ERROR # DEFINED BY DTASTS SUBROUTINE
4297 012670 004736 JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
4298 012672 000137 012716 JMP 260$ ;GO TO 260$ IF ERROR WAS FOUND
4299 012676 160$:
4300 012676 004737 041344 JSR PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
4301 012702 000405 BR 170$ ;GO TO 170$ IF NO ERROR
4302 012704 000240 ;RETURN HERE IF ERROR
4303 012706 104000 ERROR ;ERROR # DEFINED BY SECERR SUBROUTINE

```

```

4304 012710 004736 JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
4305 012712 000137 012716 JMP 260$ ;GO TO 260$ IF ERROR WAS FOUND
4306 012716 170$:
4307
4308 012716 260$:
4309
4310 ;*****
4311 ;TEST 10 FORMAT CHECK ZEROS W/ WCE ERROR
4312 ;*****
4313 012716 TST10:
4314 012716 000004 SCOPE ;SCOPE CALL
4315 012720 000240 NOP ;START OF TEST
4316 012722 012706 001100 MOV #STACK,SP ;INITIALIZE STACK POINTER
4317 012726 013700 001276 MOV $BASE,R0 ;R0=UNIBUS ADDRESS
4318 012732 013701 001450 MOV TSTQUE,R1 ;(R1) = DEVICE BLING TESTED
4319 012736 012737 000010 001226 MOV #10,$TESTN ;SET TEST NUMBER IN APT MAIL BOX
4320 012744 10$:
4321
4322 ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
4323 012744 012737 000000 001434 MOV #0,RMDC0 ;CYLINDER = 0
4324 012752 012737 000000 001406 MOV #0,RMDA0 ;TRACK = SECTOR = 0
4325 012760 012737 010000 001432 MOV #FMT16,RMOFO ;16 BIT FORMAT
4326 012766 012737 177376 001402 MOV #(<1C(2+256)+1),RMWCC ;2 + 256 WORDS
4327 012774 012737 103566 001404 MOV #BUFONE,R3A0 ;DATA BUFFER ADDRESS
4328 013002 012737 000062 001400 MOV #WH,RMCS10 ;WRITE HEADER AND DATA
4329 013010 012737 065106 001174 MOV #ZEROS,$TMP0 ;USE ALL ONES DATA PATTERN
4330 013016 012737 000001 001176 MOV #1,$TMP1
4331 013024 004737 036620 JSR PC,GENBUF ;GO GENERATE DATA BUFFER
4332 013030 20$:
4333
4334 ;PREPARE DEVICE FOR DATA TRANSFER
4335 013030 004737 034000 JSR PC,TSTPRP ;PREPARE DEVICE FOR TEST
4336 013034 154130 .WORD 154130 ;TASK DESCRIPTOR
4337 013036 000404 BR 30$ ;GO TO 30$ IF NO ERROR
4338 013040 000240 NOP ;RETURN HERE IF ERROR
4339 013042 104000 ERROR ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4340 013044 000137 013636 JMP 260$ ;GO TO 260$ IF ERROR WAS FOUND
4341 013050 30$:
4342
4343 ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
4344 013050 012737 000005 001400 MOV #SEEK!GO,RMCS10 ;CHANGE COMMAND TO SEEK
4345 013056 012702 001535 MOV #PUTINX,R2 ;WRITE REGISTER INDEX TABLE
4346 013062 112722 000006 MOVB #RMDA,(R2)+
4347 013066 112722 000034 MOVB #RMDC,(R2)+
4348 013072 112722 000032 MOVB #RMOF,(R2)+
4349 013076 112722 000000 MOVB #RMCS1,(R2)+
4350 013102 112722 000200 MOVB #200,(R2)+
4351 013106 004737 037766 JSR PC,PUT ;GO WRITE REGISTERS VIA PUT SUB
4352 013112 000404 BR 40$ ;GO TO 40$ IF NO ERROR
4353 013114 000240 NOP ;RETURN HERE IF ERROR
4354 013116 104000 ERROR ;ERROR DEFINED BY PUT SUB
4355 013120 000137 013636 JMP 260$ ;GO TO 260$ IF ERROR WAS FOUND
4356 013124 40$:
4357
4358 ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
4359 013124 004737 037432 JSR PC,GETSTS ;SETUP FOR STATUS

```

SEQ 0092

4360	013130	004737	040326
4361	013134		
4362			
4363			
4364	013134	004737	037516
4365	013140	000404	
4366	013142	000240	
4367	013144	104000	
4368	013146	000137	013636
4369	013152		
4370			
4371			
4372	013152	004737	045420
4373	013156	000405	
4374	013160	000240	
4375	013162	104000	
4376	013164	004736	
4377	013166	000137	013636
4378	013172		
4379			
4380			
4381	013172	012737	000063

```

JSR      PC,TIMOUT      ;WAIT FOR SEEK TO COMPLETE
50$:
;GO READ SEEK STATUS
JSR      PC,GET          ;GO READ REGISTERS VIA GET SUB
BR       60$             ;GO TO 60$ IF NO ERROR
NOP      ;RETURN HERE IF ERROR
ERROR    ;ERROR DEFINED BY GET SUB
JMP      260$            ;GO TO 260$ IF ERROR WAS FOUND
60$:
;VERIFY THE RESULTS OF THE SEEK COMMAND
JSR      PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
BR       70$             ;GO TO 70$ IF NO ERROR
NOP      ;RETURN HERE IF ERROR
ERROR    ;ERROR * DEFINED BY SEKSTS SUBROUTINE
JSR      PC,2(SP)+        ;GO BACK FOR MORE ERROR CHECKS
JMP      260$            ;GO TO 260$ IF ERROR WAS FOUND
70$:
;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
MOV      #WH!GO,RMCS10   ;LOAD WRITE HEADER AND DATA

```

4382	013200	012702	001540	MOV	#PUTINX+3,R2	;EXTEND REGISTER INDEX TABLE
4383	013204	112722	000002	MOVB	#RMC,(R2)+	
4384	013210	112722	000004	MOVB	#RMC2,(R2)+	
4385	013214	112722	000000	MOVB	#RMC51,(R2)+	
4386	013220	112722	0002C0	MOVB	#200,(R2)+	
4387	013224	004737	037766	JSR	PC PUT	;GO WRITE REGISTERS VIA PUT SUB
4388	013230	000404		BR	80\$;GO TO 80\$ IF NO ERROR
4389	013232	000240		NOP		;RETURN HERE IF ERROR
4390	013234	104000		ERROR		;ERROR DEFINED BY PUT SUB
4391	013236	000137	013636	JMP	260\$;GO TO 260\$ IF ERROR WAS FOUND
4392	013242					
4393						
4394						
4395	013242	004737	040326	JSR	PC,TIMOUT	;WAIT FOR COMMAND TO COMPLETE
4396	013246	004737	037516	JSR	PC,GET	;GO READ REGISTERS VIA GET SUB
4397	013252	000404		BR	90\$;GO TO 90\$ IF NO ERROR
4398	013254	000240		NOP		;RETURN HERE IF ERROR
4399	013256	104000		ERROR		;ERROR DEFINED BY GET SUB
4400	013260	000137	013636	JMP	260\$;GO TO 260\$ IF ERROR WAS FOUND
4401	013264					
4402						
4403						
4404	013264	004737	040512	JSR	PC,PRIERR	;GO CHECK FOR PRIMARY ERRORS
4405	013270	000405		BR	100\$;GO TO 100\$ IF NO ERROR
4406	013272	000240		NOP		;RETURN HERE IF ERROR
4407	013274	104000		ERROR		;ERROR # DEFINED BY PRIERR SUBROUTINE
4408	013276	004736		JSR	PC,2(SP)+	;GO BACK FOR MORE ERROR CHECKS
4409	013300	000137	013636	JMP	260\$;GO TO 260\$ IF ERROR WAS FOUND
4410	013304					
4411	013304	004737	053016	JSR	PC,DTASTS	;GO VERIFY RESULTS OF DATA TRANSFER
4412	013310	000405		BR	110\$;GO TO 110\$ IF NO ERROR
4413	013312	000240		NOP		;RETURN HERE IF ERROR
4414	013314	104000		ERROR		;ERROR # DEFINED BY DTASTS SUBROUTINE
4415	013316	004736		JSR	PC,2(SP)+	;GO BACK FOR MORE ERROR CHECKS
4416	013320	000137	013636	JMP	260\$;GO TO 260\$ IF ERROR WAS FOUND
4417	013324					
4418	013324	004737	041344	JSR	PC,SECERR	;GO CHECK FOR SECONDARY ERRORS
4419	013330	000405		BR	120\$;GO TO 120\$ IF NO ERROR
4420	013332	000240		NOP		;RETURN HERE IF ERROR
4421	013334	104000		ERROR		;ERROR # DEFINED BY SECERR SUBROUTINE
4422	013336	004736		JSR	PC,2(SP)+	;GO BACK FOR MORE ERROR CHECKS
4423	013340	000137	013636	JMP	260\$;GO TO 260\$ IF ERROR WAS FOUND
4424	013344					
4425						
4426						
4427	013344	005137	104570	COM	BUFTWO-2	;COMPLEMENT LAST DATA WORD
4428						
4429						
4430						
4431	013350	012737	000053	MOV	#WCH!GO,RMC510	;LOAD COMMAND
4432	013356	004737	037766	JSR	PC,PUT	;GO WRITE REGISTERS VIA PUT SUB
4433	013362	000404		BR	180\$;GO TO 180\$ IF NO ERROR
4434	013364	000240		NOP		;RETURN HERE IF ERROR
4435	013366	104000		ERROR		;ERROR DEFINED BY PUT SUB
4436	013370	000137	013636	JMP	260\$;GO TO 260\$ IF ERROR WAS FOUND
4437	013374					

180\$:

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 94
T10 FORMAT CHECK ZEROS W/ WCE ERROR

SEQ 0094

```

4438
4439
4440 013374 004737 040326
4441 013400 004737 037516
4442 013404 000404
4443 013406 000240
4444 013410 104000
4445 013412 000137 013636
4446 013416
4447
4448
4449 013416 004737 040512
4450 013422 000405
4451 013424 000240
4452 013426 104000
4453 013430 004736
4454 013432 000137 013636
4455 013436
4456

;WAIT FOR WRITE CHECK COMMAND TO COMPLETE AND READ STATUS
JSR PC, TIMEOUT
JSR PC, GET
BR 190$ ;GO READ REGISTERS VIA GET SUB
NOP ;GO TO 190$ IF NO ERROR
ERROR ;RETURN HERE IF ERROR
JMP 260$ ;ERROR DEFINED BY GET SUB
;GO TO 260$ IF ERROR WAS FOUND
190$:

;CHECK FOR PRIMARY ERRORS
JSR PC, PRIERR
BR 200$ ;GO CHECK J2 PRIMARY ERRORS
NOP ;GO TO 200$ IF NO ERROR
ERROR ;RETURN HERE IF ERROR
JSR PC, J(SP)+
JMP 260$ ;ERROR # DEFINED BY PRIERR SUBROUTINE
;GO BACK FOR MORE ERROR CHECKS
;GO TO 260$ IF ERROR WAS FOUND
200$:

```



```

4457 ;MAKE SURE THE WRITE CHECK ERROR WAS DETECTED
4458 013436 032737 040000 001340 BIT #WCE,RMCS2I ;IS WRITE CHECK ERROR SET??
4459 013444 001023 BNE 210$ ;YES!!
4460 013446 004737 053016 JSR PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
4461 013452 000405 BR 205$ ;GO TO 205$ IF NO ERROR
4462 013454 000240 NOP ;RETURN HERE IF ERROR
4463 013456 104000 ERROR ;ERROR # DEFINED BY DTASTS SUBROUTINE
4464 013460 004736 JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
4465 013462 000137 013636 JMP 260$ ;GO TO 260$ IF ERROR WAS FOUND
4466 013466 013737 001340 001140 205$: MOV RMCS2I,$GDDAT ;LOAD EXPECTED STATUS
4467 013474 052737 040000 001140 BIS #WCE,$GDDAT ;LOAD RECEIVED STATUS
4468 013502 013737 001340 001142 MOV RMCS2I,$BDDAT ;WRITE CHECK ERROR NOT DETECTED
4469 013510 104337 ERROR 337
4470 013512 000451 BR 260$
4471 013514
4472 210$:
4473 ;VERIFY THE ADDRESS OF THE WRITE CHECK ERROR
4474 013514 012737 104570 001134 MOV #BUFTWO-2,$GDADR ;LOAD EXPECTED ADDRESS
4475 013522 013737 001334 001136 MOV RMBAI,$BDAOR ;LOAD RECEIVED ADDRESS
4476 013530 162737 000002 001136 SUB #2,$BDAOR ;DECREMENT RECEIVED ADDRESS
4477
4478 ;GET WCE DATA AND VERIFY IT IS OK
4479 013536 112737 000022 001506 MOVB #RMOB,GETINX ;SETUP FOR READING RMOB
4480 013544 112737 000200 001507 MOVB #200,GETINX+1
4481 013552 004737 037516 JSR PC,GET ;GO READ REGISTERS VIA GET SUB
4482 013556 000404 BR 230$ ;GO TO 230$ IF NO ERROR
4483 013560 000240 NOP ;RETURN HERE IF ERROR
4484 013562 104000 ERROR ;ERROR DEFINED BY GET SUB
4485 013564 000137 013636 JMP 260$ ;GO TO 260$ IF ERROR WAS FOUND
4486 013570 013737 001352 001142 230$: MOV RMOBI,$BDDAT ;LOAD RECEIVED DATA WORD
4487 013576 013737 104570 001140 MOV BUFTWO-2,$GDDAT ;LOAD EXPECTED DATA WORD
4488 013604 005137 001140 COM $GDDAT
4489 013610 023737 001134 001136 CMP $GDADR,$BDAOR ;IS ADDRESS OK??
4490 013616 001402 BEQ 220$ ;YES!!
4491 013620 104340 ERROR 340 ;ADDRESS OF WCE INCORRECT
4492 013622 000405 BR 260$
4493 013624 023737 001140 001142 220$: CMP $GDDAT,$BDDAT ;IS DATA WORD OK??
4494 013632 001401 BEQ 260$ ;YES!!
4495 013634 104341 ERROR 341 ;UNEXPECTED WCE DATA
4496 013636
4497 260$:
4498
4499 ;*****
4500 ;*TEST 11 FORMAT ONES - 16
4501 ;*****
4502 TST11:
4503 013636 000004 SCOPE ;SCOPE CALL
4504 013640 000240 NOP ;START OF TEST
4505 013642 012706 001100 MOV #STACK,SP ;INITIALIZE STACK POINTER
4506 013646 013700 001276 MOV $BASE,R0 ;R0=UNIBUS ADDRESS
4507 013652 013701 001450 MOV TSTQUE,R1 ;R1 = DEVICE BEING TESTED
4508 013656 012737 000011 001226 MOV #11,$TESTN ;SET TEST NUMBER IN APT MAIL BOX
4509 10$:
4510 ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
4511 013664 012737 000000 001434 MOV #0,RMDCO ;CYLINDER = 0
4512 013672 012737 000000 001406 MOV #0,RMDAO ;TRACK = SECTOR = 0

```

```

4513 013700 012737 010000 001432      MOV      #FMT16,RMOFO      ;16 BIT FORMAT
4514 013706 012737 177376 001402      MOV      #(<C(2+256.)+1>),RMWCO    ;2 + 256 WORDS
4515 013714 012737 103566 001404      MOV      #BUFONE,RMBAO      ;DATA BUFFER ADDRESS
4516 013722 012737 000062 001400      MOV      #WH,RMCS10      ;WRITE HEADER AND DATA
4517 013730 012737 065044 001174      MOV      #ONES,$TMPD      ;USE ALL ONES DATA PATTERN
4518 013736 012737 000001 001176      MOV      #1,$TMP1
4519 013744 004737 036620      JSR      PC,GENBUF      ;GO GENERATE DATA BUFFER
4520 013750
4521
4522      ;PREPARE DEVICE FOR DATA TRANSFER
4523 013750 004737 034000      JSR      PC,TSTPRP      ;PREPARE DEVICE FOR TEST
4524 013754 154130      .WORD      154130      ;TASK DESCRIPTOR
4525 013756 000404      BR      30$      ;GO TO 30$ IF NO ERROR
4526 013760 000240      NOP      ;RETURN HERE IF ERROR
4527 013762 104000      ERROR      ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4528 013764 000137 014436      JMP      180$      ;GO TO 180$ IF ERROR WAS FOUND
4529 013770
4530
4531      ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
4532 013770 012737 000005 001400      MOV      #SEEK!GO,RMCS10      ;CHANGE COMMAND TO SEEK
4533 013776 012702 001535      MOV      #PUTINX,R2      ;WRITE REGISTER INDEX TABLE
4534 014002 112722 000006      MOV      #RMDA,(R2)+
4535 014006 112722 000034      MOV      #RMDC,(R2)+
4536 014012 112722 000032      MOV      #RMOF,(R2)+
4537 014016 112722 000000      MOV      #RMCS1,(R2)+
4538 014022 112722 000200      MOV      #200,(R2)+
4539 014026 004737 037766      JSR      PC,PUT      ;GO WRITE REGISTERS VIA PUT SUB
4540 014032 000404      BR      40$      ;GO TO 40$ IF NO ERROR
4541 014034 000240      NOP      ;RETURN HERE IF ERROR
4542 014036 104000      ERROR      ;ERROR DEFINED BY PUT SUB
4543 014040 000137 014436      JMP      180$      ;GO TO 180$ IF ERROR WAS FOUND
4544 014044
4545
4546      ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
4547 014044 004737 037432      JSR      PC,GETSTS      ;SETUP FOR STATUS
4548 014050 004737 040326      JSR      PC,TIMOUT      ;WAIT FOR SEEK TO COMPLETE
4549 014054
4550
4551      ;GO READ SEEK STATUS
4552 014054 004737 037516      JSR      PC,GET      ;GO READ REGISTERS VIA GET SUB
4553 014060 000404      BR      60$      ;GO TO 60$ IF NO ERROR
4554 014062 000240      NOP      ;RETURN HERE IF ERROR
4555 014064 104000      ERROR      ;ERROR DEFINED BY GET SUB
4556 014066 000137 014436      JMP      180$      ;GO TO 180$ IF ERROR WAS FOUND
4557 014072
4558
4559      ;VERIFY THE RESULTS OF THE SEEK COMMAND
4560 014072 004737 045420      JSR      PC,SEKSTS      ;GO VERIFY RESULTS OF SEEK OPERATION
4561 014076 000405      BR      70$      ;GO TO 70$ IF NO ERROR
4562 014100 000240      NOP      ;RETURN HERE IF ERROR
4563 014102 104000      ERROR      ;ERROR # DEFINED BY SEKSTS SUBROUTINE
4564 014104 004736      JSR      PC,2(SP)+      ;GO BACK FOR MORE ERROR CHECKS
4565 014106 000137 014436      JMP      180$      ;GO TO 180$ IF ERROR WAS FOUND
4566 014112
4567
4568      ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND

```

```

4569 014112 012737 000063 001400 MOV      #WH!GO,RMCS10      ;LOAD WRITE HEADER AND DATA
4570 014120 012702 001540      MOV      #PUTINX+3,R2      ;EXTEND REGISTER INDEX TABLE
4571 014124 112722 000002      MOV      #RMWC,(R2)+
4572 014130 112722 000004      MOV      #RMBB,(R2)+
4573 014134 112722 000000      MOV      #RMCS1,(R2)+
4574 014140 112722 000200      MOV      #200,(R2)+
4575 014144 004737 037766      JSR      PC,PUT      ;GO WRITE REGISTERS VIA PUT SUB
4576 014150 000404      BR      80$      ;GO TO 80$ IF NO ERROR
4577 014152 000240      NOP      ;RETURN HERE IF ERROR
4578 014154 104000      ERROR   ;ERROR DEFINED BY PUT SUB
4579 014156 000137 014436      JMP      180$      ;GO TO 180$ IF ERROR WAS FOUND
4580 014162
4581
4582      ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
4583 014162 004737 040326      JSR      PC,TIMOUT      ;WAIT FOR COMMAND TO COMPLETE
4584 014166 004737 037516      JSR      PC,GET      ;GO READ REGISTERS VIA GET SUB
4585 014172 000404      BR      90$      ;GO TO 90$ IF NO ERROR
4586 014174 000240      NOP      ;RETURN HERE IF ERROR
4587 014176 104000      ERROR   ;ERROR DEFINED BY GET SUB
4588 014200 000137 014436      JMP      180$      ;GO TO 180$ IF ERROR WAS FOUND
4589 014204
4590
4591      ;VERIFY RESULTS OF WRITE COMMAND
4592 014204 004737 040512      JSR      PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
4593 014210 000405      BR      100$      ;GO TO 100$ IF NO ERROR
4594 014212 000240      NOP      ;RETURN HERE IF ERROR
4595 014214 104000      ERROR   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4596 014216 004736      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
4597 014220 000137 014436      JMP      180$      ;GO TO 180$ IF ERROR WAS FOUND
4598 014224
4599 014224 004737 053016      JSR      PC,DASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
4600 014230 000405      BR      110$      ;GO TO 110$ IF NO ERROR
4601 014232 000240      NOP      ;RETURN HERE IF ERROR
4602 014234 104000      ERROR   ;ERROR # DEFINED BY DASTS SUBROUTINE
4603 014236 004736      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
4604 014240 000137 014436      JMP      180$      ;GO TO 180$ IF ERROR WAS FOUND
4605 014244
4606 014244 004737 041344      JSR      PC,SECERR      ;GO CHECK FOR SECONDARY ERRORS
4607 014250 000405      BR      120$      ;GO TO 120$ IF NO ERROR
4608 014252 000240      NOP      ;RETURN HERE IF ERROR
4609 014254 104000      ERROR   ;ERROR # DEFINED BY SECERR SUBROUTINE
4610 014256 004736      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
4611 014260 000137 014436      JMP      180$      ;GO TO 180$ IF ERROR WAS FOUND
4612 014264
4613
4614
4615      ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
4616 014264 012737 000073 001400 MOV      #RH!GO,RMCS10      ;READ HEADER & DATA COMMAND
4617 014272 012737 104572 001404 MOV      #BUFTWO,RMBAO      ;CHANGE BUS ADDRESS
4618 014300 004737 037766      JSR      PC,PUT      ;GO WRITE REGISTERS VIA PUT SUB
4619 014304 000404      BR      130$      ;GO TO 130$ IF NO ERROR
4620 014306 000240      NOP      ;RETURN HERE IF ERROR
4621 014310 104000      ERROR   ;ERROR DEFINED BY PUT SUB
4622 014312 000137 014436      JMP      180$      ;GO TO 180$ IF ERROR WAS FOUND
4623 014316
4624

```

```

4625 ;WAIT FOR READ TO COMPLETE AND GET STATUS
4626 JSR PC,TIMOUT ;WAIT FOR READ TO COMPLETE
4627 JSR PC,GET ;GO READ REGISTERS VIA GET SUB
4628 BR 140$ ;GO TO 140$ IF NO ERROR
4629 NOP ;RETURN HERE IF ERROR
4630 ERROR ;ERROR DEFINED BY GET SUB
4631 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
4632 140$:
4633 ;VERIFY THE RESULTS OF READ OPERATION
4634 JSR PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
4635 BR 150$ ;GO TO 150$ IF NO ERROR
4636 NOP ;RETURN HERE IF ERROR
4637 ERROR ;ERROR # DEFINED BY PRIERR SUBROUTINE
4638 JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
4639 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
4640 150$:
4641 JSR PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
4642 BR 160$ ;GO TO 160$ IF NO ERROR
4643 NOP ;RETURN HERE IF ERROR
4644 ERROR ;ERROR # DEFINED BY DTASTS SUBROUTINE
4645 JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
4646 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
4647 160$:
4648 JSR PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
4649 BR 170$ ;GO TO 170$ IF NO ERROR
4650 NOP ;RETURN HERE IF ERROR
4651 ERROR ;ERROR # DEFINED BY SECERR SUBROUTINE
4652 JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
4653 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
4654 170$:
4655 ;VERIFY DATA
4656 JSR PC,CMPBUF ;GO COMPARE WRITE, READ DATA BUFFERS
4657 .WORD BUFOne ;STARTING ADDRESS OF WRITE BUFFER
4658 .WORD BUFTwo ;STARTING ADDRESS OF READ BUFFER
4659 BR 180$ ;GO TO 180$ IF NO ERROR
4660 NOP ;RETURN HERE IF ERROR
4661 ERROR ;ERROR # DEFINED BY CMPBUF SUBROUTINE
4662 180$:
4663 ;*****
4664 ;*TEST 12 FORMAT CHECK ONES - 16
4665 ;*****
4666 TST12:
4667 SCOPE ;SCOPE CALL
4668 NOP ;START OF TEST
4669 MOV #STACK,SP ;INITIALIZE STACK POINTER
4670 MOV $BASE,R0 ;R0=UNIBUS ADDRESS
4671 MOV TSTQUE,R1 ;R1 = DEVICE BEING TESTED
4672 MOV #12,$TESTN ;SET TEST NUMBER IN APT MAIL BOX
4673 10$:
4674 ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
4675 MOV #0,RMDCO ;CYLINDER = 0
4676 MOV #0,RMDAO ;TRACK = SECTOR = 0
4677
4678 014316 004737 040326
4679 014322 004737 037516
4680 014326 000404
4681 014330 000240
4682 014332 104000
4683 014334 000137 014436
4684 014340
4685 014340 004737 040512
4686 014344 000405
4687 014346 000240
4688 014350 104000
4689 014352 004736
4690 014354 000137 014436
4691 014360
4692 014360 004737 053016
4693 014364 000405
4694 014366 000240
4695 014370 104000
4696 014372 004736
4697 014374 000137 014436
4698 014400
4699 014400 004737 041344
4700 014404 000405
4701 014406 000240
4702 014410 104000
4703 014412 004736
4704 014414 000137 014436
4705 014420
4706 014420 004737 037064
4707 014424 103566
4708 014426 104572
4709 014430 000402
4710 014432 000240
4711 014434 104000
4712 014436
4713 014436 000004
4714 014440 000240
4715 014442 012706 001100
4716 014444 013700 001276
4717 014446 013701 001450
4718 014452 012737 000012 001226
4719 014456
4720 014464
4721 014464 012737 000000 001434
4722 014472 012737 000000 001406

```

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 99
T12 FORMAT CHECK ONES - 16

SEQ 0099

```

4681 014500 012737 010000 001432      MOV      #FMT16,RMOFO      ;16 BIT FORMAT
4682 014506 012737 177376 001402      MOV      #(<C<2+256.>+1),RMWCO    ;2 + 256 WORDS
4683 014514 012737 103566 001404      MOV      #BUFONE,RMBAO      ;DATA BUFFER ADDRESS
4684 014522 012737 000062 001400      MOV      #WH,RMCS10      ;WRITE HEADER AND DATA
4685 014530 012737 065044 001174      MOV      #ONES,STMPD      ;USE ALL ONES DATA PATTERN
4686 014536 012737 000001 001176      MOV      #1,STMP1
4687 014544 004737 036620      MOV      PC,GENBUF      ;GO GENERATE DATA BUFFER
4688 014550
4689
4690      20$:
4691      ;PREPARE DEVICE FOR DATA TRANSFER
4692      JSR      PC,TSTPRP      ;PREPARE DEVICE FOR TEST
4693      .WORD    154130      ;TASK DESCRIPTOR
4694      BR       30$
4695      NOP
4696      ERROR
4697      JMP      260$      ;RETURN HERE IF ERROR
4698      ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4699      ;GO TO 260$ IF ERROR WAS FOUND
4700      30$:
4701      ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
4702      MOV      #SEEK!GO,RMCS10      ;CHANGE COMMAND TO SEEK
4703      MOV      #PUTINX,R2      ;WRITE REGISTER INDEX TABLE
4704      MOVB     #RMOA,(R2)+
4705      MOVB     #RMDC,(R2)+
4706      MOVB     #RMOF,(R2)+
4707      MOVB     #RMCS1,(R2)+
4708      MOVB     #200,(R2)+
4709      JSR      PC,PUT      ;GO WRITE REGISTERS VIA PUT SUB
4710      BR       40$      ;GO TO 40$ IF NO ERROR
4711      NOP
4712      ERROR
4713      JMP      260$      ;RETURN HERE IF ERROR
4714      ;ERROR DEFINED BY PUT SUB
4715      ;GO TO 260$ IF ERROR WAS FOUND
4716      40$:
4717      ;SETUP FOR READING STATUS AND THEN WAIT FOR SEFK TO COMPLETE
4718      JSR      PC,GETSTS      ;SETUP FOR STATUS
4719      JSR      PC,TIMOUT      ;WAIT FOR SEEK TO COMPLETE
4720      50$:
4721      ;GO READ SEEK STATUS
4722      JSR      PC,GET      ;GO READ REGISTERS VIA GET SUB
4723      BR       60$      ;GO TO 60$ IF NO ERROR
4724      NOP
4725      ERROR
4726      JMP      260$      ;RETURN HERE IF ERROR
4727      ;ERROR DEFINED BY GET SUB
4728      ;GO TO 260$ IF ERROR WAS FOUND
4729      60$:
4730      ;VERIFY THE RESULTS OF THE SEEK COMMAND
4731      JSR      PC,SEKSTS      ;GO VERIFY RESULTS OF SEEK OPERATION
4732      BR       70$      ;GO TO 70$ IF NO ERROR
4733      NOP
4734      ERROR
4735      JMP      PC,2(SP)+      ;RETURN HERE IF ERROR
4736      ;ERROR # DEFINED BY SEKSTS SUBROUTINE
4737      ;GO BACK FOR MORE ERROR CHECKS
4738      ;GO TO 260$ IF ERROR WAS FOUND
4739      70$:
4740      ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND

```

```

4737 014712 012737 000063 001400 MOV      #WH!GO,RMCS10      ;LOAD WRITE HEADER AND DATA
4738 014720 012702 001540      MOV      #PUTINX+3,R2      ;EXTEND REGISTER INDEX TABLE
4739 014724 112722 000002      MOV      #RMWC,(R2)+
4740 014730 112722 000004      MOV      #RMBB,(R2)+
4741 014734 112722 000000      MOV      #RMCS1,(R2)+
4742 014740 112722 000200      MOV      #200,(R2)+
4743 014744 004737 037766      JSR      PC,PUT      ;GO WRITE REGISTERS VIA PUT SUB
4744 014750 000404      BR      80$      ;GO TO 80$ IF NO ERROR
4745 014752 000240      NOP      ;RETURN HERE IF ERROR
4746 014754 104000      ERROR    ;ERROR DEFINED BY PUT SUB
4747 014756 000137 015212      JMP      260$      ;GO TO 260$ IF ERROR WAS FOUND
4748 014762
4749
4750      ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
4751 014762 004737 040326      JSR      PC,TIMOUT      ;WAIT FOR COMMAND TO COMPLETE
4752 014766 004737 037516      JSR      PC,GET      ;GO READ REGISTERS VIA GET SUB
4753 014772 000404      BR      90$      ;GO TO 90$ IF NO ERROR
4754 014774 000240      NOP      ;RETURN HERE IF ERROR
4755 014776 104000      ERROR    ;ERROR DEFINED BY GET SUB
4756 015000 000137 015212      JMP      260$      ;GO TO 260$ IF ERROR WAS FOUND
4757 015004
4758
4759      ;VERIFY RESULTS OF WRITE COMMAND
4760 015004 004737 040512      JSR      PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
4761 015010 000405      BR      100$      ;GO TO 100$ IF NO ERROR
4762 015012 000240      NOP      ;RETURN HERE IF ERROR
4763 015014 104000      ERROR    ;ERROR # DEFINED BY PRIERR SUBROUTINE
4764 015016 004736      JSR      PC,2(SP)+      ;GO BACK FOR MORE ERROR CHECKS
4765 015020 000137 015212      JMP      260$      ;GO TO 260$ IF ERROR WAS FOUND
4766 015024
4767 015024 004737 053016      JSR      PC,DASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
4768 015030 000405      BR      110$      ;GO TO 110$ IF NO ERROR
4769 015032 000240      NOP      ;RETURN HERE IF ERROR
4770 015034 104000      ERROR    ;ERROR # DEFINED BY DASTS SUBROUTINE
4771 015036 004736      JSR      PC,2(SP)+      ;GO BACK FOR MORE ERROR CHECKS
4772 015040 000137 015212      JMP      260$      ;GO TO 260$ IF ERROR WAS FOUND
4773 015044
4774 015044 004737 041344      JSR      PC,SECERR      ;GO CHECK FOR SECONDARY ERRORS
4775 015050 000405      BR      120$      ;GO TO 120$ IF NO ERROR
4776 015052 000240      NOP      ;RETURN HERE IF ERROR
4777 015054 104000      ERROR    ;ERROR # DEFINED BY SECERR SUBROUTINE
4778 015056 004736      JSR      PC,2(SP)+      ;GO BACK FOR MORE ERROR CHECKS
4779 015060 000137 015212      JMP      260$      ;GO TO 260$ IF ERROR WAS FOUND
4780 015064
4781
4782
4783      ;WRITE CHECK HEADER AND DATA FOR SECTOR JUST WRITTEN
4784 015064 012737 000053 001400 MOV      #WCH!GO,RMCS10      ;WRITE CHECK COMMAND
4785 015072 004737 037766      JSR      PC,PUT      ;GO WRITE REGISTERS VIA PUT SUB
4786 015076 000404      BR      130$      ;GO TO 130$ IF NO ERROR
4787 015100 000240      NOP      ;RETURN HERE IF ERROR
4788 015102 104000      ERROR    ;ERROR DEFINED BY PUT SUB
4789 015104 000137 015212      JMP      260$      ;GO TO 260$ IF ERROR WAS FOUND
4790 015110
4791
4792      ;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS

```

```

4793 015110 004737 040326      JSR    PC,TIMOUT      ;WAIT FOR READ TO COMPLETE
4794 015114 004737 037516      JSR    PC,GET        ;GO READ REGISTERS VIA GET SUB
4795 015120 000404              BR     140$          ;GO TO 140$ IF NO ERROR
4796 015122 000240              NOP                    ;RETURN HERE IF ERROR
4797 015124 104000              ERROR                   ;ERROR DEFINED BY GET SUB
4798 015126 000137 015212      JMP     260$          ;GO TO 260$ IF ERROR WAS FOUND
4799 015132
4800
4801
4802 015132 004737 040512      ;VERIFY THE RESULTS OF WRITE CHECK OPERATION
4803 015136 000405              JSR    PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
4804 015140 000240              BR     150$          ;GO TO 150$ IF NO ERROR
4805 015142 104000              NOP                    ;RETURN HERE IF ERROR
4806 015144 004736              ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4807 015146 000137 015212      JSR    PC,(SP)+      ;GO BACK FOR MORE ERROR CHECKS
4808 015152 000137 015212      JMP     260$          ;GO TO 260$ IF ERROR WAS FOUND
4809 015152 004737 053016      150$:      JSR    PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
4810 015156 000405              BR     160$          ;GO TO 160$ IF NO ERROR
4811 015160 000240              NOP                    ;RETURN HERE IF ERROR
4812 015162 104000              ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4813 015164 004736              JSR    PC,(SP)+      ;GO BACK FOR MORE ERROR CHECKS
4814 015166 000137 015212      JMP     260$          ;GO TO 260$ IF ERROR WAS FOUND
4815 015172
4816 015172 004737 041344      160$:      JSR    PC,SECERR      ;GO CHECK FOR SECONDARY ERRORS
4817 015176 000405              BR     170$          ;GO TO 170$ IF NO ERROR
4818 015200 000240              NOP                    ;RETURN HERE IF ERROR
4819 015202 104000              ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4820 015204 004736              JSR    PC,(SP)+      ;GO BACK FOR MORE ERROR CHECKS
4821 015206 000137 015212      JMP     260$          ;GO TO 260$ IF ERROR WAS FOUND
4822 015212
4823
4824 015212
4825
4826
4827
4828
4829 015212
4830 015212 000004              ;*****
4831 015214 000240              ;*TEST 13      FORMAT CHECK ONES W/ WCE ERRORS
4832 015216 012706 001100      ;*****
4833 015222 013700 001276      ;ST13:      SCOPE      ;SCOPE CALL
4834 015226 013701 001450      ;          NOP      ;START OF TEST
4835 015232 012737 000013 001226 ;          MOV    #STACK,SP ;INITIALIZE STACK POINTER
4836 015240
4837
4838
4839 015240 012737 000000 001434 ;          MOV    $BASE,RO ;RO=UNIBUS ADDRESS
4840 015246 012737 000000 001406 ;          MOV    TSTQUE,R1 ; (R1) = DEVICE BEING TESTED
4841 015254 012737 010000 001432 ;          MOV    #13,$TESTN ;SET TEST NUMBER IN APT MAIL BOX
4842 015262 012737 177376 001402 ;          ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
4843 015270 012737 103566 001404 ;          MOV    #0,RMDCO ;CYLINDER = 0
4844 015276 012737 000062 001400 ;          MOV    #0,RMDAO ;TRACK = SECTOR = 0
4845 015304 012737 065044 001174 ;          MOV    #FMT16,RMOFO ;16 BIT FORMAT
4846 015312 012737 000001 001176 ;          MOV    #<(2+256)+1>,RMWCO ;2 + 256 WORDS
4847 015320 004737 036620      ;          MOV    #BUFONE,RMBAO ;DATA BUFFER ADDRESS
4848 015324

```

```
4849
4850
4851 015324 004737 034000
4852 015330 154130
4853 015332 000404
4854 015334 000240
4855 015336 104000
4856 015340 000137 016130
4857 015344
4858
4859
4860 015344 012737 000005 001400
4861 015352 012702 001535
4862 015356 112722 000006
4863 015362 112722 000034
4864 015366 112722 000032
4865 015372 112722 000000
4866 015376 112722 000200
4867 015402 004737 037766
4868 015406 000404
4869 015410 000240
4870 015412 104000
4871 015414 000137 016130
4872 015420
4873
4874
4875 015420 004737 037432
4876 015424 004737 040326
4877 015430
4878
4879
4880 015430 004737 037516
4881 015434 000404
4882 015436 000240
4883 015440 104000
4884 015442 000137 016130
4885 015446
4886
4887 015446 004737 045420
4888 015452 000405
4889 015454 000240
4890 015456 104000
4891 015460 004736
4892 015462 000137 016130
4893 015466
4894
4895
4896
4897 015466 012737 000063 001400
4898 015474 012702 001540
4899 015500 112722 000002
4900 015504 112722 000004
4901 015510 112722 000000
4902 015514 112722 000200
4903 015520 004737 037766
4904 015524 000404

;PREPARE DEVICE FOR DATA TRANSFER
JSR PC,TSTPRP ;PREPARE DEVICE FOR TEST
WORD 154130 ;TASK DESCRIPTOR
BR 30$ ;GO TO 30$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR # DEFINED BY TSTPRP SUBROUTINE
JMP 260$ ;GO TO 260$ IF ERROR WAS FOUND

30$:
;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
MOV #SEEK!GO,RMC$10 ;CHANGE COMMAND TO SEEK
MOV #PUTINX,R2 ;WRITE REGISTER INDEX TABLE
MOVB #RMOA,(R2)+
MOVB #RMOB,(R2)+
MOVB #RMOF,(R2)+
MOVB #RMC$1,(R2)+
MOVB #200,(R2)+
JSR PC,PUT ;GO WRITE REGISTERS VIA PUT SUB
BR 40$ ;GO TO 40$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR DEFINED BY PUT SUB
JMP 260$ ;GO TO 260$ IF ERROR WAS FOUND

40$:
;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
JSR PC,GETSTS ;SETUP FOR STATUS
JSR PC,TIMOUT ;WAIT FOR SEEK TO COMPLETE

50$:
;GO READ SEEK STATUS
JSR PC,GET ;GO READ REGISTERS VIA GET SUB
BR 60$ ;GO TO 60$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR DEFINED BY GET SUB
JMP 260$ ;GO TO 260$ IF ERROR WAS FOUND

60$:
;VERIFY THE RESULTS OF THE SEEK COMMAND
JSR PC,SEKSTS ;GO VERIFY RESULTS OF SEEK OPERATION
BR 70$ ;GO TO 70$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR # DEFINED BY SEKSTS SUBROUTINE
JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
JMP 260$ ;GO TO 260$ IF ERROR WAS FOUND

70$:
;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
MOV #WH!GO,RMC$10 ;LOAD WRITE HEADER AND DATA
MOV #PUTINX+3,R2 ;EXTEND REGISTER INDEX TABLE
MOVB #RMWC,(R2)+
MOVB #RMBB,(R2)+
MOVB #RMC$1,(R2)+
MOVB #200,(R2)+
JSR PC,PUT ;GO WRITE REGISTERS VIA PUT SUB
BR 80$ ;GO TO 80$ IF NO ERROR
```


M08

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 103
T13 FORMAT CHECK ONES W/ WCE ERRORS

SEQ 0103

4905	015526	000240		NOP		;RETURN HERE IF ERROR
4906	015530	104000		ERROR		;ERROR DEFINED BY PUT SUB
4907	015532	000137	016130	JMP	260\$;GO TO 260\$ IF ERROR WAS FOUND
4908	015536					
4909						
4910						
4911	015536	004737	040326			
4912	015542	004737	037516	JSR	PC, TIMEOUT	;WAIT FOR COMMAND TO COMPLETE
4913	015546	000404		JSR	PC, GET	;GO READ REGISTERS VIA GET SUB
4914	015550	000240		BR	90\$;GO TO 90\$ IF NO ERROR
4915	015552	104000		NOP		;RETURN HERE IF ERROR
4916	015554	000137	016130	ERROR		;ERROR DEFINED BY GET SUB
4917	015560			JMP	260\$;GO TO 260\$ IF ERROR WAS FOUND
4918						
4919						
4920	015560	004737	040512			
4921	015564	000405		JSR	PC, PRIERR	;GO CHECK FOR PRIMARY ERRORS
4922	015566	000240		BR	100\$;GO TO 100\$ IF NO ERROR
4923	015570	104000		NOP		;RETURN HERE IF ERROR
4924	015572	004736		ERROR		;ERROR # DEFINED BY PRIERR SUBROUTINE
4925	015574	000137	016130	JSR	PC, 2(SP)+	;GO BACK FOR MORE ERROR CHECKS
4926	015600			JMP	260\$;GO TO 260\$ IF ERROR WAS FOUND
4927	015600	004737	053016			
4928	015604	000405		JSR	PC, DTASTS	;GO VERIFY RESULTS OF DATA TRANSFER
4929	015606	000240		BR	110\$;GO TO 110\$ IF NO ERROR
4930	015610	104000		NOP		;RETURN HERE IF ERROR
4931	015612	004736		ERROR		;ERROR # DEFINED BY DTASTS SUBROUTINE
4932	015614	000137	016130	JSR	PC, 2(SP)+	;GO BACK FOR MORE ERROR CHECKS
4933	015620			JMP	260\$;GO TO 260\$ IF ERROR WAS FOUND
4934	015620	004737	041344			
4935	015624	000405		JSR	PC, SECERR	;GO CHECK FOR SECONDARY ERRORS
4936	015626	000240		BR	120\$;GO TO 120\$ IF NO ERROR
4937	015630	104000		NOP		;RETURN HERE IF ERROR
4938	015632	004736		ERROR		;ERROR # DEFINED BY SECERR SUBROUTINE
4939	015634	000137	016130	JSR	PC, 2(SP)+	;GO BACK FOR MORE ERROR CHECKS
4940	015640			JMP	260\$;GO TO 260\$ IF ERROR WAS FOUND
4941						
4942						
4943	015640	005137	104570			
4944						
4945						
4946						
4947	015644	012737	000053			
4948	015652	004737	037766	MOV	#WCH!GO, RMCS10	;LOAD COMMAND
4949	015656	000404		JSR	PC, PUT	;GO WRITE REGISTERS VIA PUT SUB
4950	015660	000240		BR	180\$;GO TO 180\$ IF NO ERROR
4951	015662	104000		NOP		;RETURN HERE IF ERROR
4952	015664	000137	016130	ERROR		;ERROR DEFINED BY PUT SUB
4953	015670			JMP	260\$;GO TO 260\$ IF ERROR WAS FOUND
4954						
4955						
4956	015670	004737	040326			
4957	015674	004737	037516	JSR	PC, TIMEOUT	;WAIT FOR COMMAND TO COMPLETE
4958	015700	000404		JSR	PC, GET	;GO READ REGISTERS VIA GET SUB
4959	015702	000240		BR	190\$;GO TO 190\$ IF NO ERROR
4960	015704	104000		NOP		;RETURN HERE IF ERROR
				ERROR		;ERROR DEFINED BY GET SUB

```

4961 015706 000137 016130      JMP      260$      ;GO TO 260$ IF ERROR WAS FOUND
4962 015712
4963
4964      ;CHECK FOR PRIMARY ERRORS
4965 015712 004737 040512      JSR      PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
4966 015716 000405      BR      200$      ;GO TO 200$ IF NO ERROR
4967 015720 000240      NOP
4968 015722 104000      ERROR      ;RETURN HERE IF ERROR
4969 015724 004736      JSR      PC,2(SP)+      ;ERROR # DEFINED BY PRIERR SUBROUTINE
4970 015726 000137 016130      JMP      260$      ;GO BACK FOR MORE ERROR CHECKS
4971 015732
4972      200$:      ;GO TO 260$ IF ERROR WAS FOUND
4973
4974 015732 032737 040000 001340      ;MAKE SURE THE WRITE CHECK ERROR WAS DETECTED
4975 015740 001022      BIT      #WCE,RMCS2I      ;IS WRITE CHECK ERROR SET??
4976 015742 004737 053016      BNE      210$      ;YES!!
4977 015746 000405      JSR      PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
4978 015750 000240      BR      205$      ;GO TO 205$ IF NO ERROR
4979 015752 104000      NOP
4980 015754 004736      ERROR      ;RETURN HERE IF ERROR
4981 015756 000137 016130      JSR      PC,2(SP)+      ;ERROR # DEFINED BY DTASTS SUBROUTINE
4982 015762 013737 001340 001140      JMP      260$      ;GO BACK FOR MORE ERROR CHECKS
4983 015770 052737 040000 001140      205$:      MOV      RMCS2I,$GDDAT      ;GO TO 260$ IF ERROR WAS FOUND
4984 015776 013737 001340 001142      BIS      #WCE,$GDDAT      ;LOAD EXPECTED STATUS
4985 016004 104337      MOV      RMCS2I,$BDDAT      ;LOAD RECEIVED STATUS
4986 016006      ERROR      337      ;WRITE CHECK ERROR NOT DETECTED
4987
4988      210$:
4989      ;VERIFY THE ADDRESS OF THE WRITE CHECK ERROR
4990 016006 012737 104570 001134      MOV      #JFTWO-2,$GDAOR      ;LOAD EXPECTED ADDRESS
4991 016014 013737 001334 001136      MOV      #JAI,$GDAOR      ;LOAD RECEIVED ADDRESS
4992 016022 162737 000002 001136      SUB      #2,$BDAOR      ;DECREMENT RECEIVED ADDRESS
4993
4994      ;GET WCE DATA AND VERIFY IT IS OK
4995 016036 112737 000022 001506      MOV      #RMDB,GETINX      ;SETUP FOR READING RMDB
4996 016044 004737 037516 001507      MOV      #200,GETINX+1
4997 016050 000404      JSR      PC,GET      ;GO READ REGISTERS VIA GET SUB
4998 016052 000240      BR      230$      ;GO TO 230$ IF NO ERROR
4999 016054 104000      NOP
5000 016056 000137 016130      ERROR      ;RETURN HERE IF ERROR
5001 016062 013737 001352 001142      JMP      260$      ;ERROR DEFINED BY GET SUB
5002 016070 013737 104570 001140      230$:      MOV      RMDBI,$BDDAT      ;GO TO 260$ IF ERROR WAS FOUND
5003 016076 005137 001140      MOV      BUFTWO-2,$GDDAT      ;LOAD RECEIVED DATA WORD
5004 016102 023737 001134 001136      COM      $GDDAT      ;LOAD EXPECTED DATA WORD
5005 016110 001402      CMP      $GDAOR,$BDAOR      ;IS ADDRESS OK??
5006 016112 104340      BEQ      220$      ;YES!!
5007 016114 000405      ERROR      340      ;ADDRESS OF WCE INCORRECT
5008 016116      BR      260$
5009 016116 023737 001140 001142      220$:      CMP      $GDDAT,$BDDAT      ;IS DATA WORD OK??
5010 016124 001401      BEQ      260$      ;YES!!
5011 016126 104341      ERROR      341      ;UNEXPECTED WCE DATA
5012 016130
5013
5014      260$:
5015      ;*****
5016      ;TEST 14      FORMAT MULTIPLE SECTORS
5017      ;*****

```

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 105
T14 FORMAT MULTIPLE SECTORS

SEQ 0105

```

5017 016130
5018 016130 000004
5019 016132 000240
5020 016134 012706 001100
5021 016140 013700 001276
5022 016144 013701 001450
5023 016150 012737 000014 001226
5024 016156
5025
5026
5027 016156 012737 000000 001434
5028 016164 012737 000000 001406
5029 016172 012737 010000 001432
5030 016180 012737 176774 001402
5031 016206 012737 103566 001404
5032 016214 012737 000062 001400
5033 016222 012737 065106 001174
5034 016230 012737 000001 001176
5035 016236 004737 036620
5036 016242
5037
5038
5039 016242 004737 034000
5040 016246 154130
5041 016250 000404
5042 016252 000240
5043 016254 104000
5044 016256 000137 016704
5045 016262
5046
5047
5048 016262 012737 000005 001400
5049 016270 012702 001535
5050 016274 112722 000006
5051 016300 112722 000034
5052 016304 112722 000032
5053 016310 112722 000000
5054 016314 112722 000200
5055 016320 004737 037766
5056 016324 000404
5057 016326 000240
5058 016330 104000
5059 016332 000137 016704
5060 016336
5061
5062
5063 016336 004737 037432
5064 016342 004737 040326
5065 016346
5066
5067
5068 016346 004737 037516
5069 016352 000404
5070 016354 000240
5071 016356 104000
5072 016360 000137 016704

TST14:
SCOPE
NOP
MOV #STACK, SP
MOV #BASE, R0
MOV TSTQUE, R1
MOV #14, $TESTN

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

10$:
;SETUP PARAMETERS FOR GENERATING DATA BUFFER
MOV #0, RMDC0 ;CYLINDER = 0
MOV #0, RMDA0 ;TRACK = SECTOR = 0
MOV #FMT16, RMFO ;16 BIT FORMAT
MOV #((C<<(2+256.) *2)+1), RMWC0 ;WORD COUNT FOR 2 SECTORS
MOV #BUFONE, RMB0 ;DATA BUFFER ADDRESS
MOV #MH, RMCS10 ;WRITE HEADER AND DATA
MOV #ZEROS, $TMP0 ;USE ALL ZEROS DATA PATTERN
MOV #1, $TMP1
JSR PC, GENBUF ;GO GENERATE DATA BUFFER

20$:
;PREPARE DEVICE FOR DATA TRANSFER
JSR PC, TSTPRP ;PREPARE DEVICE FOR TEST
;WORD 154130 ;TASK DESCRIPTOR
BR 30$ ;GO TO 30$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR # DEFINED BY TSTPRP SUBROUTINE
JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND

30$:
;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
MOV #SEEK, GO, RMCS10 ;CHANGE COMMAND TO SEEK
MOV #PUTINX, R2 ;WRITE REGISTER INDEX TABLE
MOVB #RMDA, (R2)+
MOVB #RMDC, (R2)+
MOVB #RMFO, (R2)+
MOVB #RMCS1, (R2)+
MOVB #200, (R2)+
JSR PC, PUT ;GO WRITE REGISTERS VIA PUT SUB
BR 40$ ;GO TO 40$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR DEFINED BY PUT SUB
JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND

40$:
;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
JSR PC, GETSTS ;SETUP FOR STATUS
JSR PC, TIMEOUT ;WAIT FOR SEEK TO COMPLETE

50$:
;GO READ SEEK STATUS
JSR PC, GET ;GO READ REGISTERS VIA GET SUB
BR 60$ ;GO TO 60$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR DEFINED BY GET SUB
JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND

```

5073	016364			60\$:	
5074					
5075					;VERIFY THE RESULTS OF THE SEEK COMMAND
5076	016364	004737	045420	JSR	PC,SEKSTS ;GO VERIFY RESULTS OF SEEK OPERATION
5077	016370	000405		BR	70\$;GO TO 70\$ IF NO ERROR
5078	016372	000240		NOP	;RETURN HERE IF ERROR
5079	016374	104000		ERROR	;ERROR # DEFINED BY SEKSTS SUBROUTINE
5080	016376	004736		JSR	PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
5081	016400	000137	016704	JMP	180\$;GO TO 180\$ IF ERROR WAS FOUND
5082	016404			70\$:	
5083					;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
5084				MOV	#WH!GO,RMCS10 ;LOAD WRITE HEADER AND DATA
5085	016404	012737	000063	MOV	#PUTINX+3,R2 ;EXTEND REGISTER INDEX TABLE
5086	016412	012702	001540	MOV	#RMWC,(R2)+
5087	016416	112722	000002	MOV	#R3A,(R2)+
5088	016422	112722	000004	MOV	#RMCS1,(R2)+
5089	016426	112722	000000	MOV	#200,(R2)+
5090	016432	112722	000000	JSR	PC,PUT ;GO WRITE REGISTERS VIA PUT SUB
5091	016436	004737	037766	BR	80\$;GO TO 80\$ IF NO ERROR
5092	016442	000404		NOP	;RETURN HERE IF ERROR
5093	016444	000240		ERROR	;ERROR DEFINED BY PUT SUB
5094	016446	104000		JMP	180\$;GO TO 180\$ IF ERROR WAS FOUND
5095	016450	000137	016704	80\$:	
5096	016454				;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
5097				JSR	PC,TIMOUT ;WAIT FOR COMMAND TO COMPLETE
5098				JSR	PC,GET ;GO READ REGISTERS VIA GET SUB
5099	016454	004737	040326	BR	90\$;GO TO 90\$ IF NO ERROR
5100	016460	004737	037516	NOP	;RETURN HERE IF ERROR
5101	016464	000404		ERROR	;ERROR DEFINED BY GET SUB
5102	016466	000240		JMP	180\$;GO TO 180\$ IF ERROR WAS FOUND
5103	016470	104000		90\$:	
5104	016472	000137	016704		;VERIFY RESULTS OF WRITE COMMAND
5105	016476			JSR	PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
5106				BR	100\$;GO TO 100\$ IF NO ERROR
5107				NOP	;RETURN HERE IF ERROR
5108	016476	004737	040512	ERROR	;ERROR # DEFINED BY PRIERR SUBROUTINE
5109	016502	000405		JSR	PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
5110	016504	000240		JMP	180\$;GO TO 180\$ IF ERROR WAS FOUND
5111	016506	104000		100\$:	
5112	016510	004736		JSR	PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
5113	016512	000137	016704	BR	110\$;GO TO 110\$ IF NO ERROR
5114	016516			NOP	;RETURN HERE IF ERROR
5115	016516	004737	053016	ERROR	;ERROR # DEFINED BY DTASTS SUBROUTINE
5116	016522	000405		JSR	PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
5117	016524	000240		JMP	180\$;GO TO 180\$ IF ERROR WAS FOUND
5118	016526	104000		110\$:	
5119	016530	004736		JSR	PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
5120	016532	000137	016704	BR	120\$;GO TO 120\$ IF NO ERROR
5121	016536			NOP	;RETURN HERE IF ERROR
5122	016542	004737	041344	ERROR	;ERROR # DEFINED BY SECERR SUBROUTINE
5123	016544	000240		JSR	PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
5124	016546	104000		JMP	180\$;GO TO 180\$ IF ERROR WAS FOUND
5125	016550	004736		120\$:	
5126	016552	000137	016704		
5127	016556				
5128					

```
5129
5130
5131 ;WRITE CHECK HEADER AND DATA FOR SECTORS JUST WRITTEN
5132 016556 012737 000053 001400 MOV #WCH!GO, RMCS10 ;READ HEADER & DATA COMMAND
5133 016564 004737 037766 JSR PC, PUT ;GO WRITE REGISTERS VIA PUT SUB
5134 016570 000404 BR 130$ ;GO TO 130$ IF NO ERROR
5135 016572 000240 NOP ;RETURN HERE IF ERROR
5136 016574 104000 ERROR ;ERROR DEFINED BY PUT SUB
5137 016576 000137 016704 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
5138 016602 130$:
5139
5140 ;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS
5141 016602 004737 040326 JSR PC, TIMEOUT ;WAIT FOR WRITE CHECK TO FINISH
5142 016606 004737 037516 JSR PC, GET ;GO READ REGISTERS VIA GET SUB
5143 016612 000404 BR 140$ ;GO TO 140$ IF NO ERROR
5144 016614 000240 NOP ;RETURN HERE IF ERROR
5145 016616 104000 ERROR ;ERROR DEFINED BY GET SUB
5146 016620 000137 016704 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
5147 016624 140$:
5148
5149 ;VERIFY THE RESULTS OF WRITE CHECK OPERATION
5150 016624 004737 040512 JSR PC, PRIERR ;GO CHECK FOR PRIMARY ERRORS
5151 016630 000405 BR 150$ ;GO TO 150$ IF NO ERROR
5152 016632 000240 NOP ;RETURN HERE IF ERROR
5153 016634 104000 ERROR ;ERROR # DEFINED BY PRIERR SUBROUTINE
5154 016636 074736 JSR PC, 2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
5155 016640 000137 016704 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
5156 016644 150$:
5157 016644 004737 053016 JSR PC, DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
5158 016650 000405 BR 160$ ;GO TO 160$ IF NO ERROR
5159 016652 000240 NOP ;RETURN HERE IF ERROR
5160 016654 104000 ERROR ;ERROR # DEFINED BY DTASTS SUBROUTINE
5161 016656 004736 JSR PC, 2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
5162 016660 000137 016704 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
5163 016664 160$:
5164 016664 004737 041344 JSR PC, SECERR ;GO CHECK FOR SECONDARY ERRORS
5165 016670 000405 BR 170$ ;GO TO 170$ IF NO ERROR
5166 016672 000240 NOP ;RETURN HERE IF ERROR
5167 016674 104000 ERROR ;ERROR # DEFINED BY SECERR SUBROUTINE
5168 016676 004736 JSR PC, 2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
5169 016700 000137 016704 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
5170 016704 170$:
5171 016704 180$:
5172
5173 ;*****
5174 ;*TEST 15 FORMAT W/ HEAD SWITCHING
5175 ;*****
5176 016704 ST15:
5177 016704 000004 SCOPE ;SCOPE CALL
5178 016706 000240 NOP ;START OF TEST
5179 016710 012706 001100 MOV #STACK, SP ;INITIALIZE STACK POINTER
5180 016714 013700 001276 MOV $BASE, RO ;RO=UNIBUS ADDRESS
5181 016720 013701 001450 MOV TSTQUE, R1 ;(R1) = DEVICE BEING TESTED
5182 016724 012737 000015 001226 MOV #15, $TESTN ;SET TEST NUMBER IN APT MAIL BOX
5183 016732 10$:
5184
```

```

5185 ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
5186 016732 012737 000000 001434 MOV #0,RMDC0 ;CYLINDER = 0
5187 016740 012737 000037 001406 MOV #31,RMDA0 ;START AT LAST SECTOR
5188 016746 012737 010000 001432 MOV #FMT16,RMOF0 ;16 BIT FORMAT
5189 016754 012737 176774 001402 MOV #(<C<(2+256.)#2>+1),RMWC0 ;WORD COUNT FOR 2 SECTORS
5190 016762 012737 103566 001404 MOV #BUFONE,RMBA0 ;DATA BUFFER ADDRESS
5191 016770 012737 000062 001400 MOV #WH,RMCS10 ;WRITE HEADER AND DATA
5192 016776 012737 065106 001174 MOV #ZEROS,STMP0 ;USE ALL ZEROS DATA PATTERN
5193 017004 012737 000001 001176 MOV #1,STMP1
5194 017012 004737 036620 JSR PC,GENBUF ;GO GENERATE DATA BUFFER
5195 017016
5196
5197 ;PREPARE DEVICE FOR DATA TRANSFER
5198 017016 004737 034000 JSR PC,TSTPRP ;PREPARE DEVICE FOR TEST
5199 017022 154130 .WORD 154130 ;TASK DESCRIPTOR
5200 017024 000404 BR 30$ ;GO TO 30$ IF NO ERROR
5201 017026 000240 NOP ;RETURN HERE IF ERROR
5202 017030 104000 ERROR ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5203 017032 000137 017460 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
5204 017036
5205
5206 ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
5207 017036 012737 000005 001400 MOV #SEEK!GO,RMCS10 ;CHANGE COMMAND TO SEEK
5208 017044 012702 001535 MOV #PUTINX,R2 ;WRITE REGISTER INDEX TABLE
5209 017050 112722 000006 MOV #RMDA,(R2)+
5210 017054 112722 000034 MOV #RMDC,(R2)+
5211 017060 112722 000032 MOV #RMOF,(R2)+
5212 017064 112722 000000 MOV #RMCS1,(R2)+
5213 017070 112722 000200 MOV #200,(R2)+
5214 017074 004737 037766 JSR PC,PUT ;GO WRITE REGISTERS VIA PUT SUB
5215 017100 000404 BR 40$ ;GO TO 40$ IF NO ERROR
5216 017102 000240 NOP ;RETURN HERE IF ERROR
5217 017104 104000 ERROR ;ERROR DEFINED BY PUT SUB
5218 017106 000137 017460 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
5219 017112
5220
5221 ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
5222 017112 004737 037432 JSR PC,GETSTS ;SETUP FOR STATUS
5223 017116 004737 040326 JSR PC,TIMOUT ;WAIT FOR SEEK TO COMPLETE
5224 017122
5225
5226 ;GO READ SEEK STATUS
5227 017122 004737 037516 JSR PC,GET ;GO READ REGISTERS VIA GET SUB
5228 017126 000404 BR 60$ ;GO TO 60$ IF NO ERROR
5229 017130 000240 NOP ;RETURN HERE IF ERROR
5230 017132 104000 ERROR ;ERROR DEFINED BY GET SUB
5231 017134 000137 017460 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
5232 017140
5233
5234 ;VERIFY THE RESULTS OF THE SEEK COMMAND
5235 017140 004737 045420 JSR PC,SEKSTS ;GO VERIFY RESULTS OF SEEK OPERATION
5236 017144 000405 BR 70$ ;GO TO 70$ IF NO ERROR
5237 017146 000240 NOP ;RETURN HERE IF ERROR
5238 017150 104000 ERROR ;ERROR # DEFINED BY SEKSTS SUBROUTINE
5239 017152 004736 JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
5240 017154 000137 017460 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND

```

```

5241 017160 70$:
5242
5243
5244 017160 012737 000063 001400 ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
5245 017166 012702 001540 MOV #WH!GO, RMCS10 ;LOAD WRITE HEADER AND DATA
5246 017172 112722 000002 MOV #PUTINX+3, R2 ;EXTEND REGISTER INDEX TABLE
5247 017176 112722 000004 MOVB #RMWC, (R2)+
5248 017202 112722 000000 MOVB #RMA, (R2)+
5249 017206 112722 000200 MOVB #RMCS1, (R2)+
5250 017212 004737 037766 MOVB #200, (R2)+
5251 017216 000404 JSR PC, PUT ;GO WRITE REGISTERS VIA PUT SUB
5252 017220 000240 BR 80$ ;GO TO 80$ IF NO ERROR
5253 017222 104000 NOP ;RETURN HERE IF ERROR
5254 017224 000137 017460 ERROR ;ERROR DEFINED BY PUT SUB
5255 017230 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
5256
5257 ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
5258 017230 004737 040326 JSR PC, TIMEOUT ;WAIT FOR COMMAND TO COMPLETE
5259 017234 004737 037516 JSR PC, GET ;GO READ REGISTERS VIA GET SUB
5260 017240 000404 BR 90$ ;GO TO 90$ IF NO ERROR
5261 017242 000240 NOP ;RETURN HERE IF ERROR
5262 017244 104000 ERROR ;ERROR DEFINED BY GET SUB
5263 017246 000137 017460 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
5264 017252
5265
5266 ;VERIFY RESULTS OF WRITE COMMAND
5267 017252 004737 040512 JSR PC, PRIERR ;GO CHECK FOR PRIMARY ERRORS
5268 017256 000405 BR 100$ ;GO TO 100$ IF NO ERROR
5269 017260 000240 NOP ;RETURN HERE IF ERROR
5270 017262 104000 ERROR ;ERROR # DEFINED BY PRIERR SUBROUTINE
5271 017264 004736 JSR PC, 2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
5272 017266 000137 017460 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
5273 017272
5274 017272 004737 053016 JSR PC, DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
5275 017276 000405 BR 110$ ;GO TO 110$ IF NO ERROR
5276 017300 000240 NOP ;RETURN HERE IF ERROR
5277 017302 104000 ERROR ;ERROR # DEFINED BY DTASTS SUBROUTINE
5278 017304 004736 JSR PC, 2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
5279 017306 000137 017460 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
5280 017312
5281 017312 004737 041344 JSR PC, SECERR ;GO CHECK FOR SECONDARY ERRORS
5282 017316 000405 BR 120$ ;GO TO 120$ IF NO ERROR
5283 017320 000240 NOP ;RETURN HERE IF ERROR
5284 017322 104000 ERROR ;ERROR # DEFINED BY SECERR SUBROUTINE
5285 017324 004736 JSR PC, 2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
5286 017326 000137 017460 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
5287 017332
5288
5289
5290 ;WRITE CHECK HEADER AND DATA FOR SECTORS JUST WRITTEN
5291 017332 012737 000053 001400 MOV #WCH!GO, RMCS10 ;READ HEADER & DATA COMMAND
5292 017340 004737 037766 JSR PC, PUT ;GO WRITE REGISTERS VIA PUT SUB
5293 017344 000404 BR 130$ ;GO TO 130$ IF NO ERROR
5294 017346 000240 NOP ;RETURN HERE IF ERROR
5295 017350 104000 ERROR ;ERROR DEFINED BY PUT SUB
5296 017352 000137 017460 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND

```

```

5297 017356      130$:
5298
5299 ;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS
5300 017356 004737 040326      JSR PC,TIMOUT ;WAIT FOR WRITE CHECK TO FINISH
5301 017362 004737 037516      JSR PC,GET ;GO READ REGISTERS VIA GET SUB
5302 017366 000404          BR 140$ ;GO TO 140$ IF NO ERROR
5303 017370 000240          NOP ;RETURN HERE IF ERROR
5304 017372 104000          ERROR ;ERROR DEFINED BY GET SUB
5305 017374 000137 017460      JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
5306 017400
5307
5308 ;VERIFY THE RESULTS OF WRITE CHECK OPERATION
5309 017400 004737 040512      JSR PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
5310 017404 000405          BR 150$ ;GO TO 150$ IF NO ERROR
5311 017406 000240          NOP ;RETURN HERE IF ERROR
5312 017410 104000          ERROR ;ERROR # DEFINED BY PRIERR SUBROUTINE
5313 017412 004736          JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
5314 017414 000137 017460      JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
5315 017420
5316 017420 004737 053016      JSR PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
5317 017424 000405          BR 160$ ;GO TO 160$ IF NO ERROR
5318 017426 000240          NOP ;RETURN HERE IF ERROR
5319 017430 104000          ERROR ;ERROR # DEFINED BY DTASTS SUBROUTINE
5320 017432 004736          JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
5321 017434 000137 017460      JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
5322 017440
5323 017440 004737 041344      JSR PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
5324 017444 000405          BR 170$ ;GO TO 170$ IF NO ERROR
5325 017446 000240          NOP ;RETURN HERE IF ERROR
5326 017450 104000          ERROR ;ERROR # DEFINED BY SECERR SUBROUTINE
5327 017452 004736          JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
5328 017454 000137 017460      JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
5329 017460
5330 017460
5331
5332 ;*****
5333 ;*TEST 16 FORMAT W/ MID TRANSFER SEEK
5334 ;*****
5335 017460
5336 017460 000004          TST16:
5337 017462 000240          SCOPE ;SCOPE CALL
5338 017464 012706 001100      MOV #S'ACK,SP ;START OF TEST
5339 017470 013700 001276      MOV $BASE,R0 ;INITIALIZE STACK POINTER
5340 017474 013701 001450      MOV TSTQUE,R1 ;R0=UNIBUS ADDRESS
5341 017500 012737 000016 001226  MOV #16,$TESTN ;(R1) = DEVICE BEING TESTED
5342 017506
5343
5344 ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
5345 017506 012737 000000 001434      MOV #0,RMDC0 ;CYLINDER = 0
5346 017514 012737 002037 001406      MOV #002037,RMDAO ;START AT LAST TRACK & SECTOR
5347 017522 012737 010000 001432      MOV #FMT16,RMOFO ;16 BIT FORMAT
5348 017530 012737 176774 001402      MOV #<1C<<2+256.>*2>+1>,RMWC0 ;WORD COUNT FOR 2 SECTORS
5349 017536 012737 103566 001404      MOV #BUFONE,RMBAO ;DATA BUFFER ADDRESS
5350 017544 012737 000062 001400      MOV #WH,RMC$10 ;WRITE HEADER AND DATA
5351 017552 012737 065106 001174      MOV #ZEROS,$TMP0 ;USE ALL ZEROS DATA PATTERN
5352 017560 012737 000001 001176      MOV #1,$TMP1

```


CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 111
T16 FORMAT W/ MID TRANSFER SEEK

SEQ 0111

5353	017566	004737	036620		JSR	PC,GENBUF	;GO GENERATE DATA BUFFER
5354	017572			20\$:			
5355							
5356							
5357	017572	004737	034000		JSR	PC,TSTPRP	;PREPARE DEVICE FOR TEST
5358	017576	154130			WORD	154130	;TASK DESCRIPTOR
5359	017600	000404			BR	30\$;GO TO 30\$ IF NO ERROR
5360	017602	000240			NOP		;RETURN HERE IF ERROR
5361	017604	104000			ERROR		;ERROR # DEFINED BY TSTPRP SUBROUTINE
5362	017606	000137	020234		JMP	180\$;GO TO 180\$ IF ERROR WAS FOUND
5363	017612			30\$:			
5364							
5365							
5366	017612	012737	000005	001400			
5367	017620	012702	001535		MOV	#SEEK!GO,RMCS10	;CHANGE COMMAND TO SEEK
5368	017624	112722	000006		MOV	#PUTINX,R2	;WRITE REGISTER INDEX TABLE
5369	017630	112722	000034		MOVB	#RMOA,(R2)+	
5370	017634	112722	000032		MOVB	#RMOB,(R2)+	
5371	017640	112722	000000		MOVB	#RMOF,(R2)+	
5372	017644	112722	000200		MOVB	#RMCS1,(R2)+	
5373	017650	004737	037766		JSR	PC,PUT	;GO WRITE REGISTERS VIA PUT SUB
5374	017654	000404			BR	40\$;GO TO 40\$ IF NO ERROR
5375	017656	000240			NOP		;RETURN HERE IF ERROR
5376	017660	104000			ERROR		;ERROR DEFINED BY PUT SUB
5377	017662	000137	020234		JMP	180\$;GO TO 180\$ IF ERROR WAS FOUND
5378	017666			40\$:			
5379							
5380							
5381	017666	004737	037432		JSR	PC,GETSTS	;SETUP FOR STATUS
5382	017672	004737	040326		JSR	PC,TIMOUT	;WAIT FOR SEEK TO COMPLETE
5383	017676			50\$:			
5384							
5385							
5386	017676	004737	037516		JSR	PC,GET	;GO READ REGISTERS VIA GET SUB
5387	017702	000404			BR	60\$;GO TO 60\$ IF NO ERROR
5388	017704	000240			NOP		;RETURN HERE IF ERROR
5389	017706	104000			ERROR		;ERROR DEFINED BY GET SUB
5390	017710	000137	020234		JMP	180\$;GO TO 180\$ IF ERROR WAS FOUND
5391	017714			60\$:			
5392							
5393							
5394	017714	004737	045420		JSR	PC,SEKSTS	;GO VERIFY RESULTS OF SEEK OPERATION
5395	017720	000405			BR	70\$;GO TO 70\$ IF NO ERROR
5396	017722	000240			NOP		;RETURN HERE IF ERROR
5397	017724	104000			ERROR		;ERROR # DEFINED BY SEKSTS SUBROUTINE
5398	017726	004736			JSR	PC,2(SP)+	;GO BACK FOR MORE ERROR CHECKS
5399	017730	000137	020234		JMP	180\$;GO TO 180\$ IF ERROR WAS FOUND
5400	017734			70\$:			
5401							
5402							
5403	017734	012737	000063	001400			
5404	017742	012702	001540		MOV	#WH!GO,RMCS10	;LOAD WRITE HEADER AND DATA
5405	017746	112722	000002		MOV	#PUTINX+3,R2	;EXTEND REGISTER INDEX TABLE
5406	017752	112722	000004		MOVB	#RMWC,(R2)+	
5407	017756	112722	000000		MOVB	#RMBB,(R2)+	
5408	017762	112722	000200		MOVB	#RMCS1,(R2)+	
					MOVB	#200,(R2)+	

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 112
T16 FORMAT W/ MID TRANSFER SEEK

SEQ 0112

5409	017766	004737	037766		JSR	PC,PUT		;GO WRITE REGISTERS VIA PUT SUB
5410	017772	000404			BR	80\$;GO TO 80\$ IF NO ERROR
5411	017774	000240			NOP			;RETURN HERE IF ERROR
5412	017776	104000			ERROR			;ERROR DEFINED BY PUT SUB
5413	020000	000137	020234		JMP	180\$;GO TO 180\$ IF ERROR WAS FOUND
5414	020004			80\$:				
5415								
5416								;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
5417	020004	004737	040326		JSR	PC,TIMOUT		;WAIT FOR COMMAND TO COMPLETE
5418	020010	004737	037516		JSR	PC,GET		;GO READ REGISTERS VIA GET SUB
5419	020014	000404			BR	90\$;GO TO 90\$ IF NO ERROR
5420	020016	000240			NOP			;RETURN HERE IF ERROR
5421	020020	104000			ERROR			;ERROR DEFINED BY GET SUB
5422	020022	000137	020234		JMP	180\$;GO TO 180\$ IF ERROR WAS FOUND
5423	020026			90\$:				
5424								
5425								;VERIFY RESULTS OF WRITE COMMAND
5426	020026	004737	040512		JSR	PC,PRIERR		;GO CHECK FOR PRIMARY ERRORS
5427	020032	000405			BR	100\$;GO TO 100\$ IF NO ERROR
5428	020034	000240			NOP			;RETURN HERE IF ERROR
5429	020036	104000			ERROR			;ERROR # DEFINED BY PRIERR SUBROUTINE
5430	020040	004736			JSR	PC,2(SP)+		;GO BACK FOR MORE ERROR CHECKS
5431	020042	000137	020234		JMP	180\$;GO TO 180\$ IF ERROR WAS FOUND
5432	020046			100\$:				
5433	020046	004737	053016		JSR	PC,DTASTS		;GO VERIFY RESULTS OF DATA TRANSFER
5434	020052	000405			BR	110\$;GO TO 110\$ IF NO ERROR
5435	020054	000240			NOP			;RETURN HERE IF ERROR
5436	020056	104000			ERROR			;ERROR # DEFINED BY DTASTS SUBROUTINE
5437	020060	004736			JSR	PC,2(SP)+		;GO BACK FOR MORE ERROR CHECKS
5438	020062	000137	020234		JMP	180\$;GO TO 180\$ IF ERROR WAS FOUND
5439	020066			110\$:				
5440	020066	004737	041344		JSR	PC,SECERR		;GO CHECK FOR SECONDARY ERRORS
5441	020072	000405			BR	120\$;GO TO 120\$ IF NO ERROR
5442	020074	000240			NOP			;RETURN HERE IF ERROR
5443	020076	104000			ERROR			;ERROR # DEFINED BY SECERR SUBROUTINE
5444	020100	004736			JSR	PC,2(SP)+		;GO BACK FOR MORE ERROR CHECKS
5445	020102	000137	020234		JMP	180\$;GO TO 180\$ IF ERROR WAS FOUND
5446	020106			120\$:				
5447								
5448								
5449								;WRITE CHECK HEADER AND DATA FOR SECTORS JUST WRITTEN
5450	020106	012737	000053	001400	MOV	#WCH!GO, RMCS10		;READ HEADER & DATA COMMAND
5451	020114	004737	037766		JSR	PC,PUT		;GO WRITE REGISTERS VIA PUT SUB
5452	020120	000404			BR	130\$;GO TO 130\$ IF NO ERROR
5453	020122	000240			NOP			;RETURN HERE IF ERROR
5454	020124	104000			ERROR			;ERROR DEFINED BY PUT SUB
5455	020126	000137	020234		JMP	180\$;GO TO 180\$ IF ERROR WAS FOUND
5456	020132			130\$:				
5457								
5458								;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS
5459	020132	004737	040326		JSR	PC,TIMOUT		;WAIT FOR WRITE CHECK TO FINISH
5460	020136	004737	037516		JSR	PC,GET		;GO READ REGISTERS VIA GET SUB
5461	020142	000404			BR	140\$;GO TO 140\$ IF NO ERROR
5462	020144	000240			NOP			;RETURN HERE IF ERROR
5463	020146	104000			ERROR			;ERROR DEFINED BY GET SUB
5464	020150	000137	020234		JMP	180\$;GO TO 180\$ IF ERROR WAS FOUND

5465 020154
5466
5467
5468 020154 004737 040512
5469 020160 000405
5470 020162 000240
5471 020164 104000
5472 020166 004736
5473 020170 000137 020234
5474 020174
5475 020174 004737 053016
5476 020200 000405
5477 020202 000240
5478 020204 104000
5479 020206 004736
5480 020210 000137 020234
5481 020214
5482 020214 004737 041344
5483 020220 000405
5484 020222 000240
5485 020224 104000
5486 020226 004736
5487 020230 000137 020234
5488 020234
5489 020234
5490
5491
5492
5493
5494 020234
5495 020234 000004
5496 020236 000240
5497 020240 012706 001100
5498 020244 013700 001276
5499 020250 013701 001450
5500 020254 012737 000017 001226
5501 020262
5502
5503
5504 020262 012737 000000 001434
5505 020270 012737 000000 001406
5506 020276 012737 010000 001432
5507 020304 012737 176774 001402
5508 020312 012737 103566 001404
5509 020320 012737 000062 001400
5510 020326 012737 065106 001174
5511 020334 012737 000001 001176
5512 020342 004737 036620
5513 020346
5514
5515
5516 020346 004737 034000
5517 020352 154130
5518 020354 000404
5519 020356 000240
5520 020360 104000

140\$:
;VERIFY THE RESULTS OF WRITE CHECK OPERATION
JSR PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
BR 150\$;GO TO 150\$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR # DEFINED BY PRIERR SUBROUTINE
JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
JMP 180\$;GO TO 180\$ IF ERROR WAS FOUND
150\$:
JSR PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
BR 160\$;GO TO 160\$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR # DEFINED BY DTASTS SUBROUTINE
JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
JMP 180\$;GO TO 180\$ IF ERROR WAS FOUND
160\$:
JSR PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
BR 170\$;GO TO 170\$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR # DEFINED BY SECERR SUBROUTINE
JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
JMP 180\$;GO TO 180\$ IF ERROR WAS FOUND
170\$:
180\$:
;*****
;TEST 17 FORMAT W/ IMPLIED SEEK
;*****
TST17:
SCOPE ;SCOPE CALL
NOP ;START OF TEST
MOV #STACK,SP ;INITIALIZE STACK POINTER
MOV \$BASE,R0 ;R0=UNIBUS ADDRESS
MOV TSTQUE,R1 ;(R1) = DEVICE BEING TESTED
MOV #17,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
10\$:
;SETUP PARAMETERS FOR GENERATING DATA BUFFER
MOV #0,RMDC0 ;CYLINDER = 0
MOV #0,RMDA0 ;TRACK = SECTOR = 0
MOV #FMT16,RMCF0 ;16 BIT FORMAT
MOV #((C<<2+256.>)*2)+1,RMWC0 ;WORD COUNT FOR 2 SECTORS
MOV #BUFONE,RMBA0 ;DATA BUFFER ADDRESS
MOV #WH,RMCS10 ;WRITE HEADER AND DATA
MOV #ZEROS,\$TMP0 ;USE ALL ZEROS DATA PATTERN
MOV #1,\$TMP1
JSR PC,GENBUF ;GO GENERATE DATA BUFFER
20\$:
;PREPARE DEVICE FOR DATA TRANSFER
JSR PC,TSTPRP ;PREPARE DEVICE FOR TEST
WORD 154130 ;TASK DESCRIPTOR
BR 30\$;GO TO 30\$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR # DEFINED BY TSTPRP SUBROUTINE

```

5521 020362 000137 021032          JMP      180$          ;GO TO 180$ IF ERROR WAS FOUND
5522 020366
5523
5524          ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE OFF CYLINDER
5525 020366 013737 001434 021034      MOV      RMDCO,190$          ;SAVE CYLINDER ADDRESS
5526 020374 012737 001466 001434      MOV      #822,RMDCO          ;SEEK TO LAST CYLINDER
5527 020402 012737 000005 001400      MOV      #SEEK!GO,RMCS10      ;CHANGE COMMAND TO SEEK
5528 020410 012702 001535          MOV      #PUTINX,R2          ;WRITE REGISTER INDEX TABLE
5529 020414 112722 000006          MOV      #RMDA,(R2)+
5530 020420 112722 000034          MOV      #RMDC,(R2)+
5531 020424 112722 000032          MOV      #RMOF,(R2)+
5532 020430 112722 000000          MOV      #RMCS1,(R2)+
5533 020434 112722 000200          MOV      #200,(R2)+
5534 020440 004737 037766          JSR      PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5535 020444 000404          BR      40$          ;GO TO 40$ IF NO ERROR
5536 020446 000240          NOP          ;RETURN HERE IF ERROR
5537 020450 104000          ERROR          ;ERROR DEFINED BY PUT SUB
5538 020452 000137 021032          JMP      180$          ;GO TO 180$ IF ERROR WAS FOUND
5539 020456
5540
5541          ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
5542 020456 004737 037432          JSR      PC,GETSTS          ;SETUP FOR STATUS
5543 020462 004737 040326          JSR      PC,TIMOUT          ;WAIT FOR SEEK TO COMPLETE
5544 020466
5545
5546          ;GO READ SEEK STATUS
5547 020466 004737 037516          JSR      PC,GET          ;GO READ REGISTERS VIA GET SUB
5548 020472 000404          BR      60$          ;GO TO 60$ IF NO ERROR
5549 020474 000240          NOP          ;RETURN HERE IF ERROR
5550 020476 104000          ERROR          ;ERROR DEFINED BY GET SUB
5551 020500 000137 021032          JMP      180$          ;GO TO 180$ IF ERROR WAS FOUND
5552 020504
5553
5554          ;VERIFY THE RESULTS OF THE SEEK COMMAND
5555 020504 004737 045420          JSR      PC,SEKSTS          ;GO VERIFY RESULTS OF SEEK OPERATION
5556 020510 000405          BR      70$          ;GO TO 70$ IF NO ERROR
5557 020512 000240          NOP          ;RETURN HERE IF ERROR
5558 020514 104000          ERROR          ;ERROR # DEFINED BY SEKSTS SUBROUTINE
5559 020516 004736          JSR      PC,2(SP)+          ;GO BACK FOR MORE ERROR CHECKS
5560 020520 000137 021032          JMP      180$          ;GO TO 180$ IF ERROR WAS FOUND
5561 020524
5562
5563          ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
5564 020524 013737 021034 001434      MOV      190$,RMDCO          ;RESTORE DISK ADDRESS
5565 020532 012737 000063 001400      MOV      #WH!GO,RMCS10          ;LOAD WRITE HEADER AND DATA
5566 020540 012702 001540          MOV      #PUTINX+3,R2          ;EXTEND REGISTER INDEX TABLE
5567 020544 112722 000002          MOV      #RMMC,(R2)+
5568 020550 112722 000004          MOV      #RMBA,(R2)+
5569 020554 112722 000000          MOV      #RMCS1,(R2)+
5570 020560 112722 000200          MOV      #200,(R2)+
5571 020564 004737 037766          JSR      PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5572 020570 000404          BR      80$          ;GO TO 80$ IF NO ERROR
5573 020572 000240          NOP          ;RETURN HERE IF ERROR
5574 020574 104000          ERROR          ;ERROR DEFINED BY PUT SUB
5575 020576 000137 021032          JMP      180$          ;GO TO 180$ IF ERROR WAS FOUND
5576 020602
80$:

```

L09

```

5577      ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
5578      JSR      PC,TIMOUT      ;WAIT FOR COMMAND TO COMPLETE
5579      JSR      PC,GET          ;GO READ REGISTERS VIA GET SUB
5580      BR       90$            ;GO TO 90$ IF NO ERROR
5581      NOP                      ;RETURN HERE IF ERROR
5582      ERROR    ;ERROR DEFINED BY GET SUB
5583      JMP      180$           ;GO TO 180$ IF ERROR WAS FOUND
5584      90$:
5585      ;VERIFY RESULTS OF WRITE COMMAND
5586      JSR      PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
5587      BR       100$          ;GO TO 100$ IF NO ERROR
5588      NOP                      ;RETURN HERE IF ERROR
5589      ERROR    ;ERROR # DEFINED BY PRIERR SUBROUTINE
5590      JSR      PC,2(SP)+      ;GO BACK FOR MORE ERROR CHECKS
5591      JMP      180$           ;GO TO 180$ IF ERROR WAS FOUND
5592      100$:
5593      JSR      PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
5594      BR       110$          ;GO TO 110$ IF NO ERROR
5595      NOP                      ;RETURN HERE IF ERROR
5596      ERROR    ;ERROR # DEFINED BY DTASTS SUBROUTINE
5597      JSR      PC,2(SP)+      ;GO BACK FOR MORE ERROR CHECKS
5598      JMP      180$           ;GO TO 180$ IF ERROR WAS FOUND
5599      110$:
5600      JSR      PC,SECERR      ;GO CHECK FOR SECONDARY ERRORS
5601      BR       120$          ;GO TO 120$ IF NO ERROR
5602      NOP                      ;RETURN HERE IF ERROR
5603      ERROR    ;ERROR # DEFINED BY SECERR SUBROUTINE
5604      JSR      PC,2(SP)+      ;GO BACK FOR MORE ERROR CHECKS
5605      JMP      180$           ;GO TO 180$ IF ERROR WAS FOUND
5606      120$:
5607      ;WRITE CHECK HEADER AND DATA FOR SECTORS JUST WRITTEN
5608      MOV      @WCH!GO, RMCS10 ;READ HEADER & DATA COMMAND
5609      JSR      PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5610      BR       130$          ;GO TO 130$ IF NO ERROR
5611      NOP                      ;RETURN HERE IF ERROR
5612      ERROR    ;ERROR DEFINED BY PUT SUB
5613      JMP      180$           ;GO TO 180$ IF ERROR WAS FOUND
5614      130$:
5615      ;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS
5616      JSR      PC,TIMOUT      ;WAIT FOR WRITE CHECK TO FINISH
5617      JSR      PC,GET          ;GO READ REGISTERS VIA GET SUB
5618      BR       140$          ;GO TO 140$ IF NO ERROR
5619      NOP                      ;RETURN HERE IF ERROR
5620      ERROR    ;ERROR DEFINED BY GET SUB
5621      JMP      180$           ;GO TO 180$ IF ERROR WAS FOUND
5622      140$:
5623      ;VERIFY THE RESULTS OF WRITE CHECK OPERATION
5624      JSR      PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
5625      BR       150$          ;GO TO 150$ IF NO ERROR
5626      NOP                      ;RETURN HERE IF ERROR
5627      ERROR    ;ERROR # DEFINED BY PRIERR SUBROUTINE
5628      JSR      PC,2(SP)+      ;GO BACK FOR MORE ERROR CHECKS
5629      JMP      180$           ;GO TO 180$ IF ERROR WAS FOUND
5630      150$:
5631      ;WRITE CHECK HEADER AND DATA FOR SECTORS JUST WRITTEN
5632      MOV      @WCH!GO, RMCS10 ;READ HEADER & DATA COMMAND
5633      JSR      PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5634      BR       130$          ;GO TO 130$ IF NO ERROR
5635      NOP                      ;RETURN HERE IF ERROR
5636      ERROR    ;ERROR DEFINED BY PUT SUB
5637      JMP      180$           ;GO TO 180$ IF ERROR WAS FOUND
5638      130$:
5639      ;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS
5640      JSR      PC,TIMOUT      ;WAIT FOR WRITE CHECK TO FINISH
5641      JSR      PC,GET          ;GO READ REGISTERS VIA GET SUB
5642      BR       140$          ;GO TO 140$ IF NO ERROR
5643      NOP                      ;RETURN HERE IF ERROR
5644      ERROR    ;ERROR DEFINED BY GET SUB
5645      JMP      180$           ;GO TO 180$ IF ERROR WAS FOUND
5646      140$:
5647      ;VERIFY THE RESULTS OF WRITE CHECK OPERATION
5648      JSR      PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
5649      BR       150$          ;GO TO 150$ IF NO ERROR
5650      NOP                      ;RETURN HERE IF ERROR
5651      ERROR    ;ERROR # DEFINED BY PRIERR SUBROUTINE
5652      JSR      PC,2(SP)+      ;GO BACK FOR MORE ERROR CHECKS
5653      JMP      180$           ;GO TO 180$ IF ERROR WAS FOUND
5654      150$:

```

```

5633 020762 104000 ERROR ;ERROR # DEFINED BY PRIERR SUBROUTINE
5634 020764 004736 JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
5635 020766 000137 021032 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
5636 020772 004737 053016 150$: JSR PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
5637 020776 000405 BR 160$ ;GO TO 160$ IF NO ERROR
5638 021000 000240 NOP ;RETURN HERE IF ERROR
5639 021002 104000 ERROR ;ERROR # DEFINED BY DTASTS SUBROUTINE
5640 021004 004736 JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
5641 021006 000137 021032 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
5642 021012 004737 041344 160$: JSR PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
5643 021016 000405 BR 170$ ;GO TO 170$ IF NO ERROR
5644 021020 000240 NOP ;RETURN HERE IF ERROR
5645 021022 104000 ERROR ;ERROR # DEFINED BY SECERR SUBROUTINE
5646 021024 004736 JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
5647 021026 000137 021032 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
5648 021032 170$:
5649 021032 180$: BR 200$
5650 021032 000401 190$: .WORD ;TEMPORARY STORAGE
5651 021032 000000 200$:
5652 021036
5653
5654
5655
5656
5657 ;*****
5658 ;TEST 20 FORMAT EACH SECTOR ADDRESS
5659 ;*****
5660 021036 000004 TST20:
5661 021040 000240 SCOPE ;SCOPE CALL
5662 021042 012706 001100 NOP ;START OF TEST
5663 021046 013700 001276 MOV #STACK,SP ;INITIALIZE STACK POINTER
5664 021052 013701 001450 MOV $BASE,RO ;RO=UNIBUS ADDRESS
5665 021056 012737 000020 001226 MOV TSTQUE,R1 ;R1 = DEVICE BEING TESTED
5666 021064 10$: MOV #20,$TESTN ;SET TEST NUMBER IN APT MAIL BOX
5667
5668 ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
5669 021064 012737 000000 001434 MOV #0,RMDCO ;CYLINDER = 0
5670 021072 012737 000000 001406 MOV #0,RMDAO ;TRACK = SECTOR = 0
5671 021100 012737 010000 001432 15$: MOV #FMT16,RMOFO ;16 BIT FORMAT
5672 021106 012737 177376 001402 MOV #(<C(2+256)+1>),RMWCO ;2 + 256 WORDS
5673 021114 012737 103566 001404 MOV #BUFONE,RMBAO ;DATA BUFFER ADDRESS
5674 021122 012737 000062 001400 MOV #WH,RMCSIO ;WRITE HEADER AND DATA
5675 021130 012737 001406 001174 MOV #RMDAO,$TMPD ;USE SECTOR FOR DATA PATTERN
5676 021136 012737 000001 001176 MOV #1,$TMP1
5677 021144 004737 036620 20$: JSR PC,GENBUF ;GO GENERATE DATA BUFFER
5678 021150
5679
5680 ;PREPARE DEVICE FOR DATA TRANSFER
5681 021150 004737 034000 30$: JSR PC,TSTPRP ;PREPARE DEVICE FOR TEST
5682 021154 154130 .WORD 154130 ;TASK DESCRIPTOR
5683 021156 000404 BR 30$ ;GO TO 30$ IF NO ERROR
5684 021160 000240 NOP ;RETURN HERE IF ERROR
5685 021162 104000 ERROR ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5686 021164 000137 021562 JMP 190$ ;GO TO 190$ IF ERROR WAS FOUND
5687 021170
5688

```

```

5689      021170 012737 000063 001400 ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
5690      021170 012737 000063      MOV      #RH!GO, RMCS10      ;WRITE HEADER AND DATA
5691      021176 012702 001535      MOV      #JTIMX, R2      ;WRITE REGISTER INDEX TABLE
5692      021202 112722 000006      MOV      #DA, (R2)+
5693      021206 112722 000034      MOV      #JC, (R2)+
5694      021212 112722 000032      MOV      #RMOF, (R2)+
5695      021216 112722 000002      MOV      #JC, (R2)+
5696      021222 112722 000004      MOV      #JA, (R2)+
5697      021226 112722 000000      MOV      #RMCS1, (R2)+
5698      021232 112722 000200      MOV      #200, (R2)+
5699      021236 004737 037766      JSR      PC, PUT      ;GO WRITE REGISTERS VIA PUT SUB
5700      021242 000404      BR      80$      ;GO TO 80$ IF NO ERROR
5701      021244 000240      NOP      ;RETURN HERE IF ERROR
5702      021246 104000      ERROR     ;ERROR DEFINED BY PUT SUB
5703      021250 000137 021562      JMP      190$      ;GO TO 190$ IF ERROR WAS FOUND
5704      021254
5705
5706      ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
5707      021254 004737 037432      JSR      PC, GETSTS
5708
5709      ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
5710      021260 004737 040326      JSR      PC, TIMEOUT      ;WAIT FOR COMMAND TO COMPLETE
5711      021264 004737 037516      JSR      PC, GET      ;GO READ REGISTERS VIA GET SUB
5712      021270 000404      BR      90$      ;GO TO 90$ IF NO ERROR
5713      021272 000240      NOP      ;RETURN HERE IF ERROR
5714      021274 104000      ERROR     ;ERROR DEFINED BY GET SUB
5715      021276 000137 021562      JMP      190$      ;GO TO 190$ IF ERROR WAS FOUND
5716      021302
5717
5718      ;VERIFY RESULTS OF WRITE COMMAND
5719      021302 004737 040512      JSR      PC, PRIERR      ;GO CHECK FOR PRIMARY ERRORS
5720      021306 000405      BR      100$      ;GO TO 100$ IF NO ERROR
5721      021310 000240      NOP      ;RETURN HERE IF ERROR
5722      021312 104000      ERROR     ;ERROR # DEFINED BY PRIERR SUBROUTINE
5723      021314 004736      JSR      PC, 2(SP)+      ;GO BACK FOR MORE ERROR CHECKS
5724      021316 000137 021562      JMP      190$      ;GO TO 190$ IF ERROR WAS FOUND
5725      021322
5726      021322 004737 053016      JSR      PC, DASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
5727      021326 000405      BR      110$      ;GO TO 110$ IF NO ERROR
5728      021330 000240      NOP      ;RETURN HERE IF ERROR
5729      021332 104000      ERROR     ;ERROR # DEFINED BY DASTS SUBROUTINE
5730      021334 004736      JSR      PC, 2(SP)+      ;GO BACK FOR MORE ERROR CHECKS
5731      021336 000137 021562      JMP      190$      ;GO TO 190$ IF ERROR WAS FOUND
5732      021342
5733      021342 004737 041344      JSR      PC, SECERR      ;GO CHECK FOR SECONDARY ERRORS
5734      021346 000405      BR      120$      ;GO TO 120$ IF NO ERROR
5735      021350 000240      NOP      ;RETURN HERE IF ERROR
5736      021352 104000      ERROR     ;ERROR # DEFINED BY SECERR SUBROUTINE
5737      021354 004736      JSR      PC, 2(SP)+      ;GO BACK FOR MORE ERROR CHECKS
5738      021356 000137 021562      JMP      190$      ;GO TO 190$ IF ERROR WAS FOUND
5739      021362
5740
5741      ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
5742
5743      021362 012737 000073 001400      MOV      #RH!GO, RMCS10      ;READ HEADER & DATA COMMAND
5744      021370 012737 104572 001404      MOV      #BUFTWO, RMA0      ;CHANGE BUS ADDRESS

```

5745	021376	004737	037766	JSR	PC PUT	;GO WRITE REGISTERS VIA PUT SUB
5746	021402	000404		BR	130\$;GO TO 130\$ IF NO ERROR
5747	021404	001240		NOP		;RETURN HERE IF ERROR
5748	021406	104000		ERROR		;ERROR DEFINED BY PUT SUB
5749	021410	000137	021562	JMP	190\$;GO TO 190\$ IF ERROR WAS FOUND
5750	021414					
5751						
5752						
5753	021414	004737	040326			
5754	021420	004737	0375 6	JSR	PC TIMEOUT	;WAIT FOR READ TO COMPLETE
5755	021424	000404		JSR	PC GET	;GO READ REGISTERS VIA GET SUB
5756	021426	000240		BR	140\$;GO TO 140\$ IF NO ERROR
5757	021430	104000		NOP		;RETURN HERE IF ERROR
5758	021432	000137	021562	ERROR		;ERROR DEFINED BY GET SUB
5759	021436			JMP	190\$;GO TO 190\$ IF ERROR WAS FOUND
5760						
5761						
5762	021436	004737	040512			
5763	021442	000405		JSR	PC PRIERR	;GO CHECK FOR PRIMARY ERRORS
5764	021444	000240		BR	150\$;GO TO 150\$ IF NO ERROR
5765	021446	104000		NOP		;RETURN HERE IF ERROR
5766	021450	004736		ERROR		;ERROR # DEFINED BY PRIERR SUBROUTINE
5767	021452	000137	021562	JSR	PC 2(SP)+	;GO BACK FOR MORE ERROR CHECKS
5768	021456			JMP	190\$;GO TO 190\$ IF ERROR WAS FOUND
5769	021456	004737	053016			
5770	021462	000405		JSR	PC DTASTS	;GO VERIFY RESULTS OF DATA TRANSFER
5771	021464	000240		BR	160\$;GO TO 160\$ IF NO ERROR
5772	021466	104000		NOP		;RETURN HERE IF ERROR
5773	021470	004736		ERROR		;ERROR # DEFINED BY DTASTS SUBROUTINE
5774	021472	000137	021562	JSR	PC 2(SP)+	;GO BACK FOR MORE ERROR CHECKS
5775	021476			JMP	190\$;GO TO 190\$ IF ERROR WAS FOUND
5776	021476	004737	041344			
5777	021502	000405		JSR	PC SECERR	;GO CHECK FOR SECONDARY ERRORS
5778	021504	000240		BR	170\$;GO TO 170\$ IF NO ERROR
5779	021506	104000		NOP		;RETURN HERE IF ERROR
5780	021510	004736		ERROR		;ERROR # DEFINED BY SECERR SUBROUTINE
5781	021512	000137	021562	JSR	PC 2(SP)+	;GO BACK FOR MORE ERROR CHECKS
5782	021516			JMP	190\$;GO TO 190\$ IF ERROR WAS FOUND
5783						
5784						
5785	021516	004737	037064			
5786	021522	103566		JSR	PC CMPBUF	;GO COMPARE WRITE, READ DATA BUFFERS
5787	021524	104572		.WORD	BUFONE	;STARTING ADDRESS OF WRITE BUFFER
5788	021526	000404		.WORD	BUFTWO	;STARTING ADDRESS OF READ BUFFER
5789	021530	000240		BR	180\$;GO TO 180\$ IF NO ERROR
5790	021532	104000		NOP		;RETURN HERE IF ERROR
5791	021534	000137	021562	ERROR		;ERROR # DEFINED BY CMPBUF SUBROUTINE
5792	021540			JMP	190\$;GO TO 190\$ IF ERROR WAS FOUND
5793						
5794						
5795	021540	062737	000001 001406			
5796	021546	122737	000037 001406	ADD	#1,RMDAO	;ADVANCE SECTOR COUNT
5797	021554	103402		CMPB	#31,RMDAO	;DONE ALL SECTORS??
5798	021556	000137	021100	BLO	190\$;YES!!
5799	021562			JMP	15\$;GO DO NEXT SECTOR
5800						

C10

```

5801 021562 000004 001100
5802 021562 000240 001276
5803 021564 012706 001450
5804 021564 012706 000021 001226
5805 021562 013700 001276
5806 021564 013701 001450
5807 021564 012737 000021 001226
5808 021572 013700 001276
5809 021576 013701 001450
5810 021602 012737 000021 001226
5811 021610
5812
5813
5814 021610 012737 000000 001434
5815 021616 012737 000000 001406
5816 021624 012737 010000 001432
5817 021632 012737 177376 001402
5818 021640 012737 1C7566 001404
5819 021646 012737 000062 001400
5820 021654 012737 001436 001174
5821 021662 012737 000001 001176
5822 021670 004737 036620
5823 021674
5824
5825
5826 021674 004737 034000
5827 021700 154130
5828 021702 000404
5829 021704 000240
5830 021706 104000
5831 021710 000137 022306
5832 021714
5833
5834
5835 021714 012737 000063 001400
5836 021722 012702 001535
5837 021726 112722 000006
5838 021732 112722 000034
5839 021736 112722 000032
5840 021742 112722 000002
5841 021746 112722 000004
5842 021752 112722 000000
5843 021756 112722 000200
5844 021762 004737 037766
5845 021766 000404
5846 021770 000240
5847 021772 104000
5848 021774 000137 022306
5849 022000
5850
5851
5852 022000 004737 037432
5853
5854
5855 022004 004737 040326
5856 022010 004737 037516

;*****
;TEST 21 FORMAT EACH TRACK ADDRESS
;*****
TST21:
    SCOPE
    NOP
    MOV #STACK, SP
    MOV $BASE, R0
    MOV TSTACK, R1
    MOV #21, $TESTN
;SCOPE CALL
;START OF TEST
;INITIALIZE STACK POINTER
;R0=UNIBUS ADDRESS
;(R1) = DEVICE BEING TESTED
;;SET TEST NUMBER IN APT MAIL BOX

10$:
;SETUP PARAMETERS FOR GENERATING DATA BUFFER
    MOV #0, RMD0
    MOV #0, RMDA0
    MOV #FAT16, RMOF0
    MOV #(<C(2+256.)+1>), RMLC0
    MOV #BUFONE, RMB0
    MOV #RMC510
    MOV #RMDA0, $TMP0
    MOV #1, $TMP1
    JSR PC, GENBUF
;CYLINDER = 0
;TRACK = SECTOR = 0
;16 BIT FORMAT
;2 + 256 WORDS
;DATA BUFFER ADDRESS
;WRITE HEADER AND DATA
;USE TRACK FOR DATA PATTERN

20$:
;PREPARE DEVICE FOR DATA TRANSFER
    JSR PC, TSTPRP
    WORD 154130
    BR 30$
;PREPARE DEVICE FOR TEST
;TASK DESCRIPTOR
;GO TO 30$ IF NO ERROR
;RETURN HERE IF ERROR
;ERROR # DEFINED BY TSTPRP SUBROUTINE
;GO TO 190$ IF ERROR WAS FOUND

30$:
;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
    MOV #RMC510
    MOV #PUTINX, R2
    MOV #RMDA, (R2)+
    MOV #RMDC, (R2)+
    MOV #RMOF, (R2)+
    MOV #RMLC, (R2)+
    MOV #RMB0, (R2)+
    MOV #RMC51, (R2)+
    MOV #200, (R2)+
    JSR PC, PUT
    BR 80$
;GO WRITE REGISTERS VIA PUT SUB
;GO TO 80$ IF NO ERROR
;RETURN HERE IF ERROR
;ERROR DEFINED BY PUT SUB
;GO TO 190$ IF ERROR WAS FOUND

80$:
;SETUP INPUT REGISTER BUFFER FOR READING STATUS
    JSR PC, GETSTS

;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
    JSR PC, TIMEOUT
    JSR PC, GET
;WAIT FOR COMMAND TO COMPLETE
;GO READ REGISTERS VIA GET SUB

```

```
5857 022014 000404 BR 90$ ;GO TO 90$ IF NO ERROR
5858 022016 000240 NOP ;RETURN HERE IF ERROR
5859 022020 104000 ERROR ;ERROR DEFINED BY GET SUB
5860 022022 000137 022306 JMP 190$ ;GO TO 190$ IF ERROR WAS FOUND
5861 022026 90$:
5862 ;VERIFY RESULTS OF WRITE COMMAND
5863 JSR PC, PRIERR ;GO CHECK FOR PRIMARY ERRORS
5864 022026 004737 040512 BR 100$ ;GO TO 100$ IF NO ERROR
5865 022032 000405 NOP ;RETURN HERE IF ERROR
5866 022034 000240 ERROR ;ERROR # DEFINED BY PRIERR SUBROUTINE
5867 022036 104000 JSR PC, 2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
5868 022040 004736 JMP 190$ ;GO TO 190$ IF ERROR WAS FOUND
5869 022042 000137 022306 100$:
5870 022046 JSR PC, DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
5871 022046 004737 053016 BR 110$ ;GO TO 110$ IF NO ERROR
5872 022052 000405 NOP ;RETURN HERE IF ERROR
5873 022054 000240 ERROR ;ERROR # DEFINED BY DTASTS SUBROUTINE
5874 022056 104000 JSR PC, 2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
5875 022060 004736 JMP 190$ ;GO TO 190$ IF ERROR WAS FOUND
5876 022062 000137 022306 110$:
5877 022066 JSR PC, SECERR ;GO CHECK FOR SECONDARY ERRORS
5878 022066 004737 041344 BR 120$ ;GO TO 120$ IF NO ERROR
5879 022072 000405 NOP ;RETURN HERE IF ERROR
5880 022074 000240 ERROR ;ERROR # DEFINED BY SECERR SUBROUTINE
5881 022076 104000 JSR PC, 2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
5882 022100 004736 JMP 190$ ;GO TO 190$ IF ERROR WAS FOUND
5883 022102 000137 022306 120$:
5884 022106 ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
5885 MOV #RM!GO, RMCS10 ;READ HEADER & DATA COMMAND
5886 MOV #BUFTWO, RMBAD ;CHANGE BUS ADDRESS
5887 JSR PC, PUT ;GO WRITE REGISTERS VIA PUT SUB
5888 022106 012737 000073 001400 BR 130$ ;GO TO 130$ IF NO ERROR
5889 022114 012737 104572 001404 NOP ;RETURN HERE IF ERROR
5890 022122 004737 037766 ERROR ;ERROR DEFINED BY PUT SUB
5891 022126 000404 JMP 190$ ;GO TO 190$ IF ERROR WAS FOUND
5892 022130 000240 130$:
5893 022132 104000 ;WAIT FOR READ TO COMPLETE AND GET STATUS
5894 022134 000137 022306 JSR PC, TIMEOUT ;WAIT FOR READ TO COMPLETE
5895 022140 JSR PC, GET ;GO READ REGISTERS VIA GET SUB
5896 022140 004737 040326 BR 140$ ;GO TO 140$ IF NO ERROR
5897 022144 004737 037516 NOP ;RETURN HERE IF ERROR
5898 022150 000404 ERROR ;ERROR DEFINED BY GET SUB
5899 022152 000240 JMP 190$ ;GO TO 190$ IF ERROR WAS FOUND
5900 022154 104000 140$:
5901 022156 000137 022306 ;VERIFY THE RESULTS OF READ OPERATION
5902 022162 004737 040512 JSR PC, PRIERR ;GO CHECK FOR PRIMARY ERRORS
5903 022166 000405 BR 150$ ;GO TO 150$ IF NO ERROR
5904 022170 000240 NOP ;RETURN HERE IF ERROR
5905 022172 104000 ERROR ;ERROR # DEFINED BY PRIERR SUBROUTINE
5906 022174 004736 JSR PC, 2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
5907 022176 000137 022306 JMP 190$ ;GO TO 190$ IF ERROR WAS FOUND
```

```

5913 022202 004737 053016 150$: JSR PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
5914 022202 000405 BR 160$ ;GO TO 160$ IF NO ERROR
5915 022202 000240 NOP ;RETURN HERE IF ERROR
5916 022210 104000 ERROR ;ERROR # DEFINED BY DTASTS SUBROUTINE
5917 022212 004736 JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
5918 022214 000137 JMP 190$ ;GO TO 190$ IF ERROR WAS FOUND
5919 022216 004737 022306 160$: JSR PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
5920 022222 000405 BR 170$ ;GO TO 170$ IF NO ERROR
5921 022222 000240 NOP ;RETURN HERE IF ERROR
5922 022230 104000 ERROR ;ERROR # DEFINED BY SECERR SUBROUTINE
5923 022232 004736 JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
5924 022234 000137 JMP 190$ ;GO TO 190$ IF ERROR WAS FOUND
5925 022236 000137 022306 170$: ;VERIFY DATA
5926 022242 JSR PC,CMPBUF ;GO COMPARE WRITE, READ DATA BUFFERS
5927 022242 103566 .WORD BUFONE ;STARTING ADDRESS OF WRITE BUFFER
5928 022242 104572 .WORD BUFTWO ;STARTING ADDRESS OF READ BUFFER
5929 022242 000404 BR 180$ ;GO TO 180$ IF NO ERROR
5930 022242 000240 NOP ;RETURN HERE IF ERROR
5931 022242 104000 ERROR ;ERROR # DEFINED BY CMPBUF SUBROUTINE
5932 022242 000137 JMP 190$ ;GO TO 190$ IF ERROR WAS FOUND
5933 022242 000137 022306 180$: ;INCREMENT ADDRESS AND FORMAT NEXT TRACK
5934 022242 062737 000400 001406 ADD #400,RMDAO ;ADVANCE TRACK COUNT
5935 022242 022737 002000 001406 CMP #2000,RMDAO ;DONE ALL TRACKS??
5936 022242 103402 BLO 190$ ;YES!!
5937 022242 000137 021624 JMP 15$ ;GO DO NEXT SECTOR
5938 022242 000137 190$:
5939 022242 *****
5940 022242 ;TEST 22 FORMAT PRIME CYLINDERS
5941 022242 *****
5942 022242 TST22:
5943 022242 SCOPE ;SCOPE CALL
5944 022242 NOP ;START OF TEST
5945 022242 000004 MOV #STACK,SP ;INITIALIZE STACK POINTER
5946 022242 000240 MOV $BASE,R0 ;R0=UNIBUS ADDRESS
5947 022242 012706 001100 MOV TSTQUE,R1 ;(R1) = DEVICE BEING TESTED
5948 022242 013700 001276 MOV #22,$TESTN ;SET TEST NUMBER IN APT MAIL BOX
5949 022242 012737 000022 001226 10$: ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
5950 022242 MOV #1,RMDCO ;CYLINDER = 0
5951 022242 MOV #0,RMDAO ;TRACK = SECTOR = 0
5952 022242 012737 010000 001432 15$: MOV #FMT16,RMOFO ;16 BIT FORMAT
5953 022242 012737 177376 001402 MOV #(<1C<2+256.>+1),RMWCO ;2 + 256 WORDS
5954 022242 012737 103566 001404 MOV #BUFONE,RMBAO ;DATA BUFFER ADDRESS
5955 022242 012737 000062 001400 MOV #WH,RMC$10 ;WRITE HEADER AND DATA
5956 022242 012737 001434 001174 MOV #RMDCO,$TMP0 ;USE CYLINDER FOR DATA PATTERN
5957 022242 012737 000001 001176 MOV #1,$TMP1
5958 022242 004737 036620 JSR PC,GENBUF ;GO GENERATE DATA BUFFER
5959 022242 000001 001434 20$:
5960 022242 012737 000000 001406
5961 022242 012737 010000 001432
5962 022242 012737 177376 001402
5963 022242 012737 103566 001404
5964 022242 012737 000062 001400
5965 022242 012737 001434 001174
5966 022242 012737 000001 001176
5967 022242 004737 036620
5968 022242

```

```

5969
5970
5971 022420 004737 034000
5972 022424 154130
5973 022426 000404
5974 022430 000240
5975 022432 104000
5976 022434 000137 023030
5977 022440
5978
5979
5980 022440 012737 000063 001400
5981 022446 012702 001535
5982 022452 112722 000006
5983 022456 112722 000034
5984 022462 112722 000032
5985 022466 112722 000002
5986 022472 112722 000004
5987 022476 112722 000000
5988 022502 112722 000200
5989 022506 004737 037766
5990 022512 000404
5991 022514 000240
5992 022516 104000
5993 022520 000137 023030
5994 022524
5995
5996
5997 022524 004737 037432
5998
5999
6000 022530 004737 040326
6001 022534 004737 037516
6002 022540 000404
6003 022542 000240
6004 022544 104000
6005 022546 000137 023030
6006 022552
6007
6008
6009 022552 004737 040512
6010 022556 000405
6011 022560 000240
6012 022562 104000
6013 022564 004736
6014 022566 000137 023030
6015 022572
6016 022576 004737 053016
6017 022576 000405
6018 022600 000240
6019 022602 104000
6020 022604 004736
6021 022606 000137 023030
6022 022612
6023 022612 004737 041344
6024 022616 000405

;PREPARE DEVICE FOR DATA TRANSFER
JSR PC,TSTPRP ;PREPARE DEVICE FOR TEST
WORD 154130 ;TASK DESCRIPTOR
BR 20$ ;GO TO 30$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR # DEFINED BY TSTPRP SUBROUTINE
JMP 190$ ;GO TO 190$ IF ERROR WAS FOUND

30$:
;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
MOV #WH,GO,RMCS10 ;WRITE HEADER AND DATA
MOV #PUT,INX,R2 ;WRITE REGISTER INDEX TABLE
MOVB #RMDA,(R2)+
MOVB #RMDC,(R2)+
MOVB #RMOF,(R2)+
MOVB #RMLC,(R2)+
MOVB #R13A,(R2)+
MOVB #RMCS1,(R2)+
MOVB #200,(R2)+
JSR PC,PUT ;GO WRITE REGISTERS VIA PUT SUB
BR 80$ ;GO TO 80$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR DEFINED BY PUT SUB
JMP 190$ ;GO TO 190$ IF ERROR WAS FOUND

80$:
;SETUP INPUT REGISTER BUFFER FOR READING STATUS
JSR PC,GETSTS

;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
JSR PC,TIMOUT ;WAIT FOR COMMAND TO COMPLETE
JSR PC,GET ;GO READ REGISTERS VIA GET SUB
BR 90$ ;GO TO 90$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR DEFINED BY GET SUB
JMP 190$ ;GO TO 190$ IF ERROR WAS FOUND

90$:
;VERIFY RESULTS OF WRITE COMMAND
JSR PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
BR 100$ ;GO TO 100$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR # DEFINED BY PRIERR SUBROUTINE
JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
JMP 190$ ;GO TO 190$ IF ERROR WAS FOUND

100$:
JSR PC,DASTS ;GO VERIFY RESULTS OF DATA TRANSFER
BR 110$ ;GO TO 110$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR # DEFINED BY DASTS SUBROUTINE
JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
JMP 190$ ;GO TO 190$ IF ERROR WAS FOUND

110$:
JSR PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
BR 120$ ;GO TO 120$ IF NO ERROR

```

```

6025 022620 000240      NOP      ;RETURN HERE IF ERROR
6026 022622 104000      ERROR     ;ERROR # DEFINED BY SECERR SUBROUTINE
6027 022624 004736      JSR      PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
6028 022626 000137 023030 JMP      190$ ;GO TO 190$ IF ERROR WAS FOUND
6029 022632
6030
6031
6032 ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
6033 022632 012737 000073 001400 MOV      #RH:GO,RMC510 ;READ HEADER & DATA COMMAND
6034 022640 012737 104572 001404 MOV      #BUFTWO,RMB40 ;CHANGE BUS ADDRESS
6035 022646 004737 037766 JSR      PC,PUT ;GO WRITE REGISTERS VIA PUT SUB
6036 022652 000404 BR      130$ ;GO TO 130$ IF NO ERROR
6037 022654 000240      NOP      ;RETURN HERE IF ERROR
6038 022656 104000      ERROR     ;ERROR DEFINED BY PUT SUB
6039 022660 000137 023030 JMP      190$ ;GO TO 190$ IF ERROR WAS FOUND
6040 022664
6041
6042 ;WAIT FOR READ TO COMPLETE AND GET STATUS
6043 022664 004737 040326 JSR      PC,TIMOUT ;WAIT FOR READ TO COMPLETE
6044 022670 004737 037516 JSR      PC,GET ;GO READ REGISTERS VIA GET SUB
6045 022674 000404 BR      140$ ;GO TO 140$ IF NO ERROR
6046 022676 000240      NOP      ;RETURN HERE IF ERROR
6047 022700 104000      ERROR     ;ERROR DEFINED BY GET SUB
6048 022702 000137 023030 JMP      190$ ;GO TO 190$ IF ERROR WAS FOUND
6049 022706
6050
6051 ;VERIFY THE RESULTS OF READ OPERATION
6052 022706 004737 040512 JSR      PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
6053 022712 000405 BR      150$ ;GO TO 150$ IF NO ERROR
6054 022714 000240      NOP      ;RETURN HERE IF ERROR
6055 022716 104000      ERROR     ;ERROR # DEFINED BY PRIERR SUBROUTINE
6056 022720 004736 JSR      PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
6057 022722 000137 023030 JMP      190$ ;GO TO 190$ IF ERROR WAS FOUND
6058 022726
6059 022726 004737 053016 JSR      PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
6060 022732 000405 BR      160$ ;GO TO 160$ IF NO ERROR
6061 022734 000240      NOP      ;RETURN HERE IF ERROR
6062 022736 104000      ERROR     ;ERROR # DEFINED BY DTASTS SUBROUTINE
6063 022740 004736 JSR      PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
6064 022742 000137 023030 JMP      190$ ;GO TO 190$ IF ERROR WAS FOUND
6065 022746
6066 022746 004737 041344 JSR      PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
6067 022752 000405 BR      170$ ;GO TO 170$ IF NO ERROR
6068 022754 000240      NOP      ;RETURN HERE IF ERROR
6069 022756 104000      ERROR     ;ERROR # DEFINED BY SECERR SUBROUTINE
6070 022760 004736 JSR      PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
6071 022762 000137 023030 JMP      190$ ;GO TO 190$ IF ERROR WAS FOUND
6072 022766
6073
6074 ;VERIFY DATA
6075 022766 004737 037064 JSR      PC,CMPBUF ;GO COMPARE WRITE, READ DATA BUFFERS
6076 022772 103566 .WORD BUFO1E ;STARTING ADDRESS OF WRITE BUFFER
6077 022774 104572 .WORD BUFTWO ;STARTING ADDRESS OF READ BUFFER
6078 022776 000404 BR      180$ ;GO TO 180$ IF NO ERROR
6079 023000 000240      NOP      ;RETURN HERE IF ERROR
6080 023002 104000      ERROR     ;ERROR # DEFINED BY CMPBUF SUBROUTINE

```

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 124
T22 FORMAT PRIME CYLINDERS

SEQ 0124

```

6081 023004 000137 023030      JMP      190$      ;GO TO 190$ IF ERROR WAS FOUND
6082 023010
6083
6084      ;INCREMENT ADDRESS AND FORMAT NEXT PRIME CYLINDER
6085 023010 006337 001434      ASL      RMDCO      ;ADVANCE CYLINDER COUNT
6086 023014 022737 001466 001434      CMP      #822.,RMDCO      ;DONE ALL CYLINDERS??
6087 023022 103402      BLO      190$      ;YES!!
6088 023024 000137 022350      JMP      15$      ;GO DO NEXT CYLINDER
6089 023030
6090
6091      ;*****
6092      ;*TEST 23      FORMAT LAST SECTOR
6093      ;*****
6094      TST23:
6095 023030 000004      SCOPE      ;SCOPE CALL
6096 023032 000240      NOP      ;START OF TEST
6097 023034 012706 001100      MOV      #STACK,SP      ;INITIALIZE STACK POINTER
6098 023040 013700 001276      MOV      $BASE,R0      ;R0=UNIBUS ADDRESS
6099 023044 013701 001450      MOV      TSTQUE,R1      ;(R1) = DEVICE BEING TESTED
6100 023050 012737 000023 001226      MOV      #23,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6101 023056
6102
6103      ;PREPARE DEVICE FOR DATA TRANSFER
6104 023056 004737 034000      JSR      PC,TSTPRP      ;PREPARE DEVICE FOR TEST
6105 023062 154130      WORD      154130 ;TASK DESCRIPTOR
6106 023064 000404      BR      30$      ;GO TO 30$ IF NO ERROR
6107 023066 000240      NOP      ;RETURN HERE IF ERROR
6108 023070 104000      ERROR      ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6109 023072 000137 023326      JMP      190$      ;GO TO 190$ IF ERROR WAS FOUND
6110 023076
6111
6112      ;SETUP PARAMETERS FOR READING LAST SECTOR
6113 023076 012737 001466 001434      MOV      #822.,RMDCO      ;LAST CYLINDER
6114 023104 012737 002037 001406      MOV      #2037,RMDAO      ;TRACK = 4 SECTOR = 31.
6115 023112 012737 010000 001432      MOV      #FMT16,RMOFO      ;16 BIT FORMAT
6116 023120 012737 177376 001402      MOV      #(<1C(2+256.)+1),RMWCO      ;2 + 256 WORDS
6117 023126 012737 104572 001404      MOV      #BUFTWO,RMBAO      ;DATA BUFFER ADDRESS
6118 023134 012737 000073 001400      MOV      #RH!GO,RMCS10      ;WRITE HEADER AND DATA
6119
6120      MOV      #PUTINX,R2      ;WRITE REGISTER INDEX TABLE
6121 023146 112722 000006      MOV      #RMDA,(R2)+
6122 023152 112722 000034      MOV      #RMDC,(R2)+
6123 023156 112722 000032      MOV      #RMOF,(R2)+
6124 023162 112722 000004      MOV      #RMBA,(R2)+
6125 023166 112722 000002      MOV      #RMWC,(R2)+
6126 023172 112722 000000      MOV      #RMCS1,(R2)+
6127 023176 112722 000200      MOV      #200,(R2)+
6128
6129
6130 023202      120$:
6131
6132      ;READ HEADER AND DATA OF LAST SECTOR
6133 023202 004737 037766      JSR      PC,PUT      ;GO WRITE REGISTERS VIA PUT SUB
6134 023206 000404      BR      130$      ;GO TO 130$ IF NO ERROR
6135 023210 000240      NOP      ;RETURN HERE IF ERROR
6136 023212 104000      ERROR      ;ERROR DEFINED BY PUT SUB

```

```

6137 023214 000137 023326      JMP      190$      ;GO TO 190$ IF ERROR WAS FOUND
6138 023220
6139
6140      ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6141 023220 004737 037432      JSR      PC,GETSTS
6142      ;WAIT FOR READ TO COMPLETE AND GET STATUS
6143 023224 004737 040326      JSR      PC,TIMOUT      ;WAIT FOR READ TO COMPLETE
6144 023230 004737 037516      JSR      PC,GET      ;GO READ REGISTERS VIA GET SUB
6145 023234 000404      BR      140$      ;GO TO 140$ IF NO ERROR
6146 023236 000240      NOP      ;RETURN HERE IF ERROR
6147 023240 104000      ERROR     ;ERROR DEFINED BY GET SUB
6148 023242 000137 023326      JMP      190$      ;GO TO 190$ IF ERROR WAS FOUND
6149 023246
6150
6151      ;VERIFY THE RESULTS OF READ OPERATION
6152 023246 004737 040512      JSR      PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
6153 023252 000405      BR      150$      ;GO TO 150$ IF NO ERROR
6154 023254 000240      NOP      ;RETURN HERE IF ERROR
6155 023256 104000      ERROR     ;ERROR # DEFINED BY PRIERR SUBROUTINE
6156 023260 004736      JSR      PC,2(SP)+      ;GO BACK FOR MORE ERROR CHECKS
6157 023262 000137 023326      JMP      190$      ;GO TO 190$ IF ERROR WAS FOUND
6158 023266
6159 023266 004737 053016      JSR      PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
6160 023272 000405      BR      160$      ;GO TO 160$ IF NO ERROR
6161 023274 000240      NOP      ;RETURN HERE IF ERROR
6162 023276 104000      ERROR     ;ERROR # DEFINED BY DTASTS SUBROUTINE
6163 023300 004736      JSR      PC,2(SP)+      ;GO BACK FOR MORE ERROR CHECKS
6164 023302 000137 023326      JMP      190$      ;GO TO 190$ IF ERROR WAS FOUND
6165 023306
6166 023306 004737 041344      JSR      PC,SECERR      ;GO CHECK FOR SECONDARY ERRORS
6167 023312 000405      BR      170$      ;GO TO 170$ IF NO ERROR
6168 023314 000240      NOP      ;RETURN HERE IF ERROR
6169 023316 104000      ERROR     ;ERROR # DEFINED BY SECERR SUBROUTINE
6170 023320 004736      JSR      PC,2(SP)+      ;GO BACK FOR MORE ERROR CHECKS
6171 023322 000137 023326      JMP      190$      ;GO TO 190$ IF ERROR WAS FOUND
6172 023326
6173
6174 023326
6175
6176
6177
6178
6179 023326
6180 023326 000004
6181 023330 000240
6182 023332 012706 001100
6183 023336 013700 001276
6184 023342 013701 001450
6185 023346 012737 000024 001226
6186
6187
6188 023354 004737 034000
6189 023360 154130
6190 023362 000404
6191 023364 000240
6192 023366 104000

      ;*****
      ;TEST 24      FORMAT W/ AOE ERROR
      ;*****
      TST24:
      SCOPE      ;SCOPE CALL
      NOP      ;START OF TEST
      MOV      #STACK,SP      ;INITIALIZE STACK POINTER
      MOV      $BASE,R0      ;R0=UNIBUS ADDRESS
      MOV      TSTQUE,R1      ;(R1) = DEVICE BEING TESTED
      MOV      #24,$TESTN      ;SET TEST NUMBER IN APT MAIL BOX

      ;PREPARE DEVICE FOR DATA TRANSFER
      JSR      PC,TSTPRP      ;PREPARE DEVICE FOR TEST
      .WORD    154130      ;TASK DESCRIPTOR
      BR      30$      ;GO TO 30$ IF NO ERROR
      NOP      ;RETURN HERE IF ERROR
      ERROR     ;ERROR # DEFINED BY TSTPRP SUBROUTINE

```

6193 023370 000137 023624
6194 023374
6195
6196
6197 023374 012737 001466 001434
6198 023402 012737 002037 001406
6199 023410 012737 010000 001432
6200 023416 012737 176774 001402
6201 023424 012737 104572 001404
6202 023432 012737 000073 001400
6203 023440 012702 001535
6204 023444 112722 000006
6205 023450 112722 000034
6206 023454 112722 000032
6207 023460 112722 000002
6208 023464 112722 000004
6209 023470 112722 000000
6210 023474 112722 000200
6211 023500
6212
6213 023500
6214 023500 004737 037766
6215 023504 000404
6216 023506 000240
6217 023510 104000
6218 023512 000137 023624
6219 023516
6220
6221
6222 023516 004737 037432
6223
6224
6225 023522 004737 040326
6226 023526 004737 037516
6227 023532 000404
6228 023534 000240
6229 023536 104000
6230 023540 000137 023624
6231 023544
6232
6233
6234 023544 004737 040512
6235 023550 000405
6236 023552 000240
6237 023554 104000
6238 023556 004736
6239 023560 000137 023624
6240 023564
6241 023570 004737 053016
6242 023572 000405
6243 023574 000240
6244 023576 104000
6245 023578 004736
6246 023600 000137 023624
6247 023604
6248 023604 004737 041344

30\$: JMP 190\$;GO TO 190\$ IF ERROR WAS FOUND
;SETUP PARAMETERS FOR READING 2 SECTORS STARTING WITH LAST SECTOR
MOV #822, RMDC0 ;START AT LAST CYLINDER
MOV #2037, RMDA0 ;TRACK = 4 SECTOR = 31.
MOV #FMT16, RMFO0 ;16 BIT FORMAT
MOV #(<C<<2+256.>*2)+1, RMWCO ;FORMAT 2 SECTORS
MOV #BUFTWO, RMBA0 ;DATA BUFFER ADDRESS
MOV #RH!GO, RMCS10 ;WRITE HEADER AND DATA
MOV #PUTINX, R2 ;WRITE REGISTER INDEX TABLE
MOVB #RMDA, (R2)+
MOVB #RMDC, (R2)+
MOVB #RMFO, (R2)+
MOVB #RMWC, (R2)+
MOVB #RMBA, (R2)+
MOVB #RMCS1, (R2)+
MOVB #200, (R2)+
80\$:
125\$: JSR PC, PUT ;GO WRITE REGISTERS VIA PUT SUB
BR 130\$;GO TO 130\$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR DEFINED BY PUT SUB
JMP 190\$;GO TO 190\$ IF ERROR WAS FOUND
130\$:
;SETUP INPUT REGISTER BUFFER FOR READING STATUS
JSR PC, GETSTS
;WAIT FOR READ TO COMPLETE AND GET STATUS
JSR PC, TIMEOUT ;WAIT FOR READ TO COMPLETE
JSR PC, GET ;GO READ REGISTERS VIA GET SUB
BR 140\$;GO TO 140\$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR DEFINED BY GET SUB
JMP 190\$;GO TO 190\$ IF ERROR WAS FOUND
140\$:
;VERIFY THE RESULTS OF READ OPERATION
JSR PC, PRIERR ;GO CHECK FOR PRIMARY ERRORS
BR 150\$;GO TO 150\$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR # DEFINED BY PRIERR SUBROUTINE
JSR PC, 2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
JMP 190\$;GO TO 190\$ IF ERROR WAS FOUND
150\$: JSR PC, DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
BR 160\$;GO TO 160\$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR # DEFINED BY DTASTS SUBROUTINE
JSR PC, 2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
JMP 190\$;GO TO 190\$ IF ERROR WAS FOUND
160\$: JSR PC, SECERR ;GO CHECK FOR SECONDARY ERRORS


```

6249 023610 000405 BR 170$ ;GO TO 170$ IF NO ERROR
6250 023612 000240 NOP ;RETURN HERE IF ERROR
6251 023614 104000 ERROR ;ERROR # DEFINED BY SECERR SUBROUTINE
6252 023616 004736 JSR PC,3(SP)+ ;GO BACK FOR MORE ERROR CHECKS
6253 023620 000137 023624 JMP 190$ ;GO TO 190$ IF ERROR WAS FOUND
6254 023624 170$:
6255 023624 190$:
6256 023624
6257
6258 ;*****
6259 ;TEST 25 FORMAT INVALID SECTOR ADDRESS
6260 ;*****
6261 023624 TST25:
6262 023624 000004 SCOPE ;SCOPE CALL
6263 023626 000240 NOP ;START OF TEST
6264 023630 012706 001100 MOV #STACK,SP ;INITIALIZE STACK POINTER
6265 023634 013700 001276 MOV $BASE,R0 ;R0=UNIBUS ADDRESS
6266 023640 013701 001450 MOV TSTQUE,R1 ;(R1) = DEVICE BEING TESTED
6267 023644 012737 000025 001226 MOV #25,$TESTN ;SET TEST NUMBER IN APT MAIL BOX
6268 023652 10$:
6269
6270 ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
6271 023652 012737 000000 001434 MOV #0,RMDC0 ;CYLINDER = 0
6272 023660 012737 000036 001406 MOV #30,RMDA0 ;TRACK = 0 SECTOR = 30
6273 023666 012737 000000 001432 15$: MOV #0,RMOF0 ;18 BIT FORMAT
6274 023674 012737 177376 001402 MOV #C(2+256.)+1,RMWC0 ;2 + 256 WORDS
6275 023702 012737 103566 001404 MOV #BUFONE,RMBA0 ;DATA BUFFER ADDRESS
6276 023710 012737 000062 001400 MOV #WH,RMCS10 ;WRITE HEADER AND DATA
6277 023716 012737 001406 001174 MOV #RMDA0,$TMP0 ;USE SECTOR FOR DATA PATTERN
6278 023724 012737 000001 001176 MOV #1,$TMP1
6279 023732 004737 036620 JSR PC,GENBUF ;GO GENERATE DATA BUFFER
6280 023736 20$:
6281
6282 ;PREPARE DEVICE FOR DATA TRANSFER
6283 023736 004737 034000 JSR PC,TSTPRP ;PREPARE DEVICE FOR TEST
6284 023742 154130 .WORD 154130 ;TASK DESCRIPTOR
6285 023744 000404 BR 30$ ;GO TO 30$ IF NO ERROR
6286 023746 000240 NOP ;RETURN HERE IF ERROR
6287 023750 104000 ERROR ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6288 023752 000137 024352 JMP 190$ ;GO TO 190$ IF ERROR WAS FOUND
6289 023756 30$:
6290
6291 ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
6292 023756 012737 000063 001400 MOV #WH,GO,RMCS10 ;WRITE HEADER AND DATA
6293 023764 012702 001535 MOV #PUTINX,R2 ;WRITE REGISTER INDEX TABLE
6294 023770 112722 000006 MOVB #RMDA,(R2)+
6295 023774 112722 000034 MOVB #RMDC,(R2)+
6296 024000 112722 000032 MOVB #RMOF,(R2)+
6297 024004 112722 000002 MOVB #RMWC,(R2)+
6298 024010 112722 000004 MOVB #RMBA,(R2)+
6299 024014 112722 000000 MOVB #RMCS1,(R2)+
6300 024020 112722 000200 MOVB #200,(R2)+
6301 024024 004737 037766 JSR PC,PUT ;GO WRITE REGISTERS VIA PUT SUB
6302 024030 000404 BR 80$ ;GO TO 80$ IF NO ERROR
6303 024032 000240 NOP ;RETURN HERE IF ERROR
6304 024034 104070 ERROR ;ERROR DEFINED BY PUT SUB

```

6305	024036	000137	024352		JMP	190\$;GO TO 190\$ IF ERROR WAS FOUND
6306	024042			80\$:			
6307							
6308							
6309	024042	004737	037432		JSR	PC,GETSTS	;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6310							
6311							
6312	024046	004737	040326		JSR	PC,TIMOUT	;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
6313	024052	004737	037516		JSR	PC,GET	;WAIT FOR COMMAND TO COMPLETE
6314	024056	000404			BR	90\$;GO READ REGISTERS VIA GET SUB
6315	024060	000240			NOP		;GO TO 90\$ IF NO ERROR
6316	024062	104000			ERROR		;RETURN HERE IF ERROR
6317	024064	000137	024352		JMP	190\$;ERROR DEFINED BY GET SUB
6318	024070			90\$:			;GO TO 190\$ IF ERROR WAS FOUND
6319							
6320							
6321	024070	004737	040512		JSR	PC,PRIERR	;VERIFY RESULTS OF WRITE COMMAND
6322	024074	000405			BR	100\$;GO CHECK FOR PRIMARY ERRORS
6323	024076	000240			NOP		;GO TO 100\$ IF NO ERROR
6324	024100	104000			ERROR		;RETURN HERE IF ERROR
6325	024102	004736			JSR	PC,2(SP)+	;ERROR # DEFINED BY PRIERR SUBROUTINE
6326	024104	000137	024352		JMP	190\$;GO BACK FOR MORE ERROR CHECKS
6327	024110			100\$:			;GO TO 190\$ IF ERROR WAS FOUND
6328	024110	004737	053016		JSR	PC,DTASTS	;GO VERIFY RESULTS OF DATA TRANSFER
6329	024114	000405			BR	110\$;GO TO 110\$ IF NO ERROR
6330	024116	000240			NOP		;RETURN HERE IF ERROR
6331	024120	104000			ERROR		;ERROR # DEFINED BY DTASTS SUBROUTINE
6332	024122	004736			JSR	PC,2(SP)+	;GO BACK FOR MORE ERROR CHECKS
6333	024124	000137	024352		JMP	190\$;GO TO 190\$ IF ERROR WAS FOUND
6334	024130			110\$:			
6335	024130	004737	041344		JSR	PC,SECERR	;GO CHECK FOR SECONDARY ERRORS
6336	024134	000405			BR	120\$;GO TO 120\$ IF NO ERROR
6337	024136	000240			NOP		;RETURN HERE IF ERROR
6338	024140	104000			ERROR		;ERROR # DEFINED BY SECERR SUBROUTINE
6339	024142	004736			JSP	PC,2(SP)+	;GO BACK FOR MORE ERROR CHECKS
6340	024144	000137	024352		JMP	190\$;GO TO 190\$ IF ERROR WAS FOUND
6341	024150			120\$:			
6342							
6343							
6344	024150	004737	034000		JSR	PC,TSTPRP	;CLEAR "IAE" ERROR AND PREPARE DEVICE
6345	024154	154130			.WORD	154130	;PREPARE DEVICE FOR TEST
6346	024156	000404			BR	125\$;TASK DESCRIPTOR
6347	024160	000240			NOP		;GO TO 125\$ IF NO ERROR
6348	024162	104000			ERROR		;RETURN HERE IF ERROR
6349	024164	000137	024352		JMP	190\$;ERROR # DEFINED BY TSTPRP SUBROUTINE
6350	024170			125\$:			;GO TO 190\$ IF ERROR WAS FOUND
6351							
6352							
6353	024170	012737	000073	001400	MOV	#RH!GO,RMCS10	;READ HEADER & DATA COMMAND
6354	024176	012737	104572	001404	MOV	#BUFTWO,RMBA0	;CHANGE BUS ADDRESS
6355	024204	004737	037766		JSR	PC,PUT	;GO WRITE REGISTERS VIA PUT SUB
6356	024210	000404			BR	130\$;GO TO 130\$ IF NO ERROR
6357	024212	000240			NOP		;RETURN HERE IF ERROR
6358	024214	104000			ERROR		;ERROR DEFINED BY PUT SUB
6359	024216	000137	024352		JMP	190\$;GO TO 190\$ IF ERROR WAS FOUND
6360	024222			130\$:			

```

6361
6362
6363 024222 004737 037432      ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6364      JSR      PC,GETSTS
6365
6366      ;WAIT FOR READ TO COMPLETE AND GET STATUS
6367      JSR      PC,TIMOUT
6368      JSR      PC,GET      ;GO READ REGISTERS VIA GET SUB
6369      BR      140$      ;GO TO 140$ IF NO ERROR
6370      NOP
6371      ERROR
6372      JMP      190$      ;GO TO 190$ IF ERROR WAS FOUND
6373 140$:
6374
6375      ;VERIFY THE RESULTS OF READ OPERATION
6376      JSR      PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
6377      BR      150$      ;GO TO 150$ IF NO ERROR
6378      NOP
6379      ERROR
6380      JSR      PC,2(SP)+      ;RETURN HERE IF ERROR
6381      JMP      190$      ;ERROR # DEFINED BY PRIERR SUBROUTINE
6382      ;GO BACK FOR MORE ERROR CHECKS
6383 150$:
6384      JSR      PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
6385      BR      160$      ;GO TO 160$ IF NO ERROR
6386      NOP
6387      ERROR
6388      JSR      PC,2(SP)+      ;RETURN HERE IF ERROR
6389      JMP      190$      ;ERROR # DEFINED BY DTASTS SUBROUTINE
6390      ;GO BACK FOR MORE ERROR CHECKS
6391 160$:
6392      JSR      PC,SECERR      ;GO CHECK FOR SECONDARY ERRORS
6393      BR      170$      ;GO TO 170$ IF NO ERROR
6394      NOP
6395      ERROR
6396      JSR      PC,2(SP)+      ;RETURN HERE IF ERROR
6397      JMP      190$      ;ERROR # DEFINED BY SECERR SUBROUTINE
6398      ;GO BACK FOR MORE ERROR CHECKS
6399 170$:
6400 180$:
6401
6402      ;INCREMENT ADDRESS AND FORMAT NEXT SECTOR
6403      ADD      #1,R1      ;ADVANCE SECTOR COUNT
6404      CMP      #31,R1      ;DONE ALL SECTORS??
6405      BLO      190$      ;YES!!
6406      JMP      15$      ;GO DO NEXT SECTOR
6407 190$:
6408
6409      ;*****
6410      ;TEST 26      FORMAT INVALID TRACK ADDRESS
6411      ;*****
6412      TST26:
6413      SCOPE
6414      NOP
6415      MOV      #STACK,SP      ;SCOPE CALL
6416      MOV      $BASE,R0      ;START OF TEST
6417      MOV      TSTQUE,R1      ;INITIALIZE STACK POINTER
6418      MOV      #26,$TESTN      ;AD=UNIBLS ADDRESS
6419      ;(R1) = DEVICE BEING TESTED
6420      ;SET TEST NUMBER IN APT MAIL BOX

```

```

6417 024400 012737 000000 001434 ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
6418 024406 012737 002400 001406 MOV #0,RMD10 ;CYLINDER = 0
6419 024414 012737 010000 001406 MOV #2400,RMD40 ;TRACK = 5 SECTOR = 0
6420 024422 012737 010000 001432 15$: MOV #FMT16,RM1FO ;16 BIT FORMAT
6421 024430 012737 177376 001402 MOV #(<PC(2)+2< >+1),RMWCO ;2 + 256 WORDS
6422 024436 012737 103566 001404 MOV #BUFONE,F BAO ;DATA BUFFER ADDRESS
6423 024444 012737 000062 001406 MOV #WH,RMCS10 ;WRITE HEADER AND DATA
6424 024452 012737 001406 001174 MOV #RMD40,STMP0 ;USE SECTOR FOR DATA PATTERN
6425 024460 012737 000001 001176 MOV #1,STMP1
6426 024464 004737 036620 JSR PC,GENBUF ;GO GENERATE DATA BUFFER
6427 024464 20$:
6428
6429 ;PREPARE DEVICE FOR DATA TRANSFER
6430 024464 004737 034000 JSR PC,TSTPRP ;PREPARE DEVICE FOR TEST
6431 024470 154130 .WORD 154130 ;TASK DESCRIPTOR
6432 024472 000404 BR 30$ ;GO TO 30$ IF NO ERROR
6433 024474 000240 NOP ;RETURN HERE IF ERROR
6434 024476 104000 ERROR ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6435 024500 000137 025100 JMP 190$ ;GO TO 190$ IF ERROR WAS FOUND
6436 024504
6437
6438 ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
6439 024504 012737 000063 001400 MOV #WH,GO,RMCS10 ;WRITE HEADER AND DATA
6440 024512 012702 001535 MOV #PUTINX,R2 ;WRITE REGISTER INDEX TABLE
6441 024516 112722 000006 MOV #RMDA,(R2)+
6442 024522 112722 000034 MOV #RMDC,(R2)+
6443 024526 112722 000032 MOV #RMOF,(R2)+
6444 024532 112722 000032 MOV #RMMC,(R2)+
6445 024536 112722 000034 MOV #PMQA,(R2)+
6446 024542 112722 000000 MOV #RMCS1,(R2)+
6447 024546 112722 000200 MOV #200,(R2)+
6448 024552 004737 037766 JSR PC,PUT ;GO WRITE REGISTERS VIA PUT SUB
6449 024556 000404 BR 80$ ;GO TO 80$ IF NO ERROR
6450 024560 000240 NOP ;RETURN HERE IF ERROR
6451 024562 104000 ERROR ;ERROR DEFINED BY PUT SUB
6452 024564 000137 025100 JMP 190$ ;GO TO 190$ IF ERROR WAS FOUND
6453 024570
6454
6455 ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6456 024570 004737 037432 JSR PC,GETSTS
6457
6458 ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
6459 024574 004737 040326 JSR PC,TIMOUT ;WAIT FOR COMMAND TO COMPLETE
6460 024600 004737 037516 JSR PC,GET ;GO READ REGISTERS VIA GET SUB
6461 024604 000404 BR 90$ ;GO TO 90$ IF NO ERROR
6462 024606 000240 NOP ;RETURN HERE IF ERROR
6463 024610 104000 ERROR ;ERROR DEFINED BY GET SUB
6464 024612 000137 025100 JMP 190$ ;GO TO 190$ IF ERROR WAS FOUND
6465 024616
6466
6467 ;VERIFY RESULTS OF WRITE COMMAND
6468 024616 004737 040512 JSR PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
6469 024622 000405 BR 100$ ;GO TO 100$ IF NO ERROR
6470 024624 000240 NOP ;RETURN HERE IF ERROR
6471 024626 104000 ERROR ;ERROR # DEFINED BY PRIERR SUBROUTINE
6472 024630 004736 JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS

```

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 131
T26 FORMAT INVALID TRACK ADDRESS

SEQ 0131

6473	024632	000137	025100		JMP	190\$;GO TO 190\$ IF ERROR WAS FOUND
6474	024636			100\$:			
6475	024636	004737	053016		JSR	PC,DTASTS	;GO VERIFY RESULTS OF DATA TRANSFER
6476	024642	000405			BR	110\$;GO TO 110\$ IF NO ERROR
6477	024644	000240			NOP		;RETURN HERE IF ERROR
6478	024646	104000			ERROR		;ERROR # DEFINED BY DTASTS SUBROUTINE
6479	024650	004736			JSR	PC,2(SP)+	;GO BACK FOR MORE ERROR CHECKS
6480	024652	000137	025100		JMP	190\$;GO TO 190\$ IF ERROR WAS FOUND
6481	024656			110\$:			
6482	024656	004737	041344		JSR	PC,SECERR	;GO CHECK FOR SECONDARY ERRORS
6483	024662	000405			BR	120\$;GO TO 120\$ IF NO ERROR
6484	024664	000240			NOP		;RETURN HERE IF ERROR
6485	024666	104000			ERROR		;ERROR # DEFINED BY SECERR SUBROUTINE
6486	024670	004736			JSR	PC,2(SP)+	;GO BACK FOR MORE ERROR CHECKS
6487	024672	000137	025100		JMP	190\$;GO TO 190\$ IF ERROR WAS FOUND
6488	024676			120\$:			
6489							
6490							
6491	024676	004737	034000		JSR	PC,TSTPRP	;PREPARE DEVICE FOR TEST
6492	024702	154130			WORD	154130	;TASK DESCRIPTOR
6493	024704	000404			BR	125\$;GO TO 125\$ IF NO ERROR
6494	024706	000240			NOP		;RETURN HERE IF ERROR
6495	024710	104000			ERROR		;ERROR # DEFINED BY TSTPRP SUBROUTINE
6496	024712	000137	025100		JMP	190\$;GO TO 190\$ IF ERROR WAS FOUND
6497	024716			125\$:			
6498							
6499							
6500	024716	012737	000073	001400			
6501	024724	012737	104572	001404	MOV	#RH:GO,RMC510	;READ HEADER & DATA COMMAND
6502	024732	004737	037766		MOV	#BUFTWO,RMB40	;CHANGE BUS ADDRESS
6503	024736	000404			JSR	PC,PUT	;GO WRITE REGISTERS VIA PUT SUB
6504	024740	000240			BR	130\$;GO TO 130\$ IF NO ERROR
6505	024742	104000			NOP		;RETURN HERE IF ERROR
6506	024744	000137	025100		ERROR		;ERROR DEFINED BY PUT SUB
6507	024750				JMP	190\$;GO TO 190\$ IF ERROR WAS FOUND
6508							
6509							
6510	024750	004737	037432		JSR	PC,GETSTS	
6511							
6512							
6513	024754	004737	040326		JSR	PC,TIMOUT	;WAIT FOR READ TO COMPLETE
6514	024760	004737	037516		JSR	PC,GET	;GO READ REGISTERS VIA GET SUB
6515	024764	000404			BR	140\$;GO TO 140\$ IF NO ERROR
6516	024766	000240			NOP		;RETURN HERE IF ERROR
6517	024770	104000			ERROR		;ERROR DEFINED BY GET SUB
6518	024772	000137	025100		JMP	190\$;GO TO 190\$ IF ERROR WAS FOUND
6519	024776						
6520							
6521							
6522	024776	004737	040512		JSR	PC,PRIERR	;GO CHECK FOR PRIMARY ERRORS
6523	025002	000405			BR	150\$;GO TO 150\$ IF NO ERROR
6524	025004	000240			NOP		;RETURN HERE IF ERROR
6525	025006	104000			ERROR		;ERROR # DEFINED BY PRIERR SUBROUTINE
6526	025010	004736			JSR	PC,2(SP)+	;GO BACK FOR MORE ERROR CHECKS
6527	025012	000137	025100		JMP	190\$;GO TO 190\$ IF ERROR WAS FOUND
6528	025016						
				150\$:			

```

6529 025016 004737 053016      JSR    PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
6530 025022 000405          BR      160$          ;GO TO 160$ IF NO ERROR
6531 025024 000240          NOP          ;RETURN HERE IF ERROR
6532 025026 104000          ERROR      ;ERROR # DEFINED BY DTASTS SUBROUTINE
6533 025030 004736      JSR    PC,2(SP)+    ;GO BACK FOR MORE ERROR CHECKS
6534 025032 000137 025100      JMP      190$          ;GO TO 190$ IF ERROR WAS FOUND
6535 025036          160$:
6536 025036 004737 041344      JSR    PC,SECERR      ;GO CHECK FOR SECONDARY ERRORS
6537 025042 000405          BR      170$          ;GO TO 170$ IF NO ERROR
6538 025044 000240          NOP          ;RETURN HERE IF ERROR
6539 025046 104000          ERROR      ;ERROR # DEFINED BY SECERR SUBROUTINE
6540 025050 004736      JSR    PC,2(SP)+    ;GO BACK FOR MORE ERROR CHECKS
6541 025052 000137 025100      JMP      190$          ;GO TO 190$ IF ERROR WAS FOUND
6542 025056          170$:
6543 025056          180$:
6544
6545      ;INCREMENT ADDRESS AND FORMAT NEXT TRACK
6546 025056 062737 000400 001406      ADD     #400,RMDAO      ;ADVANCE TRACK COUNT
6547 025064 022737 003400 001406      CMP     #3400,RMDAO      ;DONE ALL TRACKS??
6548 025072 103402          BLO      190$          ;YES!!
6549 025074 000137 024414      JMP      15$          ;GO DO NEXT TRACK
6550 025100          190$:
6551
6552      ;*****
6553      ;*TEST 27      FORMAT INVALID CYLINDER ADDRESS
6554      ;*****
6555      ;TST27:
6556 025100 000004          SCOPE          ;SCOPE CALL
6557 025102 000240          NOP          ;START OF TEST
6558 025104 012706 001100      MOV     #STACK,SP      ;INITIALIZE STACK POINTER
6559 025110 013700 001276      MOV     $BASE,R0      ;R0=UNIBUS ADDRESS
6560 025114 013701 001450      MOV     TSTQUE,R1      ;(R1) = DEVICE BEING TESTED
6561 025120 012737 000027 001226      MOV     #27,$TSTIN      ;SET TEST NUMBER IN APT MAIL BOX
6562 025126          10$:
6563
6564      ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
6565 025126 012737 001467 001434      MOV     #823,RMDC0      ;START AT LAST CYLINDER + 1
6566 025134 012737 000000 001406      MOV     #0,RMDAO      ;TRACK = SECTOR = 0
6567 025142 012737 010000 001432      MOV     #FMT16,RMOF0      ;16 BIT FORMAT
6568 025150 012737 177376 001402      MOV     #(<C(2+256)+1>),RMWCO      ;2 + 256 WORDS
6569 025156 012737 103566 001404      MOV     #BUFONE,RMBAO      ;DATA BUFFER ADDRESS
6570 025164 012737 000062 001400      MOV     #WH,RMC510      ;WRITE HEADER AND DATA
6571 025172 012737 001434 001174      MOV     #RMDC0,$TMP0      ;USE CYLINDER FOR DATA PATTERN
6572 025200 012737 000001 001176      MOV     #1,$TMP1
6573 025206 004737 036520      JSR    PC,GENBUF      ;GO GENERATE DATA BUFFER
6574 025212          20$:
6575
6576      ;PREPARE DEVICE FOR DATA TRANSFER
6577 025212 004737 034000      JSR    PC,TSTPRP      ;PREPARE DEVICE FOR TEST
6578 025216 154130          .WORD      154130      ;TASK DESCRIPTOR
6579 025220 000404          BR      30$          ;GO TO 30$ IF NO ERROR
6580 025222 000240          NOP          ;RETURN HERE IF ERROR
6581 025224 104000          ERROR      ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6582 025226 000137 025626      JMP      190$          ;GO TO 190$ IF ERROR WAS FOUND
6583 025232          30$:
6584

```

D11

```

6585      025232 012737 000063
6586      025240 012702 001535
6587      025244 112722 000006
6588      025250 112722 000034
6589      025254 112722 000032
6590      025260 112722 000002
6591      025264 112722 000004
6592      025270 112722 000000
6593      025274 112722 000200
6594      025300 004737 037766
6595      025304 000404
6596      025306 000240
6597      025310 104000
6598      025312 000137 025626
6599      025316
6600
6601
6602
6603      025316 004737 037432
6604
6605
6606      025322 004737 040326
6607      025326 004737 037516
6608      025332 000404
6609      025334 000240
6610      025336 104000
6611      025340 000137 025626
6612      025344
6613
6614
6615      025344 004737 040512
6616      025350 000405
6617      025352 000240
6618      025354 104000
6619      025356 004736
6620      025360 000137 025626
6621      025364
6622      025364 004737 053016
6623      025370 000405
6624      025372 000240
6625      025374 104000
6626      025376 004736
6627      025400 000137 025626
6628      025404
6629      025404 004737 041344
6630      025410 000405
6631      025412 000240
6632      025414 104000
6633      025416 004736
6634      025420 000137 025626
6635      025424
6636
6637
6638      025424 004737 034000
6639      025430 154130
6640      025432 000404

;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
001400  MOV     $WH:GO,RMCS10      ;WRITE HEADER AND DATA
        MOV     $PUTINX,R2      ;WRITE REGISTER INDEX TABLE
        MOV     $RMDA,(R2)+
        MOV     $RMOC,(R2)+
        MOV     $RMOF,(R2)+
        MOV     $RMWC,(R2)+
        MOV     $RTA,(R2)+
        MOV     $RMCS1,(R2)+
        MOV     $200,(R2)+
        JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
        BR      80$            ;GO TO 80$ IF NO ERROR
        NOP
        ERROR   ;RETURN HERE IF ERROR
        ERROR   ;ERROR DEFINED BY PUT SUB
        JMP     190$           ;GO TO 190$ IF ERROR WAS FOUND

80$:
;SETUP INPUT REGISTER BUFFER FOR READING STATUS
      JSR     PC,GETSTS

;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
      JSR     PC,TIMOUT          ;WAIT FOR COMMAND TO COMPLETE
      JSR     PC,GET            ;GO READ REGISTERS VIA GET SUB
      BR      90$              ;GO TO 90$ IF NO ERROR
      NOP
      ERROR   ;RETURN HERE IF ERROR
      ERROR   ;ERROR DEFINED BY GET SUB
      JMP     190$           ;GO TO 190$ IF ERROR WAS FOUND

90$:
;VERIFY RESULTS OF WRITE COMMAND
      JSR     PC,PRIERR          ;GO CHECK FOR PRIMARY ERRORS
      BR      100$              ;GO TO 100$ IF NO ERROR
      NOP
      ERROR   ;RETURN HERE IF ERROR
      ERROR   ;ERROR # DEFINED BY PRIERR SUBROUTINE
      JSR     PC,$(SP)+          ;GO BACK FOR MORE ERROR CHECKS
      JMP     190$           ;GO TO 190$ IF ERROR WAS FOUND

100$:
      JSR     PC,DTASTS          ;GO VERIFY RESULTS OF DATA TRANSFER
      BR      110$              ;GO TO 110$ IF NO ERROR
      NOP
      ERROR   ;RETURN HERE IF ERROR
      ERROR   ;ERROR # DEFINED BY DTASTS SUBROUTINE
      JSR     PC,$(SP)+          ;GO BACK FOR MORE ERROR CHECKS
      JMP     190$           ;GO TO 190$ IF ERROR WAS FOUND

110$:
      JSR     PC,SECERR          ;GO CHECK FOR SECONDARY ERRORS
      BR      120$              ;GO TO 120$ IF NO ERROR
      NOP
      ERROR   ;RETURN HERE IF ERROR
      ERROR   ;ERROR # DEFINED BY SECERR SUBROUTINE
      JSR     PC,$(SP)+          ;GO BACK FOR MORE ERROR CHECKS
      JMP     190$           ;GO TO 190$ IF ERROR WAS FOUND

120$:
;CLEAR "IAE" ERROR AND PREPARE DEVICE
      JSR     PC,TSTPRP          ;PREPARE DEVICE FOR TEST
      .WORD   154130            ;TASK DESCRIPTOR
      BR      125$              ;GO TO 125$ IF NO ERROR

```

```

6641 025434 000240      NOP      ;RETURN HERE IF ERROR
6642 025436 104000      ERROR     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6643 025440 000137 025626 JMP      190$      ;GO TO 190$ IF ERROR WAS FOUND
6644 025444
6645
6646      125$:
6647      ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
6648 025444 012737 000073 001400 MOV      #RH!GO,RMC510      ;READ HEADER & DATA COMMAND
6649 025452 012737 104572 001404 MOV      #BUFTWO,RMB40      ;CHANGE BUS ADDRESS
6650 025460 004737 037766 JSR      PC,PUT      ;GO WRITE REGISTERS VIA PUT SUB
6651 025464 000404 BR      130$      ;GO TO 130$ IF NO ERROR
6652 025466 000240      NOP      ;RETURN HERE IF ERROR
6653 025470 104000      ERROR     ;ERROR DEFINED BY PUT SUB
6654 025472 000137 025626 JMP      190$      ;GO TO 190$ IF ERROR WAS FOUND
6655 025476
6656      130$:
6657      ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6658 JSR      PC,GETSTS
6659
6660      ;WAIT FOR READ TO COMPLETE AND GET STATUS
6661 JSR      PC,TIMOUT      ;WAIT FOR READ TO COMPLETE
6662 JSR      PC,GET      ;GO READ REGISTERS VIA GET SUB
6663 BR      140$      ;GO TO 140$ IF NO ERROR
6664 025512 000404      NOP      ;RETURN HERE IF ERROR
6665 025514 000240      ERROR     ;ERROR DEFINED BY GET SUB
6666 025516 104000      ERROR     ;ERROR DEFINED BY GET SUB
6667 025520 000137 025626 JMP      190$      ;GO TO 190$ IF ERROR WAS FOUND
6668 025524
6669      140$:
6670      ;VERIFY THE RESULTS OF READ OPERATION
6671 JSR      PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
6672 BR      150$      ;GO TO 150$ IF NO ERROR
6673 025532 000240      NOP      ;RETURN HERE IF ERROR
6674 025534 104000      ERROR     ;ERROR # DEFINED BY PRIERR SUBROUTINE
6675 025536 004736 JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
6676 025540 000137 025626 JMP      190$      ;GO TO 190$ IF ERROR WAS FOUND
6677 025544
6678      150$:
6679 JSR      PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
6680 BR      160$      ;GO TO 160$ IF NO ERROR
6681 025550 000405      NOP      ;RETURN HERE IF ERROR
6682 025552 000240      ERROR     ;ERROR # DEFINED BY DTASTS SUBROUTINE
6683 025554 104000      ERROR     ;GO BACK FOR MORE ERROR CHECKS
6684 025556 004736 JSR      PC,@(SP)+      ;GO TO 190$ IF ERROR WAS FOUND
6685 025560 000137 025626 JMP      190$
6686 025564
6687      160$:
6688 JSR      PC,SECERR      ;GO CHECK FOR SECONDARY ERRORS
6689 BR      170$      ;GO TO 170$ IF NO ERROR
6690 025570 000405      NOP      ;RETURN HERE IF ERROR
6691 025572 000240      ERROR     ;ERROR # DEFINED BY SECERR SUBROUTINE
6692 025574 104000      ERROR     ;GO BACK FOR MORE ERROR CHECKS
6693 025576 004736 JSR      PC,@(SP)+      ;GO TO 190$ IF ERROR WAS FOUND
6694 025600 000137 025626 JMP      190$
6695 025604
6696      170$:
6697      ;INCREMENT ADDRESS AND FORMAT NEXT CYLINDER
6698 ADD      #1,RMDC0      ;ADVANCE CYLINDER COUNT
6699 CMP      #1777,RMDC0      ;DONE ALL CYLINDERS??
6700 BLO      190$      ;YES!!
6701 JMP      15$      ;GO DO NEXT SECTOR
6702
6703      190$:

```



```

6697
6698
6699
6700
6701 025626
6702 025626 000004
6703 025630 000240
6704 025632 012706 001100
6705 025636 013700 01276
6706 025642 013701 001450
6707 025646 012737 000030 001226
6708 025654
6709
6710
6711 025654 012737 000000 001434
6712 025662 012737 000000 001406
6713 025670 012737 010000 001432
6714 025676 012737 177376 001402
6715 025704 012737 103566 001404
6716 025712 012737 000063 001400
6717 025720 012737 001406 001174
6718 025726 012737 000001 001176
6719 025734 004737 036620
6720 025740
6721
6722
6723 025740 004737 034000
6724 025744 154130
6725 025746 000404
6726 025750 000240
6727 025752 104000
6728 025754 000137 026520
6729 025760
6730
6731
6732 025760 012737 000015 001400
6733 025766 012702 001535
6734 025772 112722 000006
6735 025776 112722 000034
6736 026002 112722 000032
6737 026006 112722 000000
6738 026012 112712 000200
6739 026016 004737 037766
6740 026022 000404
6741 026024 000240
6742 026026 104000
6743 026030 000137 026520
6744 026034
6745
6746
6747 026034 012737 000063 001400
6748 026042 012702 001535
6749 026046 112722 000002
6750 026052 112722 000004
6751 026056 112722 000000
6752 026062 112722 000200

;*****
;TEST 30 FORMAT AT OFFSET
;*****
TST30:
        SCOPE
        NOP
        MOV     #STACK, SP
        MOV     $BASE, R0
        MOV     TSTQUE, R1
        MOV     #30, $TESTN
        ;SCOPE CALL
        ;START OF TEST
        ;INITIALIZE STACK POINTER
        ;R0=UNIBUS ADDRESS
        ;(R1) = DEVICE BEING TESTED
        ;SET TEST NUMBER IN APT MAIL BOX
10$:
;SETUP PARAMETERS FOR GENERATING DATA BUFFER
        MOV     #0, RMDC0
        MOV     #0, RMDA0
        MOV     #FMT16, RMOFO
        MOV     #(<IC(2+256)+1>), RMWCO
        MOV     #BUFONE, RMBAO
        MOV     #WH!GO, RMCS10
        MOV     #RMDA0, $TMPD
        MOV     #1, $TMP1
        JSR     PC, GENBUF
        ;CYLINDER = 0
        ;TRACK = SECTOR = 0
        ;16 BIT FORMAT
        ;2 + 256 WORDS
        ;DATA BUFFER ADDRESS
        ;WRITE HEADER AND DATA
        ;USE SECTOR FOR DATA PATTERN
15$:
20$:
;PREPARE DEVICE FOR DATA TRANSFER
        JSR     PC, TSTPRP
        .WORD   154130
        BR      30$
        ;PREPARE DEVICE FOR TEST
        ;TASK DESCRIPTOR
        ;GO TO 30$ IF NO ERROR
        ;RETURN HERE IF ERROR
        ;ERROR # DEFINED BY TSTPRP SUBROUTINE
        ;GO TO 190$ IF ERROR WAS FOUND
30$:
;OFFSET DEVICE FOR WRITE
        MOV     #OFFSET!GO, RMCS10
        MOV     #PUTINX, R2
        MOV     #RMDA, (R2)+
        MOV     #RMDC, (R2)+
        MOV     #RMOF, (R2)+
        MOV     #RMCS1, (R2)+
        MOV     #200, (R2)
        JSR     PC, PUT
        BR      35$
        ;CHANGE TO OFFSET COMMAND
        ;WRITE PUT INDEX TABLE
        ;TERMINATE TABLE
        ;GO WRITE REGISTERS VIA PUT SUB
        ;GO TO 35$ IF NO ERROR
        ;RETURN HERE IF ERROR
        ;ERROR DEFINED BY PUT SUB
        ;GO TO 190$ IF ERROR WAS FOUND
35$:
;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND AT OFFSET
        MOV     #WH!GO, RMCS10
        MOV     #PUTINX, R2
        MOV     #RMDC, (R2)+
        MOV     #RMDA, (R2)+
        MOV     #RMCS1, (R2)+
        MOV     #200, (R2)+
        ;WRITE HEADER AND DATA COMMAND
        ;WRITE REGISTER INDEX TABLE

```

6753	026066	004737	037766	JSR	PC,PUT	;GO WRITE REGISTERS VIA PUT SUB
6754	026072	000404		BR	80\$;GO TO 80\$ IF NO ERROR
6755	026074	000240		NOP		;RETURN HERE IF ERROR
6756	026076	104000		ERROR		;ERROR DEFINED BY PUT SUB
6757	026100	000137	026520	JMP	190\$;GO TO 190\$ IF ERROR WAS FOUND
6758	026104					
6759						
6760						
6761	026104	004737	037432	JSR	PC,GETSTS	;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6762						
6763						
6764	026110	004737	040326	JSR	PC,TIMOUT	;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
6765	026114	004737	037516	JSR	PC,GET	;WAIT FOR COMMAND TO COMPLETE
6766	026120	000404		BR	90\$;GO READ REGISTERS VIA GET SUB
6767	026122	000240		NOP		;GO TO 90\$ IF NO ERROR
6768	026124	104000		ERROR		;RETURN HERE IF ERROR
6769	026126	000137	026520	JMP	190\$;ERROR DEFINED BY GET SUB
6770	026132					;GO TO 190\$ IF ERROR WAS FOUND
6771						
6772						
6773	026132	004737	040512	JSR	PC,PRIERR	;VERIFY RESULTS OF WRITE COMMAND
6774	026136	000405		BR	100\$;GO CHECK FOR PRIMARY ERRORS
6775	026140	000240		NOP		;GO TO 100\$ IF NO ERROR
6776	026142	104000		ERROR		;RETURN HERE IF ERROR
6777	026144	004736		JSR	PC,(SP)+	;ERROR # DEFINED BY PRIERR SUBROUTINE
6778	026146	000137	026520	JMP	190\$;GO BACK FOR MORE ERROR CHECKS
6779	026152					;GO TO 190\$ IF ERROR WAS FOUND
6780	026152	004737	053016	JSR	PC,DTASTS	;GO VERIFY RESULTS OF DATA TRANSFER
6781	026156	000405		BR	110\$;GO TO 110\$ IF NO ERROR
6782	026160	000240		NOP		;RETURN HERE IF ERROR
6783	026162	104000		ERROR		;ERROR # DEFINED BY DTASTS SUBROUTINE
6784	026164	004736		JSR	PC,(SP)+	;GO BACK FOR MORE ERROR CHECKS
6785	026166	000137	026520	JMP	190\$;GO TO 190\$ IF ERROR WAS FOUND
6786	026172					
6787	026172	004737	041344	JSR	PC,SECERR	;GO CHECK FOR SECONDARY ERRORS
6788	026176	000405		BR	120\$;GO TO 120\$ IF NO ERROR
6789	026200	000240		NOP		;RETURN HERE IF ERROR
6790	026202	104000		ERROR		;ERROR # DEFINED BY SECERR SUBROUTINE
6791	026204	004736		JSR	PC,(SP)+	;GO BACK FOR MORE ERROR CHECKS
6792	026206	000137	026520	JMP	190\$;GO TO 190\$ IF ERROR WAS FOUND
6793	026212					
6794						
6795	026212	012737	000015	MOV	#OFFSET!GO,RMCS10	;CHANGE TO OFFSET COMMAND
6796	026220	012702	001535	MOV	#PUTINX,R2	;WRITE PUT INDEX TABLE
6797	026224	112722	000006	MOVB	#RMDA,(R2)+	
6798	026230	112722	000034	MOVB	#RMDC,(R2)+	
6799	026234	112722	000032	MOVB	#RMOF,(R2)+	
6800	026240	112722	000000	MOVB	#RMCS1,(R2)+	
6801	026244	112722	000200	MOVB	#200,(R2)+	
6802	026250	004737	037766	JSR	PC,PUT	;GO WRITE REGISTERS VIA PUT SUB
6803	026254	000404		BR	125\$;GO TO 125\$ IF NO ERROR
6804	026256	000240		NOP		;RETURN HERE IF ERROR
6805	026260	104000		ERROR		;ERROR DEFINED BY PUT SUB
6806	026262	000137	026520	JMP	190\$;GO TO 190\$ IF ERROR WAS FOUND
6807	026266					
6808						

```

6809
6810
6811 026266 012737 000073 001400 ;READ HEADER AND DATA AT OFFSET
6812 026274 012737 104572 001404 MOV #RH,GO,RMCS10 ;READ HEADER & DATA COMMAND
6813 026302 012702 001535 MOV #BUFTWO,RMBA0 ;CHANGE BUS ADDRESS
6814 026306 112722 000002 MOV #PUTINX,R2 ;WRITE PUT INDEX TABLE
6815 026312 112722 000004 MOVB #RMWC,(R2)+
6816 026316 112722 000000 MOVB #RMBA,(R2)+
6817 026322 112722 000200 MOVB #RMCS1,(R2)+
6818 026326 004737 037766 MOVB #200,(R2)+
6819 026332 000404 JSR PC,PUT ;GO WRITE REGISTERS VIA PUT SUB
6820 026334 000240 BR 130$ ;GO TO 130$ IF NO ERROR
6821 026336 104000 NOP ;RETURN HERE IF ERROR
6822 026340 000137 026520 ERROR ;ERROR DEFINED BY PUT SUB
6823 026344 JMP 190$ ;GO TO 190$ IF ERROR WAS FOUND
6824
6825
6826 026344 004737 040326 ;WAIT FOR READ TO COMPLETE AND GET STATUS
6827 026350 004737 037516 JSR PC,TIMOUT ;WAIT FOR READ TO COMPLETE
6828 026354 000404 JSR PC,GET ;GO READ REGISTERS VIA GET SUB
6829 026356 000240 BR 140$ ;GO TO 140$ IF NO ERROR
6830 026360 104000 NOP ;RETURN HERE IF ERROR
6831 026362 000137 026520 ERROR ;ERROR DEFINED BY GET SUB
6832 026366 JMP 190$ ;GO TO 190$ IF ERROR WAS FOUND
6833
6834
6835 026366 004737 040512 ;VERIFY THE RESULTS OF READ OPERATION
6836 026372 000405 JSR PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
6837 026374 000240 BR 150$ ;GO TO 150$ IF NO ERROR
6838 026376 104000 NOP ;RETURN HERE IF ERROR
6839 026400 004736 ERROR ;ERROR # DEFINED BY PRIERR SUBROUTINE
6840 026402 000137 026520 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
6841 026406 000137 026520 JMP 190$ ;GO TO 190$ IF ERROR WAS FOUND
6842 026406 004737 053016 150$: JSR PC,DSTATS ;GO VERIFY RESULTS OF DATA TRANSFER
6843 026412 000405 BR 160$ ;GO TO 160$ IF NO ERROR
6844 026414 000240 NOP ;RETURN HERE IF ERROR
6845 026416 104000 ERROR ;ERROR # DEFINED BY DSTATS SUBROUTINE
6846 026420 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
6847 026422 000137 026520 JMP 190$ ;GO TO 190$ IF ERROR WAS FOUND
6848 026426 004737 041344 160$: JSR PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
6849 026432 000405 BR 170$ ;GO TO 170$ IF NO ERROR
6850 026434 000240 NOP ;RETURN HERE IF ERROR
6851 026436 104000 ERROR ;ERROR # DEFINED BY SECERR SUBROUTINE
6852 026440 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
6853 026442 000137 026520 JMP 190$ ;GO TO 190$ IF ERROR WAS FOUND
6854 026446
6855
6856
6857
6858 026446 004737 037064 ;VERIFY DATA
6859 026452 103566 JSR PC,CMPBUF ;GO COMPARE WRITE, READ DATA BUFFERS
6860 026454 104572 .WORD BUFTWO ;STARTING ADDRESS OF WRITE BUFFER
6861 026456 000404 .WORD BUFTWO ;STARTING ADDRESS OF READ BUFFER
6862 026460 000240 BR 180$ ;GO TO 180$ IF NO ERROR
6863 026462 104000 NOP ;RETURN HERE IF ERROR
6864 026464 000137 026520 ERROR ;ERROR # DEFINED BY CMPBUF SUBROUTINE
JMP 190$ ;GO TO 190$ IF ERROR WAS FOUND

```

```

6865 026470
6866 026470 032737 000200 001432 180$: BIT #OFD,RMOFO ;DONE BOTH OFFSETS??
6867 026476 001010 190$ BNE 190$ ;YES!!
6868 026500 052737 000200 001432 BIS #OFD,RMOFO ;SET FORWARD OFFSET
6869 026506 012737 103566 001404 MOV #BUFONE,RMBA0 ;CHANGE DATA BUFFER
6870 026514 000137 025740 JMP 20$
6871 026520
6872
6873
6874
6875
6876 026520
6877 026520 000004
6878 026522 000240
6879 026524 012706 001100
6880 026530 013700 001276
6881 026534 013701 001450
6882 026540 012737 000031 001226 190$:
6883 026546
6884
6885
6886 026546 012737 000000 001434 ;SCOPE CALL
6887 026554 012737 000000 001406 ;START OF TEST
6888 026562 012737 010000 001432 ;INITIALIZE STACK POINTER
6889 026570 012737 177376 001402 ;RD=UNIBUS ADDRESS
6890 026576 012737 103566 001404 ;(R1) = DEVICE BEING TESTED
6891 026604 012737 000062 001400 ;SET TEST NUMBER IN APT MAIL BOX
6892 026612 012737 001406 001174
6893 026620 012737 000001 001176
6894 026626 004737 036620
6895 026632
6896
6897
6898 026632 004737 034000 10$:
6899 026636 154130 ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
6900 026640 000404 MOV #0,RMDC0 ;CYLINDER = 0
6901 026642 000240 MOV #0,RMOA0 ;TRACK = SECTOR = 0
6902 026644 104000 MOV #FMT16,RMOFO ;16 BIT FORMAT
6903 026646 000137 027326 MOV #(<C(2+256.)+1>),RMWCC ;2 + 256 WORDS
6904 026652
6905
6906
6907 026652 012737 000001 001424 ;DATA BUFFER ADDRESS
6908 026660 112737 000024 001535 ;WRITE HEADER AND DATA
6909 026666 112737 000200 001536 ;USE SECTOR FOR DATA PATTERN
6910 026674 004737 037766
6911 026700 000404
6912 026702 000240
6913 026704 104000
6914 026706 000137 027326
6915 026712
6916
6917
6918 026712 012737 000063 001400 15$:
6919 026720 012737 000000 001424 20$:
6920 026726 012702 001535

```

6921	026732	112722	000024	MOV	#RMMR1,(R2)+	
6922	026736	112722	000006	MOV	#RMDA,(R2)+	
6923	026742	112722	000034	MOV	#RMDC,(R2)+	
6924	026746	112722	000032	MOV	#RMOF,(R2)+	
6925	026752	112722	000002	MOV	#RMWC,(R2)+	
6926	026756	112722	000004	MOV	#RMBR,(R2)+	
6927	026762	112722	000000	MOV	#RMCS1,(R2)+	
6928	026766	112722	000200	MOV	#200,(R2)+	
6929	026772	004737	037766	JSR	PC,PUT	;GO WRITE REGISTERS VIA PUT SUB
6930	026776	000404		BR	80\$;GO TO 80\$ IF NO ERROR
6931	027000	000240		NOP		;RETURN HERE IF ERROR
6932	027002	104000		ERROR		;ERROR DEFINED BY PUT SUB
6933	027004	000137	027326	JMP	190\$;GO TO 190\$ IF ERROR WAS FOUND
6934	027010					
6935						
6936						
6937	027010	004737	037432	JSR	PC,GETSIS	;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6938						
6939						
6940	027014	004737	040326	JSR	PC,TIMOUT	;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
6941	027020	004737	037516	JSR	PC,GET	;WAIT FOR COMMAND TO COMPLETE
6942	027024	000404		BR	90\$;GO READ REGISTERS VIA GET SUB
6943	027026	000240		NOP		;GO TO 90\$ IF NO ERROR
6944	027030	104000		ERROR		;RETURN HERE IF ERROR
6945	027032	000137	027326	JMP	190\$;ERROR DEFINED BY GET SUB
6946	027036					;GO TO 190\$ IF ERROR WAS FOUND
6947						
6948						
6949	027036	004737	040512	JSR	PC,PRIERR	;VERIFY RESULTS OF WRITE COMMAND
6950	027042	000405		BR	100\$;GO CHECK FOR PRIMARY ERRORS
6951	027044	000240		NOP		;GO TO 100\$ IF NO ERROR
6952	027046	104000		ERROR		;RETURN HERE IF ERROR
6953	027050	004736		JSR	PC,2(SP)+	;ERROR # DEFINED BY PRIERR SUBROUTINE
6954	027052	000137	027326	JMP	190\$;GO BACK FOR MORE ERROR CHECKS
6955	027056					;GO TO 190\$ IF ERROR WAS FOUND
6956	027056	032737	010000	BIT	#IVC,RMER2I	;IS IVC STATUS SET??
6957	027064	001012	001372	BNE	110\$;YES!!
6958	027066	013737	001142	MOV	RMER2I,\$BDDAT	;RECEIVED STATUS FOR TYPEOUT
6959	027074	013737	001372	MOV	RMER2I,\$GDDAT	;EXPECTED STATUS
6960	027102	052737	010000	BIS	#IVC,\$GDDAT	
6961	027110	104342		ERROR	342	;IVC NOT SET DURING FORMAT
6962	027112					
6963	027112	004737	041344	JSR	PC,SECERR	;GO CHECK FOR SECONDARY ERRORS
6964	027116	000405		BR	120\$;GO TO 120\$ IF NO ERROR
6965	027120	000240		NOP		;RETURN HERE IF ERROR
6966	027122	104000		ERROR		;ERROR # DEFINED BY SECERR SUBROUTINE
6967	027124	004736		JSR	PC,2(SP)+	;GO BACK FOR MORE ERROR CHECKS
6968	027126	000137	027326	JMP	190\$;GO TO 190\$ IF ERROR WAS FOUND
6969	027132					
6970						
6971						
6972	027132	004737	034000	JSR	PC,TSTPRP	;CLEAR IVC ERROR
6973	027136	040100		.WORD	040100	;PREPARE DEVICE FOR TEST
6974	027140	000404		BR	125\$;TASK DESCRIPTOR
6975	027142	000240		NOP		;GO TO 125\$ IF NO ERROR
6976	027144	104000		ERROR		;RETURN HERE IF ERROR
						;ERROR # DEFINED BY TSTPRP SUBROUTINE

```

6977 027146 000137 027326      JMP      190$      ;GO TO 190$ IF ERROR WAS FOUND
6978 027152
6979
6980      ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
6981 027152 012737 000073 001400      MOV      #RH!GO,RMC$IO      ;READ HEADER & DATA COMMAND
6982 027160 012737 104572 001404      MOV      #BUFTWO,RMB$O      ;CHANGE BUS ADDRESS
6983 027166 004737 037766      JSR      PC,PUT      ;GO WRITE REGISTERS VIA PUT SUB
6984 027172 000404      BR      130$      ;GO TO 130$ IF NO ERROR
6985 027174 000240      NOP      ;RETURN HERE IF ERROR
6986 027176 104000      ERROR    ;ERROR DEFINED BY PUT SUB
6987 027200 000137 027326      JMP      190$      ;GO TO 190$ IF ERROR WAS FOUND
6988 027204
6989
6990      ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6991 027204 004737 037432      JSR      PC,GETSTS
6992
6993      ;WAIT FOR READ TO COMPLETE AND GET STATUS
6994 027210 004737 040326      JSR      PC,TIMOUT      ;WAIT FOR READ TO COMPLETE
6995 027214 004737 037516      JSR      PC,GET      ;GO READ REGISTERS VIA GET SUB
6996 027220 000404      BR      140$      ;GO TO 140$ IF NO ERROR
6997 027222 000240      NOP      ;RETURN HERE IF ERROR
6998 027224 104000      ERROR    ;ERROR DEFINED BY GET SUB
6999 027226 000137 027326      JMP      190$      ;GO TO 190$ IF ERROR WAS FOUND
7000 027232
7001
7002      ;VERIFY THE RESULTS OF READ OPERATION
7003 027232 004737 040512      JSR      PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
7004 027236 000405      BR      150$      ;GO TO 150$ IF NO ERROR
7005 027240 000240      NOP      ;RETURN HERE IF ERROR
7006 027242 104000      ERROR    ;ERROR # DEFINED BY PRIERR SUBROUTINE
7007 027244 004736      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
7008 027246 000137 027326      JMP      190$      ;GO TO 190$ IF ERROR WAS FOUND
7009 027252
7010 027252 032737 010000 001372 150$:      BIT      #IVC,RMER2I      ;IS IVC STATUS SET??
7011 027260 001012      BNE      160$      ;YES!!
7012 027262 013737 001372 001142      MOV      RMER2I,$BDDAT      ;RECEIVED STATUS FOR TYPEOUT
7013 027270 013737 001372 001140      MOV      RMER2I,$GDDAT      ;EXPECTED STATUS
7014 027276 052737 010000 001140      BIS      #IVC,$GDDAT
7015 027304 104342      ERROR    342      ;IVC NOT SET DURING FORMAT
7016 027306
7017 027306 004737 041344 160$:      JSR      PC,SECERR      ;GO CHECK FOR SECONDARY ERRORS
7018 027312 000405      BR      170$      ;GO TO 170$ IF NO ERROR
7019 027314 000240      NOP      ;RETURN HERE IF ERROR
7020 027316 104000      ERROR    ;ERROR # DEFINED BY SECERR SUBROUTINE
7021 027320 004736      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
7022 027322 000137 027326      JMP      190$      ;GO TO 190$ IF ERROR WAS FOUND
7023 027326
7024 027326
7025
7026      ;*****
7027      ;*TEST 32      FORMAT ERROR TEST - 18
7028      ;*****
7029 027326      TST32:
7030 027326 000004      SCOPE    ;SCOPE CALL
7031 027330 000240      NOP      ;START OF TEST
7032 027332 012706 001100      MOV      #STACK,SP      ;INITIALIZE STACK POINTER

```

```
7033 027336 013700 001276      MOV    $BASE,R0      ;RO=UNIBUS ADDRESS
7034 027342 013701 001450      MOV    TSTQUE,R1      ;(R1) = DEVICE BEING TESTED
7035 027346 012737 000032 001226  MOV    #32,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7036 027354
7037
7038      10$:
;SETUP PARAMETERS FOR GENERATING DATA BUFFER
7039 027354 012737 000000 001434      MOV    #0,RMOC0      ;CYLINDER = 0
7040 027362 012737 000000 001406      MOV    #0,RMOA0      ;TRACK = SECTOR = 0
7041 027370 012737 000000 001432      MOV    #0,RMOF0      ;18 BIT FORMAT
7042 027376 012737 177376 001402      MOV    #(<C<2+256.>+1),RMWC0 ;2 + 256 WORDS
7043 027404 012737 103566 001404      MOV    #BUFONE,RMBA0 ;DATA BUFFER ADDRESS
7044 027412 012737 000062 001400      MOV    #WH,RMCS10    ;WRITE HEADER AND DATA
7045 027420 012737 065106 001174      MOV    #ZEROS,$TMP0   ;USE ALL ZEROS DATA PATTERN
7046 027426 012737 000001 001176      MOV    #1,$TMP1
7047 027434 004737 036620      JSR     PC,GENBUF      ;GO GENERATE DATA BUFFER
7048 027440
7049
7050      20$:
;PREPARE DEVICE FOR DATA TRANSFER
7051 027440 004737 034000      JSR     PC,TSTPRP      ;PREPARE DEVICE FOR TEST
7052 027444 154130
7053 027446 000404      .WORD 154130 ;TASK DESCRIPTOR
7054 027450 000240      BR     30$           ;GO TO 30$ IF NO ERROR
7055 027452 104000      NOP
7056 027454 000137 030050      ERROR 180$          ;RETURN HERE IF ERROR
7057 027460      JMP     180$          ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7058
7059      30$:
;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
7060 027460 012737 000063 001400      MOV    #WH,GO,RMCS10 ;WRITE HEADER AND DATA
7061 027466 012702 001535      MOV    #PUTINX,R2    ;WRITE REGISTER INDEX TABLE
7062 027472 112722 000006      MOV     #RMDA,(R2)+
7063 027476 112722 000034      MOV     #RMOC,(R2)+
7064 027502 112722 000032      MOV     #RMOF,(R2)+
7065 027506 112722 000002      MOV     #RMLC,(R2)+
7066 027512 112722 000004      MOV     #RMBA,(R2)+
7067 027516 112722 000000      MOV     #RMCS1,(R2)+
7068 027522 112722 000200      MOV     #200,(R2)+
7069 027526 004737 037766      JSR     PC,PUT        ;GO WRITE REGISTERS VIA PUT SUB
7070 027530 000404      BR     80$           ;GO TO 80$ IF NO ERROR
7071 027534 000240      NOP
7072 027536 104000      ERROR 180$          ;RETURN HERE IF ERROR
7073 027540 000137 030050      JMP     180$          ;ERROR DEFINED BY PUT SUB
7074 027544
7075
7076      80$:
;SETUP INPUT REGISTER BUFFER FOR READING STATUS
7077 027544 004737 037432      JSR     PC,GETSTS
7078
7079      ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
7080 027550 004737 040326      JSR     PC,TIMOUT    ;WAIT FOR COMMAND TO COMPLETE
7081 027554 004737 037516      JSR     PC,GET        ;GO READ REGISTERS VIA GET SUB
7082 027560 000404      BR     90$           ;GO TO 90$ IF NO ERROR
7083 027562 000240      NOP
7084 027564 104000      ERROR 180$          ;RETURN HERE IF ERROR
7085 027566 000137 030050      JMP     180$          ;ERROR DEFINED BY GET SUB
7086 027572
7087
7088      90$:
;VERIFY RESULTS OF WRITE COMMAND
```

```

7089 027572 004737 040512      JSR    PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
7090 027576 000405              BR      100$          ;GO TO 100$ IF NO ERROR
7091 027600 000240              NOP                     ;RETURN HERE IF ERROR
7092 027602 104000              ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7093 027604 004736              JSR    PC,2(SP)+        ;GO BACK FOR MORE ERROR CHECKS
7094 027606 000137 030050      JMP      180$          ;GO TO 180$ IF ERROR WAS FOUND
7095 027612              100$:
7096 027612 004737 053016      JSR    PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
7097 027616 000405              BR      110$          ;GO TO 110$ IF NO ERROR
7098 027620 000240              NOP                     ;RETURN HERE IF ERROR
7099 027622 104000              ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7100 027624 004736              JSR    PC,2(SP)+        ;GO BACK FOR MORE ERROR CHECKS
7101 027626 000137 030050      JMP      180$          ;GO TO 180$ IF ERROR WAS FOUND
7102 027632              110$:
7103 027632 004737 041344      JSR    PC,SECERR      ;GO CHECK FOR SECONDARY ERRORS
7104 027636 000405              BR      120$          ;GO TO 120$ IF NO ERROR
7105 027640 000240              NOP                     ;RETURN HERE IF ERROR
7106 027642 104000              ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7107 027644 004736              JSR    PC,2(SP)+        ;GO BACK FOR MORE ERROR CHECKS
7108 027646 000137 030050      JMP      180$          ;GO TO 180$ IF ERROR WAS FOUND
7109 027652              120$:
7110
7111
7112      ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN AT 16 BIT FORMAT
7113 027652 012737 010000 001432      MOV     #ENT16,RMOEO      ;CHANGE TO 16 BIT FORMAT
7114 027656 012737 000073 001400      MOV     #AH,GO,RHCS10    ;READ HEADER & DATA COMMAND
7115 027666 012737 104572 001404      MOV     #BUFTWO,RMBAO    ;CHANGE BUS ADDRESS
7116 027674 004737 037766              JSR    PC,PUT      ;GO WRITE E-16 BYTES VIA PUT SUB
7117 027700 000404              BR      130$          ;GO TO 130$ IF NO ERROR
7118 027702 000240              NOP                     ;RETURN HERE IF ERROR
7119 027704 104000              ERROR                   ;ERROR DEFINED BY PUT SUB
7120 027706 000137 030050      JMP      180$          ;GO TO 180$ IF ERROR WAS FOUND
7121 027712              130$:
7122
7123      ;WAIT FOR READ TO COMPLETE AND GET STATUS
7124 027712 004737 040326      JSR    PC,TIMOUT      ;WAIT FOR READ TO COMPLETE
7125 027716 004737 037516      JSR    PC,GET      ;GO READ REGISTERS VIA GET SUB
7126 027722 000404              BR      140$          ;GO TO 140$ IF NO ERROR
7127 027724 000240              NOP                     ;RETURN HERE IF ERROR
7128 027726 104000              ERROR                   ;ERROR DEFINED BY GET SUB
7129 027730 000137 030050      JMP      180$          ;GO TO 180$ IF ERROR WAS FOUND
7130 027734              140$:
7131
7132      ;VERIFY THE RESULTS OF READ OPERATION
7133 027734 004737 040512      JSR    PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
7134 027740 000405              BR      150$          ;GO TO 150$ IF NO ERROR
7135 027742 000240              NOP                     ;RETURN HERE IF ERROR
7136 027744 104000              ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7137 027746 004736              JSR    PC,2(SP)+        ;GO BACK FOR MORE ERROR CHECKS
7138 027750 000137 030050      JMP      180$          ;GO TO 180$ IF ERROR WAS FOUND
7139 027754              150$:
7140
7141      ;VERIFY THAT FORMAT ERROR IS SET
7142 027754 032737 000020 001344      BIT     #FER,RMER11    ;IS FORMAT ERROR SET??
7143 027762 001022              BNE     160$          ;YES!!
7144 027764 004737 053016      JSR    PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER

```



```
7145 027770 000405 BR 155$ ;GO TO 155$ IF NO ERROR
7146 027772 000240 NOP ;RETURN HERE IF ERROR
7147 027774 104000 ERROR ;ERROR # DEFINED BY DTASTS SUBROUTINE
7148 027776 004736 JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
7149 030000 000137 030050 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
7150 030004 013737 001344 001142 155$: MOV R11,$BODAT ;BAD DATA FOR TYPEOUT
7151 030012 013737 001344 001140 MOV R11,$GDDAT ;EXPECTED DATA
7152 030020 052737 000020 001140 BIS #FER,$GDDAT
7153 030026 104343 ERROR 343 ;FORMAT ERROR NOT SET
7154 030030 160$: JSR PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
7155 030030 004737 041344 BR 170$ ;GO TO 170$ IF NO ERROR
7156 030034 000405 NOP ;RETURN HERE IF ERROR
7157 030036 000240 ERROR ;ERROR # DEFINED BY SECERR SUBROUTINE
7158 030040 104000 JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
7159 030042 004736 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
7160 030044 000137 030050 170$:
7161 030050 180$:
7162 030050
7163
7164 ;*****
7165 ;TEST 33 FORMAT ERROR TEST - 16
7166 ;*****
7167 TST33:
7168 030050 000004 SCOPE ;SCOPE CALL
7169 030052 000240 NOP ;START OF TEST
7170 030054 012706 001100 MOV #STACK,SP ;INITIALIZE STACK POINTER
7171 030060 013700 001276 MOV $BASE,R0 ;R0=UNIBUS ADDRESS
7172 030064 013701 001450 MOV TSTQUE,R1 ;(R1) = DEVICE BEING TESTED
7173 030070 012737 000033 001226 MOV #33,$TESTN ;SET TEST NUMBER IN APT MAIL BOX
7174 030076 10$:
7175 ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
7176 MOV #0,RMDC0 ;CYLINDER = 0
7177 030076 012737 000000 001434 MOV #0,RMDA0 ;TRACK = SECTOR = 0
7178 030104 012737 000000 001406 MOV #FMT16,RMOFO ;16 BIT FORMAT
7179 030112 012737 010000 001432 MOV #<1C(2+256.)+1>,RMWCO ;2 + 256 WORDS
7180 030120 012737 177376 001402 MOV #BUFONE,RMBAO ;DATA BUFFER ADDRESS
7181 030126 012737 103566 001404 MOV #WH,RMCS10 ;WRITE HEADER AND DATA
7182 030134 012737 000062 001400 MOV #ZEROS,$TMP0 ;USE ALL ZEROS DATA PATTERN
7183 030142 012737 065106 001174 MOV #1,$TMP1
7184 030150 012737 000001 001176 JSR PC,GENBUF ;GO GENERATE DATA BUFFER
7185 030156 004737 036620 20$:
7186 030162
7187
7188 ;PREPARE DEVICE FOR DATA TRANSFER
7189 030162 004737 034000 JSR PC,TSTPRP ;PREPARE DEVICE FOR TEST
7190 030166 154130 .WORD 154130 ;TASK DESCRIPTOR
7191 030170 000404 BR 30$ ;GO TO 30$ IF NO ERROR
7192 030172 000240 NOP ;RETURN HERE IF ERROR
7193 030174 104000 ERROR ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7194 030176 000137 030572 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
7195 030202 30$:
7196
7197 ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
7198 030202 012737 000063 001400 MOV #WH!GO,RMCS10 ;WRITE HEADER AND DATA
7199 030210 012702 001535 MOV #PUTINX,R2 ;WRITE REGISTER INDEX TABLE
7200 030214 112722 000006 MOVB #RMDA,(R2)+
```

B12

7201 030220 112722 000034
7202 030220 112722 000032
7203 030220 112722 000002
7204 030220 112722 000004
7205 030220 112722 000000
7206 030220 112722 000200
7207 030220 004737 037766
7208 030220 000404
7209 030220 000240
7210 030220 104000
7211 030220 000137 030572
7212 030266
7213
7214
7215 030266 004737 037432
7216
7217
7218 030272 004737 040326
7219 030272 004737 037516
7220 030302 000404
7221 030304 000240
7222 030306 104000
7223 030310 000137 030572
7224 030314
7225
7226
7227 030314 004737 040512
7228 030320 000405
7229 030322 000240
7230 030324 104000
7231 030326 004736
7232 030330 000137 030572
7233 030334
7234 030334 004737 053016
7235 030340 000405
7236 030342 000240
7237 030344 104000
7238 030346 004736
7239 030350 000137 030572
7240 030354
7241 030354 004737 041344
7242 030360 000405
7243 030362 000240
7244 030364 104000
7245 030366 004736
7246 030370 000137 030572
7247 030374
7248
7249
7250
7251 030374 012737 000000 001432
7252 030402 012737 000073 001400
7253 030410 012737 104572 001404
7254 030416 004737 037766
7255 030422 000404
7256 030424 000240

MOVW #RMC0, (R2)+
MOVW #RMC1, (R2)+
MOVW #RMC2, (R2)+
MOVW #RMC3, (R2)+
MOVW #RMC4, (R2)+
MOVW #RMC5, (R2)+
MOVW #200, (R2)+
JSR PC, PUT ;GO WRITE REGISTERS VIA PUT SUB
BR 80\$;GO TO 80\$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR DEFINED BY PUT SUB
JMP 180\$;GO TO 180\$ IF ERROR WAS FOUND
80\$:
;SETUP INPUT REGISTER BUFFER FOR READING STATUS
JSR PC, GETSYS
;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
JSR PC, TIMEOUT ;WAIT FOR COMMAND TO COMPLETE
JSR PC, GET ;GO READ REGISTERS VIA GET SUB
BR 90\$;GO TO 90\$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR DEFINED BY GET SUB
JMP 180\$;GO TO 180\$ IF ERROR WAS FOUND
90\$:
;VERIFY RESULTS OF WRITE COMMAND
JSR PC, PRIERR ;GO CHECK FOR PRIMARY ERRORS
BR 100\$;GO TO 100\$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR # DEFINED BY PRIERR SUBROUTINE
JSR PC, 2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
JMP 180\$;GO TO 180\$ IF ERROR WAS FOUND
100\$:
JSR PC, DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
BR 110\$;GO TO 110\$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR # DEFINED BY DTASTS SI ROL E
JSR PC, 2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
JMP 180\$;GO TO 180\$ IF ERROR WAS FOUND
110\$:
JSR PC, SECERR ;GO CHECK FOR SECONDARY ERRORS
BR 120\$;GO TO 120\$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
ERROR ;ERROR # DEFINED BY SECERR SUBROUTINE
JSR PC, 2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
JMP 180\$;GO TO 180\$ IF ERROR WAS FOUND
120\$:
;READ HEADER AND DATA FOR SECTOR JUST WRITTEN AT 18 BIT FORMAT
MOV #0, RMC0 ;SET FOR 18 BIT FORMAT
MOV #RA, RMC10 ;READ HEADER & DATA COMMAND
MOV #BUFTWO, RMC10 ;CHANGE BUS ADDRESS
JSR PC, PUT ;GO WRITE REGISTERS VIA PUT SUB
BR 130\$;GO TO 130\$ IF NO ERROR
NOP ;RETURN HERE IF ERROR

```

7257 030426 104000          ERROR          ;ERROR DEFINED BY PUT SUB
7258 030430 000137 030572    JMP          180$    ;GO TO 180$ IF ERROR WAS FOUND
7259 030434
7260
7261          ;WAIT FOR READ TO COMPLETE AND GET STATUS
7262 030434 004737 040326    JSR          PC,TIMOUT          ;WAIT FOR READ TO COMPLETE
7263 030440 004737 037516    JSR          PC,GET          ;GO READ REGISTERS VIA GET SUB
7264 030444 000404          BR          140$    ;GO TO 140$ IF NO ERROR
7265 030446 000240          NOP          ;RETURN HERE IF ERROR
7266 030450 104000          ERROR          ;ERROR DEFINED BY GET SUB
7267 030452 000137 030572    JMP          180$    ;GO TO 180$ IF ERROR WAS FOUND
7268 030456
7269
7270          ;VERIFY THE RESULTS OF READ OPERATION
7271 030456 004737 040512    JSR          PC,PRIERR          ;GO CHECK FOR PRIMARY ERRORS
7272 030462 000405          BR          150$    ;GO TO 150$ IF NO ERROR
7273 030464 000240          NOP          ;RETURN HERE IF ERROR
7274 030466 104000          ERROR          ;ERROR # DEFINED BY PRIERR SUBROUTINE
7275 030470 004736          JSR          PC,(SP)+          ;GO BACK FOR MORE ERROR CHECKS
7276 030472 000137 030572    JMP          180$    ;GO TO 180$ IF ERROR WAS FOUND
7277 030476
7278
7279          ;VERIFY THAT FORMAT ERROR IS SET
7280 030476 032737 000020 001344 BIT          #FER,RMER11          ;IS FORMAT ERROR SET??
7281 030476 001022          BNE          160$    ;YES!!
7282 030476 004737 053016    JSR          PC,DTASTS          ;GO VERIFY RESULTS OF DATA TRANSFER
7283 030476 000405          BR          155$    ;GO TO 155$ IF NO ERROR
7284 030476 000240          NOP          ;RETURN HERE IF ERROR
7285 030476 000240          ERROR          ;ERROR # DEFINED BY DTASTS SUBROUTINE
7286 030476 004736          JSR          PC,(SP)+          ;GO BACK FOR MORE ERROR CHECKS
7287 030476 000137 030572    JMP          180$    ;GO TO 180$ IF ERROR WAS FOUND
7288 030476 013737 001344 001142 155$: MOV          RMER11,$BNDAT          ;RECEIVED STATUS FOR TYPEOUT
7289 030476 013737 001344 001140 MOV          RMER11,$CDDAT          ;EXPECTED STATUS
7290 030476 052737 000020 001140 BIS          #FER,$CDDAT
7291 030476 104343          ERROR          343          ;FORMAT ERROR NOT SET
7292
7293          ;GO CHECK FOR SECONDARY ERRORS
7294 030476 004737 041344    JSR          PC,SECERR          ;GO CHECK FOR SECONDARY ERRORS
7295 030476 000405          BR          170$    ;GO TO 170$ IF NO ERROR
7296 030476 000240          NOP          ;RETURN HERE IF ERROR
7297 030476 000240          ERROR          ;ERROR # DEFINED BY SECERR SUBROUTINE
7298 030476 004736          JSR          PC,(SP)+          ;GO BACK FOR MORE ERROR CHECKS
7299 030476 000137 030572    JMP          180$    ;GO TO 180$ IF ERROR WAS FOUND
7300 030572
7301
7302          ;*****
7303          ;TEST 34          FORMAT HCE, FIRST HEADER WORD
7304          ;*****
7305          ;TST34:
7306          SCOPE
7307          NOP
7308 030576 012706 001100    MOV          #STACK,SP          ;SCOPE CALL
7309 030576 013700 001276    MOV          $BASE,R0          ;START OF TEST
7310 030606 013701 001450    MOV          TSTACK,R1          ;INITIALIZE STACK POINTER
7311 030612 012737 000034 001226 MOV          #34,$TESTN          ;(R1) = DEVICE BEING TESTED
7312 030620 012737 000001 031576 MOV          #1,190$          ;SET TEST NUMBER IN APT MAIL BOX
                          ;INIT BIT POSITION

```

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 146
T34 FORMAT HCE, FIRST HEADER WORD

SEQ 0146

7313	030626				10\$:	
7314						
7315						;SETUP PARAMETERS FOR GENERATING DATA BUFFER
7316	030626	012737	000000	001434	MOV	#0,RMOC0 ;CYLINDER = 0
7317	030634	012737	000000	001406	MOV	#0,RMDA0 ;TRACK = SECTOR = 0
7318	030642	012737	010000	001432	MOV	#FMT16,RMOF0 ;16 BIT FORMAT
7319	030650	012737	177376	001402	MOV	#(PC(2,256)+1),RMWC0 ;2 + 256 WORDS
7320	030656	012737	103566	001404	MOV	#BUFONE,RMBA0 ;DATA BUFFER ADDRESS
7321	030664	012737	000062	001400	MOV	#WH,RMCS10 ;WRITE HEADER AND DATA
7322	030672	012737	065106	001174	MOV	#ZEROS,STMP0 ;USE ALL ZEROS DATA PATTERN
7323	030700	012737	000001	001176	MOV	#1,STMP1
7324	030706	004737	036620		JSR	PC,GENBUF ;GO GENERATE DATA BUFFER
7325	030712					
7326					20\$:	
7327						;PREPARE DEVICE FOR DATA TRANSFER
7328	030712	004737	034000		JSR	PC,TSTPRP ;PREPARE DEVICE FOR TEST
7329	030716	154130			WORD	154130 ;TASK DESCRIPTOR
7330	030720	000404			BR	30\$;GO TO 30\$ IF NO ERROR
7331	030722	000240			NOP	;
7332	030724	104000			ERROR	;
7333	030726	000137	031750		JMP	240\$;GO TO 240\$ IF ERROR WAS FOUND
7334	030732					
7335					30\$:	
7336						;COMPLEMENT DATA BIT IN FIRST HEADER WORD
7337	030732	033737	031576	103566	BIT	190\$,BUFONE ;IS BIT IN HEADER ON??
7338	030740	001404			BEQ	31\$;NO!!
7339	030742	043737	031576	103566	BIC	190\$,BUFONE ;RESET BIT IN HEADER
7340	030750	000403			BR	32\$
7341	030752	053737	031576	103566	31\$:	BIS 190\$,BUFONE ;SET BIT IN HEADER
7342						
7343	030760				32\$:	
7344						
7345						;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
7346	030760	012737	000063	001400	MOV	#WH,GO,RMCS10 ;WRITE HEADER AND DATA
7347	030766	012702	001535		MOV	#PUTINX,R2 ;WRITE REGISTER INDEX TABLE
7348	030772	112722	000006		MOVB	#RMDA,(R2)+
7349	030776	112722	000034		MOVB	#RMOC,(R2)+
7350	031002	112722	000032		MOVB	#RMOF,(R2)+
7351	031006	112722	000002		MOVB	#RMWC,(R2)+
7352	031012	112722	000004		MOVB	#RMBA,(R2)+
7353	031016	112722	000000		MOVB	#RMCS1,(R2)+
7354	031022	112722	000200		MOVB	#200,(R2)+
7355	031026	004737	037766		JSR	PC,PUT ;GO WRITE REGISTERS VIA PUT SUB
7356	031032	000404			BR	80\$;GO TO 80\$ IF NO ERROR
7357	031034	000240			NOP	;
7358	031036	104000			ERROR	;
7359	031040	000137	031574		JMP	180\$;GO TO 180\$ IF ERROR WAS FOUND
7360	031044					
7361					80\$:	
7362						;SETUP INPUT REGISTER BUFFER FOR READING STATUS
7363	031044	004737	037432		JSR	PC,GETSTS
7364						
7365						;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
7366	031050	004737	040326		JSR	PC,TIMOUT ;WAIT FOR COMMAND TO COMPLETE
7367	031054	004737	037516		JSR	PC,GET ;GO READ REGISTERS VIA GET SUB
7368	031060	000404			BR	90\$;GO TO 90\$ IF NO ERROR

```

7369 031062 000240      NOP      ;RETURN HERE IF ERROR
7370 031064 104000      ERROR     ;ERROR DEFINED BY GET SUB
7371 031066 000137 031574 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
7372 031072
7373
7374 ;VERIFY RESULTS OF WRITE COMMAND
7375 031072 004737 040512 JSR PC, PRIERR ;GO CHECK FOR PRIMARY ERRORS
7376 031076 000405      BR 100$ ;GO TO 100$ IF NO ERROR
7377 031100 000240      NOP      ;RETURN HERE IF ERROR
7378 031102 104000      ERROR     ;ERROR # DEFINED BY PRIERR SUBROUTINE
7379 031104 004736      JSR PC, 2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
7380 031106 000137 031574 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
7381 031112
7382 031112 004737 053016 JSR PC, DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
7383 031116 000405      BR 110$ ;GO TO 110$ IF NO ERROR
7384 031120 000240      NOP      ;RETURN HERE IF ERROR
7385 031122 104000      ERROR     ;ERROR # DEFINED BY DTASTS SUBROUTINE
7386 031124 004736      JSR PC, 2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
7387 031126 000137 031574 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
7388 031132
7389 031132 004737 041344 JSR PC, SECERR ;GO CHECK FOR SECONDARY ERRORS
7390 031136 000405      BR 120$ ;GO TO 120$ IF NO ERROR
7391 031140 000240      NOP      ;RETURN HERE IF ERROR
7392 031142 104000      ERROR     ;ERROR # DEFINED BY SECERR SUBROUTINE
7393 031144 004736      JSR PC, 2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
7394 031146 000137 031574 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
7395 031152
7396
7397 ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
7398 MOV #RH!GO, RMCSIO ;READ HEADER & DATA COMMAND
7399 031152 012737 000073 001400 MOV #BUFTWO, RMBAO ;CHANGE BUS ADDRESS
7400 031160 012737 104572 001404 JSR PC, PUT ;GO WRITE REGISTERS VIA PUT SUB
7401 031166 004737 037766 BR 130$ ;GO TO 130$ IF NO ERROR
7402 031172 000404      NOP      ;RETURN HERE IF ERROR
7403 031174 000240      ERROR     ;ERROR DEFINED BY PUT SUB
7404 031176 104000      JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
7405 031200 000137 031574
7406 031204
7407
7408 ;WAIT FOR READ TO COMPLETE AND GET STATUS
7409 031204 004737 040326 JSR PC, TIMEOUT ;WAIT FOR READ TO COMPLETE
7410 031210 004737 037516 JSR PC, GET ;GO READ REGISTERS VIA GET SUB
7411 031214 000404      BR 140$ ;GO TO 140$ IF NO ERROR
7412 031216 000240      NOP      ;RETURN HERE IF ERROR
7413 031220 104000      ERROR     ;ERROR DEFINED BY GET SUB
7414 031222 000137 031574 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
7415 031226
7416
7417 ;VERIFY THE RESULTS OF READ OPERATION
7418 031226 004737 040512 JSR PC, PRIERR ;GO CHECK FOR PRIMARY ERRORS
7419 031232 000405      BR 150$ ;GO TO 150$ IF NO ERROR
7420 031234 000240      NOP      ;RETURN HERE IF ERROR
7421 031236 104000      ERROR     ;ERROR # DEFINED BY PRIERR SUBROUTINE
7422 031240 004736      JSR PC, 2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
7423 031242 000137 031574 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
7424 031246

```

```

7425 031246 032737 140000 031576 BIT #MSE!USE,190$ ;IS THIS BSE ??
7426 031254 001033 BNE 152$ ;YES !!
7427 031256 032737 010000 031576 BIT #FM16,190$ ;IS THIS FER ??
7428 031264 001466 BEQ 155$ ;NO !!
7429
7430 ;VERIFY THAT FER IS SET
7431 031266 032737 000020 001344 BIT #FER,RMER11 ;IS FER SET ??
7432 031274 001122 BNE 160$ ;YES !!
7433 031276 004737 053016 JSR PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
7434 031302 000405 BR 151$ ;GO TO 151$ IF NO ERROR
7435 031304 000240 NOP ;RETURN HERE IF ERROR
7436 031306 104000 ERROR # DEFINED BY DTASTS SUBROUTINE
7437 031310 004736 JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
7438 031312 000137 031574 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
7439 031316 013737 001344 001142 151$: MOV R11,$B00AT ;RECEIVED STATUS
7440 031324 013737 001344 001140 MOV RMER11,$C00AT ;EXPECTED STATUS
7441 031332 052737 000020 001140 BIS #FER,$C00AT
7442 031340 104343 ERROR 343 ;FORMAT ERROR NOT SET
7443 031342 000514 BR 180$
7444 031344
7445 152$: ;VERIFY THAT BAD SECTOR ERROR IS SET
7446 031344 032737 100000 001372 BIT #B E,RMER21 ;IS BSE SET ??
7447 031352 001073 BNE 160$ ;YES !!
7448 031354 004737 053016 JSR PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
7449 031360 000405 BR 153$ ;GO TO 153$ IF NO ERROR
7450 031362 000240 NOP ;RETURN HERE IF ERROR
7451 031364 104000 ERROR # DEFINED BY DTASTS SUBROUTINE
7452 031366 004736 JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
7453 031370 000137 031574 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
7454 031374
7455 153$: MOV R11,$B00AT ;RECEIVED STATUS
7456 031374 013737 001372 001142 MOV RMER21,$C00AT ;EXPECTED STATUS
7457 031402 013737 001372 001140 BIS #B E,$C00AT
7458 031410 052737 104700 MOV BUFO1,$TMP0 ;WRITTEN HEADER
7459 031416 013737 104700 001174 MOV BUFTWO,$TMP1 ;READ HEADER WORD
7460 031424 013737 104572 001176 MOV 190$,$TMP2 ;FAILING BIT POSITION
7461 031432 013737 031576 001200 MOV 345 ;FAILED TO DETECT BSE
7462 031440 000455 BR 180$ ;SKIP REST OF TEST
7463 031442
7464 155$:
7465 ;VERIFY THAT HEADER COMPARE ERROR IS SET
7466 031442 032737 000200 001344 BIT #HCE,RMER11 ;IS "HCE" SET??
7467 031450 001034 BNE 160$ ;YES !!
7468 031452 004737 053016 JSR PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
7469 031456 000405 BR 156$ ;GO TO 156$ IF NO ERROR
7470 031460 000240 NOP ;RETURN HERE IF ERROR
7471 031462 104000 ERROR # DEFINED BY DTASTS SUBROUTINE
7472 031464 004736 JSR PC,2(SP)+ ;GO BACK FOR MORE ERROR CHECKS
7473 031466 000137 031574 JMP 180$ ;GO TO 180$ IF ERROR WAS FOUND
7474 031472 013737 001344 001142 156$: MOV R11,$B00AT ;RECEIVED STATUS FOR TYPEOUT
7475 031500 013737 001344 001140 MOV RMER11,$C00AT ;EXPECTED STATUS
7476 031506 052737 000200 001140 BIS #HCE,$C00AT
7477 031514 013737 103566 001174 MOV BUFO1,$TMP0 ;WRITTEN HEADER
7478 031522 013737 104572 001176 MOV BUFTWO,$TMP1 ;READ HEADER WORD
7479 031530 013737 031576 001200 MOV 190$,$TMP2 ;FAILING BIT POSITION
7480 031536 104344 ERROR 344 ;FORMAT ERROR NOT SET

```

CZRMDB0 RM03/2 FCTNL TST 2
CZRMDB.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 149
T34 FORMAT HCE, FIRST HEADER WORD

SEQ 0149

7481	031540	000415			BR	180\$	
7482							
7483	031542				160\$:		
7484							
7485					;CHECK FOR OTHER ERRORS		
7486	031542	004737	041344		JSR	PC, SECERR	;GO CHECK FOR SECONDARY ERRORS
7487	031546	000405			BR	165\$;GO TO 165\$ IF NO ERROR
7488	031550	000240			NOP		;RETURN HERE IF ERROR
7489	031552	104000			ERROR		;ERROR # DEFINED BY SECERR SUBROUTINE
7490	031554	004736			JSR	PC, 2(SP)+	;GO BACK FOR MORE ERROR CHECKS
7491	031556	000137	031574		JMP	180\$;GO TO 180\$ IF ERROR WAS FOUND
7492	031562				165\$:		
7493							
7494							
7495							
7496					;ADVANCE THE BIT POSITION AND FORMAT AGAIN IF NOT DONE		
7497	031562	006337	031576		ASL	190\$;SHIFT TO NEXT BIT POSITION
7498	031566	001404			BEQ	200\$;EXIT IF DONE
7499	031570	000137	030626		JMP	10\$;GO DO NEXT SECTOR
7500	031574				180\$:		
7501	031574	000465			BR	240\$;JUMP OVER STORAGE
7502	031576				190\$:		
7503	031576	000000			.WORD		;STORAGE FOR BIT POSITION
7504	031600				200\$:		
7505					MOV	#0, RMDCO	;REFORMAT THE SECTOR IN 16 BIT MODE
7506					MOV	#0, RMDAO	
7507	031600	012737	010000	001432	MOV	#FMT16, RMOFO	;16 BIT MODE
7508	031606	012737	177776	001402	MOV	#-2, RMCWC	;ONLY TWO HEADER WORDS
7509	031614	012737	103566	001404	MOV	#BUFONE, RMBAG	;SELECT THE BUFFER
7510	031622	012737	000062	001400	MOV	#WH, RMCS10	;WRITE HEAD AND DATA COMMAND
7511	031630	004737	036620		JSR	PC, GENBUF	;GENERATE THE BUFFER PATTERN
7512							
7513	031634	004737	034000		JSR	PC, TSTPRP	;PREPARE DEVICE FOR TEST
7514	031640	154130			.WORD	154130	;TASK DESCRIPTOR
7515	031642	000404			BR	210\$;GO TO 210\$ IF NO ERROR
7516	031644	000240			NOP		;RETURN HERE IF ERROR
7517	031646	104000			ERROR		;ERROR # DEFINED BY TSTPRP SUBROUTINE
7518	031650	000137	031654		JMP	210\$;GO TO 210\$ IF ERROR WAS FOUND
7519	031654	012737	000063	001400	MOV	#WH!GO, RMCS10	;SET THE GO BIT
7520	031662	012702	001535		MOV	#FUTINX, R2	;PUT THE REGISTER ADDRESS INTO TABLE
7521	031666	112722	000006		MOVB	#RMDA, (R2)+	
7522	031672	112722	000034		MOVB	#RMDC, (R2)+	
7523	031676	112722	000032		MOVB	#RMOF, (R2)+	
7524	031702	112722	000002		MOVB	#RMWC, (R2)+	
7525	031706	112722	000004		MOVB	#RMBB, (R2)+	
7526	031712	112722	000000		MOVB	#RMCS1, (R2)+	
7527	031716	112722	000200		MOVB	#200, (R2)+	;TERMINATOR
7528	031722	004737	037766		JSR	PC, PUT	;GO WRITE REGISTERS VIA PUT SUB
7529	031726	000404			BR	220\$;GO TO 220\$ IF NO ERROR
7530	031730	000240			NOP		;RETURN HERE IF ERROR
7531	031732	104000			ERROR		;ERROR DEFINED BY PUT SUB
7532	031734	000137	031750		JMP	240\$;GO TO 240\$ IF ERROR WAS FOUND
7533	031740	000240			220\$:		
7534	031742	004737	040326		NOP		
7535	031746	000240			JSR	PC, TIMEOUT	;WAIT UNTIL IT FINISH
7536	031750				240\$:		

```

7537
7538
7539
7540
7541 031750
7542 031750 000004
7543 031752 000240
7544 031754 012706 001100
7545 031760 013700 001276
7546 031764 013701 001450
7547 031770 012737 000035 001226
7548 031776 012737 000001 032560
7549 032004
7550
7551
7552 032004 012737 000000 001434
7553 032012 012737 000000 001406
7554 032020 012737 010000 001432
7555 032026 012737 177376 001402
7556 032034 012737 103566 001404
7557 032042 012737 000062 001400
7558 032050 012737 065106 001174
7559 032056 012737 000001 001176
7560 032064 004737 036620
7561 032070
7562
7563
7564 032070 004737 034000
7565 032074 154130
7566 032076 000404
7567 032100 000240
7568 032102 104000
7569 032104 000137 032732
7570 032110
7571
7572
7573 032110 033737 032560 103570
7574 032116 001404
7575 032120 043737 032560 103570
7576 032126 000403
7577 032130 053737 032560 103570
7578
7579 032136
7580
7581
7582 032136 012737 000063 001400
7583 032144 012702 001535
7584 032150 112722 000006
7585 032154 112722 000034
7586 032160 112722 000032
7587 032164 112722 000002
7588 032170 112722 000004
7589 032174 112722 000000
7590 032200 112722 000200
7591 032204 004737 037766
7592 032210 000404

;*****
;TEST 35 FORMAT HCE, SECOND HEADER WORD
;*****
TST35:
        SCOPE
        NOP
        MOV #STACK, SP
        MOV $BASE, R0
        MOV TSTQUE, R1
        MOV #35, $TESTN
        MOV #1, $190$

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

10$:
;SETUP PARAMETERS FOR GENERATING DATA BUFFER
        MOV #0, RMDC0
        MOV #0, RMDA0
        MOV #FMT16, RMOFO
        MOV #(<1C<2+256>+1>), RMWC0
        MOV #BUFONE, RMBA0
        MOV #WH, RMCS10
        MOV #ZEROS, $TMP0
        MOV #1, $TMP1
        JSR PC, GENBUF

;CYLINDER = 0
;TRACK = SECTOR = 0
;16 BIT FORMAT
;2 + 256 WORDS
;DATA BUFFER ADDRESS
;WRITE HEADER AND DATA
;USE ALL ZEROS DATA PATTERN
;GO GENERATE DATA BUFFER

20$:
;PREPARE DEVICE FOR DATA TRANSFER
        JSR PC, TSTPRP
        .WORD 154130
        BR 30$
        NOP
        ERROR
        JMP 240$

;PREPARE DEVICE FOR TEST
;TASK DESCRIPTOR
;GO TO 30$ IF NO ERROR
;RETURN HERE IF ERROR
;ERROR # DEFINED BY TSTPRP SUBROUTINE
;GO TO 240$ IF ERROR WAS FOUND

30$:
;COMPLEMENT DATA BIT IN SECOND HEADER WORD
        BIT 190$, BUFONE+2
        BEQ 31$
        BIC 190$, BUFONE+2
        BR 32$
        BIS 190$, BUFONE+2

;IS BIT IN HEADER ON??
;NO!!
;RESET BIT IN HEADER
;SET BIT IN HEADER

31$:
32$:
;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
        MOV #WH, GO, RMCS10
        MOV #PUTINX, R2
        MOV #RMDA, (R2)+
        MOV #RMDC, (R2)+
        MOV #RMOF, (R2)+
        MOV #RMWC, (R2)+
        MOV #RMBA, (R2)+
        MOV #RMCS1, (R2)+
        MOV #200, (R2)+
        JSR PC, PUT
        BR 80$

;WRITE HEADER AND DATA
;WRITE REGISTER INDEX TABLE
;GO WRITE REGISTERS VIA PUT SUB
;GO TO 80$ IF NO ERROR

```



```

7593 032212 000240      NOP      ;RETURN HERE IF ERROR
7594 032214 104000      ERROR     ;ERROR DEFINED BY PUT SUB
7595 032216 000137 032556 JMP      180$    ;GO TO 180$ IF ERROR WAS FOUND
7596 032222
7597
7598
7599 032222 004737 037432 ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
7600      JSR      PC,GETSTS
7601
7602 032226 004737 040326 ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
7603 032232 004737 037516 JSR      PC,TIMOUT ;WAIT FOR COMMAND TO COMPLETE
7604 032236 000404      JSR      PC,GET ;GO READ REGISTERS VIA GET SUB
7605 032240 000240      BR      90$    ;GO TO 90$ IF NO ERROR
7606 032242 104000      NOP      ;RETURN HERE IF ERROR
7607 032244 000137 032556 ERROR     ;ERROR DEFINED BY GET SUB
7608 032250      JMP      180$    ;GO TO 180$ IF ERROR WAS FOUND
7609
7610
7611 032250 004737 040512 ;VERIFY RESULTS OF WRITE COMMAND
7612 032254 000405      JSR      PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
7613 032256 000240      BR      100$   ;GO TO 100$ IF NO ERROR
7614 032260 104000      NOP      ;RETURN HERE IF ERROR
7615 032262 004736      ERROR     ;ERROR # DEFINED BY PRIERR SUBROUTINE
7616 032264 000137 032556 JSR      PC,(SP)+ ;GO BACK FOR MORE ERROR CHECKS
7617 032270      JMP      180$   ;GO TO 180$ IF ERROR WAS FOUND
7618 032270 004737 053016 100$:      JSR      PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
7619 032274 000405      BR      110$   ;GO TO 110$ IF NO ERROR
7620 032276 000240      NOP      ;RETURN HERE IF ERROR
7621 032300 104000      ERROR     ;ERROR # DEFINED BY DTASTS SUBROUTINE
7622 032302 004736      JSR      PC,(SP)+ ;GO BACK FOR MORE ERROR CHECKS
7623 032304 000137 032556 JMP      180$   ;GO TO 180$ IF ERROR WAS FOUND
7624 032310
7625 032310 004737 041344 110$:      JSR      PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
7626 032314 000405      BR      120$   ;GO TO 120$ IF NO ERROR
7627 032316 000240      NOP      ;RETURN HERE IF ERROR
7628 032320 104000      ERROR     ;ERROR # DEFINED BY SECERR SUBROUTINE
7629 032322 004736      JSR      PC,(SP)+ ;GO BACK FOR MORE ERROR CHECKS
7630 032324 000137 032556 JMP      180$   ;GO TO 180$ IF ERROR WAS FOUND
7631 032330
7632
7633
7634
7635 032330 012737 000073 001400 ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
7636 032336 012737 104572 001404 MOV      #RH!GO,RMC510 ;READ HEADER & DATA COMMAND
7637 032344 004737 037766 MOV      #BUFTWO,RMBA0 ;CHANGE BUS ADDRESS
7638 032350 000404      JSR      PC,PUT ;GO WRITE REGISTERS VIA PUT SUB
7639 032352 000240      BR      130$   ;GO TO 130$ IF NO ERROR
7640 032354 104000      NOP      ;RETURN HERE IF ERROR
7641 032356 000137 032556 ERROR     ;ERROR DEFINED BY PUT SUB
7642 032362      JMP      180$   ;GO TO 180$ IF ERROR WAS FOUND
7643
7644
7645 032362 004737 040326 ;WAIT FOR READ TO COMPLETE AND GET STATUS
7646 032366 004737 037516 JSR      PC,TIMOUT ;WAIT FOR READ TO COMPLETE
7647 032372 000404      JSR      PC,GET ;GO READ REGISTERS VIA GET SUB
7648 032374 000240      BR      140$   ;GO TO 140$ IF NO ERROR
      NOP      ;RETURN HERE IF ERROR

```

```

7649 032376 104000          ERROR          ;ERROR DEFINED BY GET SUB
7650 032400 000137 032556    JMP          180$      ;GO TO 180$ IF ERROR WAS FOUND
7651 032404          140$:
7652
7653          ;VERIFY THE RESULTS OF READ OPERATION
7654 032404 004737 040512    JSR          PC, PRIERR      ;GO CHECK FOR PRIMARY ERRORS
7655 032410 000405          BR          150$      ;GO TO 150$ IF NO ERROR
7656 032412 000240          NOP          ;RETURN HERE IF ERROR
7657 032414 104000          ERROR          ;ERROR # DEFINED BY PRIERR SUBROUTINE
7658 032416 004736          JSR          PC, 2(SP)+    ;GO BACK FOR MORE ERROR CHECKS
7659 032420 000137 032556    JMP          180$      ;GO TO 180$ IF ERROR WAS FOUND
7660 032424          150$:
7661
7662          ;VERIFY THAT HEADER COMPARE ERROR IS SET
7663 032424 032737 000200 001344 BIT          #HCE, RMER11      ;IS "HCE" SET??
7664 032432 001034          BNE          160$      ;YES!!
7665 032434 004737 053016    JSR          PC, DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
7666 032440 000405          BR          155$      ;GO TO 155$ IF NO ERROR
7667 032442 000240          NOP          ;RETURN HERE IF ERROR
7668 032444 104000          ERROR          ;ERROR # DEFINED BY DTASTS SUBROUTINE
7669 032446 004736          JSR          PC, 2(SP)+    ;GO BACK FOR MORE ERROR CHECKS
7670 032450 000137 032556    JMP          180$      ;GO TO 180$ IF ERROR WAS FOUND
7671 032454 013737 001344 001142 155$: MOV          RMER11, $BDDAT      ;RECEIVED STATUS FOR TYPEOUT
7672 032462 013737 001344 001140      MOV          RMER11, $GDDAT      ;EXPECTED STATUS
7673 032470 052737 000200 001140      BIS          #HCE, $GDDAT
7674 032476 013737 103570 001174      MOV          BUFOFF+2, $TMP0      ;WRITTEN HEADER
7675 032504 013737 104574 001176      MOV          BUFTWO+2, $TMP1      ;READ HEADER WORD
7676 032512 013737 032560 001200      MOV          190$, $TMP2      ;FAILING BIT POSITION
7677 032520 104344          ERROR          344      ;FORMAT ERROR NOT SET
7678 032522 000415          BR          180$
7679
7680 032524          160$:
7681
7682          ;CHECK FOR OTHER ERRORS
7683 032524 004737 041344      JSR          PC, SECERR      ;GO CHECK FOR SECONDARY ERRORS
7684 032530 000405          BR          165$      ;GO TO 165$ IF NO ERROR
7685 032532 000240          NOP          ;RETURN HERE IF ERROR
7686 032534 104000          ERROR          ;ERROR # DEFINED BY SECERR SUBROUTINE
7687 032536 004736          JSR          PC, 2(SP)+    ;GO BACK FOR MORE ERROR CHECKS
7688 032540 000137 032556    JMP          180$      ;GO TO 180$ IF ERROR WAS FOUND
7689 032544          165$:
7690
7691          ;ADVANCE THE BIT POSITION AND FORMAT NEXT SECTOR IF NOT DONE
7692          ASL          190$      ;SHIFT TO NEXT BIT POSITION
7693 032544 006337 032560      BEQ          200$      ;EXIT IF DONE
7694 032550 001404          JMP          10$      ;GO DO NEXT SECTOR
7695 032552 000137 032004      180$: BR          240$      ;JUMP OVER STORAGE
7696 032556          190$:
7697 032556 000465          .WORD          ;STORAGE FOR BIT POSITION
7698 032560          200$:
7699 032560 000000          ;
7700 032562          ;
7701          MOV          #0, RMDCO      ;REFORMAT THE SECTOR IN 16 BIT MODE
7702          MOV          #0, RMDAO      ;SECTOR 0, TRACK 0, CYL 0
7703 032562 012737 010000 001432      MOV          #FMT16, RMOFO      ;16 BIT MODE
7704 032570 012737 177776 001402      MOV          #-2, RMWCO      ;2 HEADER WORDS ONLY

```

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 153
T35 FORMAT HCE, SECOND HEADER WORD

SEQ 0153

```

7705 032576 012737 103566 001404 MOV #BUFONE,RMB00 ;BUFFER ADDRESS
7706 032604 012737 000062 001400 MOV #WH,RMCS10 ;WRITE HEAD AND DATA COMMAND
7707 032612 004737 036620 JSR PC,GENBUF ;SET UP THE BUFFER
7708 032616 004737 034000 JSR PC,TSTPRP ;PREPARE DEVICE FOR TEST
7709 032622 154130 .WORD 154130 ;TASK DESCRIPTOR
7710 032624 000404 BR 210$ ;GO TO 210$ IF NO ERROR
7711 032626 000240 NOP ;RETURN HERE IF ERROR
7712 032630 104000 ERROR ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7713 032632 000137 032636 JMP 210$ ;GO TO 210$ IF ERROR WAS FOUND
7714 032636 012737 000063 001400 210$: MOV #WH!GO,RMCS10 ;SET GO BIT
7715 032644 012702 001535 MOV #PUTINX,R2 ;TABLE ADDRESS
7716 032650 112722 000006 MOV8 #RMDA,(R2)+
7717 032654 112722 000034 MOV8 #RMOC,(R2)+
7718 032660 112722 000032 MOV8 #RMOF,(R2)+
7719 032664 112722 000004 MOV8 #RMDA,(R2)+
7720 032670 112722 000002 MOV8 #RMWC,(R2)+
7721 032674 112722 000000 MOV8 #RMCS1,(R2)+
7722 032700 112722 000200 MOV8 #200,(R2)+ ;TERMINATOR
7723 032704 004737 037766 JSR PC,PUT ;GO WRITE REGISTERS VIA PUT SUB
7724 032710 000404 BR 220$ ;GO TO 220$ IF NO ERROR
7725 032712 000240 NOP ;RETURN HERE IF ERROR
7726 032714 104000 ERROR ;ERROR DEFINED BY PUT SUB
7727 032716 000137 032732 JMP 240$ ;GO TO 240$ IF ERROR WAS FOUND
7728 032722 000240 220$: NOP
7729 032724 004737 040326 JSR PC,TIMOUT ;WAIT FOR TIME OUT
7730 032730 000240 NOP
7731 032732 240$:
7732
7733 ;PUT NEWTEST HERE

```

```

7734 032732          SEOSP:
7735 032732          NOP
7736 032734 000240      MOV      TSTQUE,RO
7737 032740 013700 001450  ADD      #2,RO
7738 032744 062700 000002  MOV      RO,TSTQUE
7739 032750 010037 001450  MOV      (RO)
7740 032752 005710      TST
7741 032752 001402      BEQ      1$
7742 032754 000137 007124  JMP      READY
7743 032760 012737 001452 001450 1$: MOV      #TSTQUE+2,TSTQUE
7744          SBTTL      END OF PASS ROUTINE
7745          ;*****
7746          ;INCREMENT THE PASS NUMBER ($PASS)
7747          ;TYPE "END PASS XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY"
7748          ;WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
7749          ;IF THERES A MONITOR GO TO IT
7750          ;IF THERE ISN'T JUMP TO SHUT
7751
7752 032766          SEOP:
7753 032766 000240      NOP
7754 032770 005037 001116  CLR      $TSTNM          ;;ZERO THE TEST NUMBER
7755 032774 005037 001206  CLR      $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
7756 033000 005237 001230  INC      $PASS          ;;INCREMENT THE PASS NUMBER
7757 033004 042737 100000 001230  BIC      #100000,$PASS          ;;DON'T ALLOW A NEG. NUMBER
7758 033012 005327          DEC      (PC)+          ;;LOOP?
7759 033014 000001  SEOPCT: .WORD      1
7760 033016 003063      BGT      $DOAGN          ;;YES
7761 033020 012737      MOV      (PC)+,$2(PC)+      ;;RESTORE COUNTER
7762 033022 000001  SENDCT: .WORD      1
7763 033024 033014      SEOPCT
7764 033026 104401 033034      TYPE      65$          ;;TYPE ASCIZ STRING
7765 033032 000407      BR      64$          ;;GET OVER THE ASCIZ
7766          ;;65$: .ASCIZ      <12><15>/END PASS #/
7767          64$:
7768 033052 013746 001230      MOV      $PASS,-($P)          ;;SAVE $PASS FOR TYPEOUT
7769          ;;TYPE PASS NUMBER
7770 033056 104405          TYPDS          ;;GO TYPE--DECIMAL ASCII WITH SIGN
7771 033060 104401 033066      TYPE      67$          ;;TYPE ASCIZ STRING
7772 033064 000421      BR      66$          ;;GET OVER THE ASCIZ
7773          ;;67$: .ASCIZ      / TOTAL ERRORS SINCE LAST REPORT /
7774          66$:
7775 033130          MOV      $ERTTL,-($P)          ;;SAVE $ERTTL FOR TYPEOUT
7776          ;;TOTAL NUMBER OF ERRORS
7777 033134 104405          TYPDS          ;;GO TYPE--DECIMAL ASCII WITH SIGN
7778 033136 104401          TYPE      $CRLF          ;;TYPE CARRIAGE RETURN, LINE FEED
7779 033142 005037 001126  CLR      $ERTTL          ;;CLEAR ERROR TOTAL
7780 033146 013700 000042  MOV      #42,RO          ;;GET MONITOR ADDRESS
7781 033152 001405      BEQ      $DOAGN          ;;BRANCH IF NO MONITOR
7782 033154 000005      RESET          ;;CLEAR THE WORLD
7783 033156 004710      JSR      PC,(RO)          ;;GO TO MONITOR
7784 033160 000240      NOP          ;;SAVE ROOM
7785 033162 000240      NOP          ;;FOR
7786 033164 000240      NOP          ;;ACT11
7787          $DOAGN:
7788          JMP      2(PC)+          ;;RETURN
7789 033170 057364  SRTNAD: .WORD      SHUT

```

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 155
END OF PASS ROUTINE

M12

SEQ 0155

7790 033172 377 377
7791 033176

000 \$ENULL: .BYTE -1,-1,0 ; ; NULL CHARACTER STRING
.EVEN

```

7792 .SBTTL SUBROUTINES
7793
7794 ;*****
7795 .SBTTL ERROR TYPEOUT ROUTINE
7796
7797 ;*THE ERROR TYPEOUT ROUTINE ASSEMBLES AND PRINTS INFORMATION
7798 ;*REGARDING THE DETECTION OF AN ERROR AS FOLLOWS:
7799 ;*
7800 ;*   UNIT NUMBER, TEST NUMBER, ERROR NUMBER AND PROGRAM COUNTER ARE
7801 ;*   PRINTED ON THE FIRST LINE;
7802 ;*   ERROR MESSAGE IS ASSEMBLED, FORMATTED AND PRINTED ON
7803 ;*   ONE OR MORE SUCCEEDING LINES;
7804 ;*   PAIRED LINES OF ERROR HEADERS AND ERROR DATA
7805 ;*   ARE PRINTED AFTER THE ERROR MESSAGE.
7806
7807 ERRTP:
7808     SAVREG
7809     BIT     #SW13,SWR      ;INHIBIT TYPEOUTS??
7810     BEQ     1$            ;NO!!
7811     JMP     21$           ;YES!!
7812
7813     ;TYPE UNIT NUMBER, TEST NUMBER, ERROR NUMBER, AND PROGRAM COUNTER
7814     1$:    TYPE  $CRLF
7815           TYPE  $ERTY00
7816           MOV   $UNIT,-(SP)
7817           TYP0S
7818           .BYTE 3
7819           .BYTE 0
7820           CLR   TSTNMB
7821           MOV   $TSTN,TSTNMB
7822           TYPE  $ERTY01
7823           MOV   $TSTNMB,-(SP)
7824
7825           TYP0S
7826           .BYTE 3
7827           .BYTE 0
7828           CLR   ERRNMB
7829           MOVB  $ITEMB,ERRNMB
7830           BEQ   2$
7831           TYPE  $ERTY02
7832           MOV   $ERRNMB,-(SP)
7833
7834           TYP0S
7835           .BYTE 3
7836           .BYTE 0
7837           TYPE  $ERTY03
7838           MOV   $ERRPC,-(SP)
7839
7840           TYP0S
7841           .BYTE 6
7842           .BYTE 1
7843           ;GENERATE POINTER TO ERROR TABLE
7844           3$:  TST   ERRNMB
7845                 BEQ   21$
7846                 TYPE  $CRLF
7847                 CLPB  $BOTFLG

```

;INHIBIT TYPEOUTS??

;NO!!

;YES!!

;TYPE UNIT NUMBER, TEST NUMBER, ERROR NUMBER, AND PROGRAM COUNTER

;TYPE "UNIT"

;SAVE \$UNIT FOR TYPEOUT

;TYPE UNIT NUMBER

;GO TYPE--OCTAL ASCII

;TYPE 3 DIGIT(S)

;SUPPRESS LEADING ZEROS

;LOAD TEST NUMBER FOR

;TYPE "TST"

;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 "ERR"

;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

;WAS AN ERROR CALLED?

;NO!!

;YES-TYPE CRLF

;CLEAR BOT FLAG

7848	033344	105037	033741	CLRB	CHRCNT	; CLEAR CHARACTER COUNTER
7849	033350	013700	033734	MOV	ERRNMB,R0	; R0 POINTS TO FIRST OF
7850	033354	006300		ASL	R0	; FOUR ENTRIES IN ERROR
7851	033356	006300		ASL	R0	; TABLE
7852	033360	006300		ASL	R0	
7853	033362	062700	001554	ADD	#ERRTB-B.,R0	
7854	033366	011001		MOV	(R0),R1	; R1 POINTS TO ERROR MESSAGE
7855						; TABLE
7856	033370	001507		BEQ	13\$; BRANCH IF NO ERROR MESSAGE
7857				; TYPE THE ERROR MESSAGE		
7858	033372	012102		4\$: MOV	(R1)+,R2	; R2=ADDRESS OF MESSAGE STRING
7859	033374	001505		BEQ	12\$; BRANCH IF END OF MESSAGE
7860	033376	010237	033544	MOV	R2,11\$; LOAD ADDRESS OF STRING
7861	033402	005037	033736	CLR	BOTADR	; CLEAR BOT ADDRESS
7862	033406	112203		5\$: MOV	(R2)+,R3	; END OF STRING??
7863	033410	001454		BEQ	10\$; YES!!
7864	033412	122703	000015	CMPB	#CR,R3	; CARRIAGE RETURN??
7865	033416	001003		BNE	6\$; NO!!
7866	033420	105037	033741	CLRB	CHRCNT	; YES-CLEAR CHAR COUNT
7867	033424	000770		BR	5\$; GET NEXT CHARACTER
7868	033426	122703	000012	6\$: CMPB	#LF,R3	; LINE FEED??
7869	033432	001765		BEQ	5\$; YES-GET NEXT CHARACTER
7870	033434	122703	000011	CMPB	#HT,R3	; HORIZONTAL TAB??
7871	033440	001007		BNE	8\$; NO!!
7872	033442	105237	033741	7\$: INCB	CHRCNT	; ADJUST CHARACTER COUNT
7873	033446	132737	000007	033741	BITB	#7,CHRCNT
7874	033454	001372		BNE	7\$	
7875	033456	000407		BR	9\$	
7876	033460	105237	033741	8\$: INCB	CHRCNT	; INCREMENT CHARACTER COUNT
7877	033464	122703	000040	CMPB	#',R3	; SPACE??
7878	033470	001002		BNE	9\$; NO!!
7879	033472	010237	033736	MOV	R2,BOTADR	; SAVE ADDRESS OF SPACE
7880	033476	122737	000100	033741	9\$: CMPB	#64.,CHRCNT
7881	033504	103340		BHIS	5\$; END OF LINE??
7882	033506	013704	033736	MOV	BOTADR,R4	; GET ADDRESS OF LAST SPACE
7883	033512	001007		BNE	90\$; BRANCH IF SPACE DETECTED
7884	033514	104401	001217	TYPE	#CRLF	; TYPE CRLF
7885	033520	105037	033741	CLRB	CHRCNT	; CLEAR CHARACTER COUNT
7886	033524	013702	033544	MOV	11\$,R2	; SET UP R2 FOR TESTING
7887	033530	000726		BR	5\$	
7888	033532	105044		90\$: CLRB	-(R4)	; REPLACE SPACE
7889	033534	112737	177777	033740	MOV	#-1,BOTFLG
7890	033542	104401		10\$: TYPE		; SET BOT FLAG
7891	033544	000000		11\$: .WORD		; TYPE ERROR MESSAGE STRING
7892	033546	105737	033740	TSTB	BOTFLG	; STRING ADDRESS GOES HERE
7893	033552	001707		BEQ	4\$; WAS STRING TRUNCATED??
7894	033554	104401	001217	TYPE	#CRLF	; YES-TYPE CRLF
7895	033560	105037	033740	CLRB	BOTFLG	; CLEAR BOT FLAG
7896	033564	105037	033741	CLRB	CHRCNT	; CLEAR CHARACTER COUNT
7897	033570	013702	033736	MOV	BOTADR,R2	; SETUP R2 FOR TESTING
7898	033574	010237	033544	MOV	R2,11\$; SETUP 11\$ FOR TYPING
7899	033600	112742	000040	MOV	#',-(R2)	; RESTORE SPACE
7900	033604	105722		TSTB	(R2)+	; RESTORE R2
7901	033606	000677		BR	5\$; TYPE REST OF STRING
7902	033610			12\$:		
7903				; TYPE ERROR HEADER AND ERROR DATA		

```

7904 033610
7905 033610 016001 000002
7906 033614 001444
7907 033616 104401 001217
7908 033622 016002 000004
7909 033626 016003 000006
7910 033632 012137 033642
7911 033636 001433
7912
7913 033640 104401
7914 033642 000000
7915 033644 104401 001217
7916 033650 005702
7917 033652 001767
7918 033654 012204
7919 033656 012205
7920 033660 105725
7921 033662 100407
7922 033664 001403
7923 033666 013446
7924 033670 104405
7925 033672 000405
7926 033674 013446
7927 033676 104402
7928 033700 000402
7929 033702 013446
7930 033704 104406
7931 033706 005714
7932 033710 001403
7933 033712 104401 033775
7934 033716 000760
7935 033720 104401 001217
7936 033724 000742
7937 033726 104415
7938 033730 000207
7939
7940 033732 000000
7941 033734 000000
7942 033736 000000
7943 033740 000
7944 033741 000
7945
7946 033742 047125 052111 000043
7947 033750 020054 042524 052123
7948 033756 000043
7949 033760 020054 051105 021522
7950 033766 000
7951 033767 054 050040 036503
7952 033774 000
7953 033775 040 000040
7954
7955

13$: MOV 2(R0),R1 ;R1 POINTS TO ERROR HEADER TABLE
      BEQ 21$ ;BRANCH IF NO HEADER
      TYPE $CRLF ;(ASSUME NO DATA)
      MOV 4(R0),R2 ;R2 POINTS TO DATA ADDRESS TABLE
      MOV 6(R0),R3 ;R3 POINTS TO FORMAT TABLE
14$: MOV (R1)+,15$ ;PUT HEADER ADDRESS FOR TYPE
      BEQ 21$ ;BRANCH IF END OF HEADERS
      ;(ASSUME END OF DATA)

15$: .WORD 0 ;HEADER ADDRESS GOES HERE
      TYPE $CRLF
      TST R2 ;DATA WITH HEADER??
      BEQ 14$ ;NO!!
      MOV (R2)+,R4 ;R4 POINTS TO DATA ADDRESS
      MOV (R3)+,R5 ;R5 POINTS TO FORMAT
16$: TSTB (R5)+ ;WHAT KIND OF DATA??
      BMI 18$ ;BINARY
      BEQ 17$ ;OCTAL
      MOV 2(R4)+,-(SP) ;DECIMAL

17$: MOV 2(R4)+,-(SP)
      TYPDC
      BR 19$

18$: MOV 2(R4)+,-(SP)
      TYPBN

19$: TST (R4) ;MORE DATA??
      BEQ 20$ ;NO!!
      TYPE ERTY04 ;YES-TYPE 2 SPACES
      BR 16$ ;AND CONTINUE
20$: TYPE $CRLF ;TYPE ONE BLANK LINE
      BR 14$ ;BEFORE NEXT HEADER

21$: RESREG
      RTS PC

TSTNMB: .WORD 0 ;TEST NUMBER
ERRNMB: .WORD 0 ;ERROR NUMBER
BOTADR: .WORD 0 ;BEGINNING OF TEXT ADDRESS
BOTFLG: .BYTE 0 ;BOT FLAG
CHRCNT: .BYTE 0 ;CHARACTER COUNT

ERTY00: .ASCIZ 2UNIT#2
ERTY01: .ASCIZ 2, TEST#2
ERTY02: .ASCIZ 2, ERR#2
ERTY03: .ASCIZ 2, PC=2
ERTY04: .ASCIZ 2 2
      .EVEN

```


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
8001
8002
8003
8004
8005
8006
8007
8008
8009
8010
8011

034000

034000 017637 000000 034712
034006 062716 000006
034012 105076 000000
034016 162716 000004

034022 032737 100000 034712
034030 001414
034032 004737 045206
034036 000411
034040 000401
034042 000000
034044 062716 000004
034050 113776 034042 000000
034056 000137 034700
034062

034062 032737 040000 034712
034070 001453

.SBTTL TEST PREPARATION MODULE

; THIS MODULE PREPARES THE RM03 SUBSYSTEM FOR THE EXECUTION OF A TEST,
; REPORTING AN ERROR TO THE USER IF AN ERROR IS DETECTED. THE USER
; SPECIFIES TASKS TO BE PERFORMED, WHICH THE MODULE EXECUTES
; USING SUBROUTINES.

;CALL:

JSR PC,TSTPRP
.WORD
BR ??
NOP
ERROR

TASK/VERIFY DESCRIPTOR
RETURN HERE IF NO ERROR
RETURN HERE IF ERROR
ERROR DEFINED BY MODULE

;TASK/VERIFY DESCRIPTOR

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

TSTPRP:

;STORE TASK DESCRIPTOR AND CLEAR USER'S ERROR CALL

MOV 2(SP),500\$;STORE DESCRIPTOR
ADD #6,(SP) ;MOVE SP TO USERS ERROR CALL
CLRB 2(SP) ;CLEAR ERROR CALL
SUB #4,(SP) ;MOVE SP TO NO ERROR RETURN
;*****
;SELECT DEVICE AND VERIFY DEVICE AVAILABLE IF BIT 15 SET IN TASK
BIT #BIT15,500\$;SELECT DEVICE??
BEQ 30\$;NO!!
JSR PC,DEVSEL ;GO SELECT DEVICE
BR 30\$;NO ERROR - CONTINUE
BR 20\$

10\$: .WORD ;ERROR NUMBER FROM DEVSEL
20\$: ADD #4,(SP) ;TRANSFER ERROR TO USER
MOVW 10\$,2(SP)
JMP 400\$

30\$:

;*****

;CLEAR CONTROLLER IF BIT 14 IS SET IN TASK

BIT #BIT14,500\$;CLEAR CONTROLLER??
BEQ 120\$;NO!!

8012	034072	004737	046660		JSR	PC,CNTCLR	;GO CLEAR CONTROLLER
8013	034076	000411			BR	60\$;CONTINUE - NO ERROR
8014	034100	000401			BR	50\$	
9015	034102	000000		40\$:	.WORD		;ERROR NUMBER FROM CNTCLR
8016	034104	062716	000004	50\$:	ADD	#4,(SP)	;TRANSFER ERROR TO USER
8017	034110	113776	034102	000000	MOVB	40\$,2(SP)	
8018	034116	000137	034700		JMP	400\$	
8019	034122			60\$:			
8020					*****		
8021					;VERIFY CONTROLLER CLEAR IF BIT6 SET IN TASK		
8022	034122	032737	000100	034712	BIT	#BIT6,500\$;VERIFY??
8023	034130	001433			BEG	120\$;NO!!
8024	034132	004737	037432		JSR	PC,GETSTS	;SETUP INDEX TABLE
8025	034136	004737	037516		JSR	PC,GET	;GO GET STATUS
8026	034142	000411			BR	90\$;NO ERROR GETTING STATUS
9027	034144	000401			BR	80\$	
8028	034146	000000		70\$:	.WORD		;ERROR FROM GETTING STATUS
8029	034150	062716	000004	80\$:	ADD	#4,(SP)	;TRANSFER ERROR TO USER
8030	034154	113776	034146	000000	MOVB	70\$,2(SP)	
8031	034162	000137	034700		JMP	400\$	
8032	034166	004737	046776	90\$:	JSR	PC,CLRSTS	;GO VERIFY STATUS CLEAR
8033	034172	000412			BR	120\$;NO ERROR IN CLEAR
8034	034174	000401			BR	110\$	
8035	034176	000000		100\$:	.WORD		;ERROR IN STATUS CLEAR
8036	034200	005726		110\$:	TST	(SP)+	;STRIP RETURN ADDRESS TO
8037	034202	062716	000004		ADD	#4,(SP)	;SUBROUTINE AND TRANSFER
8038	034206	113776	034176	000000	MOVB	100\$,2(SP)	;ERROR TO USER
8039	034214	000137	034700		JMP	400\$	
8040	034220			120\$:			
8041					*****		
8042					;EXECUTE PACK ACKNOWLEDGE IF BIT12 SET IN TASK AND VOLUME IS		
8043					;NOT VALID		
8044					BIT	#BIT12,500\$;PACK ACKNOWLEDGE??
8045	034220	032737	010000	034712	BEG	240\$;NO!!
8046	034226	001511			MOVB	#RMD5,GETINX	;GET RMD5
8047	034230	112737	000012	001506	MOVB	#200,GETINX+1	
8048	034236	112737	000200	001507	JSR	PC,GET	
8049	034244	004737	037516		BR	150\$;NO ERROR GETTING RMD5
8050	034250	000411			BR	140\$	
8051	034252	000401					
8052	034254	000000		130\$:	.WORD		
8053	034256	062716	000004	140\$:	ADD	#4,(SP)	;TRANSFER ERROR TO USER
8054	034262	113776	034254	000000	MOVB	130\$,2(SP)	
8055	034270	000137	034700		JMP	400\$	
8056	034274	032737	000100	001342	150\$:	BIT	#VV,RMDSI
8057	034302	001063			BNE	240\$;IS VOLUME VALID??
8058	034304	005037	001474		CLR	MEDEMB	;YES!!
8059	034310	012737	000023	001400	MOV	#PAKACK!GO,RMCSI	;CLEAR MEDIA ENABLE
8060	034316	112737	000000	001535	MOVB	#RMCSI,PUTINX	;LOAD PACK ACK COMMAND
8061	034324	112737	000200	001536	MOVB	#200,PUTINX+1	;SETUP REGISTER INDEX TABLE
8062	034332	004737	037766		JSR	PC,PUT	;GO WRITE COMMAND
8063	034336	000410			BR	180\$;NO ERROR LOADING REGISTER
8064	034340	000401			BR	170\$	
8065	034342	000000		160\$:	.WORD		;ERROR FROM PUT SUB
8066	034344	062716	000004	170\$:	ADD	#4,(SP)	;TRANSFER ERROR TO USER
8067	034350	113776	034342	000000	MOVB	160\$,2(SP)	

```

8068 034356 000550
8069 034360
8070
8071
8072
8073 034360 032737 000020 034712
8074 034366 001431
8075 034370 004737 037432
8076 034374 004737 037516
8077 034400 000410
8078 034402 000401
8079 034404 000000
8080 034406 062716 000004
8081 034412 113776 034404 000000
8082 034420 000527
8083 034422 004737 047656
8084 034426 000411
8085 034430 000401
8086 034432 000000
8087 034434 005726
8088 034436 062716 000004
8089 034442 113776 034432 000000
8090 034450 000513
8091 034452
8092
8093
8094
8095 034452 032737 004000 034712
8096 034460 001513
8097 034462 112737 000012 001506
8098 034470 112737 000200 001507
8099 034476 004737 037516
8100 034502 000410
8101 034504 000401
8102 034506 000000
8103 034510 062716 000004
8104 034514 113776 034506 000000
8105 034522 000466
8106 034524 032737 020000 001342
8107 034532 001466
8108 034534 012737 000007 001400
8109 034542 112737 000000 001535
8110 034550 112737 000200 001536
8111 034556 004737 037766
8112 034562 000410
8113 034564 000401
8114 034566 000000
8115 034570 062716 000004
8116 034574 113776 034566 000000
8117 034602 000436
8118 034604 004737 037432
8119 034610 004737 040326
8120
8121
8122
8123 034614 032737 000010 034712

180$: BR 400$
;*****
;VERIFY PACK ACKNOWLEDGE IF #BIT4 SET IN TASK
BIT #BIT4,500$ ;VERIFY PACK ACKNOWLEDGE??
BEQ 240$ ;NO!!
JSR PC,GETSTS ;SETUP FOR STATUS
JSR PC,GET ;GO GET STATUS
BR 210$ ;NO ERROR GETTING STATUS
BR 200$
190$: .WORD ;ERROR FROM GET SUB
200$: ADD #4,(SP) ;TRANSFER ERROR TO USER
MOVB 190$,2(SP)
BR 400$
210$: JSR PC,ACKSTS ;GO CHECK ACKNOWLEDGE
BR 240$ ;NO ERROR
BR 230$
220$: .WORD ;PACK ACKNOWLEDGE ERROR
230$: TST (SP)+ ;STRIP RETURN TO SUB AND
ADD #4,(SP) ;TRANSFER ERROR TO USER
MOVB 220$,2(SP)
BR 400$
240$:
;*****
;RECALIBRATE DRIVE IF BIT 11 SET IN TASK AND PIP IS ACTIVE
BIT #BIT11,500$ ;RECALIBRATE??
BEQ 410$ ;NO!!
MOVB #RMD5,GETINX ;LOAD REGISTER INDEX TABLE
MOVB #200,GETINX+1
JSR PC,GET ;GO GET RMD5
BR 270$ ;NO ERROR GETTING RMD5
BR 260$
250$: .WORD ;ERROR FROM GET SUB
260$: ADD #4,(SP) ;TRANSFER ERROR TO USER
MOVB 250$,2(SP)
BR 400$
270$: BIT #PIP,RMD5I ;IS PIP ACTIVE??
BEQ 410$ ;NO!!
MOV #RECAL!GO,RMCS10 ;LOAD RECALIBRATE COMMAND
MOVB #RMCS1,PUTINX ;AND REGISTER INDEX
MOVB #200,PUTINX+1
JSR PC,PUT ;GO ISSUE RECALIBRATE
BR 300$ ;NO ERROR
BR 290$
280$: .WORD ;ERROR IN REGISTER TRANSFER
290$: ADD #4,(SP) ;TRANSFER ERROR TO USER
MOVB 280$,2(SP)
BR 400$
300$: JSR PC,GETSTS ;SETUP FOR STATUS
JSR PC,TIMOUT ;WAIT FOR COMPLETION
;*****
;VERIFY RECALIBRATE IF BIT 3 SET IN TASK
BIT #BIT3,500$ ;VERIFY RECALIBRATE??

```

8124	034622	001432				BEG	410\$;NO!!
8125	034624	004737	037516			JSR	PC,GET		;GO GET STATUS
8126	034630	000410				BR	330\$;NO ERROR GETTING STATUS
8127	034632	000401				BR	320\$		
8128	034634	000000			310\$:	.WORD			;ERROR FROM GET
8129	034636	062716	000004		320\$:	ADD	#4,(SP)		;TRANSFER ERROR TO USER
8130	034642	113776	034634	000000		MOVB	310\$,2(SP)		
8131	034650	000413				BR	400\$		
8132	034652	004737	050452		330\$:	JSR	PC,RCLSTS		;GO CHECK RECALIBRATE
8133	034656	000414				BR	410\$;NO ERROR DURING RECALIBRATE
8134	034660	000401				BR	350\$		
8135	034662	000000			340\$:	.WORD			;ERROR DURING RECALIBRATE
8136	034664	005726			350\$:	TST	(SP)+		;STRIP RETURN TO SUB AND
8137	034666	062716	000004			ADD	#4,(SP)		;TRANSFER ERROR TO USER
8138	034672	113776	034662	000000		MOVB	340\$,2(SP)		
8139	034700	162716	000002		400\$:	SUB	#2,(SP)		;MOVE SP BACK BEFORE ERROR
8140	034704	000240				NOP			
8141	034706	000240				NOP			
8142	034710	000207			410\$:	RTS	PC		;RETURN TO USER
8143									
8144	034712	000000			500\$:	.WORD			;TASK/VERIFY DESCRIPTOR

```

8145 .SBTTL BAD SECTOR MODULE
8146
8147 ;THE BAD SECTOR MODULE PERFORMS TWO MAJOR FUNCTIONS, I.E.,
8148 ;RECOVER THE BAD SECTOR FILE FROM THE LAST TRACK, AND
8149 ;APPROVE THE USAGE OF A SECTOR BASED ON INFORMATION IN
8150 ;THE BAD SECTOR FILE OR ASSIGN A NEW SECTOR IF THE ONE ELECTED IS
8151 ;NOT AVAILABLE FOR USE.
8152
8153 ;INFORMATION REQUIRED BY THE MODULE INCLUDES
8154 ;.RMDCO, THE DESIRED CYLINDER,
8155 ;.RMDAO, THE TRACK AND SECTOR ADDRESS,
8156 ;.RMWCO, THE WORD COUNT,
8157 ;.RMCSIO, THE COMMAND.
8158
8159 ;MODULE CALL IS AS FOLLOWS,
8160
8161 JSR PC,BADSCT
8162 BR ??? ;RETURN HERE IF NO ERROR
8163 TYPE ,MESSAGE ;RETURN HERE IF THE BAD SECTOR FILE
8164 ;CANNOT BE RECOVERED
8165 ERROR ;THE MODULE DEFINES THE ERROR NUMBER
8166
8167 ;THE MODULE IS INTENDED TO BE CALLED PRIOR TO CALLING THE BUFFER
8168 ;GENERATOR SUBROUTINE AND PRESERVES THE "PUT BUFFER" SO THAT THE
8169 ;BUFFER NEED ONLY BE FILLED ONCE FOR THE EXECUTION OF A FORMAT
8170 ;OPERATION.
8171
8172 ;THE MODULE RETURNS TO THE CALLING TEST WITH THE APPROVED OR ASSIGNED
8173 ;SECTOR IN THE PUT BUFFER AND ALSO IN LOCATIONS "ASNOA" AND "ASNDC"
8174 ;SO THAT A REFERENCE IS AVAILABLE TO THE TEST OUTSIDE OF THE PUT BUFFER.
8175
8176
8177 034714 BADSCT:
8178
8179 ;TEST "MEDIA ENABLE" TO DETERMINE WHETHER OR NOT THE BAD SECTOR FILES
8180 ;HAVE BEEN RECOVERED.
8181 TST MEDENB
8182 BEQ 10$
8183 JMP 300$ ;BAD SECTOR FILE IS AVAILABLE
8184
8185 ;*****
8186 ;RECOVER THE BAD SECTOR FILE FROM THE LAST TRACK
8187 10$:
8188
8189 ;SAVE THE USER'S PUT BUFFER
8190 MOV RO,-(SP) ;;PUSH RO ON STACK
8191 CLR RO
8192 MOV PUTBUF(RO),BUFFER(RO)
8193 ADD #2,RO ;ADVANCE TO NEXT BUFFER POSITION
8194 CMP #46,RO ;END OF BUFFER
8195 BHS 20$ ;NO !!
8196
8197 ;SET RETRY COUNT AND LOAD PUT BUFFER AND REGISTER INDEX TABLE
8198 MOV #3,500$ ;RETRY COUNT
8199 MOV #002000,RMDAO ;STARTING SECTOR ADDRESS
8200 MOV #822.,RMDCO ;DESIRED CYLINDER

```

```

8201 034774 012737 177376 001402      MOV      #(<C258.+1>,RMWCO      ;WORD COUNT
8202 035002 012737 010000 001432      MOV      #FMT16,RMOF0      ;16 BIT FORMAT
8203 035010 012737 101546 001404      MOV      #MFGFIL,RMBA0      ;BUFFER ADDRESS
8204
8205 035016 012700 001535      MOV      #PUTINX,RO      ;RO POINTS TO REGISTER INDEX TABLE
8206 035022 112720 000006      MOV      #RMDA,(RO)+
8207 035026 112720 000034      MOV      #RMDC,(RO)+
8208 035032 112720 000032      MOV      #RMWC,(RO)+
8209 035036 112720 000032      MOV      #RMOF,(RO)+
8210 035042 112720 000004      MOV      #RMBA,(RO)+
8211 035046 112720 000000      MOV      #RMCS1,(RO)+
8212 035052 112720 000200      MOV      #200,(RO)+
8213 035056 012600      MOV      (SP)+,RO      ;;POP STACK INTO RO
8214
8215      ;SET GET INDEX TABLE FOR READING STATUS
8216 035060 004737 037432      JSR      PC,GETSTS      ;SETUP GET INDEX REGISTER FOR STATUS
8217 035064
8218
8219      ;CLEAR THE DEVICE USING DRIVE CLEAR COMMAND
8220 035064 012737 000011 001400      MOV      #DRVCLR!GO,RMCS10      ;LOAD COMMAND IN PUT BUFFER
8221 035072 004737 037766      JSR      PC,PUT      ;OUTPUT COMMAND
8222 035076 000411      BR      45$      ;RETURN HERE IF NO ERROR
8223 035100 000401      BR      40$      ;GET AROUND ERROR #
8224 035102 000000      35$:      .WORD      ;ERROR # GOES HERE
8225
8226 035104 062716 000006      40$:      ADD      #6,(SP)      ;MOVE SP TO USERS ERROR CALL
8227 035110 113776 035102 000000      MOV      35$,2(SP)      ;MOVE ERROR NUMBER TO USER
8228 035116 000137 035616      JMP      215$
8229 035122 004737 037516      45$:      JSR      PC,GET      ;GO GET STATUS
8230 035126 000411      BR      60$      ;RETURN HERE IF NO ERROR
8231 035130 000401      BR      55$      ;GET AROUND ERROR #
8232 035132 000000      50$:      .WORD      ;ERROR # GOES HERE
8233
8234 035134 062716 000006      55$:      ADD      #6,(SP)      ;MOVE SP TO USERS ERROR CALL
8235 035140 113776 035132 000000      MOV      50$,2(SP)      ;MOVE ERROR # TO USERS ERROR CALL
8236 035146 000137 035616      JMP      215$
8237
8238 035152 004737 052214      60$:      JSR      PC,DRVSTS      ;GO VERIFY DRIVE CLEAR COMMAND
8239 035156 000412      BR      75$      ;RETURN HERE IF NO ERROR
8240 035160 000401      BR      70$      ;GET AROUND ERROR
8241 035162 000000      65$:      .WORD      ;ERROR # GOES HERE
8242
8243 035164 005726      70$:      TST      (SP)+      ;STRIP RETURN TO SUBROUTINE
8244 035166 062716 000006      ADD      #6,(SP)      ;MOVE SP TO USERS ERROR CALL
8245 035172 113776 035162 000000      MOV      65$,2(SP)      ;MOVE ERROR # TO USER CALL
8246 035200 000137 035616      JMP      215$
8247
8248 035204      75$:
8249
8250      ;ISSUE A PACK ACKNOWLEDGE IF VOLUME VALID IS RESET
8251 035204 032737 000100 001342      BIT      #VV,RMDSI      ;IS VV RESET ??
8252 035212 001050      BNE      120$      ;NO !!
8253 035214 012737 000023 001400      MOV      #PACACK!GO,RMCS10      ;LOAD COMMAND
8254 035222 004737 037766      JSR      PC,PUT      ;GO PUT COMMAND TO DRIVE
8255 035226 000411      BR      90$      ;RETURN HERE IF NO OUTPUT ERROR
8256 035230 000401      BR      85$      ;GET AROUND ERROR #

```

8257	035232	000000		80\$:	.WORD		;ERROR # GOES HERE
8258							
8259	035234	062716	000006	85\$:	ADD #6,(SP)		;MOVE SP TO USERS ERROR CALL
8260	035240	113776	035232	000000	MOV 80\$,2(SP)		;MOVE ERROR # TO ERROR CALL
8261	035246	000137	035616		JMP 215\$		
8262	035252	004737	037516	90\$:	JSR PC,GET		;GO GET STATUS FOR PACK ACK
8263	035256	000411			BR 105\$;RETURN HERE IF NO ERROR
8264	035260	000401			BR 100\$;GET AROUND ERROR #
8265	035262	000000		95\$:	.WORD		;ERROR # GOES HERE
8266							
8267	035264	062716	000006	100\$:	ADD #6,(SP)		;MOVE SP TO USERS ERROR CALL
8268	035270	113776	035262	000000	MOV 95\$,2(SP)		;MOVE ERROR # TO CALL
8269	035276	000137	035616		JMP 215\$		
8270							
8271	035302	004737	047656	105\$:	JSR PC,ACKSTS		;GO VERIFY ACKNOWLEDGE STATUS
8272	035306	000412			BR 120\$;RETURN HERE IF NO ERROR
8273	035310	000401			BR 115\$;GET AROUND ERROR #
8274	035312	000000		110\$:	.WORD		;ERROR # GOES HERE
8275							
8276	035314	005726		115\$:	TST (SP)+		;STRIP RETURN TO SUBROUTINE
8277	035316	062716	000006		ADD #6,(SP)		;MOVE SP TO USERS ERROR CALL
8278	035322	113776	035312	000000	MOV 110\$,2(SP)		;MOVE ERROR # TO USERS ERROR CALL
8279	035330	000137	035616		JMP 215\$		
8280							
8281	035334			120\$:			
8282							
8283							
8284	035334	032737	020000	001342	;RECALIBRATE THE DRIVE IF PIP IS SET		
8285	035342	001452			BIT #PIP,RMSI		;IS PIP SET ??
8286					BEQ 165\$;NO !!
8287	035344	012737	000007	001400	MOV #RECAL!GO,RMCS10		;LOAD RECALIBRATE COMMAND
8288	035352	004737	037766		JSR PC,PUT		;PUT THE RECAL COMMAND
8289	035356	000411			BR 135\$;RETURN HERE IF NO ERROR
8290	035360	000401			BR 130\$;GET AROUND ERROR #
8291	035362	000000		125\$:	.WORD		;ERROR # GOES HERE
8292							
8293	035364	062716	000006		130\$:	ADD #6,(SP)	;MOVE SP TO USERS ERROR CALL
8294	035370	113776	035362	000000	MOV 125\$,2(SP)		;MOVE ERROR # TO USERS CALL
8295	035376	000137	035616		JMP 215\$		
8296	035402			135\$:			
8297	035402	004737	040326		JSR PC,TIMOUT		;WAIT FOR RECALIBRATE TO COMPLETE
8298	035406	004737	037516		JSR PC,GET		;GO GET RECAL STATUS
8299	035412	000411			BR 150\$;RETURN HERE IF NO ERROR
8300	035414	000401			BR 145\$;GET AROUND ERROR #
8301	035416	000000		140\$:	.WORD		;ERROR # GOES HERE
8302							
8303	035420	062716	000006		145\$:	ADD #6,(SP)	;MOVE SP TO USERS ERROR CALL
8304	035424	113776	035416	000000	MOV 140\$,2(SP)		;MOVE ERROR TO USERS CALL
8305	035432	000137	035616		JMP 215\$		
8306							
8307	035436	004737	050452	150\$:	JSR PC,RCLSTS		;GO VERIFY RECALIBRATE STATUS
8308	035442	000412			BR 165\$;RETURN HERE IF NO ERROR
8309	035444	000401			BR 160\$;GET AROUND ERROR #
8310	035446	000000		155\$:	.WORD		;ERROR # GOES HERE
8311							
8312	035450	005726		160\$:	TST (SP)+		;STRIP RETURN TO SUBROUTINE

```

8313 035452 062716 000006      ADD      #6,(SP)      ;MOVE SP TO USERS ERROR CALL
8314 035456 113776 035446 000000      MOV      155$,2(SP)    ;MOVE ERROR # TO USERS CALL
8315 035464 000137 035616      JMP      215$
8316
8317 035470      165$:
8318
8319      ;READ THE SECTOR IDENTIFIED BY RMDAO, INCLUDING HEADER AND DATA
8320
8321 035470 012737 000073 001400      MOV      #RH!GO,RMC510    ;LOAD READ HEADER AND DATA COMMAND
8322 035476 004737 037766      JSR      PC,PUT      ;PUT COMMAND
8323 035502 000411      BR      180$      ;RETURN HERE IF NO ERROR
8324 035504 000401      BR      175$      ;GET AROUND ERROR #
8325 035506 000000      170$: .WORD      ;ERROR # GOES HERE
8326
8327 035510 062716 000006      175$: ADD      #6,(SP)      ;MOVE SP TO USERS ERROR CALL
8328 035514 113776 035506 000000      MOV      170$,2(SP)    ;MOVE ERROR # TO USERS ERROR CALL
8329 035522 000137 035616      JMP      215$
8330
8331 035526 004737 040326      180$: JSR      PC,TIMOUT    ;WAIT FOR READ OPERATION TO COMPLETE
8332 035532 004737 037516      JSR      PC,GET      ;GO GET STATUS FOR READ OPERATION
8333 035536 000411      BR      195$      ;RETURN HERE IF NO ERROR
8334 035540 000401      BR      190$      ;GET AROUND ERROR #
8335 035542 000000      185$: .WORD      ;ERROR # GOES HERE
8336
8337 035544 062716 000006      190$: ADD      #6,(SP)      ;MOVE SP TO USERS ERROR CALL
8338 035550 113776 035542 000000      MOV      185$,2(SP)    ;MOVE ERROR # TO CALL
8339 035556 000137 035616      JMP      215$
8340
8341 035562 004737 053016      195$: JSR      PC,DASTS      ;GO VERIFY RESULTS OF READ OPERATION
8342 035566 000412      BR      210$      ;RETURN HERE IF NO ERROR
8343 035570 000401      BR      205$      ;GET AROUND ERROR #
8344 035572 000000      200$: .WORD      ;ERROR # GOES HERE
8345
8346 035574 005726      205$: TST      (SP)+      ;STRIP RETURN ADDRESS TO SUBROUTINE
8347 035576 062716 000006      ADD      #6,(SP)      ;MOVE SP TO USERS ERROR CALL
8348 035602 113776 035572 000000      MOV      200$,2(SP)    ;MOVE ERROR # TO USERS CALL
8349 035610 000137 035616      JMP      215$
8350
8351 035614 000456      210$: BR      240$
8352
8353 035616      215$:
8354
8355      ;AN ERROR HAS BEEN DETECTED IN TRYING TO READ THE BAD SECTOR FILE.
8356      ;THE SECTOR WILL BE RETRIED IF POSSIBLE.
8357
8358 035616 032737 010000 001342      BIT      #MOL,RMDSI      ;IS MEDIUM ON LINE ??
8359 035624 001446      BEQ      230$      ;YES !!
8360
8361 035626 005337 036606      DEC      500$      ;DECREMENT RETRY COUNT
8362 035632 100037      BPL      225$      ;RETRY IF COUNT NOT NEG
8363
8364      ;THE RETRY COUNT HAS EXPIRED - SEE IF THE ERROR IS MEDIA RELATED
8365 035634 013746 001344      MOV      #RMER1,-(SP)    ;GET ER1
8366 035640 042716 100720      BIC      #DCK!MCRC!MCE!FER!ECH,(SP)
8367 035644 032737 000010 001372      BIT      #DPE,RMER2I      ;WAS THERE A DATA PARITY ERROR ??
8368 035652 001402      BEQ      220$      ;NO !!

```



```
8369 035654 042716 000010      BIC      #PAR, (SP)      ; YES - CLEAR PAR
8370 035660 005726      220$: TST      (SP)+      ; ARE THERE ANY ERRORS NOT DUE TO MEDIA ?
8371 035662 001027      BNE      230$      ; YES !!
8372 035664 013746 001372      MOV      RMER2I, -(SP)      ; GET ER2
8373 035670 042726 100010      BIC      #BSE!DPE, (SP)+      ; CLEAR MEDIA ERRORS
8374 035674 001022      BNE      230$      ; BRANCH IF NONMEDIA ERRORS DETECTED
8375
8376      ; THE ERRORS DETECTED WHILE TRYING TO RECOVER THE BAD SECTOR FILE ARE
8377      ; DUE TO THE MEDIA. SEE IF THE BAD SECTOR FILE CAN BE RECOVERED FROM
8378      ; ANOTHER AREA ON THE LAST TRACK
8379
8380 035676 062737 000002 001406      ADD      #2, RMDAO      ; ADVANCE SECTOR ADDRESS
8381 035704 122737 000012 001406      CMPB     #10, RMDAO      ; QUIT IF ALL MFG SECTORS HAVE BEEN
8382 035712 001413      BEQ      230$      ; TRIED
8383 035714 122737 000040 001406      CMPB     #32, RMDAO      ; QUIT IF ALL USER SECTORS HAVE
8384 035722 001407      BEQ      230$      ; BEEN TRIED
8385 035724 012737 000003 036606      MOV      #3, 500$      ; REINSTATE RETRY COUNT FOR THIS SECTOR
8386 035732 162716 000006      225$: SUB      #6, (SP)      ; MOVE SP BACK TO NO ERROR
8387 035736 000137 035064      JMP      30$      ; RETRY THE READ OPERATION
8388
8389 035742      230$:
8390
8391      ; THE BAD SECTOR FILE CANNOT BE READ
8392 035742 162716 000004      SLB      #4, (SP)      ; MOVE SP TO ERROR RETURN
8393 035746 000137 036602      JMP      410$      ; GO TO MODULE EXIT
8394
8395 035752      240$:
8396
8397      ; THE SECTOR WAS RECOVERED WITHOUT ERROR -
8398      ; READ THE USER FILE IF THIS IS THE MFG FILE
8399      ; OR ELSE DONE
8400
8401 035752 122737 000011 001406      CMPB     #9, RMDAO      ; WAS THE USER FILE READ ??
8402 035760 103451      BLO      260$      ; YES - READ IS COMPLETE
8403
8404      ; IF 144 IS IMPLEMENTED THEN READ THE USER FILE -
8405      ; ELSE DUMMY THE BAD SECTOR FILE
8406 035762 032737 140000 101546      BIT      #MSE!USE, MFGFIL ; ARE THE BAD SECTOR FLAGS OFF ??
8407 035770 001410      BEQ      250$      ; YES - 144 IS DISABLED
8408
8409 035772 012737 002012 001406      MOV      #002012, RMDAO ; READ THE USER FILE
8410 036000 012737 000003 036606      MOV      #3, 500$      ; RELOAD THE RETRY COUNT FOR THIS SECTOR
8411 036006 000137 035064      JMP      30$      ; GO READ THE USER FILE
8412 036012
8413      250$:
8414
8415      ; DUMMY THE BAD SECTOR FILES
8416      MOV      R0, -(SP)      ; PUSH R0 ON STACK
8417 036014 010146      MOV      R1, -(SP)      ; PUSH R1 ON STACK
8418 036016 012701 000374      MOV      #252, R1      ; R1 = NUMBER OF ENTRIES IN FILES
8419 036022 012700 000014      MOV      #14, R0      ; R0 = ACCESS INDEX TO FILE STORAGE
8420 036026 012760 177777 101546 255$: MOV      #-1, MFGFIL(R0) ; ENTER ALL ONES IN MFG FILE
8421 036034 012760 177777 102556      MOV      #-1, USRFIL(R0) ; ENTER ALL ONES IN USER FILE
8422 036042 062700 000002      ADD      #2, R0      ; ADVANCE ADDRESS
8423 036046 005301      DEC      R1      ; DECREMENT COUNT
8424 036050 001366      BNE      255$      ; CONTINUE IF NOT DONE
      MOV      #6, R1      ; CLEAR ID, SERIAL NUMBER ETC
```

Address	Hex	Dec	Label	Instruction	Comments
036056	005000			CLR R0	
036060	005060	101546	256\$:	CLR MFGFIL(R0)	
036064	005060	102556		CLR USRFIL(R0)	
036070	062700	000002		ADD #2,R0	
036074	005301			DEC R1	
036076	001370			BNE 256\$	
036100	012601			MOV (SP)+,R1	::POP STACK INTO R1
036102	012600			MOV (SP)+,R0	::POP STACK INTO R0
036104			260\$:		
036104	012737	177777		MOV #1,MEDENB	::SET MEDIA ENABLE AND RESTORE THE USERS PUT BUFFER
036112	010046			MOV R0,-(SP)	::PUSH R0 ON STACK
036114	005000			CLR R0	::R0 IS REGISTER INDEX
036116	016060	103566	001400	MOV BUFFER(R0),PUTBUF(R0)	
036124	062700	000002	265\$:	ADD #2,R0	::ADVANCE R0
036130	022700	000046		CMP #46,R0	::DONE ??
036134	103370			BHIS 265\$	
036136	012600			MOV (SP)+,R0	::POP STACK INTO R0
036140			270\$:		
036140			300\$:		
036140					::*****
036140					::VERIFY THAT THE DESIRED SECTOR IS NOT IN THE MFG BAD SECTOR FILE
036140					::AND NOT IN THE USERS BAD SECTOR FILE. ASSIGN A NEW SECTOR IF THE
036140					::SECTOR IS IN EITHER FILE.
036140					::LOAD INITIAL VARIABLES AND COMPUTE THE NUMBER OF SECTORS
036140	010046			MOV R0,-(SP)	::PUSH R0 ON STACK
036142	010146			MOV R1,-(SP)	::PUSH R1 ON STACK
036144	010246			MOV R2,-(SP)	::PUSH R2 ON STACK
036146	013737	001434	001476	MOV RMDC0,ACNDC	::LOAD REQUESTED CYLINDER, TRACK,
036154	013737	001406	001500	MOV RMDAO,ASNOA	::AND SECTOR ADDRESS IN ASSIGNED STORAGE
036162	005002			CLR R2	::R2 = NUMBER OF SECTORS
036164	013700	001402		MOV RMDC0,R0	::R0 = WORD COUNT
036170	005300			DEC R0	::CONVERT FROM 2'S COMPLEMENT
036172	005100			COM R0	
036174	012701	000400		MOV #256,R1	::R1 = NUMBER OF WORDS PER SECTOR
036176	032737	000002	001400	BIT #11,RMCS10	::IS THIS A HEADER AND DATA COMMAND ??
036180	001402			BEQ 305\$::NO !!
036182	012701	000402		MOV #258,R1	::CHANGE WORDS PER SECTOR
036184	020100			CMP R1,R0	::IS THERE A FULL SECTOR ??
036186	101404			BLOS 310\$::YES !!
036188	005700			YST R0	::IS R0 ZERO ??
036190	001405			BEQ 315\$::YES !!
036192	005202			INC R2	::INCREMENT FOR PARTIAL SECTOR
036194	000403			BR 315\$	
036196	160100			SUB R1,R0	::SUBTRACT ONE SECTOR FROM WORD COUNT
036198	005202			INC R2	::INCREMENT SECTOR COUNT
036200	000767			BR 305\$	
036202	010237	036606	315\$:	MOV R2,500\$::SAVE SECTOR COUNT

```

8481 036242
8482
8483
8484
8485
8486
8487 036242 012737 101562 036616      MOV      #MFGFIL+14,540$ ;MOVE THE STARTING ADDRESS OF MFG
8488                                     ;FILE TO BASE ADDRESS STORAGE
8489 036250 013737 001476 036612 320$: MOV      ASND0,520$ ;LOAD COMPARING CYLINDER ADDRESS
8490 036256 013737 001500 036614      MOV      ASNDA,530$ ;LOAD COMPARING TRACK SECTOR ADDRESS
8491 036264 013737 036606 036610      MOV      S00$,510$ ;LOAD NUMBER OF SECTORS TO CONFIRM
8492
8493 ;SETUP FOR A BINARY SEARCH OF THE CURRENT FILE FOR THE COMPARING
8494 ;CYLINDER, TRACK AND SECTOR ADDRESS
8495 036272 013700 036616 325$: MOV      540$,R0 ;LOAD THE BASE ADDRESS IN R0
8496 036276 012701 000376      MOV      #((127.*4)/2),R1 ;R1 = LENGTH OF FILE
8497 036302 060100      ADD      R1,R0 ;START BINARY DIVIDE AT CENTER
8498 036304 021037 036612 330$: CMP      (R0),520$ ;DOES TABLE ENTRY = COMPARING CYLINDER ?
8499 036310 001405      BEQ      345$ ;YES !!
8500 036312 101002      BHI      340$ ;BRANCH IF ENTRY > COMPARING CYLINDER
8501 036314
8502
8503 ;FILE ENTRY IS LESS THAN COMPARING CYLINDER (OR TRACK AND SECTOR),
8504 ;SO ADD DISPLACEMENT TO CURRENT POSITION
8505 036314 060100      ADD      R1,R0
8506 036316 000410      BR      350$
8507 036320
8508
8509 ;FILE ENTRY IS GREATER THAN COMPARING CYLINDER SO SUBTRACT DISPLACEMENT
8510 ;FROM CURRENT POSITION
8511 036320 160100      SUB      R1,R0
8512 036322 000406      BR      345$
8513 036324
8514
8515 ;FILE ENTRY EQUALS COMPARING CYLINDER. SEE IF THE NEXT ENTRY EQUALS
8516 ;THE COMPARING TRACK AND SECTOR.
8517 036324 026037 000002 036614      CMP      2(R0),530$ ;ARE THEY EQUAL ??
8518 036332 001413      BEQ      350$ ;YES !!
8519 036334 101371      BHI      340$ ;BRANCH IF HIGHER
8520 036336 000766      BR      335$ ;BRANCH IF LOWER
8521 036340
8522
8523 ;THE POINTER (R0) HAS BEEN ADJUSTED. HALVE THE DISPLACEMENT AND
8524 ;CONTINUE THE SEARCH IF DISPLACEMENT NOT ZERO
8525 036340 005701      TST      R1 ;IS DISPLACEMENT ZERO ??
8526 036342 001440      BEQ      370$ ;YES !!
8527 036344 006201      ASR      R1 ;HALVE THE DISPLACEMENT
8528 036346 042701 000003      BIC      #3,R1
8529 036352 020037 036616      CMP      R0,540$ ;IS THE NEW POINTER WITHIN BOUNDS ??
8530 036356 103432      BLO      370$ ;NO !!
8531 036360 000751      BR      330$ ;CONTINUE SEARCH
8532 036362
8533
8534 ;THE COMPARING CYLINDER, TRACK AND SECTOR IS IN THE BAD SECTOR FILE.
8535 ;ADVANCE THE ASSIGNED SECTOR AND START THE SEARCH ALL OVER.
8536 036362 105237 001500      INCB     ASNDA ;INCREMENT SECTOR

```

```

8537 036436 122737 000037 001500      CMPB      #31.,ASNOA      ;SECTOR OK ??
8538      C      103022      BHIS      365$      ;YES !!
8539 036437 103037 001500      CLRB      ASNOA      ;CLEAR SECTOR AND ADVANCE TRACK
8540 036402 103037 001501      INCB      ASNOA+1
8541 036406 122737 000004 001501      CMPB      #4,ASNOA+1      ;TRACK OK ??
8542 036414 103012      BHIS      365$      ;YES !!
8543 036416 005037 001500      CLR      ASNOA      ;CLEAR TRACK AND SECTOR
8544 036422 005037 001476      INC      ASNOC      ;INCREMENT CYLINDER
8545 036426 022737 001466 001476      CMP      #822.,ASNOC      ;CYLINDER OK ??
8546 036434 103002      BHIS      365$      ;YES !!
8547 036436 005037 001476      CLR      ASNOC
8548 036442 000677      BR      316$      ;REPEAT SEARCH
8549
8550 036444      370$:
8551
8552      ;THE COMPARING SECTOR IS NOT IN THE BAD SECTOR FILES. DECREMENT THE
8553      ;NUMBER OF SECTORS TO COMPARE AND SEARCH THE NEXT SECTOR IF THE NUMBER
8554      ;IS NOT ZERO.
8555
8556 036444 005337 036610      DEC      510$      ;DECREMENT NUMBER OF SECTORS TO COMPARE
8557 036450 001432      BEQ      380$      ;DONE IF ZERO
8558 036452 105237 036614      INCB      530$      ;INCREMENT THE COMPARING SECTOR
8559 036456 122737 000037 036614      CMPB      #31.,530$      ;SECTOR OK ??
8560 036464 103022      BHIS      375$      ;YES !!
8561 036466 105037 036614      CLRB      530$      ;CLEAR SECTOR
8562 036472 105237 036615      INCB      530$+1      ;INCREMENT TRACK
8563 036476 122737 000004 036615      CMPB      #4,530$+1      ;TRACK OK ??
8564 036504 103012      BHIS      375$      ;YES !!
8565 036506 005037 036614      CLR      530$      ;CLEAR SECTOR TRACK
8566 036512 005237 036612      INC      520$      ;INCREMENT CYLINDER
8567 036516 022737 001466 036612      CMP      #822.,520$      ;CYLINDER OK ??
8568 036524 103002      BHIS      375$      ;YES !!
8569 036526 005037 036612      CLR      520$      ;CLEAR CYLINDER
8570 036532 000137 036272      JMP      325$      ;SEARCH NEXT SECTOR
8571
8572 036536      380$:
8573
8574      ;THE ASSIGNED SECTOR (AND REQUIRED ADJACENT SECTORS) ARE NOT IN
8575      ;THE BAD SECTOR FILE JUST SEARCHED. SEARCH THE USER FILE IF IT
8576      ;HAS NOT YET BEEN SEARCHED, ELSE ASSIGN THE SECTOR AND RETURN TO USER.
8577
8578 036536 022737 102572 036616      CMP      #USRFIL+14,540$ ;TEST BASE ADDRESS
8579 036544 001405      BEQ      400$      ;DONE IF BASE = USER
8580 036546 012737 102572 036616      MOV      #USRFIL+14,540$ ;LOAD BASE ADDRESS
8581 036554 000137 036250      JMP      320$      ;SEARCH USER FILE
8582
8583
8584 036560      400$:
8585
8586      ;ASSIGN THE SECTOR AND RETURN TO USER
8587 036560 013737 001476 001434      MOV      ASNOC,RMDCO      ;LOAD CYLINDER
8588 036566 013737 001500 001406      MOV      ASNOA,RMDAO      ;LOAD TRACK AND SECTOR
8589 036574 012602      MOV      (SP)+,R2      ;POP STACK INTO R2
8590 036576 012601      MOV      (SP)+,R1      ;POP STACK INTO R1
8591 036600 012600      MOV      (SP)+,R0      ;POP STACK INTO R0
8592 036602 000240      NOP      410$:

```

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 171
BAD SECTOR MODULE

SEQ 0171

8593 036604 000207

RTS PC

8594

8595

8596

8597 036606 000000

8598 036610 000000

8599 036612 000000

8600 036614 000000

8601 036616 000000

;THE FOLLOWING ARE STORAGE LOCATIONS FOR THE MODULE

500\$: .WORD ;RETRY COUNT/ NUMBER OF SECTORS REQUIRED

510\$: .WORD ;NUMBER OF SECTORS TO COMPARE

520\$: .WORD ;COMPARING CYLINDER

530\$: .WORD ;COMPARING TRACK AND SECTOR

540\$: .WORD ;BASE ADDRESS OF BAD SECTOR FILE BEING SEARCHED

.SBTTL BUFFER GENERATOR SUBROUTINE

```

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

```

```

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

```

```

GENBUF:
MOV R0,-(SP)      ; PUSH R0 ON STACK
MOV R1,-(SP)      ; PUSH R1 ON STACK
MOV R2,-(SP)      ; PUSH R2 ON STACK
MOV R3,-(SP)      ; PUSH R3 ON STACK
MOV R4,-(SP)      ; PUSH R4 ON STACK
MOV RMDA0,R0      ; LOAD DATA BUFFER ADDRESS
MOV RMWC0,R1      ; LOAD WORD COUNT
MOV RMDC0,60$     ; LOAD STARTING CYLINDER ADDRESS
MOV RMDA0,65$     ; LOAD STARTING TRACK, SECTOR ADDRESS
10$: BIT #BIT1,RMC$10 ; WRITE HEADER & DATA??
    BEQ 25$      ; NO!!
    MOV 60$,(R0) ; WRITE HEADER WORD #1
    BIC #1C$YLMSK,(R0) ; SET (DISABLE) BAD SECTOR FLAGS
    BIS #MSE$USE,(R0) ; R2 = MAXIMUM SECTOR ADDRESS
    MOV #29,R2    ; 16 BIT FORMAT??
    BIT #FMT16,RMOF0 ; NO!!
    BEQ 15$      ; SET FORMAT BIT IN HEADER
    BIS #FMT16,(R0) ; CHANGE MAXIMUM SECTOR ADDRESS
    MOV #31,R2    ; INCREMENT WORD COUNT
    INC R1        ; EXIT IF DONE
    BEQ 50$      ; MOVE R0 TO HEADER WORD #2
    ADD #2,R0     ; WRITE HEADER WORD #2
    MOV 65$,(R0)+ ; INCREMENT WORD COUNT AND
    INC R1        ; EXIT IF DONE
    BEQ 50$      ; ADVANCE SECTOR ADDRESS
    MOV #65$,R3   ; SECTOR OVERFLOW ??
    INCB (R3)     ; NO !!
    CMPB R2,(R3)  ; YES - CLEAR SECTOR ADDRESS
    BHIS 25$     ; ADVANCE TRACK ADDRESS
    CLRB (R3)     ; TRACK OVERFLOW ??
    INCB 1(R3)    ; NO !!
    CMPB #4,1(R3) ; YES - CLEAR TRACK ADDRESS
    BHIS 25$     ; ADVANCE CYLINDER ADDRESS
    CLRB 1(R3)    ; LOAD SECTOR DATA COUNT
    INCB 60$     ; LOAD PATTERN ADDRESS
    MOV #256,R4   ; LOAD PATTERN COUNT
    MOV $TMP0,R2  ; WRITE DATA PATTERN
    MOV $TMP1,R3  ; INCREMENT WORD COUNT AND
    INC R1        ; EXIT IF DONE
    BEQ 50$      ; DECREMENT SECTOR COUNT
    DEC R4
25$: MOV #256,R4
30$: MOV $TMP0,R2
    MOV $TMP1,R3
40$: MOV (R2)+,(R0)+
    INC R1
    BEQ 50$
    DEC R4

```

```

8602
8603
8604
8605
8606
8607
8608
8609
8610
8611
8612
8613
8614
8615 036620
8616 036620 010046
8617 036622 010146
8618 036624 010246
8619 036626 010346
8620 036630 010446
8621 036632 013700 001404
8622 036636 013701 001402
8623 036642 013737 001434 037060
8624 036650 013737 001406 037062
8625 036656 032737 000002 001400 10$:
8626 036664 001450
8627 036666 013710 037060
8628 036672 042710 176000
8629 036676 052710 140000
8630 036682 012702 000035
8631 036686 032737 010000 001432
8632 036690 001404
8633 036694 052710 010000
8634 036698 012702 000037
8635 036702 005201 15$:
8636 036706 001444
8637 036710 062700 000002
8638 036714 013720 037062
8639 036718 005201
8640 036722 001436
8641 036726 012703 037062
8642 036730 105213
8643 036734 120213
8644 036738 103013
8645 036742 105013
8646 036746 105263 000001
8647 036750 122763 000004 000001
8648 036754 103004
8649 036758 105063 000001
8650 036762 105237 037060
8651 036766 012704 000400
8652 036770 013702 001174
8653 036774 013703 001176
8654 036778 012220
8655 036782 005201
8656 036786 001405
8657 036790 005304

```

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 173
BUFFER GENERATOR SUBROUTINE

SEQ 0173

8658	037032	001711	BEQ	10\$; START NEXT SECTOR IF 0
8659	037034	005303	DEC	R3	; DECREMENT PATTERN COUNT
8660	037036	001765	BEQ	30\$; RESTART PATTERN IF 0
8661	037040	000770	BR	40\$; CONTINUE DATA PATTERN
8662					
8663	037042		50\$:		
8664	037042	012604	MOV	(SP)+,R4	; POP STACK INTO R4
8665	037044	012603	MOV	(SP)+,R3	; POP STACK INTO R3
8666	037046	012602	MOV	(SP)+,R2	; POP STACK INTO R2
8667	037050	012601	MOV	(SP)+,R1	; POP STACK INTO R1
8668	037052	012600	MOV	(SP)+,R0	; POP STACK INTO R0
8669	037054	000240	NOP		
8670	037056	000207	RTS	PC	
8671					
8672	037060	000000	60\$:	.WORD	; CYLINDER ADDRESS STORAGE
8673	037062	000000	65\$:	.WORD	; TRACK, SECTOR ADDRESS STORAGE

8674
8675
8676
8677
8678
8679
8680
8681
8682
8683
8684
8685
8686
8687
8688
8689
8690 037064
8691 037064 010046
8692 037066 010146
8693 037070 010246
8694 037072 010346
8695
8696 037074 005037 037430
8697
8698 037100 032737 100000 001344
8699 037106 001465
8700 037110 032737 000100 001344
8701 037116 001061
8702 037120 033737 004000 001362
8703 037126 001055
8704 037130 032737 010000 001362
8705 037136 001451
8706
8707
8708 037140 013700 001404
8709 037144 013701 001374
8710 037150 052737 100000 037430
8711
8712
8713 037156 022701 000020
8714 037162 103005
8715 037164 162701 000020
8716 037170 062700 000002
8717 037174 000770
8718 037176 012702 000001
8719 037202 010203
8720
8721
8722 037204 020301
8723 037206 001403
8724 037210 006302
8725 037212 005203
8726 037214 000773
8727 037216 012703 000013
8728
8729

.SBTTL COMPARE BUFFER SUBROUTINE

; THIS SUBROUTINE COMPARES THE CONTENTS OF BUFONE AND BUFTWO
; ASSUMING THAT BUFONE IS THE BUFFER FROM WHICH DATA WAS WRITTEN
; AND BUFTWO IS THE BUFFER TO WHICH DATA WAS READ. ERRORS IN BUFFER
; COMPARISON ARE REPORTED TO THE USER VIA THE USER'S ERROR CALL.

; CALL:

(1) JSR PC,CMPBUF
(2) .WORD WRITE BUFFER ADDRESS
 .WORD READ BUFFER ADDRESS
(3) BR ?? RETURN HERE IF NO ERROR
(4) NOP RETURN HERE IF ERROR
(5) ERROR ERROR DEFINED BY SUBROUTINE
(6) ???

CMPBUF:

MOV R0,-(SP) ; PUSH R0 ON STACK
MOV R1,-(SP) ; PUSH R1 ON STACK
MOV R2,-(SP) ; PUSH R2 ON STACK
MOV R3,-(SP) ; PUSH R3 ON STACK

CLR 150\$; CLEAR CORRECTION FLAG
; DETERMINE IF DATA SHOULD BE CORRECTED
BIT #DCK,RMER11 ; WAS THERE A DATA CHECK ??
BEQ 80\$; NO !!
BIT #ECH,RMER11 ; IS IT A HARD ERROR ??
BNE 80\$; YES !!
BIT #ECI,RMOFI ; WAS ECC CORRECTION ALLOWED ??
BNE 80\$; NO !!
BIT #FMT16,RMOFI ; IS THIS 16 BIT FORMAT ??
BEQ 80\$; NO !!

; CORRECT DATA USING ECC INFORMATION
MOV RMB00,R0 ; R0 = STARTING BUFFER ADDRESS
MOV RMEC11,R1 ; R1 = ECC POSITION
BIS #BIT15,150\$; SET CORRECTION FLAG
; MOVE R0 TO WORD BOUNDARY OF ERROR BURST
10\$: CMP #16.,R1 ; IS BIT POSITION > 1 WORD
BHS 20\$; NO !!
SUB #16.,R1 ; SUBTRACT 1 WORDS WORTH
ADD #2,R0 ; ADVANCE BUFFER ADDRESS 1 WORD
BR 10\$
20\$: MOV #1,R2 ; R2 = BIT POINTER
 MOV R2,R3 ; R3 = BIT NUMBER
; MOVE R2 TO STARTING BIT OF ERROR BURST
30\$: CMP R3,R1 ; IS R3 SAME AS R1 ??
BEQ 35\$; YES !!
ASL R2 ; SHIFT BIT POINTER
INC R3 ; INCREMENT BIT NUMBER
BR 30\$
35\$: MOV #11.,R3 ; R3 = LENGTH OF ERROR BURST
; CORRECT THE ERROR BURST


```

8730 037222 030237 001376      40$: BIT      R2,RMEC2I      ;IS THIS BIT SET IN ECC PATTERN ??
8731 037226 001405              BEQ      60$      ;NO - DO NOT CORRECT THIS BIT
8732 037230 030210              BIT      R2,(R0)    ;IS THE BIT PRESENTLY SET ??
8733 037232 001402              BEQ      50$      ;NO
8734 037234 040210              BIC      R2,(R0)    ;RESET THE BIT
8735 037236 000401              BR       60$
8736 037240 050210      50$: BIS      R2,(R0)    ;SET THE BIT
8737 037242 006302      60$: ASL      R2          ;SHIFT TO NEXT BIT
8738 037244 001004              BNE      70$
8739 037246 012702 000001      MOV      #1,R2      ;CONTINUE WITH FIRST BIT OF NEXT WORD
8740 037252 062700 000002      ADD      #2,R0
8741 037256 005303      70$: DEC      R3          ;END OF BURST ??
8742 037260 001360              BNE      40$      ;NO !!
8743 037262 017600 000010      80$: MOV      #10(SP),R0      ;R0 = WRITE BUFFER
8744 037266 062766 000002 000010      ADD      #2,10(SP)      ;MOVE SP TO READ ADDRESS
8745 037274 017601 000010      MOV      #10(SP),R1      ;R1 = READ BUFFER
8746 037300 062766 000002 000010      ADD      #2,10(SP)      ;MOVE SP TO RETURN ADDRESS
8747 037306 013702 001332      MOV      RMWCI,R2      ;R2 = NUMBER OF WORDS TRANSFER
8748 037312 163702 001402      SUB      RMWCO,R2
8749 037316 022021      90$: CMP      (R0)+,(R1)+      ;COMPARE DATA WORDS
8750 037320 001003              BNE      100$      ;EXIT IF NOT EQUAL
8751 037322 005302              DEC      R2          ;DECREMENT WORD COUNT
8752 037324 001374              BNE      90$      ;CONTINUE IF NOT DONE
8753 037326 000433              BR       110$      ;DONE COMPARE - NO ERROR
8754 037330
8755 037330 014037 001140      100$: MOV      -(R0),SGDAT      ;STORE GOOD DATA FOR TYPEOUT
8756 037334 014137 001142      MOV      -(R1),SBDAT      ;STORE BAD DATA FOR TYPEOUT
8757 037340 010037 001134      MOV      R0,SGADR      ;STORE ADDRESS OF GOOD DATA
8758 037344 010137 001136      MOV      R1,SBADR      ;STORE ADDRESS OF BAD DATA
8759 037350 010237 001174      MOV      R2,$TMPD      ;STORE WORD COUNT OF ERROR
8760 037354 062766 000004 000010      ADD      #4,10(SP)      ;MOVE SP TO USER'S ERROR CALL
8761 037362 112776 000336 000010      MOVB     #336, #10(SP)      ;WRITE ERROR NUMBER IN CALL
8762
8763
8764 037370 032737 100000 037430 ;CHANGE ERROR NUMBER IF ECC CORRECTION FAILED
8765 037376 001403              BIT      #BIT15,150$      ;WAS ECC CORRECTION USED ??
8766 037400 112776 000163 000010      BEQ      105$      ;NO !!
8767 037406 162766 000002 000010      MOVB     #163, #10(SP)      ;ECC CORRECTION FAILED
8768 037414 000240      105$: SUB      #2,10(SP)      ;MOVE SP TO RETURN IF ERROR
8769 037416
8770 037416 012603      110$: MOV      (SP)+,R3      ;POP STACK INTO R3
8771 037420 012602      MOV      (SP)+,R2      ;POP STACK INTO R2
8772 037422 012601      MOV      (SP)+,R1      ;POP STACK INTO R1
8773 037424 012600      MOV      (SP)+,R0      ;POP STACK INTO R0
8774 037426 000207      RTS      PC          ;RETURN TO USER
8775
8776 037430 000000      150$: .WORD      ;ECC CORRECTION FLAG

```

```

8777 .SBTTL GET STATUS SUBROUTINE
8778 ;THIS SUBROUTINE SETS UP THE "GET INDEX TABLE" AND THE "GET
8779 ;BUFFER" FOR READING ALL SUBSYSTEM REGISTERS VIA THE GET SUBROUTINE
8780 ;AND THEN RETURNS TO THE USER.
8781 ;CALL: JSR PC,GETSTS
8782 ;??? RETURN HERE
8783
8784 GETSTS:
8785     MOV R0,-(SP) ;: PUSH R0 ON STACK
8786     MOV R1,-(SP) ;: PUSH R1 ON STACK
8787     MOV R2,-(SP) ;: PUSH R2 ON STACK
8788     MOV #GETINX,R0 ;: R0= ADDRESS OF INDEX TABLE
8789     MOV #RMEC2I+2,R1 ;: R1 = ADDRESS OF GET BUFFER
8790     MOV #RMEC2,R2 ;: R2 = REGISTER INDE
8791 2$: MOV R2,(R0)+ ;: WRITE REGISTER INDEX IN TABLE
8792     CLR -(R1) ;: CLEAR CORRESPONDING LOCATION
8793 3$: SUB #2,R2 ;: DECREMENT TO NEXT INDEX
8794     BMI 4$ ;: BRANCH OUT IF DONE
8795     CMP #RMOB,R2 ;: DONT WRITE RMOB INDEX
8796     BNE 2$
8797     CLR -(R1)
8798     BR 3$
8799 4$: MOVB #200,(R0)+ ;: WRITE TERMINATOR
8800     MOV (SP)+,R2 ;: POP STACK INTO R2
8801     MOV (SP)+,R1 ;: POP STACK INTO R1
8802     MOV (SP)+,R0 ;: POP STACK INTO R0
8803     NOP
8804     RTS PC
8805
8806 037432 010046
8807 037432 010146
8808 037434 010246
8809 037436 010246
8810 037440 012700 001506
8811 037444 012701 001400
8812 037450 012702 000046
8813 037454 110220
8814 037456 005041
8815 037460 162702 000002
8816 037464 100405
8817 037466 022702 000022
8818 037472 001370
8819 037474 005041
8820 037476 000770
8821 037500 112720 000200
8822 037504 012602
8823 037506 012601
8824 037510 012600
8825 037512 000240
8826 037514 000207
8827

```

```
8808 .SBTTL GET SUBROUTINE
8809
8810 ;THIS SUBROUTINE READS THE REGISTERS WHICH ARE LISTED IN THE
8811 ;"GET INDEX TABLE" AND STORES THEIR VALUES IN THE CORRESPONDING
8812 ;LOCATION IN THE "GET REGISTER BUFFER". FOR EXAMPLE, AN
8813 ;ENTRY OF 04 IN THE TABLE WILL CAUSE THE SUBROUTINE TO
8814 ;READ "R04" AND STORE ITS CONTENTS AT THE LOCATION IN
8815 ;THE BUFFER ASSIGNED TO THAT REGISTER. THE NUMBER OF
8816 ;REGISTERS TO BE READ IS VARIABLE FROM 1 TO 22; THE INDEX
8817 ;TABLE MUST BE TERMINATED WITH A CONTROL BYTE (200)
8818 ;WHICH SHOULD FOLLOW THE LAST ENTRY.
8819
8820 SUBROUTINE CALL:
8821 (1) "GET INDEX TABLE" HAS BEEN LOADED WITH REGISTER INDEX
8822     VALUES AND TERMINATED WITH A CONTROL BYTE
8823 (2) "GET INPUT BUFFER" IS AVAILABLE FOR USE. (NOTE THAT
8824     UNUSED LOCATIONS, I.E., ENTRIES IN BUFFER CORRESPONDING
8825     TO REGISTERS NOT READ, ARE NOT CHANGED.)
8826 (3) JSR PC GET
8827     BR ??? RETURN HERE IF NO ERROR FOUND
8828     NOP RETURN HERE IF ANY ERROR FOUND
8829     ERROR SUB DEFINES ERROR NUMBER
8830     ???
8831
8832 GET: NOP
8833     ADD #4,(SP) ;CLEAR ERROR NUMBER IN USER'S
8834     CLRB 2(SP) ;ERROR CALL
8835     SUB #4,(SP)
8836     MOV R0,-(SP) ;PUSH R0 ON STACK
8837     MOV R1,-(SP) ;PUSH R1 ON STACK
8838     MOV R2,-(SP) ;PUSH R2 ON STACK
8839     MOV R3,-(SP) ;PUSH R3 ON STACK
8840     MOV R4,-(SP) ;PUSH R4 ON STACK
8841     MOV ERRVEC,-(SP) ;PUSH ERRVEC ON STACK
8842     MOV ERRVEC+2,-(SP) ;PUSH ERRVEC+2 ON STACK
8843     MOV $BASE,R0
8844     MOV #GETBUF,R2
8845     MOV #GETINX,R4
8846     MOV #5$ ,ERRVEC ;SETUP FOR TIMEOUT
8847     MOV #PR6 ,ERRVEC+2
8848     MOV RMCS2(R0),STMP0 ;GET "NED" STATUS
8849     MOV RMCS1(R0),STMP1 ;GET "DVA" STATUS
8850     BIT #DVA,STMP1 ;DEVICE AVAILABLE??
8851     BNE 3$ ;YES!!
8852     ADD #4,16(SP) ;WRITE ERROR NUMBER IN USER'S
8853     MOVB #112,216(SP) ;ERROR CALL
8854     BR 7$
8855     TSTB (R4) ;DONE??
8856     BMI 9$ ;YES!!
8857     MOVB (R4),R1 ;R1 = REGISTER ADDRESS
8858     BIC #1C10XMSK,R1 ;CLEAR ANY SIGN EXTENSION
8859     ADD R0,R1
8860     MOVB (R4)+,R3 ;R3 = STORAGE ADDRESS FOR REGISTER
8861     BIC #1C10XMSK,R3 ;CLEAR ANY SIGN EXTENSION
8862     ADD R2,R3
8863     MOV (R1),(R3) ;READ REGISTER
```

037516	000240				
037520	062716	000004			
037524	105076	000000			
037530	162716	000004			
037534	010046				
037536	010146				
037540	010246				
037542	010346				
037544	010446				
037546	013746	000004			
037552	013746	000006			
037556	013700	001276			
037562	012702	001330			
037566	012704	001506			
037572	012737	037700	000004		
037600	012737	000300	000006		
037606	016037	000010	001174	1\$:	
037614	016037	000000	001176		
037622	032737	004000	001176		
037630	001007				
037632	062766	000004	000016		
037640	112776	000112	000016		
037646	000423				
037650	105714			3\$:	
037652	100433				
037654	111401				
037656	042701	177700			
037662	060001				
037664	112403				
037666	042703	177700			
037672	060203				
037674	011113				

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY1 30(1046) 23-NOV-77 12:38 PAGE 178
GET SUBROUTINE

SEQ 0178

8864	037676	000764		BR	3\$	
8865						
8866	037700	022626		5\$:	CMP	(SP)+, (SP)+ ; RESTORE STACK
8867	037702	062766	000004 000016		ADD	#4, 16(SP) ; WRITE ERROR NUMBER IN
8868	037710	112776	000007 000016		MOVB	#7, 316(SP) ; USER'S ERROR CALL
8869	037716	162766	000002 000016	7\$:	SUB	#2, 16(SP)
8870	037724	105714		8\$:	TSTB	(R4) ; DONE CLEARING??
8871	037726	100405			BMI	9\$; YES!!
8872	037730	005003			CLR	R3 ; CLEAR REMAINING STORAGE
8873	037732	112403			MOVB	(R4)+, R3 ; LOCATIONS
8874	037734	060203			ADD	R2, R3
8875	037736	005013			CLR	(R3)
8876	037740	000771			BR	8\$;
8877	037742			9\$:		
8878	037742	012637	000006		MOV	(SP)+, ERRVEC+2 ; POP STACK INTO ERRVEC+2
8879	037746	012637	000004		MOV	(SP)+, ERRVEC ; POP STACK INTO ERRVEC
8880	037752	012604			MOV	(SP)+, R4 ; POP STACK INTO R4
8881	037754	012603			MOV	(SP)+, R3 ; POP STACK INTO R3
8882	037756	012602			MOV	(SP)+, R2 ; POP STACK INTO R2
8883	037760	012601			MOV	(SP)+, R1 ; POP STACK INTO R1
8884	037762	012600			MOV	(SP)+, R0 ; POP STACK INTO R0
8885	037764	000207			RTS	PC ; RETURN
8886						

K14

SEQ 0179

```
8887 .SBTTL PUT SUBROUTINE
8888
8889 ; THIS SUBROUTINE WRITES THE REGISTERS WHICH ARE LISTED IN THE
8890 ; "PUT INDEX TABLE" WITH THE CONTENTS OF THE CORRESPONDING
8891 ; LOCATION IN THE "PUT REGISTER BUFFER". THE NUMBER OF
8892 ; REGISTERS WRITTEN IS VARIABLE; THE INDEX TABLE MUST
8893 ; BE TERMINATED WITH A CONTROL BYTE (200) WHICH SHOULD
8894 ; FOLLOW THE LAST ENTRY.
8895
8896 ; SUBROUTINE CALL:
8897
8898 ; (1) "PUT INDEX TABLE" HAS BEEN LOADED WITH INDEX VALUES
8899 ; OF REGISTERS TO BE WRITTEN.
8900 ; (2) "PUT REGISTER BUFFER" CONTAINS CONTENTS OF EACH
8901 ; REGISTER TO BE WRITTEN.
8902 ; (3) JSR PC,PUT
8903 ; BR ??? RETURN HERE IF NO ERROR FOUND
8904 ; NOP RETURN HERE IF ANY ERROR FOUND
8905 ; ERROR SUB DEFINES ERROR NUMBER
8906 ; ???
8907
8908 PUT: NOP
8909 MOV R0,-(SP) ; PUSH R0 ON STACK
8910 MOV R1,-(SP) ; PUSH R1 ON STACK
8911 MOV R2,-(SP) ; PUSH R2 ON STACK
8912 MOV R3,-(SP) ; PUSH R3 ON STACK
8913 MOV R4,-(SP) ; PUSH R4 ON STACK
8914 MOV ERRVEC,-(SP) ; PUSH ERRVEC ON STACK
8915 MOV ERRVEC+2,-(SP) ; PUSH ERRVEC+2 ON STACK
8916 SB=SE,R0
8917 #PUTBUF,R2
8918 #PUTINX,R4
8919 MOV #55,ERRVEC ; SETUP FOR TIMEOUT
8920 #PA6,ERRVEC+2
8921 1$: MOV RMCS2(R0),$TMP0 ; GET "MED" STATUS
8922 MOV RMCS1(R0),$TMP1 ; GET "DVA" STATUS
8923 BIT #DVA,$TMP1 ; DEVICE AVAILABLE??
8924 BNE 3$ ; YES!!
8925 ADD #4,16(SP) ; WRITE ERROR NUMBER IN
8926 MOV #112,216(SP) ; USER'S ERROR CALL
8927 BR 7$
8928 3$: TSTB (R4) ; DONE??
8929 BMI 9$ ; YES!!
8930 MOV (R4),R1 ; R1 = REGISTER ADDRESS
8931 BIC #CIDXMSK,R1 ; CLEAR ANY SIGN EXTENSION
8932 ADD R0,R1
8933 MOV (R4)+,R3 ; R3 = STORAGE ADDRESS
8934 BIC #CIDXMSK,R3 ; CLEAR ANY SIGN EXTENSION
8935 ADD R2,R3
8936 MOV (R3),(R1) ; WRITE REGISTER
8937 BR 3$
8938
8939 5$: CMP (SP)+,(SP)+ ; ADJUST STACK
8940 ADD #4,16(SP) ; WRITE ERROR NUMBER IN
8941 MOV #7,216(SP) ; USER'S ERROR CALL
8942 7$: SUB #2,16(SP)
```

Address	Index	Register	Value	Index	Register	Value
037766	000240					
037770	010046					
037772	010146					
037774	010246					
037776	010346					
040000	010446					
040002	013746	000004				
040006	013746	000006				
040012	013700	001276				
040016	012702	001400				
040022	012704	001535				
040026	012737	040134	000004			
040034	012737	000300	000006			
040042	016037	000010	001174			
040050	016037	000000	001176			
040056	032737	004000	001176			
040064	001007					
040066	062766	000004	000016			
040074	112776	000112	000016			
040102	000423					
040104	105714					
040106	100424					
040110	111401					
040112	042701	177700				
040116	060001					
040120	112403					
040122	042703	177700				
040126	060203					
040130	011311					
040132	000764					
040134	022626					
040136	062766	000004	000016			
040144	112776	000007	000016			
040152	162766	000002	000016			

L14

CZRM080 RM03/2 FCTNL IST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 180
PUT SUBROUTINE

SEQ 0180

8943					
8944	040160			95:	
8945	040160	012637	000006	MOV	(SP)+,ERRVEC+2
8946	040164	012637	000004	MOV	(SP)+,ERRVEC
8947	040170	012604		MOV	(SP)+,R4
8948	040172	012603		MOV	(SP)+,R3
8949	040174	012602		MOV	(SP)+,R2
8950	040176	012601		MOV	(SP)+,R1
8951	040200	012600		MOV	(SP)+,R0
8952	040202	000207		RTS	PC
8953					

					::POP STACK INTO ERRVEC+2
					::POP STACK INTO ERRVEC
					::POP STACK INTO R4
					::POP STACK INTO R3
					::POP STACK INTO R2
					::POP STACK INTO R1
					::POP STACK INTO R0
					::RETURN

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 181
SIZE CLOCK SUBROUTINE

SEQ 0181

```

8954      .SBTTL  SIZE CLOCK SUBROUTINE
8955
8956      040204      013746      000004      000004      MOV      ERRVEC, -(SP)      ;; PUSH ERRVEC ON STACK
8957      040204      013746      000006      000006      MOV      ERRVEC+2, -(SP)      ;; PUSH ERRVEC+2 ON STACK
8958      040210      013746      000006      000006      MOV      #15, ERRVEC      ;; SET UP FOR BUS TIMEOUT
8959      040214      012737      040252      000004      MOV      #PR6, ERRVEC+2      ;; SET UP FOR BUS TIMEOUT
8960      040222      012737      000300      000006      MOV      #177546, CLKADR      ;; LOAD ADDRESSES FOR KW11-L
8961      040230      012737      177546      001502      MOV      #100, CLKVCT      ;; LOAD ADDRESSES FOR KW11-L
8962      040236      012737      000100      001504      MOV      #100, CLKVCT      ;; LOAD ADDRESSES FOR KW11-L
8963      040244      005777      141232      001504      TST      CLKADR      ;; TEST FOR KW11-L PRESENT
8964      040250      000421      000004      001504      BR      3$      ;; YES - KW11-L IS PRESENT
8965      040254      022626      040304      000004      1$: CMP      (SP)+, (SP)+      ;; RESTORE SP
8966      040254      012737      040304      000004      MOV      #25, ERRVEC      ;; SET UP FOR BUS TIMEOUT
8967      040262      012737      172540      001502      MOV      #172540, CLKADR      ;; LOAD ADDRESSES FOR KW11-P CLOCK
8968      040270      012737      000104      001504      MOV      #104, CLKVCT      ;; LOAD ADDRESSES FOR KW11-P CLOCK
8969      040276      005777      141200      001504      TST      CLKADR      ;; TEST FOR KW11-P PRESENT
8970      040302      000404      000004      001504      BR      3$      ;; YES - KW11-P IS PRESENT
8971      040304      022626      000002      000004      2$: CMP      (SP)+, (SP)+      ;; RESTORE SP
8972      040306      062766      000002      000004      ADD      #2, 4(SP)      ;; MOVE RETURN TO ERROR
8973      040314      012637      000006      000004      3$: MOV      (SP)+, ERRVEC+2      ;; POP STACK INTO ERRVEC+2
8974      040314      012637      000004      000004      MOV      (SP)+, ERRVEC      ;; POP STACK INTO ERRVEC
8975      040320      012637      000004      000004      RTS      PC      ;; RETURN TO USER
8976      040324      000207      000004      000004

```

N14

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19MACY11 30(1046) 23-NOV-77 12:38 PAGE 182
TIMEOUT SUBROUTINE

SEQ 0182

```

8977 .SBTTL TIMEOUT SUBROUTINE
8978
8979 ;THIS SUBROUTINE WAITS FOR GO TO RESET OR FOR A TIMEOUT GREATER THAN
8980 ;500 MS, WHICH EVER OCCURS FIRST, AND THEN RETURNS.
8981
8982 ;CALL: JSR PC, TIMEOUT
8983 ;???
8984 ;RETURN HERE
8985
8986 TIMEOUT:
8987 MOV R0, -(SP) ;: PUSH R0 ON STACK
8988 MOV R1, -(SP) ;: PUSH R1 ON STACK
8989 MOV R2, -(SP) ;: PUSH R2 ON STACK
8990 MOV ERRVEC, -(SP) ;: PUSH ERRVEC ON STACK
8991 MOV ERRVEC+2, -(SP) ;: PUSH ERRVEC+2 ON STACK
8992 MOV #45, ERRVEC ;: SETUP FOR BUS TIMEOUT - 04 TRAP
8993 MOV #PR6, ERRVEC+2
8994 MOV $BASE, R0 ;: R0=RM03 ADDRESS
8995 MOV CLKAOR, R1 ;: R1=CLOCK ADDRESS
8996 MOV #31, R2 ;: R2=NUMBER OF CLOCK CYCLES
8997 ;KW11-P CLOCK??
8998 ;NO!!
8999 MOV #1, 2(R1) ;: SET COUNTER
9000 MOV #BIT2:BIT0, (R1) ;: START COUNTER
9001
9002 1$: CMP R1, #172540 ;: KW11-P CLOCK??
9003 BNE 2$ ;: NO!!
9004 MOV #1, 2(R1) ;: SET COUNTER
9005 2$: MOV #BIT2:BIT0, (R1) ;: START COUNTER
9006
9007 3$: MOV RMCS1(R0), -(SP) ;: GET STATUS
9008 BIC #C<RDY!GO>, (SP)
9009 CMP #RDY, (SP)+ ;: RDY=1, GO=0??
9010 BEQ 5$ ;: YES!!
9011 BIT #BIT7, (R1) ;: TIMER DONE??
9012 BEQ 3$ ;: NO!!
9013 DEC R2 ;: DEC NUMBER OF CYCLES
9014 BNE 1$ ;: CONTINUE IF NOT DONE
9015 BR 5$
9016
9017 4$: CMP (SP)+, (SP)+ ;: ADJUST STACK
9018 ADD #4, 12(SP) ;: MOVE SP TO USER'S CALL
9019 MOVB #7, 212(SP) ;: WRITE ERROR NUMBER
9020 SUB #2, 12(SP)
9021
9022 5$: MOV (SP)+, ERRVEC+2 ;: POP STACK INTO ERRVEC+2
9023 MOV (SP)+, ERRVEC ;: POP STACK INTO ERRVEC
9024 MOV (SP)+, R2 ;: POP STACK INTO R2
9025 MOV (SP)+, R1 ;: POP STACK INTO R1
9026 MOV (SP)+, R0 ;: POP STACK INTO R0
9027 RTS PC ;: RETURN TO USER

```



```

9023
9024
9025
9026
9027
9028
9029
9030
9031
9032
9033
9034
9035
9036
9037
9038
9039
9040
9041
9042
9043
9044
9045
9046
9047
9048
9049
9050
9051
9052
9053
9054
9055
9056
9057
9058
9059
9060
9061 040512
9062
9063
9064 040512 062716 000004
9065 040516 105076 000000
9066 040522 162716 000004
9067
9068
9069
9070 040526 013737 001340 001142
9071 040534 042737 177770 001142
9072 040542 013737 001234 001140
9073 040550 042737 177770 001140
9074 040556 123737 001140 001142
9075
9076 040564 001415
9077 040566 062716 000004
9078 040572 112776 000001 000000

```

```

.SBTTL PRIMARY ERROR CHECK SUBROUTINE

```

```

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

```

```

;CURRENT UNIT IS SELECTED, I.E., THE UNIT SELECT BITS OF RMCS2
;(BITS 0-2) EQUAL THE UNIT BEING TESTED;

```

```

;SELECTED UNIT IS AVAILABLE, I.E., DVA (BIT 11 OF RMCS1) IS SET
AND NED (BIT 12 OF RMCS2) IS RESET;

```

```

;LAST COMMAND WAS COMPLETED, I.E., THE MASSBUS CONTROLLER IS
READY (BIT 7 OF RMCS1) AND THE GO BIT IS RESET (BIT 0 OF RMCS1) OR THE
DRIVE READY BIT (BIT 7 OF RMDS) IS SET.

```

```

;NO PARITY ERROR OCCURRED WHEN READING REMOTE REGISTERS,
I.E., MCPE = 0.

```

```

;NO PARITY ERROR OCCURRED WHEN WRITING REMOTE REGISTERS,
I.E., PAR = 0, OR, PAR = DPE = 1

```

```

;THE SUBROUTINE ASSUMES THAT:

```

```

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

```

```

;($UNIT) CONTAINS THE DRIVE NUMBER

```

```

;THE SUBROUTINE IS CALLED AS FOLLOWS:

```

```

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

```

```

PRIERR:

```

```

;CLEAR USER'S ERROR CALL

```

```

ADD    #4,(SP)          ;MOVE (SP) TO ERROR CALL
CLRB   #2(SP)           ;CLEAR ERROR NUMBER
SUB     #4,(SP)          ;MOVE (SP) TO NO ERROR RETURN

```

```

;REPORT AN ERROR IF THE WRONG UNIT IS SELECTED

```

```

MOV     RMCS2I,$BDDAT   ;CORRECT UNIT SELECTED??
BIC     #1CUNTMSK,$BDDAT
MOV     $UNIT,$GDDAT    ;GOOD DATA FOR TYPEOUT
BIC     #1CUNTMSK,$GDDAT
CMPB    $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
;DRIVE NUMBERS
BEQ     1$              ;YES!!
ADD     #4,(SP)
MOVB    #1,$(SP)        ;ERROR 1

```

9079	040600	162716	000002		JSR	#2,(SP)	;MOVE SP TO RETURN FOR ERROR
9080	040604	004736			JSR	PC,#2(SP)+	;REPORT WRONG UNIT SELECTED
9081	040606	162716	000010		SUB	#10,(SP)	;RESTORE (SP)
9082	040612	000240			NOP		
9083	040614	000137	041334		JMP	10\$;SKIP OTHER CHECKS
9084	040620						
9085							
9086							
9087							
9088	040620	032737	004000	001330			
9089	040626	001045			BIT	#DVA,RMCS11	;DEVICE AVAILABLE??
9090	040630	013737	001330	001140	BNE	5\$;YES!!
9091	040636	052737	004000	001140	MOV	RMCS11,\$GDDAT	;EXPECTED STATUS
9092	040644	013737	001330	001142	BIS	#DVA,\$GDDAT	
9093	040652	062716	000004		MOV	RMCS11,\$BDDAT	;RECEIVED STATUS
9094	040656	112776	000002	000000	ADD	#4,(SP)	
9095	040664	032737	010000	001340	MOVB	#2,#2(SP)	;ERROR #2
9096	040672	001414			BIT	#MED,RMCS21	;WAS MED SET??
9097	040674	013737	001340	001140	BEQ	2\$;NO!!
9098	040702	013737	001340	001142	MOV	RMCS21,\$GDDAT	;EXPECTED STATUS
9099	040710	042737	010000	001140	MOV	RMCS21,\$BDDAT	;RECEIVED STATUS
9100	040716	112776	000003	000000	BIC	#MED,\$GDDAT	
9101	040724	162716	000002		MOVB	#3,#3(SP)	;YES - CHANGE ERROR NUMBER
9102	040730	004736			SUB	#2,(SP)	;MOVE SP TO RETURN FOR ERROR
9103	040732	162716	000010		JSR	PC,#2(SP)+	;REPORT DEVICE NOT AVAILABLE
9104	040736	000240			SUB	#10,(SP)	;RESTORE (SP)
9105	040740	000575			NOP		
9106	040742				BR	10\$;SKIP OTHER CHECKS
9107							
9108							
9109	040742	032737	000200	001330			
9110	040750	001030			BIT	#RDY,RMCS11	;CONTROLLER READY??
9111	040752	013737	001330	001140	BNE	7\$;YES!!
9112	040760	052737	000200	001140	MOV	RMCS11,\$GDDAT	;EXPECTED STATUS
9113	040766	042737	160001	001140	BIS	#RDY,\$GDDAT	
9114	040774	013737	001330	001142	BIC	#SC:TR:MCPE:GO,\$GDDAT	
9115	041002	062716	000004		MOV	RMCS11,\$BDDAT	;RECEIVED STATUS
9116	041006	112776	000004	000000	ADD	#4,(SP)	
9117	041014	162716	000002		MOVB	#4,#4(SP)	;ERROR #4
9118	041020	004736			SUB	#2,(SP)	;MOVE SP TO RETURN FOR ERROR
9119	041022	162716	000010		JSR	PC,#2(SP)+	;REPORT CONTROLLER NOT READY
9120	041026	000240			SUB	#10,(SP)	;RESTORE (SP)
9121	041030	000541			NOP		
9122	041032				BR	10\$;SKIP OTHER CHECKS
9123							
9124							
9125	041032	032737	000001	001330			
9126	041040	001431			BIT	#GO,RMCS11	;GO RESET??
9127	041042	032737	000200	001342	BEQ	8\$;YES!!
9128	041050	001025			BIT	#DRY,RMDSI	;DRIVE READY??
9129	041052	013737	001330	001140	BNE	8\$;YES!!
9130	041060	042737	160001	001140	MOV	RMCS11,\$GDDAT	;EXPECTED STATUS
9131	041066	013737	001330	001142	BIC	#SC:TR:MCPE:GO,\$GDDAT	
9132	041074	062716	000004		MOV	RMCS11,\$BDDAT	;RECEIVED STATUS
9133	041100	112776	000005	000000	ADD	#4,(SP)	
9134	041106	162716	000002		MOVB	#5,#5(SP)	;ERROR #5
					SUB	#2,(SP)	;MOVE SP TO RETURN FOR ERROR

```

9135 041112 004736          JSR    PC,2(SP)+      ;REPORT DRIVE NOT READY
9136 041114 162716 000010  SUB    #10,(SP)      ;RESTORE (SP)
9137 041120 000240          NOP
9138 041122 000504          BR     10$
9139 041124
9140
9141          8$:
9142          ;REPORT AN ERROR IF THE RM CONTROLLER DETECTED BAD
9143          ;PARITY ON THE MASSBUS CONTROL BUS
9144 041124 032737 020000 001330 BIT    #MCPE,RMC511      ;PARITY ERROR ??
9145 041132 001425          BEQ     9$              ;NO!!
9146 041134 013737 001330 001140 MOV    RMC511,$GDDAT      ;EXPECTED STATUS
9147 041142 042737 160001 001140 BIC    #SC:TR:MCPE:GO,$GDDAT
9148 041150 013737 001330 001142 MOV    RMC511,$BDDAT      ;RECEIVED STATUS
9149 041156 062716 000004          ADD    #4,(SP)      ;MOVE STACK TO USER'S ERROR
9150 041162 112776 000013 000000 MOVB   #13,2(SP)      ;ERROR #47
9151 041170 162716 000002          SUB    #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
9152 041174 004736          JSR    PC,2(SP)+      ;REPORT ERROR VIA USER
9153 041176 162716 000010  SUB    #10,(SP)      ;RESTORE STACK
9154 041202 000240          NOP
9155 041204 000453          BR     10$
9156 041206
9157          9$:
9158          ;REPORT AN ERROR IF THE RM03 DETECTED A CONTROL BUS PARITY ERROR
9159 041206 032737 000010 001344 BIT    #PAR,RMER11      ;WAS THERE A PARITY ERROR??
9160 041214 001451          BEQ     11$            ;NO!!
9161 041216 032737 000010 001372 BIT    #DPE,RMER21      ;WAS IT THE CONTROL BUS??
9162 041224 001045          BNE     11$            ;NOT SURE!!
9163 041226 032737 000010 001414 BIT    #PAR,RMER10      ;DID TEST SET PAR ??
9164 041234 001413          BEQ     93$            ;NO!!
9165 041236 010046          MOV     R0,-(SP)        ;PUSH R0 ON STACK
9166 041240 012700 001535          MOV    #PUTINX,R0      ;R0 POINTS TO INDEX TABLE
9167 041244 122710 000014          CMPB   #RMER1,(R0)    ;SEARCH TABLE FOR RMER1
9168 041250 001002          BNE     92$
9169 041252 012600          MOV     (SP)+,R0        ;POP STACK INTO R0
9170 041254 000431          BR     11$            ;PAR WAS SET BY TEST
9171 041256 105720          TSTB   (R0)+          ;END OF TABLE??
9172 041260 100371          BPL     91$            ;NO!!
9173 041262 012600          MOV     (SP)+,R0        ;POP STACK INTO R0
9174 041264 013737 001344 001140 93$: MOV    RMER11,$GDDAT      ;EXPECTED STATUS
9175 041272 042737 000010 001140 BIC    #PAR,$GDDAT
9176 041300 013737 001344 001142 MOV    RMER11,$BDDAT      ;RECEIVED STATUS
9177 041306 062716 000004          ADD    #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
9178 041312 112776 000050 000000 MOVB   #50,2(SP)      ;WRITE THE ERROR NUMBER
9179 041320 162716 000002          SUB    #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
9180 041324 004736          JSR    PC,2(SP)+      ;REPORT THE ERROR
9181 041326 162716 000010  SUB    #10,(SP)      ;MOVE SP TO NO ERROR RETURN
9182 041332 000240          NOP
9183 041334 062716 000010 10$: ADD    #10,(SP)      ;RETURN TO ERROR
9184 041340 000240          NOP                    ;RETURN TO NO ERROR
9185 041342 000207          RTS     PC

```

```

9186 .SBTTL SECONDARY ERROR CHECK SUBROUTINE
9187
9188 ;THE ERROR CHECK SUBROUTINE PROVIDES DETECTION OF SECONDARY ERRORS
9189 ;SUCH AS UNEXPECTED ERRORS AND UNEXPECTED REGISTER
9190 ;CONTENTS. THESE ERRORS ARE DEEMED
9191 ;SECONDARY IN THAT THEY ARE NOT NECESSARILY ASSOCIATED WITH THE OPERATION
9192 ;BEING PERFORMED. WHEN THE SUBROUTINE IDENTIFIES SUCH AN ERROR, IT MOVES
9193 ;THE ERROR NUMBER TO THE ERROR CALL IN THE TEST ROUTINE AND THEN RETURNS
9194 ;TO THE TEST ROUTINE WHICH MAKES THE ERROR CALL. AFTER THE TEST ROUTINE
9195 ;MAKES THE ERROR CALL, IT RETURNS TO THE SUBROUTINE WHICH THEN LOOKS FOR
9196 ;OTHER ERRORS. WHEN ALL ERRORS HAVE BEEN REPORTED, THE SUBROUTINE
9197 ;RETURNS TO THE ADDRESS FOLLOWING THE SUBROUTINE CALL.
9198
9199 CALL: JSR PC,SECERR
9200 BR ??? RETURN HERE IF NO ERROR
9201 NOP RETURN HERE TO REPORT AN ERROR
9202 ERROR ERROR NUMBER DEFINED BY SUB
9203 JSR PC,2(SP)+ GO BACK TO SUB FOR MORE ERROR CHECKS
9204 ??? RETURN HERE IF NO MORE ERRORS
9205
9206 ;NOTE: THE SUBROUTINE ASSUMES THAT REGISTERS HAVE BEEN STORED AT THE
9207 ;INPUT REGISTER BUFFER.
9208
9209 SECERR:
9210
9211 ;*****
9212 ;STORE FUNCTION CODE AND CLEAR USER'S ERROR NUMBER
9213 MOV RMCS10,515$ ;STORE FUNCTION CODE
9214 BIC #1C(F0!F1!F2!F3!F4),515$
9215 ADD #4,(SP) ;MOVE (SP) TO ERROR CALL
9216 CLRB 2(SP) ;CLEAR ERROR NUMBER
9217 SUB #4,(SP) ;MOVE (SP) TO NO ERROR RETURN
9218
9219 ;*****
9220 ;CHECK SECONDARY ERRORS COMMON TO ALL COMMANDS
9221
9222 ;REPORT ERROR IF DRIVE IS NOT READY, I.E. IF DRY = 0
9223 BIT #DRY,RMDSI ;DRIVE READY??
9224 BNE 5$ ;YES!!
9225 MOV RMDSI,$BDDAT ;BAD DATA FOR TYPEOUT
9226 BIC #1CDRY,$BDDAT
9227 MOV #DRY,$GDDAT ;GOOD DATA FOR TYPEOUT
9228 ADD #4,(SP)
9229 MOVB #10,2(SP) ;ERROR NUMBER
9230 SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
9231 JSR PC,2(SP)+ ;REPORT NOT READY
9232 SUB #10,(SP) ;RESTORE (SP) TO ERROR N
9233 NOP
9234
9235 ;REPORT ERROR IF GO BIT IS NOT RESET
9236 5$: BIT #GO,RMCS11 ;GO BIT RESET??
9237 BEQ 10$ ;YES!!
9238 MOV RMCS11,$BDDAT ;BAD DATA FOR TYPEOUT
9239 BIC #1CGO,$BDDAT
9240 CLR $GDDAT ;GOOD DATA FOR TYPEOUT
9241 ADD #4,(SP)

```

```

9242 041510 112776 000011 000000      MOVB    #11,2(SP)      ;ERROR NUMBER
9243 041516 162716 000002      SUB      #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
9244 041522 004736      JSR      PC,2(SP)+  ;REPORT DEVICE NOT AVAILABLE
9245 041524 162716 000010      SUB      #10,(SP)     ;RESTORE (SP)
9246 041530 000240      NOP
9247
9248      ;REPORT ERROR IF FUNCTION CODE READ FROM DEVICE IS NOT CORRECT
9249 041532 013737 001330 001142 10$:  MOV      RMCS1I,$BDDAT ;IS FUNCTION CODE CORRECT??
9250 041540 042737 177701 001142      BIC      #1C76,$BDDAT
9251 041546 013737 045204 001140      MOV      S155,$GDDAT ;EXPECTED FUNCTION CODE
9252 041554 023737 001142 001140      CMP      $BDDAT,$GDDAT
9253 041562 001413      BEQ      15$      ;YES!!
9254 041564 062716 000004      ADD      #4,(SP)
9255 041570 112776 000012 000000      MOVB    #12,2(SP)     ;ERROR NUMBER
9256 041576 162716 000002      SUB      #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
9257 041602 004736      JSR      PC,2(SP)+  ;REPORT WRONG FUNCTION CODE
9258 041604 162716 000010      SUB      #10,(SP)     ;RESTORE (SP)
9259 041610 000240      NOP
9260 041612
9261      15$:
9262      ;REPORT AN ERROR IF COMPOSITE ERROR IS SET AND NO OTHER
9263      ;ERRORS ARE SET OR IF COMPOSITE ERROR IS NOT SET AND
9264      ;OTHER ERRORS ARE SET
9265 041616 005037 001140      CLR      $GDDAT      ;EXPECT "ERR"=0
9266 041622 001003 001344      TST      RMER1I      ;IS RMER1 =0??
9267 041624 005737 001372      BNE      20$      ;NO!!
9268 041630 001403      TST      RMER2I      ;IS RMERZ=0??
9269 041632 052737 040000 001140 20$:  BEQ      25$      ;YES!!
9270 041640 013737 001342 001142 25$:  BIS      #ERR,$GDDAT ;"ERR" SHOULD BE SET
9271 041646 042737 137777 001142      MOV      RMDSI,$BDDAT
9272 041654 023737 001140 001142      BIC      #1CERA,$BDDAT
9273 041662 001412      CMP      $GDDAT,$BDDAT ;IS "ERR" OK??
9274 041664 062716 000004      BEQ      30$      ;YES!!
9275 041670 112776 000047 000000      ADD      #4,(SP)     ;MOVE SP TO USER'S ERROR
9276 041676 162716 000002      MOVB    #4,2(SP)     ;WRITE ERROR NUMBER
9277 041702 004736      SUB      #2,(SP)      ;MOVE SP TO ERROR RETURN
9278 041704 162716 000010      JSR      PC,2(SP)+  ;REPORT INVALID COMP ERROR
9279      SUB      #10,(SP)
9280
9281      ;REPORT AN ERROR IF "TRE" IS SET AND NONE OF THE BITS WHICH SET
9282      ;TRE IS SET, OR IF TRE IS NOT SET AND ONE OR MORE BITS WHICH
9283      ;SET TRE IS SET
9284 041710 005037 001140 30$:  CLR      $GDDAT      ;EXPECT "TRE" =0
9285 041714 013746 001340      MOV      RMCS2I,-(SP) ;WAS DLT, WCE, UPE, NED, NEM
9286 041720 042726 000377      BIC      #377,(SP)+  ;PGE, MXF OR MOPE SET
9287 041724 001710      BNE      35$      ;YES!!
9288 041726 032737 040000 001342      BIT      #ERR,RMDSI ;WAS EXCEPTION RECEIVED??
9289 041734 001407      BEQ      40$      ;NO!!
9290 041736 022737 000030 045204      CMP      #SEARCH,S155 ;WAS DATA TRANSFERRED??
9291 041744 103003      BHIS     40$      ;NO!!
9292 041746 052737 040000 001140 35$:  BIS      #TRE,$GDDAT ;"TRE" SHOULD BE SET
9293 041754 013737 001330 001142 40$:  MOV      RMCS1I,$BDDAT ;BAD DATA FOR TYPEOUT
9294 041762 042737 137777 001142      BIC      #1CTRE,$BDDAT
9295 041770 023737 001140 001142      CMP      $GDDAT,$BDDAT ;IS "TRE" OK??
9296 041776 001413      BEQ      45$      ;YES!!
9297 042000 062716 000004      ADD      #4,(SP)     ;MOVE SP TO USER'S ERROR CALL
9298 042004 112776 000014 000000      MOVB    #14,2(SP)   ;WRITE ERROR NUMBER

```

9298	042012	162716	000002		SUB	#2,(SP)	;MOVE SP TO RETURN FOR ERROR
9299	042016	004736			JSR	PC,2(SP)+	;REPORT THE ERROR
9300	042020	162716	000010		SUB	#10,(SP)	;RESTORE (SP)
9301	042024	000240			NOP		
9302	042026						
9303					45\$:		
9304							
9305							
9306							
9307							
9308							
9309							
9310							
9311							
9312							
9313	042026	010046					
9314	042030	013700	045204		MOV	RO,-(SP)	;PUSH RO ON STACK
9315	042034	016037	064674	045176	MOV	515\$,RO	;RO = FUNCTION CODE
9316	042042	012600			MOV	FNCDB(RO),500\$;STORE ENTRY
9317					MOV	(SP)+,RO	;POP STACK INTO RO
9318							
9319							
9320	042044	013737	045176	001140			
9321	042052	032737	040000	001342			
9322	042060	001403					
9323	042062	052737	100000	001140			
9324	042070	042737	077777	001140			
9325	042076	013737	001342	001142	50\$:		
9326	042104	042737	077777	001142			
9327	042112	023737	001140	001142			
9328	042120	001413					
9329	042122	062716	000004				
9330	042126	112776	000006	000000			
9331	042134	162716	000002				
9332	042140	004736					
9333	042142	162716	000010				
9334	042146	000240					
9335	042150						
9336							
9337							
9338							
9339	042150	013737	045176	001140			
9340	042156	042737	177776	001140			
9341	042164	013737	001344	001142			
9342	042172	042737	177776	001142			
9343	042200	023737	001140	001142			
9344	042206	001412					
9345	042210	062716	000004				
9346	042214	112776	000254	000000			
9347	042222	162716	000002				
9348	042226	004736					
9349	042230	162716	000010				
9350	042234	005037	001140				
9351							
9352							
9353	042240	013746	045176				

;*****
 ;USING THE FUNCTION CODE TABLE, CHECK FOR THE FOLLOWING ERRORS:
 ;STATUS BITS NOT SET THAT SHOULD BE SET, E.G., ATA AND ILF
 ;STATUS BITS SET THAT SHOULD NOT BE SET, E.G., WCE AND ECH
 ;NOTE THAT SOME ERROR BITS ARE CONDITIONAL ON THE COMMAND AND OTHER
 ;STATUS CONDITIONS, E.G., WRITE LOCK ERROR SHOULD ONLY BE SET IF
 ;WRITE LOCK IS ON AND THE COMMAND IS A WRITE.
 ;GET AND STORE THE ENTRY FROM THE FUNCTION CODE TABLE
 ;ATA IS NOT SET AND SHOULD BE SET.
 ;REPORT AN ERROR IF AN UNEXPECTED ATTENTION OCCURRED OR IF
 ;ATA IS NOT SET AND SHOULD BE SET.
 ;GET EXPECTED ATA STATUS
 ;IS COMPOSITE ERROR SET ??
 ;NO !!
 ;EXPECT AN ATTENTION
 ;STRIP DONT CARES
 ;GET RECEIVED ATA
 ;STRIP DONT CARES
 ;IS ATA OK ??
 ;YES !!
 ;MOVE SP TO USERS ERROR CALL
 ;LOAD ERROR # IN CALL
 ;MOVE SP TO ERROR RETURN
 ;REPORT ERROR
 ;RESTORE SP
 ;YES !!
 ;GET EXPECTED ILF
 ;CLEAR ALL OTHER BITS
 ;GET RECEIVED ILF
 ;CLEAR ALL OTHER BITS
 ;IS ILF OK ??
 ;YES !!
 ;MOVE SP TO USERS ERROR CALL
 ;WRITE ERROR NUMBER IN CALL
 ;MOVE SP TO ERROR RETURN
 ;REPORT ERROR AND RETURN
 ;MOVE SP TO NO ERROR
 ;CLEAR EXPECTED STATUS
 ;REPORT AN ERROR IF WCE IS SET AND SHOULD NOT BE SET
 ;GET WCE STATUS ENABLE

9354	042244	052716	137777		BIS	#1CWCE,(SP)	:SET ALL OTHER BITS
9355	042250	013737	001340	001142	MOV	RMCS21,\$BDDAT	:RECEIVED STATUS
9356	042256	042637	001142		BIC	(SP)+,\$BDDAT	:CLEAR WCE IF ENABLED
9357	042262	001412			BEQ	90\$:BRANCH IF WCE OK
9358	042264	062716	000004		ADD	#4,(SP)	:MOVE SP TO USER'S ERROR CALL
9359	042270	112776	000026	000000	MOVB	#26,2(SP)	:WRITE ERROR NUMBER
9360	042276	162716	000002		SUB	#2,(SP)	:MOVE SP TO ERROR RETURN
9361	042302	004736			JSR	PC,2(SP)+	:REPORT ERROR
9362	042304	162716	000010		SUB	#10,(SP)	:RESTORE ERROR
9363	042310						
9364							
9365							
9366	042310	013746	045176				
9367	042314	052716	157777		MOV	500\$,-(SP)	:GET OPI STATUS ENABLE
9368	042320	013737	001344	001142	BIS	#1COP1,(SP)	:SET ALL OTHER BITS
9369	042326	042637	001142		MOV	RMR11,\$BDDAT	:GET RECEIVED STATUS
9370	042332	001412			BIC	(SP)+,\$BDDAT	:CLEAR OPI IF ENABLED
9371	042334	062716	000004		BEQ	100\$:BRANCH IF OPI OK
9372	042340	112776	000164	000000	ADD	#4,(SP)	:MOVE SP TO USER'S ERROR CALL
9373	042346	162716	000002		MOVB	#164,2(SP)	:WRITE ERROR NUMBER IN CALL
9374	042352	004736			SUB	#2,(SP)	:MOVE SP TO ERROR RETURN
9375	042354	162716	000010		JSR	PC,2(SP)+	:REPORT ERROR
9376	042360				SUB	#10,(SP)	:RESTORE SP
9377							
9378							
9379							
9380	042360	013746	045176				
9381	042364	032737	000100	001342	MOV	500\$,-(SP)	:GET IVC STATUS ENABLE
9382	042372	001402			BIT	#VV,RMSI	:IS VV SET
9383	042374	042716	010000		BEQ	105\$:NO !!
9384	042400	052716	167777		BIC	#IVC,(SP)	:YES - IVC SHOULD BE 0
9385	042404	013737	001372	001142	BIS	#1CIVC,(SP)	:SET ALL OTHER BITS
9386	042412	042637	001142		MOV	RMR21,\$BDDAT	:GET RECEIVED STATUS
9387	042416	001412			BIC	(SP)+,\$BDDAT	:CLEAR IVC IF ENABLED
9388	042420	062716	000004		BEQ	110\$:BRANCH IF IVC OK
9389	042424	112776	000165	000000	ADD	#4,(SP)	:MOVE SP TO USER'S ERROR CALL
9390	042432	162716	000002		MOVB	#165,2(SP)	:WRITE ERROR NUMBER IN CALL
9391	042436	004736			SUB	#2,(SP)	:MOVE SP TO ERROR RETURN
9392	042440	162716	000010		JSR	PC,2(SP)+	:REPORT ERROR
9393	042444				SUB	#10,(SP)	:RESTORE SP TO NO ERROR
9394							
9395							
9396							
9397							
9398							
9399							
9400							
9401							
9402							
9403							
9404							
9405	042444	012746	177777				
9406	042450	032737	004000	045176	MOV	#-1,-(SP)	:ASSUME WRITE ERRORS ENABLED
9407	042456	001404			BIT	#WLE,500\$:ARE WRITE ERRORS ENABLED ??
9408	042460	032737	004000	001342	BEQ	115\$:NO !!
9409	042466	001002			BIT	#WAL,RMSI	:IS THE DRIVE WRITE PROTECTED ??
					BNE	120\$:YES !!

9410	042470	042716	004000		115\$:	BIC	#WLE (SP)	;RESET WLE ENABLE
9411	042474	013737	001344	001142	120\$:	MOV	RMR11,\$BDDAT	;GET RECEIVED STATUS
9412	042502	042637	001142			BIC	(SP)+,\$BDDAT	;CLEAR WLE IF ENABLED
9413	042506	001412				BEQ	125\$;BRANCH IF WLE OK
9414	042510	062716	000004			ADD	#4 (SP)	;MOVE SP TO USERS ERROR CALL
9415	042514	112776	000023	000000		MOVB	#23,2(SP)	;WRITE ERROR NUMBER IN CALL
9416	042522	162716	000002			SUB	#2,(SP)	;MOVE SP TO ERROR RETURN
9417	042526	004736				JSR	PC,2(SP)+	;REPORT ERROR AND RETURN
9418	042530	162716	000010			SUB	#10,(SP)	;RESTORE SP TO NO ERROR
9419	042534				125\$:			
9420								
9421								
9422	042534	012746	177777					;REPORT ERROR IF WCF IS SET AND WRITE ERRORS ARE NOT ENABLED
9423	042540	032737	004000	045176		MOV	#-1, -(SP)	;ASSUME WRITE ERRORS ENABLED
9424	042546	001002				BIT	#WLE,500\$;ARE WRITE ERRORS ENABLED ??
9425	042550	042716	000040			BNE	130\$;YES !!
9426	042554	013737	001344	001142	130\$:	BIC	#WCF (SP)	;DISABLE WCF ERROR
9427	042562	042637	001142			MOV	RMR11,\$BDDAT	;GET RECEIVED STATUS
9428	042566	001412				BIC	(SP)+,\$BDDAT	;RESET WCF IF ENABLED
9429	042570	062716	000004			BEQ	135\$;BRANCH IF WCF OK
9430	042574	112776	000025	000000		ADD	#4 (SP)	;MOVE SP TO USERS ERROR CALL
9431	042602	162716	000002			MOVB	#25,2(SP)	;WRITE ERROR NUMBER IN CALL
9432	042606	004736				SUB	#2,(SP)	;MOVE SP TO ERROR RETURN
9433	042610	162716	000010			JSR	PC,2(SP)+	;REPORT ERROR
9434	042614					SUB	#10,(SP)	;RESTORE SP TO NO ERROR
9435					135\$:			
9436								
9437	042614	012746	177777					;REPORT ERROR IF DPE IS SET AND WRITE ERRORS ARE NOT ENABLED
9438	042620	032737	004000	045176		MOV	#-1, -(SP)	;ASSUME WRITE ERRORS ARE ENABLED
9439	042626	001002				BIT	#WLE,500\$;ARE WRITE ERRORS ENABLED ??
9440	042630	042716	000010			BNE	140\$;YES !!
9441	042634	013737	001372	001142	140\$:	BIC	#DPE (SP)	;RESET DPE ENABLE
9442	042642	042637	001142			MOV	RMR21,\$BDDAT	;GET RECEIVED STATUS
9443	042646	001412				BIC	(SP)+,\$BDDAT	;RESET DPE IF ENABLED
9444	042650	062716	000004			BEQ	145\$;BRANCH IF DPE OK
9445	042654	112776	000040	000000		ADD	#4 (SP)	;MOVE SP TO USERS ERROR CALL
9446	042662	162716	000002			MOVB	#40,2(SP)	;WRITE ERROR NUMBER IN CALL
9447	042666	004736				SUB	#2,(SP)	;MOVE SP TO ERROR RETURN
9448	042670	162716	000010			JSR	PC,2(SP)+	;REPORT ERROR
9449	042674					SUB	#10,(SP)	;RESTORE SP TO NO ERROR
9450					145\$:			
9451								
9452	042674	012746	177777					;REPORT AN ERROR IF UPE IS SET AND WRITE ERRORS ARE NOT ENABLED
9453	042700	032737	004000	045176		MOV	#-1, -(SP)	;ASSUME WRITE ERRORS ARE ENABLED
9454	042706	001002				BIT	#WLE,500\$;ARE WRITE ERRORS ENABLED ??
9455	042710	042716	020000			BNE	150\$;YES !!
9456	042714	013737	001340	001142	150\$:	BIC	#UPE (SP)	;DISABLE UPE ERROR
9457	042722	042637	001142			MOV	RMC521,\$BDDAT	;GET RECEIVED STATUS
9458	042726	001412				BIC	(SP)+,\$BDDAT	;RESET UPE IF ENABLED
9459	042730	062716	000004			BEQ	155\$;BRANCH IF UPE OK
9460	042734	112776	000024	000000		ADD	#4 (SP)	;MOVE SP TO USERS ERROR CALL
9461	042742	162716	000002			MOVB	#24,2(SP)	;WRITE ERROR NUMBER IN CALL
9462	042746	004736				SUB	#2,(SP)	;MOVE SP TO ERROR RETURN
9463	042750	162716	000010			JSR	PC,2(SP)+	;REPORT ERROR AND RETURN
9464	042754					SUB	#10,(SP)	;MOVE SP TO NO ERROR
9465					155\$:			


```

9466      042754 013746 045176      ;REPORT AN ERROR IF IAE IS SET AND IS NOT ENABLED
9467      042760 052716 175777      MOV 500$,-(SP)      ;GET IAE ENABLE
9468      042764 013737 001344      BIS #CIAE,(SP)      ;SET ALL OTHER BITS
9469      042772 042637 001142      MOV RMER1,$BDDAT      ;GET RECEIVED STATUS
9470      042776 001412 000004      BIC (SP)+,$BDDAT      ;CLEAR IAE IF ENABLED
9471      043000 062716 000004      BEQ 160$      ;BRANCH IF IAE IS OK
9472      043004 112776 000166      ADD #4,(SP)      ;MOVE SP TO USERS ERROR CALL
9473      043012 162716 000002      MOVB #166,2(SP)      ;WRITE ERROR NUMBER
9474      043016 004736 000010      SUB #2,(SP)      ;MOVE SP TO ERROR RETURN
9475      043020 162716 000010      JSR PC,2(SP)+      ;REPORT ERROR AND RETURN
9476      043024      SUB #10,(SP)      ;MOVE SP TO NO ERROR
9477      160$:
9478
9479      ;BIT 09 (AOE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
9480      ;ALL READ/WRITE ERRORS, I.E.,
9481      RMCS1 - TRE
9482      RMCS2 - DLT,NEM,MXF
9483      RMD5 - LBT
9484      RMER1 - AOE
9485      ;NOTE:
9486      ;LBT IS NOT CHECKED BECAUSE IT ONLY RESETS WHEN THE DESIRED
9487      ;CYLINDER REGISTER IS WRITTEN
9488      ;NOTE:
9489      ;AOE CANNOT BE SET IF LBT IS NOT ALSO SET
9490      ;NOTE:
9491      ;TRE IS CHECKED AS A FUNCTION OF OTHER ERROR CONDITONS ABOVE
9492
9493
9494      ;REPORT AN ERROR IF DLT IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
9495      043024 012746 177777      MOV #-1,-(SP)      ;ASSUME ERRORS ARE ENABLED
9496      043030 032737 001000      BIT #AOE,500$      ;ARE ERRORS ENABLED ??
9497      043036 001002 045176      BNE 165$      ;YES !!
9498      043040 042716 100000      BIC #DLT,(SP)      ;RESET DLT ENABLE
9499      043044 013737 001340      MOV RMCS2I,$BDDAT      ;GET RECEIVED STATUS
9500      043052 042637 001142      BIC (SP)+,$BDDAT      ;CLEAR DLT IF ENABLED
9501      043056 001412 000004      BEQ 170$      ;BRANCH IF DLT IS OK
9502      043060 062716 000004      ADD #4,(SP)      ;MOVE SP TO USERS ERROR CALL
9503      043064 112776 000032      MOVB #32,2(SP)      ;WRITE ERROR NUMBER IN CALL
9504      043072 162716 000002      SUB #2,(SP)      ;MOVE SP TO ERROR RETURN
9505      043076 004736 000010      JSR PC,2(SP)+      ;REPORT ERROR AND RETURN
9506      043100 162716 000010      SUB #10,(SP)      ;MOVE SP TO NO ERROR
9507      170$:
9508
9509      ;REPORT ERROR IF NEM IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
9510      043104 012746 177777      MOV #-1,-(SP)      ;ASSUME ERRORS ARE ENABLED
9511      043110 032737 001000      BIT #AOE,500$      ;ARE ERRORS ENABLED ??
9512      043116 001002 045176      BNE 175$      ;YES !!
9513      043120 042716 004000      BIC #NEM,(SP)      ;DISABLE NEM
9514      043124 013737 001340      MOV RMCS2I,$BDDAT      ;GET RECEIVED STATUS
9515      043132 042637 001142      BIC (SP)+,$BDDAT      ;CLEAR NEM IF ENABLED
9516      043136 001412 000004      BEQ 180$      ;BRANCH IF NEM IS OK
9517      043140 062716 000004      ADD #4,(SP)      ;MOVE SP TO USERS ERROR CALL
9518      043144 112776 000167      MOVB #167,2(SP)      ;WRITE ERROR NUMBER IN CALL
9519      043152 162716 000002      SUB #2,(SP)      ;MOVE SP TO ERROR RETURN
9520      043156 004736 000010      JSR PC,2(SP)+      ;REPORT ERROR AND RETURN
9521      043160 162716 000010      SUB #10,(SP)      ;MOVE SP TO NO ERROR

```

```
9522 043164 180$:
9523
9524 ;REPORT ERROR IF MXF IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
9525 043164 012746 177777 MOV #1, -(SP) ;ASSUME ERRORS ARE ENABLED
9526 043170 032737 001000 045176 BIT #AOE, 500$ ;ARE DATA ERRORS ENABLED ??
9527 043176 001002 BNE 185$ ;YES !!
9528 043200 042716 001000 BIC #MXF, (SP) ;DISABLE MXF ERROR
9529 043204 013737 001340 001142 185$: MOV RMCS2I, $BDDAT ;GET RECEIVED STATUS
9530 043212 042637 001142 BIC (SP)+, $BDDAT ;CLEAR MXF IF ENABLED
9531 043216 001412 BEQ 190$ ;BRANCH IF MXF IS OK
9532 043220 062716 000004 ADD #4, (SP) ;MOVE SP TO USERS ERROR CALL
9533 043224 112776 000033 000000 MOVB #33, 2(SP) ;WRITE ERROR NUMBER IN CALL
9534 043232 162716 000002 SUB #2, (SP) ;MOVE SP TO ERROR RETURN
9535 043236 004736 JSR PC, 2(SP)+ ;REPORT ERROR AND RETURN
9536 043240 162716 000010 SUB #10, (SP) ;MOVE SP TO NO ERROR
9537 043244 190$:
9538
9539 ;REPORT ERROR IF AOE IS SET AND DATA ERRORS ARE NOT ENABLED
9540 043244 012746 177777 MOV #1, -(SP) ;ASSUME DATA ERRORS ARE ENABLED
9541 043250 032737 001000 045176 BIT #AOE, 500$ ;ARE DATA ERRORS ENABLED ??
9542 043256 001404 BEQ 191$ ;NO !!
9543 043260 032737 002000 001342 BIT #LBT, RMOSI ;IS LBT ALSO SET ??
9544 043266 001002 BNE 195$ ;YES !!
9545 043270 042716 001000 191$: BIC #AOE, (SP) ;DISABLE AOE
9546 043274 013737 001344 001142 195$: MOV RMER1I, $BDDAT ;GET RECEIVED STATUS
9547 043302 042637 001142 BIC (SP)+, $BDDAT ;CLEAR AOE IF ENABLED
9548 043306 001412 BEQ 200$ ;BRANCH IF AOE IS OK
9549 043310 062716 000004 ADD #4, (SP) ;MOVE SP TO USERS ERROR CALL
9550 043314 112776 000020 000000 MOVB #20, 2(SP) ;WRITE ERROR NUMBER
9551 043322 162716 000002 SUB #2, (SP) ;MOVE SP TO ERROR RETURN
9552 043326 004736 JSR PC, 2(SP)+ ;REPORT ERROR AND RETURN
9553 043330 162716 000010 SUB #10, (SP) ;MOVE SP TO NO ERROR
9554 043334 200$:
9555
9556 ;BIT 07 (HCE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
9557 ;HEADER ERRORS, I.E.,
9558 ; RMER1 - HCRC, HCE, FER
9559 ; RMER2 - BSE
9560
9561 ;RESET THE ENABLING BIT (HCE) IF HEADER COMPARE INHIBIT IS SET
9562 043334 032737 002000 001362 BIT #HCI, RMOFI ;IS HCI SET ??
9563 043342 001403 BEQ 201$ ;NO !!
9564 043344 042737 000200 045176 BIC #HCE, 500$ ;YES - DISABLE ALL HEADER ERRORS
9565 043352 201$:
9566
9567 ;REPORT AN ERROR IF HCRC IS SET AND HEADER ERRORS ARE NOT ENABLED
9568 043352 012746 177777 MOV #1, -(SP) ;ASSUME ERRORS ENABLED
9569 043356 032737 000200 045176 BIT #HCE, 500$ ;ARE HEADER ERRORS ENABLED ??
9570 043364 001002 BNE 205$ ;YES !!
9571 043366 042716 000400 BIC #HCRC, (SP) ;DISABLE HCRC
9572 043372 013737 001344 001142 205$: MOV RMER1I, $BDDAT ;GET RECEIVED STATUS
9573 043400 042637 001142 BIC (SP)+, $BDDAT ;RESET HCRC IF ENABLED
9574 043404 001412 BEQ 210$ ;BRANCH IF HCRC IS OK
9575 043406 062716 000004 ADD #4, (SP) ;MOVE SP TO USERS ERROR CALL
9576 043412 112776 000035 000000 MOVB #35, 2(SP) ;WRITE ERROR NUMBER IN CALL
9577 043420 162716 000002 SUB #2, (SP) ;MOVE SP TO ERROR RETURN
```

```
9578 043424 004736          JSR    PC,2(SP)+      ;REPORT ERROR AND RETURN
9579 043426 162716 000010    SUB     #10,(SP)      ;MOVE SP TO NO ERROR
9580 043432          210$:
9581          ;REPORT ERROR IF HCE IS SET AND HEADER ERRORS ARE NOT ENABLED
9582          MOV     #-1,-(SP)      ;ASSUME ERRORS ENABLED
9583 043432 012746 177777      BIT     #HCE,500$      ;ARE ERRORS ENABLED ??
9584 043436 032737 000200 045176 BNE     215$      ;YES !!
9585 043444 001002          BNE     215$      ;DISABLE HCE
9586 043446 042716 000200      BIC     #HCE,(SP)      ;GET RECEIVED STATUS
9587 043452 013737 001344 001142 215$: MOV     RMER1I,$BDDAT ;CLEAR HCE IF ENABLED
9588 043460 042637 001142      BIC     (SP)+,$BDDAT ;BRANCH IF HCE IS OK
9589 043464 001412          BEQ     220$      ;MOVE SP TO USERS ERROR CALL
9590 043466 062716 000004      ADD     #4,(SP)      ;WRITE ERROR NUMBER IN CALL
9591 043472 112776 000036 000000      MOVB   #36,2(SP)      ;MOVE SP TO ERROR RETURN
9592 043500 162716 000002      SUB     #2,(SP)      ;REPORT ERROR AND RETURN
9593 043504 004736          JSR     PC,2(SP)+      ;MOVE SP TO NO ERROR
9594 043506 162716 000010    SUB     #10,(SP)
9595 043512          220$:
9596          ;REPORT ERROR IF FER IS SET AND HEADER ERRORS ARE NOT ENABLED
9597          MOV     #-1,-(SP)      ;ASSUME FER IS ENABLED
9598 043512 012746 177777      BIT     #HCE,500$      ;ARE HEADER ERRORS ENABLED ??
9599 043516 032737 000200 045176 BNE     225$      ;YES !!
9600 043524 001002          BNE     225$      ;DISABLE FER
9601 043526 042716 000020      BIC     #FER,(SP)      ;GET RECEIVED STATUS
9602 043532 013737 001344 001142 225$: MOV     RMER1I,$BDDAT ;RESET FER IF ENABLED
9603 043540 042637 001142      BIC     (SP)+,$BDDAT ;BRANCH IF FER OK
9604 043544 001412          BEQ     230$      ;MOVE SP TO USERS ERROR CALL
9605 043546 062716 000004      ADD     #4,(SP)      ;WRITE ERROR NUMBER IN CALL
9606 043552 112776 000037 000000      MOVB   #37,2(SP)      ;MOVE SP TO ERROR RETURN
9607 043560 162716 000002      SUB     #2,(SP)      ;REPORT ERROR AND RETURN
9608 043564 004736          JSR     PC,2(SP)+      ;MOVE SP TO NO ERROR
9609 043566 162716 000010    SUB     #10,(SP)
9610 043572          230$:
9611          ;REPORT ERROR IF BSE IS SET AND HEADER ERRORS ARE NOT ENABLED
9612          MOV     #-1,-(SP)      ;ASSUME ERRORS ENABLED
9613 043572 012746 177777      BIT     #HCE,500$      ;ARE THEY ENABLED ??
9614 043576 032737 000200 045176 BNE     235$      ;YES !!
9615 043604 001002          BNE     235$      ;DISABLE BSE
9616 043606 042716 100000      BIC     #BSE,(SP)      ;GET RECEIVED STATUS
9617 043612 013737 001372 001142 235$: MOV     RMER2I,$BDDAT ;CLEAR BSE IF ENABLED
9618 043620 042637 001142      BIC     (SP)+,$BDDAT ;BRANCH IF BSE OK
9619 043624 001412          BEQ     240$      ;MOVE SP TO USERS ERROR CALL
9620 043626 062716 000004      ADD     #4,(SP)      ;WRITE ERROR NUMBER
9621 043632 112776 000354 000000      MOVB   #354,2(SP)      ;MOVE SP TO ERROR RETURN
9622 043640 162716 000002      SUB     #2,(SP)      ;REPORT ERROR AND RETURN
9623 043644 004736          JSR     PC,2(SP)+      ;MOVE SP TO NO ERROR
9624 043646 162716 000010    SUB     #10,(SP)
9625 043652          240$:
9626          ;BIT 06 OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR DATA
9627          ;FIELD ERRORS, I.E.
9628          ;RMCS2 - MOPE
9629          ;RMER1 - DCK,ECH
9630          ;NOTE:
9631          ;ECH CANNOT SET UNLESS IT IS ENABLED AND ECI IS RESET AND
9632          ;DCK IS SET.
```

```

9634
9635 ;REPORT ERROR IF MOPE IS SET AND IS NOT ENABLED
9636 043652 012746 177777      MOV    #-1,-(SP)      ASSUME ENABLED
9637 043656 032737 000100 045176 BIT    #ECH,500$      ARE DATA FIELD ERRORS ENABLED ??
9638 043659 001002          BNE    245$      YES !!
9639 043666 042716 000400      BIC    #MOPE,(SP)      DISABLE MOPE
9640 043672 013737 001340 001142 245$: MOV    #MCS21,$BDDAT      GET RECEIVED STATUS
9641 043700 042637 001142      BIC    (SP)+,$BDDAT      CLEAR MOPE IF ENABLED
9642 043704 001412          BEQ    250$      BRANCH IF MOPE OK
9643 043706 062716 000004      ADD    #4,(SP)      MOVE SP TO USERS ERROR CALL
9644 043712 112776 000027 000000      MOVB  #27,2(SP)      WRITE ERROR NUMBER IN CALL
9645 043720 162716 000002      SUB    #2,(SP)      MOVE SP TO ERROR RETURN
9646 043724 004736          JSR    PC,2(SP)+      REPORT ERROR AND RETURN
9647 043726 162716 000010      SUB    #10,(SP)      MOVE SP TO NO ERROR
9648 043732          250$:
9649
9650 ;REPORT ERROR IF DCK IS SET AND DATA FIELD ERRORS ARE NOT ENABLED
9651 043732 012746 177777      MOV    #-1,-(SP)      ASSUME ENABLED
9652 043736 032737 000100 045176 BIT    #ECH,500$      ARE THEY ENABLED ??
9653 043744 001002          BNE    255$      YES !!
9654 043746 042716 100000      BIC    #DCK,(SP)      DISABLE DCK
9655 043752 013737 001344 001142 255$: MOV    #MER11,$BDDAT      GET RECEIVED STATUS
9656 043760 042637 001142      BIC    (SP)+,$BDDAT      CLEAR DCK IF ENABLED
9657 043764 001412          BEQ    260$      BRANCH IF DCK IS OK
9658 043766 062716 000004      ADD    #4,(SP)      MOVE SP TO USERS ERROR CALL
9659 043772 112776 000030 000000      MOVB  #30,2(SP)      WRITE ERROR NUMBER IN CALL
9660 044000 162716 000002      SUB    #2,(SP)      MOVE SP TO ERROR RETURN
9661 044004 004736          JSR    PC,2(SP)+      REPORT ERROR AND RETURN
9662 044006 162716 000010      SUB    #10,(SP)      MOVE SP TO NO ERROR
9663 044012          260$:
9664
9665 ;REPORT ERROR IF ECH IS SET AND
9666 ;DATA FIELD ERRORS ARE NOT ENABLED, OR
9667 ;ECI IS SET, OR
9668 ;DCK IS NOT SET
9669 044012 012746 177777      MOV    #-1,-(SP)      ASSUME ENABLED
9670 044016 032737 000100 045176 BIT    #ECH,500$      ARE ERRORS ENABLED ??
9671 044024 001410          BEQ    265$      NO !!
9672 044026 032737 004000 001362 BIT    #ECI,RMOFI      IS ECI SET ??
9673 044034 001004          BNE    265$      YES !!
9674 044036 032737 100000 001344 BIT    #DCK,RMER11      IS DCK ALSO SET ??
9675 044044 001002          BNE    270$      YES !!
9676 044046 042716 000100 265$: BIC    #ECH,(SP)      DISABLE ECH
9677 044052 013737 001344 270$: MOV    #MER11,$BDDAT      GET RECEIVED STATUS
9678 044060 042637 001142      BIC    (SP)+,$BDDAT      CLEAR ECH IF ENABLED
9679 044064 001412          BEQ    275$      BRANCH IF ECH IS OK
9680 044066 062716 000004      ADD    #4,(SP)      MOVE SP TO USERS ERROR CALL
9681 044072 112776 000031 001000      MOVB  #31,2(SP)      WRITE ERROR NUMBER IN CALL
9682 044100 162716 000002      SUB    #2,(SP)      MOVE SP TO ERROR RETURN
9683 044104 004736          JSR    PC,2(SP)+      REPORT ERROR AND RETURN
9684 044106 162716 000010      SUB    #10,(SP)      MOVE SP TO NO ERROR
9685 044112          275$:
9686
9687 ;*****
9688 ;PERFORM THE REMAINING ERROR CHECKS ONLY FOR DATA TRANSFER COMMANDS
9689

```

N15

CZRMDB0 RM03/2 FCTNL TST 2
CZRMDB.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 195
SECONDARY ERROR CHECK SUBROUTINE

SEQ 0195

9690	044112	022737	000030	045204	CMP	SEARCH,515\$;WAS DATA TRANSFERRED??
9691	044120	103402			BLO	280\$;YES!!
9692	044122	000137	045150		JMP	355\$	
9693							

CZRMOB0 RMO3/2 FCTNL TST 2
CZRMOB.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 196
SECONDARY ERROR CHECK SUBROUTINE

SEQ 0196

```

9694      ;THE FOLLOWING STATUS CHECKS ARE FOR DATA TRANSFER COMMANDS ONLY
9695      ;REPORT ERROR IF RMC NOT ZERO AND TRE IS ZERO
9696      280$: MOV RMC1,$B00AT      ;WORD COUNT ZERO??
9697      BEQ 281$      ;YES
9698      BIT TRE,RMC$11      ;TRANSFER ERROR DETECTED??
9699      BNE 285$      ;YES!!
9700      ADD #4,(SP)
9701      MOV #15,2(SP)      ;ERROR NUMBER
9702      CLR $C00AT      ;GOOD DATA FOR TYPEOUT
9703      SUB #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
9704      JSR PC,2(SP)+      ;REPORT WORD COUNT NOT ZERO
9705      SUB #10,(SP)      ;RESTORE (SP)
9706      NOP
9707      ;REPORT ERROR IF RMB IS NOT CORRECT
9708      285$: MOV RMC1,$C00AT      ;NUMBER OF WORDS TRANSFERRED
9709      SUB RMC0,$C00AT
9710      ASL $C00AT
9711      ADD RMB0,$C00AT      ;EXPECTED BUS ADDRESS
9712      BIT RBA1,RMC$21      ;WAS BAI SET ??
9713      BEQ 290$      ;NO !!
9714      MOV RMB0,$C00AT      ;ADDRESS SHOULD NOT HAVE CHANGED
9715      290$: CMP $C00AT,RBA1      ;BUS ADDRESS OK??
9716      BEQ 295$      ;YES!!
9717      MOV RBA1,$B00AT      ;BAD DATA FOR TYPEOUT
9718      ADD #4,(SP)
9719      MOV #16,2(SP)      ;ERROR NUMBER
9720      SUB #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
9721      JSR PC,2(SP)+      ;REPORT UNEXPECTED ADDRESS
9722      SUB #10,(SP)      ;RESTORE (SP)
9723      NOP
9724      ;COMPUTE NUMBER OF SECTORS TRANSFERRED
9725      295$: CLR -(SP)      ;NUMBER OF SECTORS TRANSFERRED
9726      MOV RMC1,-(SP)      ;NUMBER OF WORDS TRANSFERRED
9727      SUB RMC0,(SP)
9728      MOV #256,-(SP)      ;NUMBER OF WORDS PER SECTOR
9729      BIT #111,RMC$10      ;HEADER & DATA COMMAND ??
9730      BEQ 300$      ;NO !!
9731      MOV #258,(SP)      ;YES - CHANGE WORDS PER SECTOR
9732      300$: INC 4(SP)      ;INCREMENT SECTOR COUNT
9733      SUB (SP),2(SP)      ;SUBTRACT ONE SECTOR'S WORTH
9734      BGT 300$      ;CONTINUE IF NOT DONE
9735      CMP (SP)+,(SP)+      ;STRIP 2 FROM STACK
9736      ;COMPUTE EXPECTED SECTOR TRACK AND CYLINDER ADDRESS FROM NUMBER OF SECTORS
9737      MOV RMD0,510$      ;STORE ORIGINAL SECTOR
9738      MOV RMD0,505$      ;STORE ORIGINAL TRACK
9739      MOV RMC0,500$      ;STORE ORIGINAL CYLINDER
9740      BIC #1C37,510$
9741      SWAB 505$
9742      BIC #1C7,505$
9743      ADD (SP)+,510$
9744
9745      305$: CMP 510$,#32.      ;SECTOR OVEFLOWED??
9746      BLO 315$      ;NO!!
9747      CMP 510$,#(5*32.)      ;CYLINDERS WORTH??
9748      BLO 310$      ;NO!!
9749      INC 500$      ;YES INCREMENT CYLINDER

```

```

9750 044450 162737 000240 045202 SUB #(<5*32.),510$ ;ADJUST SECTOR
9751 044456 000762 BR 305$ ;TRY AGAIN
9752 044460 005237 045200 310$: INC 505$ ;INCREMENT TRACK
9753 044464 162737 000040 045202 SUB #32.,510$ ;ADJUST SECTOR
9754 044472 000754 BR 305$ ;TRY AGAIN
9755
9756 044474 023727 045200 000005 315$: CMP 505$,#5 ;TRACK OVERFLOWED??
9757 044502 103406 BLO 320$ ;NO!!
9758 044504 005237 045176 INC 500$ ;INCREMENT CYLINDER
9759 044510 162737 000005 045200 SUB #5,505$ ;ADJUST TRACK
9760 044516 000766 BR 315$ ;TRY AGAIN
9761 ;REPORT ERROR IF "LBT" IS NOT CORRECT
9762 044520 005037 001140 320$: CLR $GDDAT ;SET GOOD DATA FOR LBT=0
9763 044524 022737 001467 045176 CMP #823.,500$ ;SHOULD LBT BE SET??
9764 044532 101007 BHI 325$ ;NO!!
9765 044534 032737 002000 001344 BIT #IAE,RMER11 ;WAS IAE SET ??
9766 044542 001003 BNE 325$ ;YES - LBT SHOULD NOT BE SET
9767 044544 012737 002000 001140 MOV #LBT,$GDDAT ;SET GOOD DATA FOR LBT=1
9768 044552 013737 001342 001142 325$: MOV RMDS1,$BDDAT ;BAD DATA FOR TYPEOUT
9769 044560 042737 175777 001142 BIC #CLBT,$BDDAT
9770 044566 023737 001140 001142 CMP $GDDAT,$BDDAT ;IS LBT CORRECT??
9771 044574 001413 BEQ 330$ ;YES!!
9772 044576 062716 000004 ADD #4,(SP)
9773 044602 112776 000017 000000 MOVB #17,2(SP) ;ERROR NUMBER
9774 044610 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
9775 044614 004736 JSR PC,2(SP)+ ;REPORT LBT IS WRONG
9776 044616 162716 000010 SUB #10,(SP) ;RESTORE (SP)
9777 044622 000240 NOP
9778 ;REPORT ERROR IF "AOE" IS INCORRECT
9779 044624 005037 001140 330$: CLR $GDDAT ;SET FOR AOE=0
9780 044630 032737 002000 001344 BIT #IAE,RMER11 ;WAS "IAE" DETECTED??
9781 044636 001031 BNE 340$ ;YES-"IAE" SHOULD BE ZERO
9782 044640 022737 001467 045176 CMP #823.,500$ ;SHOULD AOE BE SET??
9783 044646 101025 BHI 340$ ;NO!!
9784 044650 005737 045200 TST 505$ ;MAYBE
9785 044654 001012 BNE 335$ ;YES
9786 044656 005737 045202 TST 510$
9787 044662 001007 BNE 335$ ;YES !!
9788 044664 032737 000010 045204 BIT #F2,515$ ;WAS THIS READ OR WRITE CHECK ??
9789 044672 001413 BEQ 340$ ;NO !!
9790 044674 005737 001332 TST RMWCI ;WAS ALL DATA TRANSFERRED ??
9791 044700 001410 BEQ 340$ ;YES !!
9792 044702 012737 001000 001140 335$: MOV #AOE,$GDDAT ;SET FOR AOE=1
9793 044710 005037 045200 CLR 505$ ;CLEAR EXPECTED TRACK
9794 044714 012737 000001 045202 MOV #1,510$ ;EXPECT SECTOR=1
9795 044722 013737 001344 001142 340$: MOV RMER11,$BDDAT ;BAD DATA FOR TYPEOUT
9796 044730 042737 176777 001142 BIC #CAOE,$BDDAT
9797 044736 023737 001140 001142 CMP $GDDAT,$BDDAT ;IS AOE CORRECT??
9798 044744 001413 BEQ 345$ ;YES!!
9799 044746 062716 000004 ADD #4,(SP)
9800 044752 112776 000020 000000 MOVB #20,2(SP) ;ERROR NUMBER
9801 044760 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
9802 044764 004736 JSR PC,2(SP)+ ;REPORT AOE IS WRONG
9803 044766 162716 000010 SUB #10,(SP) ;RESTORE (SP)
9804 044772 000240 NOP
5805 ;REPORT ERROR IF RMDA IS NOT CORRECT

```

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 198
SECONDARY ERROR CHECK SUBROUTINE

SEQ 0198

9806	044774	032737	002000	001344	345\$:	BIT	#IAE,RMER11	;WAS THERE AN IAE ERROR ??
9807	045002	001062				BNE	355\$;YES - DONT CHECK RMDA,RMDC
9808	045004	013737	045200	001140		MOV	505\$,SGDDAT	;SETUP EXPECTED DISK ADDRESS
9809	045012	000337	001140			SWAB	SGDDAT	
9810	045016	113737	045202	001140		MOVB	510\$,SGDDAT	
9811	045024	013737	001336	001142		MOV	RMDAT,\$BDDAT	;SETUP RECEIVED DISK ADDRESS
9812	045032	023737	001140	001142		CMP	SGDDAT,\$BDDAT	;COMPARE EXPECTED & RECEIVED
9813	045040	001413				BEQ	350\$;BRANCH IF EQUAL
9814	045042	062716	000004			ADD	#4,(SP)	
9815	045046	112776	000021	000000		MOVB	#21,2(SP)	;ERROR NUMBER
9816	045054	162716	000002			SUB	#2,(SP)	;MOVE SP TO RETURN FOR ERROR
9817	045060	004736				JSR	PC,2(SP)+	;REPORT BAD DISK ADDRESS
9818	045062	162716	000010			SUB	#10,(SP)	;RESTORE (SP)
9819	045066	000240				NOP		
9820					:REPORT	ERROR IF RMDC IS INCORRECT		
9821	045070	013737	045176	001140	350\$:	MOV	500\$,SGDDAT	;SETUP EXPECTED CYLINDER
9822	045076	042737	176000	001140		BIC	#1C1777,SGDDAT	
9823	045104	013737	001364	001142		MOV	RMDCI,\$BDDAT	;SETUP RECEIVED CYLINDER
9824	045112	023737	001140	001142		CMP	SGDDAT,\$BDDAT	;COMPARE CYLINDERS
9825	045120	001413				BEQ	355\$;BRANCH IF EQUAL
9826	045122	062716	000004			ADD	#4,(SP)	
9827	045126	112776	000022	000000		MOVB	#22,2(SP)	;ERROR NUMBER
9828	045134	162716	000002			SUB	#2,(SP)	;MOVE SP TO RETURN FOR ERROR
9829	045140	004736				JSR	PC,2(SP)+	;REPORT BAD CYLINDER
9830	045142	162716	000010			SUB	#10,(SP)	;RESTORE (SP)
9831	045146	000240				NOP		

E16

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 199
SECONDARY ERROR CHECK SUBROUTINE

SEQ 0199

9832	045150	062716	000004	355\$:	ADD	#4,(SP)	;MOVE (SP) TO ERROR CALL
9833	045154	105776	000000		TSTB	2(SP)	;WAS ERROR FOUND??
9834	0451F3	001403			BEQ	360\$	
9835	045162	062716	000004		ADD	#4,(SP)	;MOVE (SP) TO ERROR RETURN
9836	045166	000402			BR	365\$	
9837	045170	162716	000004	360\$:	SUB	#4,(SP)	;MOVE (SP) TO NO ERROR RETURN
9838	045174	000207		365\$:	RTS	PC	
9839							
9840	045176	000000		500\$:	.WORD	0	;CYLINDER
9841	045200	000000		505\$:	.WORD	0	;TRACK
9842	045202	000000		510\$:	.WORD	0	;SECTOR
9843	045204	000000		515\$:	.WORD	0	;FUNCTION CODE
9844							
9845							

```

9846 .SBTTL DEVICE SELECT SUBROUTINE
9847
9848 ; THIS SUBROUTINE SELECTS THE DEVICE, GETTING THE DEVICE NUMBER FROM THE
9849 ; TEST QUEUE.
9850
9851 ;CALL:
9852 ;(1) JSR PC,DEVSEL
9853 ;(2) BR ??
9854 ;(3) NOP
9855 ;(4) ERROR
9856
9857 045206 DEVSEL:
9858
9859 ;CLEAR USER'S ERROR CALL
9860 045206 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR
9861 045212 105076 000000 CLRB #2(SP) ;CLEAR LOW ORDER BYTE OF CALL
9862 045216 162716 000004 SUB #4,(SP) ;MOVE SP BACK
9863 ;SAVE USER'S INFORMATION AND SETUP REGISTERS
9864 045222 013746 000004 MOV ERRVEC,-(SP) ;PUSH ERRVEC ON STACK
9865 045226 013746 000006 MOV ERRVEC+2,-(SP) ;PUSH ERRVEC+2 ON STACK
9866 045232 010046 MOV RO,-(SP) ;PUSH RO ON STACK
9867 045234 010146 MOV R1,-(SP) ;PUSH R1 ON STACK
9868 045236 012737 045356 000004 MOV #20$,ERRVEC ;SETUP FOR BUS TIMEOUT
9869 045244 012737 000300 000006 MOV #PR6,ERRVEC+2
9870 045252 013700 001276 MOV $BASE,RO ;RO = UNIBUS ADDRESS
9871 045256 013701 001450 MOV !STQUE,R1 ;R1 POINTS TO DEVICE NUMBER
9872
9873 ;SELECT DEVICE AND VERIFY THAT DEVICE IS AVAILABLE
9874 045262 111160 000010 MOVB (R1),RMC52(RO) ;WRITE UNIT SELECT BITS
9875 045266 016037 000000 001176 MOV RMC51(RO),STMP1 ;GET "DVA" STATUS
9876 045274 016037 000010 001174 MOV RMC52(RO),STMP0 ;GET "MED" STATUS
9877 045302 032737 010000 001174 BIT #MED,STMP0 ;IS DEVICE NONEXISTENT??
9878 045310 001407 BEQ 10$ ;NO!!
9879 045312 062766 000004 000010 ADD #4,10(SP) ;MOVE SP TO USERS ERROR CALL
9880 045320 112776 000111 000010 MOVB #111,10(SP) ;WRITE ERROR NUMBER
9881 045326 000422 BR 30$
9882 045330 032737 004000 001176 10$: BIT #DVA,STMP1 ;IS DEVICE AVAILABLE??
9883 045336 001021 BNE 35$ ;YES!!
9884 045340 062766 000004 000010 ADD #4,10(SP)
9885 045346 112776 000112 000010 MOVB #112,10(SP)
9886 045354 000407 BR 30$
9887
9888 045356 20$:
9889
9890 ;HANDLE BUS TIMEOUT
9891 045356 022626 CMP (SP)+,(SP)+ ;ADJUST SP
9892 045360 062766 000004 000010 ADD #4,10(SP)
9893 045366 112776 000113 000010 MOVB #113,10(SP)
9894 045374 162766 000002 000010 30$: SUB #2,10(SP) ;MOVE SP TO RETURN IF ERROR
9895
9896 045402 35$:
9897
9898 ;RESTORE USERS DATA AND RETURN TO ADDRESS ON STACK
9899
9900 045402 012601 MOV (SP)+,R1 ;POP STACK INTO R1
9901 045404 012600 MOV (SP)+,RO ;POP STACK INTO RO

```

G16

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 201
DEVICE SELECT SUBROUTINE

SEQ 0201

9902 045406 012637 000006
9903 045412 012637 000004
9904 045416 000207

MOV (SP)+,ERRVEC+2 ; POP STACK INTO ERRVEC+2
MOV (SP)+,ERRVEC ; POP STACK INTO ERRVEC
RTS PC ; EXIT

```

9905 .SBTTL SEEK STATUS CHECK SUBROUTINE
9906
9907 ;THIS SUBROUTINE VERIFIES THE RESULTS OF SEEK TESTS USING STATUS
9908 ;STORED IN THE GET BUFFER AND TEST PARAMETERS STORED IN THE PUT BUFFER.
9909
9910 ;THE SUBROUTINE RETURNS TO THE CALLING ROUTINE IF AN ERROR IS DETECTED
9911 ;AFTER HAVING LOADED THE APPROPRIATE ERROR NUMBER IN THE "ERROR" TRAP
9912 ;OF THE CALLING ROUTINE. SEEK STATUS IS CHECKED AS FOLLOWS:
9913
9914 CALL:
9915 (1) JSR PC,SEKSTS
9916 BR ??? RETURN HERE IF NO ERROR
9917 NOP RETURN HERE TO REPORT AN ERROR
9918 ERROR ERROR NUMBER DEFINED BY SUB
9919 JSR PC,2(SP)+ GO BACK TO SUB FOR MORE ERROR CHECKS
9920 ??? RETURN HERE IF NO MORE ERRORS
9921
9922
9923 045420 SEKSTS:
9924
9925 ;CLEAR USER'S ERROR CALL
9926 045420 000240 NOP
9927 045422 062716 000004 ADD #4,(SP) ;MOVE (SP) TO ERROR CALL
9928 045426 105076 000000 CLRB 2(SP) ;CLEAR ERROR NUMBER
9929 045432 162716 000004 SUB #4,(SP) ;MOVE (SP) TO NO ERROR RETURN
9930 045436 005037 046656 CLR 300$ ;CLEAR ERROR FLAGS
9931
9932 ;TEST FOR MASSBUS CONTROL BUS PARITY ERROR WHEN WRITING
9933 ;LOCAL REGISTERS, I.E. "PAR" = 1 AND "OPE" = 0
9934 045442 032737 000010 001344 BIT #PAR,RMER1I ;WAS PARITY ERROR DETECTED??
9935 045450 001424 BEQ 1$ ;NO!!
9936 045452 032737 000010 001372 BIT #OPE,RMER2I ;WAS IT DUE TO CONTROL BUS??
9937 045460 001020 BNE 1$ ;NOT SURE!!
9938
9939 ;REPORT REGISTER PARITY ERROR VIA USER'S ERROR CALL
9940 045462 005037 001140 CLR $GDDAT ;EXPECTED STATUS
9941 045466 013737 001344 001142 MOV RMER1I,$BDDAT ;RECEIVED STATUS
9942 045474 062716 000004 ADD #4,(SP) ;MOVE STACK TO USER'S ERROR
9943 045500 112776 000050 000000 MOVB #50,2(SP) ;ERROR #50
9944 045506 162716 00002 SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
9945 045512 004736 JSR PC,2(SP)+
9946 045514 162716 000010 SUB #10,(SP) ;RESTORE STACK
9947 045520 000437 BR 3$ ;IAE SHOULD BE ZERO
9948
9949
9950 ;DETERMINE THE VALUE OF "IAE" STATUS BASED ON TRACK,SECTOR AND
9951 ;CYLINDER ADDRESS USED DURING SEEK OPERATION
9952 045522 012737 002000 001140 1$: MOV #IAE,$GDDAT ;SET UP FOR IAE = 1
9953 045530 052737 040000 046656 BIS #SKI,300$ ;SET SKI ERROR FLAG
9954 045536 023727 001400 001466 CMP RMDCO,#822. ;CYLINDER > 822??
9955 045544 101025 BHI 3$ ;YES!!
9956 045546 042737 040000 046656 BIC #SKI,300$ ;CLEAR SKI ERROR FLAG
9957 045554 123727 001406 000037 CMPB RMDAO,#31. ;SECTOR > 31??
9958 045562 101016 BHI 3$ ;YES!!
9959 045564 123727 001406 000035 CMPB RMDAO,#29. ;SECTOR > 29 ??
9960 045572 101404 BLOS 2$ ;NO!!

```

```

9961 045574 032737 010000 001432      BIT      #FMT16,RMOFO      ;30 SECTOR FORMAT??
9962 045602 001406      BEQ      3$              ;YES!!
9963 045604 123727 001407 000004 2$:      CMPB     RMDAO+1,#4.      ;TRACK >4??
9964 045612 101002      BHI      3$              ;YES!!
9965 045614 005037 001140      CLR      $GDDAT      ;"IAE" SHOULD BE 0
9966
9967      ;COMPARE EXPECTED AND RECEIVED "IAE" STATUS
9968 045620 013737 001344 001142 3$:      MOV      RMR1I,$BDDAT      ;IS IAE OK??
9969 045626 042737 175777 001142      BIC      #ICIAE,$BDDAT
9970 045634 023737 001140 001142      CMP      $GDDAT,$BDDAT
9971 045642 001004      BNE      35$          ;IAE IN ERROR
9972 045644 042737 040000 046656      BIC      #SKI,300$      ;CLEAR SKI FLAG
9973 045652 000413      BR       5$              ;GO CHECK NEXT ERROR
9974 045654
9975      35$:
9976 045654 062716 000004      ;REPORT INCORRECT "IAE" STATUS VIA USER'S ERROR CALL
9977 045660 112776 000051 000000      ADD      #4,(SP)
9978 045666 162716 000002      MOVVB     #51,2(SP)      ;ERROR 51
9979 045672 004736      SUB      #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
9980 045674 162716 000010      JSR      PC,2(SP)+      ;REPORT INCORRECT IAE
9981 045700 000240      SUB      #10,(SP)      ;RESTORE (SP)
9982 045702
9983
9984      5$:
9985      ;REPORT ANY IVC ERROR AS
9986      ; IVC ERROR WITH VOLUME VALID ZERO
9987 045702 032737 010000 001372      ; ERRONEOUS IVC ERROR, VOLUME VALID IS SET
9988 045710 001427      BIT      #IVC,RMR2I      ; IVC ERROR??
9989 045712 005037 001140      BEQ      52$          ;NO!!
9990 045716 013737 001372 001142      CLR      $GDDAT      ;EXPECTED STATUS
9991 045724 062716 000004      MOV      RMR2I,$BDDAT      ;RECEIVED STATUS
9992 045730 112776 000060 000000      ADD      #4,(SP)      ;MOVE SP TO USER'S ERROR
9993 045736 032737 000100 001342      MOVVB     #60,2(SP)      ;ERROR 60 IF VV = 0
9994 045744 001403      BIT      #VV,RMO5I
9995 045746 112776 000061 000000      BEQ      51$          ;ERROR 61 IF VV = 1
9996 045754 162716 000002 51$:      MOVVB     #61,2(SP)      ;MOVE SP TO RETURN FOR ERROR
9997 045760 004736      SUB      #2,(SP)      ;REPORT ERROR VIA USER
9998 045762 162716 000010      JSR      PC,2(SP)+      ;RESTORE SP
9999 045766 000240      SUB      #10,(SP)
10000
10001 045770 013737 001372 001142 52$:      MOV      RMR2I,$BDDAT      ;RECEIVED STATUS
10002 045776 042737 137777 001142      BIC      #ICSKI,$BDDAT      ;CLEAR DONT CARES
10003 046004 013737 046656 001140      MOV      300$,$GDDAT      ;GET EXPECTED SKI STATUS
10004 046012 042737 137777 001140      BIC      #ICSKI,$GDDAT      ;CLEAR DONT CARES
10005 046020 001417      BEQ      53$          ;BRANCH IF 0 EXPECTED
10006
10007      ;REPORT ERROR IF SKI IS NOT SET (IAE WAS NOT DETECTED)
10008 046022 032737 040000 001142      BIT      #SKI,$BDDAT      ;WAS SKI DETECTED ??
10009 046030 001032      BNE      54$          ;YES !!
10010 046032 062716 000004      ADD      #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
10011 046036 112776 000267 000000      MOVVB     #267,2(SP)      ;WRITE ERROR NUMBER
10012 046044 162716 000002      SUB      #2,(SP)      ;MOVE SP TO ERROR RETURN
10013 046050 004736      JSR      PC,2(SP)+      ;REPORT ERROR AND RETURN
10014 046052 162716 000010      SUB      #10,(SP)      ;MOVE SP TO NO ERROR
10015 046056 000443      BR       6$              ;GO TO NEXT ERROR CHECK
10016 046060
10017      53$:

```

```
10017
10018
10019 046060 032737 040000 001142 ;REPORT ERROR IF SKI IS SET
10020 046066 001413 BIT #SKI,$BDDAT ;IS SKI SET??
10021 046070 062716 000004 BEQ 54$ ;NO - SKI IS OK
10022 046074 112776 000054 000000 ADD #4,(SP) ;MOVE (SP) TO ERROR
10023 046102 162716 000002 MOVB #54,2(SP) ;LOAD ERROR NUMBER
10024 046106 004736 SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
10025 046110 162716 000010 JSR PC,2(SP)+ ;REPORT SEEK ERROR
10026 046114 000240 SUB #10,(SP) ;RESTORE (SP)
10027
10028
10029 046116 032737 000200 001372 ;REPORT ANY DEVICE CHECK
10030 046124 001420 54$: BIT #DVC,R-CR2I ;WAS THERE DVC DURING SEEK??
10031 046126 005037 001140 BEQ 6$ ;NO!!
10032 046132 013737 001372 001142 CLR $GDDAT ;EXPECTED STATUS
10033 046140 062716 000004 MOV RMR1I,$BDDAT ;RECEIVED STATUS
10034 046144 112776 000055 000000 ADD #4,(SP)
10035 046152 162716 000002 MOVB #55,2(SP) ;ERROR #55
10036 046156 004736 SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
10037 046160 162716 000010 JSR PC,2(SP)+ ;REPORT ERROR VIA USER
10038 046164 000240 SUB #10,(SP) ;RESTORE SP
10039
10040
10041 ;REPORT ANY "OPI" ERROR AS OPI WITH MOL =0, OR OPI
10042 ;BECAUSE ON CYLINDER LATCH DIDN'T RESET
10043 046166 032737 020000 001344 6$: BIT #OPI,RMR1I ;"OPI" ERROR??
10044 046174 001427 BEQ 8$ ;NO!!
10045 046176 005037 001140 CLR $GDDAT ;EXPECTED STATUS
10046 046202 013737 001344 001142 MOV RMR1I,$BDDAT ;RECEIVED STATUS
10047 046210 062716 000004 ADD #4,(SP) ;MOVE (SP) TO ERROR
10048 046214 112776 000052 000000 MOVB #52,2(SP) ;LOAD ERROR NUMBER
10049 046222 032737 010000 001342 BIT #MOL,RMDSI ;WAS MEDIUM ON LINE??
10050 046232 112776 000053 000000 BEQ 7$ ;NO!!
10051 046240 162716 000002 MOVB #53,2(SP) ;YES - CHANGE ERROR NUMBER
10052 046244 004736 SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
10053 046246 162716 000010 JSR PC,2(SP)+ ;REPORT "OPI" ERROR
10054 046252 000240 SUB #10,(SP) ;RESTORE (SP)
10055
10056
10057 046254 013746 001342 ;SEE IF "PIP" IS 0, AND "ATA","MOL"AND"VV" =1
10058 046260 042716 047677 8$: MOV RMDSI,-(SP)
10059 046264 022726 110100 BIC #C(ATA:PIP:MOL:VV), (SP)
10060 046270 001002 CMP #ATA:MOL:VV, (SP)+
10061 046272 000137 046626 BNE 9$ ;ERROR IN RMDS
10062 JMP 14$ ;RMDS IS OK
10063
10064
10065 046276 032737 010000 001342 ;REPORT ERROR IF MOL = 0 AND OPI = 0
10066 046304 001030 9$: BIT #MOL,RMDSI ;IS MOL RESET??
10067 046306 032737 020000 001344 BNE 10$ ;NO - MOL IS SET
10068 046314 001024 BIT #OPI,RMR1I ;WAS OPI SET
10069 046316 013737 001342 001140 BNE 10$ ;YES - DONT REPORT ERROR
10070 046324 052737 010000 001140 MOV RMDSI,$GDDAT ;EXPECTED STATUS
10071 046332 013737 001342 001142 BIS #MOL,$GDDAT
10072 046340 062716 000004 MOV RMDSI,$BDDAT ;RECEIVED STATUS
10073 046344 112776 000062 000000 ADD #4,(SP)
10074 MOVB #62,2(SP)
```

K16

SEQ 0205

10073	046352	162716	000002		SUB	#2, (SP)	; MOVE SP TO RETURN FOR ERROR
10074	046356	004736			JSR	PC, 2(SP)+	; REPORT ERROR VIA USER
10075	046360	162716	000010		SUB	#10, (SP)	
10076	046364	000240			NOP		
10077							
10078							
10079	046366	032737	020000	001342	105:	BIT	#PIP, RMOSI
10080	046374	001430			BEQ	115	; IS "PIP" STILL SET??
10081	046376	032737	040000	001372	BIT	#SKI, RMER2I	; NO!!
10082	046404	001024			BNE	115	; WAS "SKI" SET??
10083	046406	013737	001342	001140	MOV	RMOSI, \$GDOAT	; YES-DONT REPORT PIP
10084	046414	042737	020000	001142	BIC	#PI6, \$BDOAT	; EXPECTED STATUS
10085	046422	013737	001342	001142	MOV	RMOSI, \$BDOAT	
10086	046430	062716	000004		ADD	#4, (SP)	; RECEIVED STATUS
10087	046434	112776	000056	000000	MOVB	#56, 2(SP)	; MOVE (SP) TO ERROR
10088	046442	162716	000002		SUB	#2, (SP)	; LOAD ERROR NUMBER
10089	046446	004736			JSR	PC, 2(SP)+	; MOVE SP TO RETURN FOR ERROR
10090	046450	162716	000010		SUB	#10, (SP)	; REPORT "PIP" SET AFTER SEEK
10091	046454	000240			NOP		; RESTORE (SP)
10092							
10093							
10094	046456	032737	100000	001342	115:	BIT	#ATA, RMOSI
10095	046464	001024			BNE	135	; WAS "ATA" SET ??
10096	046466	013737	001342	001140	MOV	RMOSI, \$GDOAT	; YES!!
10097	046474	052737	110600	001140	BIS	#ATA!MOL!DPR!DRY, \$GDOAT	; EXPECTED STATUS
10098	046502	013737	001342	001142	MOV	RMOSI, \$BDOAT	
10099	046510	062716	000004		ADD	#4, (SP)	; RECEIVED STATUS
10100	046514	112776	000057	000000	MOVB	#57, 2(SP)	; MOVE (SP) TO ERROR
10101	046522	162716	000002		SUB	#2, (SP)	; LOAD ERROR NUMBER
10102	046526	004736			JSR	PC, 2(SP)+	; MOVE SP TO RETURN FOR ERROR
10103							; REPORT ATTENTION NOT SET DURING
10104	046530	162716	000010		SUB	#10, (SP)	; SEEK TEST
10105	046534	000240			NOP		; RESTORE (SP)
10106							
10107							
10108	046536	032737	000100	001342	135:	BIT	#VV, RMOSI
10109	046544	001030			BNE	145	; IS VV = 0 ??
10110	046546	032737	010000	001372	BIT	#IVC, RMER2I	; NO!!
10111	046554	001024			BNE	145	; IS IVC ALSO 0 ??
10112	046556	013737	001342	001140	MOV	RMOSI, \$GDOAT	; NO - IVC IS SET
10113	046564	052737	000100	001140	BIS	#VV, \$GDOAT	; EXPECTED STATUS
10114	046572	013737	001342	001142	MOV	RMOSI, \$BDOAT	
10115	046600	062716	000004		ADD	#4, (SP)	; RECEIVED STATUS
10116	046604	112776	000064	000000	MOVB	#64, 2(SP)	; ERROR #64
10117	046612	162716	000002		SUB	#2, (SP)	; MOVE SP TO RETURN FOR ERROR
10118	046616	004736			JSR	PC, 2(SP)+	
10119	046620	162716	000010		SUB	#10, (SP)	
10120	046624	000240			NOP		
10121	046626						
10122							
10123							
10124	046626	000240			145:		
10125	046630	062716	000004		NOP		
10126	046634	105776	000000		ADD	#4, (SP)	; MODIFY THE RETURN ADDRESS IF AN ERROR WAS DETECTED
10127	046640	001403			TSTB	2(SP)	; MOVE (SP) TO ERROR CALL
10128	046642	062716	000004		BEQ	155	; WAS ERROR CALLED??
					ADD	#4, (SP)	; NO!!
							; MOVE TO ERROR RETURN

L16

CZPM080 RM03/2 FCTNL TST 2
CZPM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 206
SEEK STATUS CHECK SUBROUTINE

SEQ 0206

10129	046646	000402	BR	16\$	
10130					
10131	046650	162716	15\$: SUB	#4, (SP)	; MOVE (SP) TO NO ERROR RETURN
10132	046654	000207	16\$: RTS	PC	; RETURN
10133					
10134	046656	000000	300\$: .WORD	0	; ERROR FLAGS
10135					

.SBTTL CONTROLLER CLEAR SUBROUTINE

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

CALL: JSR PC,CNTCLR
BR
NOP
ERROR
RETURN HERE IF NO ERROR FOUND
RETURN HERE IF ANY ERROR FOUND
SUB DEFINES ERROR NUMBER

CNTCLR:

MOV RO,-(SP) ;: PUSH RO ON STACK
MOV R1,-(SP) ;: PUSH R1 ON STACK
MOV ERRVEC,-(SP) ;: PUSH ERRVEC ON STACK
MOV ERRVEC+2,-(SP) ;: PUSH ERRVEC+2 ON STACK
MOV #10\$,ERRVEC ;: SETUP FOR BUS TIMEOUT
MOV #PR6,ERRVEC+2
MOV \$BASE,RO ;: RO=UNIBUS ADDRESS
MOV #CLR,AMCS2(RO) ;: CLEAR MASSBUS
MOV TSTQUE,R1
MOVB (R1),AMCS2(RO) ;: SELECT DEVICE
BR 20\$
10\$: CMP (SP)+,(SP)+ ;: ADJUST STACK
ADD #4,10(SP) ;: MOVE SP TO USER'S ERROR CALL
MOVB #7,10(SP) ;: WRITE THE ERROR NUMBER
SUB #2,10(SP)
20\$: MOV (SP)+,ERRVEC+2 ;: POP STACK INTO ERRVEC+2
MOV (SP)+,ERRVEC ;: POP STACK INTO ERRVEC
MOV (SP)+,R1 ;: POP STACK INTO R1
MOV (SP)+,RO ;: POP STACK INTO RO
RTS PC

10136
10137
10138
10139
10140
10141
10142
10143
10144
10145
10146
10147 046660
10148 046660 010046
10149 046662 010146
10150 046664 013746 000004
10151 046670 013746 000006
10152 046674 012737 046734 000004
10153 046702 0127 000300 000006
10154 046710 01370 001276
10155 046714 012760 000040 000010
10156 046722 013701 01450
10157 046726 111160 000010
10158 046732 000412
10159 046734 022626
10160 046736 062766 000004 000010
10161 046744 112776 000007 000010
10162 046752 162766 000002 000010
10163 046760
10164 046760 012637 000006
10165 046764 012637 000004
10166 046770 012601
10167 046772 012600
10168 046774 000207
10169

10170
10171
10172
10173
10174
10175
10176
10177
10178
10179
10180
10181
10182
10183
10184
10185
10186
10187
10188
10189
10190
10191
10192
10193
10194
10195
10196
10197
10198
10199
10200
10201
10202
10203
10204
10205
10206
10207
10208
10209
10210
10211
10212
10213
10214
10215
10216
10217
10218
10219
10220
10221
10222
10223
10224
10225

046776

046776 062716 000004
047002 105076 000000
047006 162716 000004
047012 013737 001330 001142
047020 042737 100000 001142
047026 012737 004200 001140
047034 023737 001140 001142
047042 001413
047044 062716 000004
047050 112776 000126 000000
047056 162716 000002
047062 004736
047064 162716 000010
047070 000240
047072 005037 001140
047076 013737 001334 001142
047104 001413
047106 062716 000004
047112 112776 000127 000000
047120 162716 000002
047124 004736
047126 162716 000010
047132 000240
047134 013737 001340 001142
047142 010146
047144 005046
047146 013701 001450
047152 111116
047154 052716 000100

.SBTTL CONTROLLER CLEAR STATUS CHECK SUBROUTINE

: THIS SUBROUTINE VERIFIES THAT THE RMD3 SUBSYSTEM IS INITIALIZED BASED ON
: STATUS STORED IN THE GET BUFFER. THIS SUBROUTINE SHOULD ONLY BE
: USED FOLLOWING A CONTROLLER CLEAR OPERATION, I.E., WRITING A 1 IN BIT
: 5 OF RMCS2, BECAUSE THE ERROR MESSAGES ARE BASED ON THAT CONDITION.

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

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

:CALL:

(1) JSR PC,CLRSTS
BR ???
NOP
ERROR
JSR PC,2(SP)+
???

RETURN HERE IF NO ERROR
RETURN HERE TO REPORT AN ERROR
ERROR NUMBER DEFINED BY SUB
GO BACK TO SUB FOR MORE ERROR CHECKS
RETURN HERE IF NO MORE ERRORS

CLRSTS:

:CLEAR USER'S ERROR CALL

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

:REPORT ERROR IF RMBA NOT RESET

\$S: CLR \$GDDAT ;VERIFY RMBA IS ZERO
MOV RMBAI,\$BDDAT
BEQ \$S ;BRANCH IF ZERO
ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
MOVB #127,2(SP) ;WRITE ERROR NUMBER IN CALL
SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
JSR PC,2(SP)+ ;REPORT ERROR VIA USER
SUB #10,(SP) ;MOVE SP BACK TO NO ERROR
NOP

:REPORT ERROR IF RMCS2 NOT INITIALIZED

7\$: MOV RMCS2I,\$BDDAT ;VERIFY RMCS2
MOV R1,-(SP) ;PUSH R1 ON STACK
CLR -(SP) ;EXPECT IR & UNIT NUMBER
MOV TSTQUE,R1 ;R1 = ADDRESS OF TEST QUE
MOVB (R1),(SP)
BIS #IR,(SP)

10226	047160	012637	001140		MOV	(CP)+,\$GDDAT	
10227	047164	012601			MOV	(SP)+,R1	;POP STACK INTO R1
10228	047166	023737	001140	001142	CMP	\$GDDAT,\$BDDAT	;COMPARE EXPECTED & RECEIVED
10229	047174	001413			BEQ	9\$;BRANCH IF EQUAL
10230	047176	062716	000004		ADD	#4,(SP)	;MOVE SP TO USER'S ERROR CALL
10231	047202	112776	000130	000000	MOVB	#130,2(SP)	;WRITE ERROR NUMBER IN CALL
10232	047210	162716	000002		SUB	#2,(SP)	;MOVE SP TO RETURN FOR ERROR
10233	047214	004736			JSR	PC,2(SP)+	;REPORT ERROR VIA USER
10234	047216	162716	000010		SUB	#10,(SP)	;MOVE SP BACK TO NO ERROR
10235	047222	000240			NOP		
10236					;REPORT ERROR IF RMER1 NOT RESET-IGNORE UNS		
10237	047224	005037	001140		9\$:	CLR	\$GDDAT ;VERIFY RMER1
10238	047230	013737	001344	001142	MOV	RMER1,\$BDDAT	
10239	047236	042737	040000	001142	BIC	#UNS,\$BDDAT	;IGNORE UNSAFE
10240	047244	001413			BEQ	13\$;BRANCH IF ZERO
10241	047246	062716	000004		ADD	#4,(SP)	;MOVE SP TO USER'S ERROR CALL
10242	047252	112776	000131	000000	MOVB	#131,2(SP)	;WRITE ERROR NUMBER IN CALL
10243	047260	162716	000002		SUB	#2,(SP)	;MOVE SP TO RETURN FOR ERROR
10244	047264	004736			JSR	PC,2(SP)+	;REPORT ERROR VIA USER
10245	047266	162716	000010		SUB	#10,(SP)	;MOVE SP BACK TO NO ERROR
10246	047272	000240			NOP		
10247					;REPORT ERROR IF RMMR1 NOT INITIALIZED-IGNORE WC,LS,LST		
10248	047274	013737	001354	001142	13\$:	MOV	RMMR1,\$BDDAT ;VERIFY RMMR1
10249	047302	042737	000046	001142	BIC	#WC!LS!LST,\$BDDAT	;IGNORE WORD CLOCK, SCT, TRK
10250	047310	012737	000010	001140	MOV	#MWO,\$GDDAT	;EXPECT WRITE DATA BIT
10251	047316	023737	001140	001142	CMP	\$GDDAT,\$BDDAT	;COMPARE EXPECTED AND RECEIVED
10252	047324	001413			BEQ	17\$;BRANCH IF 0
10253	047326	062716	000004		ADD	#4,(SP)	;MOVE SP TO USER'S ERROR CALL
10254	047332	112776	000133	000000	MOVB	#133,2(SP)	;WRITE ERROR NUMBER IN CALL
10255	047340	162716	000002		SUB	#2,(SP)	;MOVE SP TO RETURN FOR ERROR
10256	047344	004736			JSR	PC,2(SP)+	;REPORT ERROR VIA USER
10257	047346	162716	000010		SUB	#10,(SP)	;MOVE SP BACK TO NO ERROR
10258	047352	000240			NOP		
10259					;REPORT AN ERROR IF RMEC2 IS NOT RESET		
10260	047354	005037	001140		17\$:	CLR	\$GDDAT ;EXPECT ZEROS
10261	047360	013737	001376	001142	MOV	RMEC2,\$BDDAT	;VERIFY RMEC2=0
10262	047366	001413			BEQ	19\$	
10263	047370	062716	000004		ADD	#4,(SP)	;MOVE SP TO USER'S ERROR CALL
10264	047374	112776	000135	000000	MOVB	#135,2(SP)	;WRITE ERROR NUMBER IN CALL
10265	047402	162716	000002		SUB	#2,(SP)	;MOVE SP TO RETURN FOR ERROR
10266	047406	004736			JSR	PC,2(SP)+	;REPORT ERROR VIA USER
10267	047410	162716	000010		SUB	#10,(SP)	;MOVE SP BACK TO NO ERROR
10268	047414	000240			NOP		
10269					;REPORT ERROR IF RMMR2 NOT INITIALIZED-IGNORE RQA,RQB		
10270	047416	013737	001370	001142	19\$:	MOV	RMMR2,\$BDDAT ;VERIFY RMMR2
10271	047424	042737	140000	001142	BIC	#RQA!RQB,\$BDDAT	
10272	047432	012737	011777	001140	MOV	#TST!177,\$GDDAT	;EXPECT TEST BIT ON
10273	047440	023737	001140	001142	CMP	\$GDDAT,\$BDDAT	
10274	047446	001413			BEQ	21\$	
10275	047450	062716	000004		ADD	#4,(SP)	;MOVE SP TO USER'S ERROR CALL
10276	047454	112776	000136	000000	MOVB	#136,2(SP)	;WRITE ERROR NUMBER IN CALL
10277	047462	162716	000002		SUB	#2,(SP)	;MOVE SP TO RETURN FOR ERROR
10278	047466	004736			JSR	PC,2(SP)+	;REPORT ERROR VIA USER
10279	047470	162716	000010		SUB	#10,(SP)	;MOVE SP BACK TO NO ERROR
10280	047474	000240			NOP		
10281					;REPORT ERROR IF RMER2 NOT RESET-IGNORE SKI,DVC		

CZRMDB0 RM03/2 FCTNL TST 2
CZRMDB.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 210
CONTROLLER CLEAR STATUS CHECK SUBROUTINE

SEQ 0210

10282	047476	005037	001140		21\$:	CLR	\$GDDAT	: EXPECT ALL ZEROS
10283	047502	013737	001372	001142		MOV	RMR2I,\$BDDAT	: VERIFY RMR2
10284	047510	042737	040200	001142		BIC	#SKI!DVC,\$BDDAT	: IGNORE DEVICE ERRORS
10285	047516	001413				BEQ	215\$: BRANCH IF OTHER BITS 0
10286	047520	062716	000004			ADD	#4,(SP)	: MOVE SP TO USER'S ERROR CALL
10287	047524	112776	000174	000000		MOVB	#174,2(SP)	: WRITE ERROR NUMBER IN CALL
10288	047532	162716	000002			SUB	#2,(SP)	: MOVE SP TO RETURN FOR ERROR
10289	047536	004736				JSR	PC,2(SP)+	: REPORT ERROR VIA USER
10290	047540	162716	000010			SUB	#10,(SP)	: MOVE SP BACK TO NO ERROR
10291	047544	000240				NOP		
10292						: REPORT	ERROR IF RMD5 NOT INITIALIZED	
10293	047546	013737	001342	001142	215\$:	MOV	RMD5I,\$BDDAT	: TEST DRIVE STATUS REGISTER
10294	047554	042737	177177	001142		BIC	#1C<DRY!DPR>,\$BDDAT	
10295	047562	012737	000600	001140		MOV	#DPR!DRY,\$GDDAT	: EXPECTED DRIVE STATUS
10296	047570	023737	001140	001142		CMF	\$GDDAT,\$BDDAT	: COMPARE EXPECTED & RECEIVED
10297	047576	001413				BEQ	22\$: BRANCH IF EQUAL
10298	047600	062716	000004			ADD	#4,(SP)	: MOVE SP TO USER'S ERROR CALL
10299	047604	112776	000134	000000		MOVB	#134,2(SP)	: WRITE ERROR NUMBER
10300	047612	162716	000002			SUB	#2,(SP)	: MOVE SP TO RETURN FOR ERROR
10301	047616	004736				JSR	PC,2(SP)+	: REPORT ERROR TO USER
10302	047620	162716	000010			SUB	#10,(SP)	: MOVE SP BACK TO NO ERROR
10303	047624	000240				NOP		
10304	047626	062716	000004		22\$:	ADD	#4,(SP)	: MOVE SP TO ERROR CALL
10305	047632	105776	000000			TSTB	2(SP)	: WAS AN ERROE DETECTED??
10306	047636	001403				BEQ	23\$: NO!!
10307	047640	062716	000004			ADD	#4,(SP)	: YES - MOVE TO ERROR RETURN
10308	047644	000402				BR	24\$	
10309	047646	162716	000004		23\$:	SUB	#4,(SP)	: MOVE SP TO NO ERROR RETURN
10310	047652	000240			24\$:	NOP		
10311	047654	000207				RTS	PC	

EO1

CZRMO80 RM03/2 FCTNL TST 2
CZRMO8.P11 23-NOV-77 12:19MACY11 30(1046) 23-NOV-77 12:38 PAGE 211
PACK ACKNOWLEDGE STATUS CHECK

SEQ 0211

10312
10313
10314
10315
10316
10317
10318
10319
10320
10321
10322
10323
10324
10325
10326
10327
10328
10329
10330
10331
10332
10333
10334
10335
10336
10337
10338
10339
10340
10341
10342
10343
10344
10345
10346
10347
10348
10349
10350
10351
10352
10353
10354
10355
10356
10357
10358
10359
10360
10361
10362
10363
10364
10365
10366
10367

047656

047656 062716 000004
047662 105076 000000
047666 162716 000004047672 032737 000100 001342
047700 001024
047702 013737 001342 001140
047710 052737 000100 001140
047716 013737 001342 001142
047724 062716 000004
047730 112776 000155 000000
047736 162716 000002
047742 004736
047744 162716 000010
047750 000240
047752047752 032737 010000 001342
047760 001024
047762 013737 001342 001140
047770 052737 010000 001140
047776 013737 001342 001142
050004 062716 000004
050010 112776 000041 000000
050016 162716 000002
050022 004 30
050024 162716 000010
050030 000240
050032050032 032737 060007 001344
050040 001570
050042 032737 040000 001344
050050 001424

.SBTTL PACK ACKNOWLEDGE STATUS CHECK

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

; CALL:

; (1) JSR PC,ACKSTS
BR ???
NOP
ERROR
JSR PC,2(SP)+
???RETURN HERE IF NO ERROR
RETURN HERE TO REPORT AN ERROR
ERROR NUMBER DEFINED BY SUB
GO BACK TO SUB FOR MORE ERROR CHECKS
RETURN HERE IF NO MORE ERRORS

ACKSTS:

; CLEAR USER'S ERROR CALL

ADD #4,(SP)
CLRB 2(SP)
SUB #4,(SP); MOVE SP TO ERROR CALL
; CLEAR LOW ORDER BYTE
; MOVE SP BACK

; REPORT AN ERROR IF "VV" IS 0

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

1\$:

; REPORT AN ERROR IF "MOL" IS 0

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

2\$:

; SEE IF "UNS", "OPI", "RMR", "ILR", OR "ILF" IS SET

BIT #UNS!OPI!RMR!ILR!ILF,RMER1I
BEQ 7\$

; REPORT AN ERROR IF "UNS" IS SET

BIT #UNS,RMER1I
BEQ 3\$; WAS UNS SET??
; NO!!

F01

CZRMD80 RM03/2 FCTNL TST 2
CZRMD8.P11 23-NOV-77 12:19MACY11 30(1046) 23-NOV-77 12:38 PAGE 212
PACK ACKNOWLEDGE STATUS CHECK

SEQ 0212

10368	050052	013737	001344	001142	MOV	RMER11,\$BDDAT	;RECEIVED STATUS
10369	050060	013737	001344	001140	MOV	RMER11,\$GDDAT	;EXPECTED STATUS
10370	050066	042737	040000	001140	BIC	#UNS,\$GDDAT	
10371	050074	062716	000004		ADD	#4,(SP)	;MOVE SP TO ERROR CALL
10372	050100	112776	000042	000000	MOVB	#42,2(SP)	;WRITE NUMBER OF ERROR IN CALL
10373	050106	162716	000002		SUB	#2,(SP)	;MOVE SP TO RETURN FOR ERROR
10374	050112	004736			JSR	PC,2(SP)+	;REPORT THE ERROR VIA USER
10375	050114	162716	000010		SUB	#10,(SP)	;MOVE SP TO NO ERROR RETURN
10376	050120	000240			NOP		
10377	050122				3\$:		
10378							
10379					;REPORT ANY OPI ERROR		
10380	050122	032737	020000	001344	BIT	#OPI,RMER11	;WAS OPI SET??
10381	050130	001424			BEQ	4\$;NO!!
10382	050132	013737	001344	001142	MOV	RMER11,\$BDDAT	;RECEIVED STATUS
10383	050140	013737	001344	001140	MOV	RMER11,\$GDDAT	;EXPECTED STATUS
10384	050146	042737	020000	001140	BIC	#OPI,\$GDDAT	
10385	050154	062716	000004		ADD	#4,(SP)	;MOVE SP TO ERROR CALL
10386	050160	112776	000043	000000	MOVB	#43,2(SP)	;WRITE NUMBER OF ERROR IN CALL
10387	050166	162716	000002		SUB	#2,(SP)	;MOVE SP TO RETURN FOR ERROR
10388	050172	004736			JSR	PC,2(SP)+	;REPORT THE ERROR VIA USER
10389	050174	162716	000010		SUB	#10,(SP)	;MOVE SP TO NO ERROR RETURN
10390	050200	000240			NOP		
10391	050202				4\$:		
10392							
10393					;REPORT ANY RMR ERROR		
10394	050202	032737	000004	001344	BIT	#RMR,RMER11	;WAS RMR SET??
10395	050210	001424			BEQ	5\$;NO!!
10396	050212	013737	001344	001142	MOV	RMER11,\$BDDAT	;RECEIVED STATUS
10397	050220	013737	001344	001140	MOV	RMER11,\$GDDAT	;EXPECTED STATUS
10398	050226	042737	000004	001140	BIC	#RMR,\$GDDAT	
10399	050234	062716	000004		ADD	#4,(SP)	;MOVE SP TO ERROR CALL
10400	050240	112776	000044	000000	MOVB	#44,2(SP)	;WRITE NUMBER OF ERROR IN CALL
10401	050246	162716	000002		SUB	#2,(SP)	;MOVE SP TO RETURN FOR ERROR
10402	050252	004736			JSR	PC,2(SP)+	;REPORT THE ERROR VIA USER
10403	050254	162716	000010		SUB	#10,(SP)	;MOVE SP TO NO ERROR RETURN
10404	050260	000240			NOP		
10405	050262				5\$:		
10406							
10407					;REPORT ANY ILR ERROR		
10408	050262	032737	000002	001344	BIT	#ILR,RMER11	;WAS ILR SET??
10409	050270	001424			BEQ	6\$;NO!!
10410	050272	013737	001344	001142	MOV	RMER11,\$BDDAT	;RECEIVED STATUS
10411	050300	013737	001344	001140	MOV	RMER11,\$GDDAT	;EXPECTED STATUS
10412	050306	042737	000002	001140	BIC	#ILR,\$GDDAT	
10413	050314	062716	000004		ADD	#4,(SP)	;MOVE SP TO ERROR CALL
10414	050320	112776	000045	000000	MOVB	#45,2(SP)	;WRITE NUMBER OF ERROR IN CALL
10415	050326	162716	000002		SUB	#2,(SP)	;MOVE SP TO RETURN FOR ERROR
10416	050332	004736			JSR	PC,2(SP)+	;REPORT THE ERROR VIA USER
10417	050334	162716	000010		SUB	#10,(SP)	;MOVE SP TO NO ERROR RETURN
10418	050340	000240			NOP		
10419	050342				6\$:		
10420							
10421					;REPORT ANY ILF ERROR		
10422	050342	032737	000001	001344	BIT	#ILF,RMER11	;WAS ILF SET??
10423	050350	001424			BEQ	7\$;NO!!

GO1

CZRMDB0 RM03/2 FCTNL TST 2
CZRMDB.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 213
PACK ACKNOWLEDGE STATUS CHECK

SEQ 0213

10424	050352	013737	001344	001142	MOV	RMER11,\$BDDAT	;RECEIVED STATUS
10425	050360	013737	001344	001140	MOV	RMER11,\$GDDAT	;EXPECTED STATUS
10426	050366	042737	000001	001140	BIC	#1LF,\$GDDAT	
10427	050374	062716	000004		ADD	#4,(SP)	;MOVE SP TO ERROR CALL
10428	050400	112776	000046	000000	MOVB	#46,2(SP)	;WRITE NUMBER OF ERROR IN CALL
10429	050406	162716	000002		SUB	#2,(SP)	;MOVE SP TO RETURN FOR ERROR
10430	050412	004736			JSR	PC,2(SP)+	;REPORT THE ERROR VIA USER
10431	050414	162716	000010		SUB	#10,(SP)	;MOVE SP TO NO ERROR RETURN
10432	050420	000240			NOP		
10433	050422						
10434							
10435							
10436	050422	062716	000004		ADD	#4,(SP)	;MOVE SP TO ERROR CALL
10437	050426	105776	000000		TSTB	2(SP)	;WAS ERROR FOUND??
10438	050432	001403			BEQ	8\$;NO!!
10439	050434	062716	000004		ADD	#4,(SP)	;YES - MOVE TO ERROR RETURN
10440	050440	000402			BR	9\$	
10441	050442	162716	000004	8\$:	SUB	#4,(SP)	;MOVE SP TO NO ERROR RETURN
10442	050446	000240		9\$:	NOP		
10443	050450	000207			R~S	PC	
10444							

H01

CZRMD80 RM03/2 FCTNL TST 2
CZRMD8.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 214
RECALIBRATE STATUS CHECK SUBROUTINE

SEQ 0214

```

10445 .SBTTL RECALIBRATE STATUS CHECK SUBROUTINE
10446 ;THIS SUBROUTINE CHECKS THE RESULTS OF A RECALIBRATE OPERATION
10447 ;USING THE STATUS STORED IN THE GET BUFFER.
10448
10449 ;CALL:
10450
10451 ;(1) JSR PC,RCLSTS ;CALL SUBROUTINE
10452 BR ??? ;RETURN HERE IF NO ERROR
10453 NOP ;RETURN HERE TO REPORT AN ERROR
10454 ERROR ;ERROR NUMBER DEFINED BY SUB
10455 JSR PC,2(SP)+ ;GO BACK TO SUB FOR MORE ERROR CHECKS
10456 ??? ;RETURN HERE IF NO MORE ERRORS
10457
10458 RCLSTS:
10459 ;CLEAR USER'S ERROR NUMBER
10460 ADD #4,(SP)
10461 CLRB 2(SP) ;CLEAR USER'S ERROR CALL
10462 SUB #4,(SP) ;MOVE SP BACK TO BRANCH
10463
10464 ;SEE IF "PAR" OR "ILF" OR "OPI" OR "IAE" IS SET
10465 BIT #OPI,PAR,ILF,IAE,RMER11
10466 BEQ 4$ ;NONE ARE SET - GO TO NEXT CHECK
10467
10468 ;REPORT ANY MASSBUS CONTROL BUS PARITY ERROR, I.E.,
10469 ;"PAR" = 1 AND "DPE" = 0
10470 BIT #PAR,RMER11 ;WAS "PAR" SET??
10471 BEQ 1$ ;NO!!
10472 BIT #DPE,RMER21 ;WAS "DPE" SET??
10473 BNE 1$ ;YES - NOT A REGISTER ERROR
10474 MOV RMER11,$GDDAT ;EXPECTED STATUS
10475 BIC #PAR,$GDDAT
10476 MOV RMER11,$BDDAT ;RECEIVED STATUS
10477 ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
10478 MOVB #50,2(SP) ;WRITE ERROR NUMBER IN CALL
10479 SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
10480 JSR PC,2(SP)+ ;GO REPORT ERROR
10481 SUB #10,(SP) ;MOVE SP BACK TO BRANCH
10482 NOP
10483
10484 1$:
10485 ;REPORT ANY "ILF" ERROR
10486 BIT #ILF,RMER11 ;WAS "ILF" SET??
10487 BEQ 2$ ;NO!!
10488 MOV RMER11,$GDDAT ;EXPECTED STATUS
10489 BIC #ILF,$GDDAT
10490 MOV RMER11,$BDDAT ;RECEIVED STATUS
10491 ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
10492 MOVB #71,2(SP) ;WRITE ERROR NUMBER IN CALL
10493 SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
10494 JSR PC,2(SP)+ ;REPORT ERROR VIA USER
10495 SUB #10,(SP) ;MOVE SP BACK TO BRANCH
10496 NOP
10497
10498 2$:
10499
10500

```



```

10501
10502
10503
10504
10505 050646 032737 020000 001344
10506 050654 001433
10507 050656 013737 001344 001140
10508 050664 042737 020000 001140
10509 050672 013737 001344 001142
10510 050700 062716 000004
10511 050704 112776 000072 000000
10512 050712 032737 010000 001342
10513 050720 001403
10514 050722 112776 000073 000000
10515 050730 162716 000002
10516 050734 004736
10517 050736 162716 000010
10518 050742 000240
10519 050744
10520
10521
10522 050744 032737 002000 001344
10523 050752 001424
10524 050754 013737 001344 001140
10525 050762 042737 002000 001140
10526 050770 013737 001344 001142
10527 050776 062716 000004
10528 051002 112776 000070 000000
10529 051010 162716 000002
10530 051014 004736
10531 051016 162716 000010
10532 051022 000240
10533 051024
10534
10535
10536 051024 032737 050200 001372
10537 051032 001517
10538
10539
10540
10541
10542
10543 051034 032737 010000 001372
10544 051042 001433
10545 051044 013737 001372 001140
10546 051052 042737 010000 001140
10547 051060 013737 001372 001142
10548 051066 062716 000004
10549 051072 112776 000074 000000
10550 051100 032737 000100 001342
10551 051106 001403
10552 051110 112776 000075 000000
10553 051116 162716 000002
10554 051122 004736
10555 051124 162716 000010
10556 051130 000240

;REPORT ANY "OPI" ERROR AS
; . OPI DUE TO "MOL" = 0
; . OPI BECAUSE ON CYLINDER LATCH DIDN'T RESET
;
; BIT #OPI,RMER11 ; WAS OPI SET??
; BEQ 31$ ; NO!!
; MOV RMER11,$GDDAT ; EXPECTED STATUS
; BIC #OPI,$GDDAT
; MOV RMER11,$BDDAT ; RECEIVED STATUS
; ADD #4,(SP) ; MOVE SP TO USER'S ERROR CALL
; MOVB #72,2(SP) ; WRITE ERROR NUMBER IN USER'S CALL
; BIT #MOL,RMDSI ; WAS "MOL" = 0??
; BEQ 3$ ; YES!!
; MOVB #73,2(SP) ; NO - CHANGE ERROR NUMBER
; SUB #2,(SP) ; MOVE SP TO RETURN FOR ERROR
; JSR PC,2(SP)+ ; REPORT ERROR VIA USER
; SUB #10,(SP) ; MOVE SP BACK TO BRANCH
; NOP

31$:
3$:
3$:

;REPORT AN ERROR IF "IAE" IS SET
; BIT #IAE,RMER11 ; IS "IAE" SET??
; BEQ 4$ ; NO!!
; MOV RMER11,$GDDAT ; EXPECTED STATUS
; BIC #IAE,$GDDAT
; MOV RMER11,$BDDAT ; RECEIVED STATUS
; ADD #4,(SP) ; MOVE SP TO ERROR CALL
; MOVB #70,2(SP) ; WRITE ERROR NUMBER IN USER'S CALL
; SUB #2,(SP) ; MOVE SP TO RETURN FOR ERROR
; JSR PC,2(SP)+ ; REPORT ERROR
; SUB #10,(SP) ; MOVE SP BACK TO NO ERROR RETURN
; NOP

4$:

;SEE IF "SKI" OR "IVC" OR "DVC" IS SET
; BIT #SKI!IVC!DVC,RMER2I
; BEQ 8$ ; NONE OF THE BITS ARE SET

;REPORT ANY "IVC" ERROR AS
; . IVC WITH VV = 0
; . ERRONEOUS IVC ERROR
;
; BIT #IVC,RMER2I ; WAS IVC SET??
; BEQ 6$ ; NO!!
; MOV RMER2I,$GDDAT ; EXPECTED STATUS
; BIC #IVC,$GDDAT
; MOV RMER2I,$BDDAT ; RECEIVED STATUS
; ADD #4,(SP) ; MOVE SP TO USER'S ERROR CALL
; MOVB #74,2(SP) ; WRITE ERROR NUMBER IN CALL
; BIT #VV,RMDSI ; WAS VV = 0??
; BEQ 5$ ; YES!!
; MOVB #75,2(SP) ; NO - CHANGE ERROR NUMBER
; SUB #2,(SP) ; MOVE SP TO RETURN FOR ERROR
; JSR PC,2(SP)+ ; REPORT ERROR VIA USER
; SUB #10,(SP) ; MOVE SP BACK TO BRANCH
; NOP

5$:

```

```

10557 051132          6$:
10558
10559          ;REPORT ANY "SKI" ERROR
10560 051132 032737 040000 001372      BIT      #SKI,RMER2I      ;WAS SKI SET??
10561 051140 001424          BEQ      7$      ;NO!!
10562 051142 013737 001372 001140      MOV      RMER2I,$GDDAT      ;EXPECTED STATUS
10563 051150 042737 040000 001140      BIC      #SKI,$GDDAT
10564 051156 013737 001372 001142      MOV      RMER2I,$BDDAT      ;RECEIVED STATUS
10565 051164 062716 000004          ADD      #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
10566 051170 112776 000076 000000      MOV      #76,2(SP)      ;WRITE ERROR NUMBER
10567 051176 162716 000002          SUB      #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
10568 051202 004736          JSR      PC,2(SP)+      ;REPORT ERROR VIA USER
10569 051204 162716 000010          SUB      #10,(SP)      ;MOVE SP TO BRANCH
10570 051210 000240          NOP
10571 051212          7$:
10572
10573          ;REPORT ANY "DVC" ERROR
10574 051212 032737 000200 001372      BIT      #DVC,RMER2I      ;WAS "DVC" SET??
10575 051220 001424          BEQ      8$      ;NO!!
10576 051222 013737 001372 001140      MOV      RMER2I,$GDDAT      ;EXPECTED STATUS
10577 051230 042737 000200 001140      BIC      #DVC,$GDDAT
10578 051236 013737 001372 001142      MOV      RMER2I,$BDDAT      ;RECEIVED STATUS
10579 051244 062716 000004          ADD      #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
10580 051250 112776 000077 000000      MOV      #77,2(SP)      ;WRITE ERROR NUMBER
10581 051256 162716 000002          SUB      #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
10582 051262 004736          JSR      PC,2(SP)+      ;REPORT ERROR VIA USER
10583 051264 162716 000010          SUB      #10,(SP)      ;MOVE SP TO USER'S BRANCH
10584 051270 000240          NOP
10585 051272          8$:
10586
10587          ;SEE IF "PIP" AND "OM" ARE 0, AND "ATA" "MOL" AND "VV" ARE 1
10588 051272 013746 001342          MOV      RMDSI,-(SP)      ;PUT RMDS ON STACK
10589 051276 042716 047676          BIC      #1C<PIP!MOL!VV!OM!ATA>,(SP)
10590 051302 022726 110100          CMP      #ATA!MOL!VV,(SP)+
10591 051306 001002          BNE      85$
10592 051310 000137 051724          JMP      13$
10593 051314          85$:
10594
10595          ;REPORT AN ERROR IF MOL = 0 AND OPI = 0, I.E., MEDIUM WENT OFF
10596          ;LINE AFTER RECALIBRATE WAS INITIATED
10597 051314 032737 010000 001342      BIT      #MOL,RMDSI      ;DID MOL DROP??
10598 051322 001030          BNE      9$      ;NO!!
10599 051324 032737 020000 001344      BIT      #OPI,RMER1I      ;WAS OPI ERROR REPORTED??
10600 051332 001024          BNE      9$      ;YES - DON'T REPORT MOL=0
10601 051334 013737 001342 001140      MOV      RMDSI,$GDDAT      ;EXPECTED STATUS
10602 051342 052737 010000 001140      BIC      #MOL,$GDDAT
10603 051350 013737 001342 001142      MOV      RMDSI,$BDDAT      ;RECEIVED STATUS
10604 051356 062716 000004          ADD      #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
10605 051362 112776 000100 000000      MOV      #100,2(SP)      ;WRITE ERROR NUMBER
10606 051370 162716 000002          SUB      #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
10607 051374 004736          JSR      PC,2(SP)+      ;REPORT ERROR VIA USER
10608 051376 162716 000010          SUB      #10,(SP)      ;MOVE SP BACK TO USER'S BRANCH
10609 051402 000240          NOP
10610 051404          9$:
10611
10612          ;REPORT AN ERROR IF "VV" = 0 AND "IVC" = 0

```

K01

CZRMO80 RM03/2 FCTNL TST 2 MACY11 30(1046) 23-NOV-77 12:38 PAGE 217
 CZRMO8.P11 23-NOV-77 12:19 RECALIBRATE STATUS CHECK SUBROUTINE

SEQ 0217

10613	051404	032737	000100	001342	BIT	#VV,RMSI	;DID "VV" DROP??
10614	051412	001030			BNE	10\$;NO!!
10615	051414	032737	010000	001372	BIT	#IVC,RMER2I	;WAS THERE A IVC ERROR??
10616	051422	001024			BNE	10\$;YES - DONT REPORT VV=0
10617	051424	013737	001342	001140	MOV	RMSI,\$GDDAT	;EXPECTED STATUS
10618	051432	013737	001342	001142	MOV	RMSI,\$BDDAT	;RECEIVED STATUS
10619	051440	052737	000100	001140	BIS	#VV,\$GDDAT	
10620	051446	062716	000004		ADD	#4,(SP)	;MOVE SP TO USER'S ERROR CALL
10621	051452	112776	000101	000000	MOVB	#101,2(SP)	;WRITE ERROR NUMBER IN CALL
10622	051460	162716	000002		SUB	#2,(SP)	;MOVE SP TO RETURN FOR ERROR
10623	051464	004736			JSR	PC,2(SP)+	
10624	051466	162716	000010		SUB	#10,(SP)	;MOVE SP BACK TO USER'S BRANCH
10625	051472	000240			NOP		
10626	051474						
10627					10\$:		
10628							
10629	051474	032737	100000	001342			;REPORT AN ERROR IF ATA IS NOT SET
10630	051502	001024			BIT	#ATA,RMSI	;WAS ATA SET DURING RECALIBRATE??
10631	051504	013737	001342	001140	BNE	11\$;YES!!
10632	051512	052737	100000	001140	MOV	RMSI,\$GDDAT	;EXPECTED STATUS
10633	051520	013737	001342	001142	BIS	#ATA,\$GDDAT	
10634	051526	062716	000004		MOV	RMSI,\$BDDAT	;RECEIVED STATUS
10635	051532	112776	000102	000000	ADD	#4,(SP)	;MOVE SP TO USER'S ERROR CALL
10636	051540	162716	000002		MOVB	#102,2(SP)	;WRITE ERROR NUMBER IN CALL
10637	051544	004736			SUB	#2,(SP)	
10638	051546	162716	000010		JSR	PC,2(SP)+	
10639	051552	000240			SUB	#10,(SP)	;MOVE SP TO USER'S BRANCH
10640					NOP		
10641	051554						
10642					11\$:		
10643							
10644							;REPORT AN ERROR IF "OM" IS NOT ZERO BECAUSE RECALIBRATE SHOULD
10645	051554	032737	000001	001342	;ALWAYS		
10646	051562	001424			BIT	#OM,RMSI	;WAS "OM" RESET??
10647	051564	013737	001342	001140	BEQ	12\$;YES!!
10648	051572	042737	000001	001140	MOV	RMSI,\$GDDAT	;EXPECTED STATUS
10649	051600	013737	001342	001142	BIC	#OM,\$GDDAT	
10650	051606	062716	000004		MOV	RMSI,\$BDDAT	;RECEIVED STATUS
10651	051612	112776	000103	000000	ADD	#4,(SP)	;MOVE SP TO USER'S ERROR CALL
10652	051620	162716	000002		MOVB	#103,2(SP)	;WRITE ERROR NUMBER
10653	051624	004736			SUB	#2,(SP)	;MOVE SP TO RETURN FOR ERROR
10654	051626	162716	000010		JSR	PC,2(SP)+	;REPORT ERROR VIA USER
10655	051632	000240			SUB	#10,(SP)	;MOVE SP TO USER'S BRANCH
10656	051634				NOP		
10657					12\$:		
10658							
10659							;REPORT AN ERROR IF "PIP" IS STIL ON, I.E., DRIVE NOT ON
10660	051634	032737	020000	001342	;CYLINDER		
10661	051642	001430			BIT	#PIP,RMSI	;IS DRIVE OFF CYLINDER??
10662	051644	032737	040000	001372	REQ	13\$;NO!!
10663	051652	001024			BIT	#SKI,RMER2I	;WAS "SKI" DETECTED??
10664	051654	013737	001342	001140	BNE	13\$;YES-DONT REPORT "PIP"
10665	051662	042737	020000	001140	MOV	RMSI,\$GDDAT	;EXPECTED STATUS
10666	051670	013737	001342	001142	BIC	#PIP,\$GDDAT	
10667	051676	062716	000004		MOV	RMSI,\$BDDAT	;RECEIVED STATUS
10668	051702	112776	000104	000000	ADD	#4,(SP)	;MOVE SP TO USER'S ERROR CALL
					MOVB	#104,2(SP)	;WRITE ERROR NUMBER

Address	Instruction	Comment
10669	051710 162716 000002	SUB #2, (SP) ; MOVE SP TO RETURN FOR ERROR
10670	051714 004736	JSR PC, 2(SP) +
10671	051716 162716 000010	SUB #10, (SP) ; MOVE SP BACK TO USER'S BRANCH
10672	051722 000240	NOP
10673	051724	13\$:
10674		
10675		; SEE IF "ILR" OR "RMR" OR "UNS" IS SET
10676	051724 032737 040006 001344	BIT #ILR!RMR!UNS, RMER11
10677	051732 001514	BEQ 16\$
10678		
10679		; REPORT AN ERROR IF "ILR" IS SET
10680	051734 032737 000002 001344	BIT #ILR, RMER11 ; WAS ILR SET DURING RECALIBRATE??
10681	051742 001424	BEQ 14\$; NO!!
10682	051744 013737 001344 001140	MOV RMER11, \$GDDAT ; EXPECTED STATUS
10683	051752 042737 000002 001140	BIC #ILR, \$GDDAT
10684	051760 013737 001344 001142	MOV RMER11, \$BDDAT ; RECEIVED STATUS
10685	051766 062716 000004	ADD #4, (SP) ; MOVE SP TO USER'S ERROR CALL
10686	051772 112776 000105 000000	MOVB #105, 2(SP) ; WRITE ERROR NUMBER IN CALL
10687	052000 162716 000002	SUB #2, (SP) ; MOVE SP TO RETURN FOR ERROR
10688	052004 004736	JSR PC, 2(SP) +
10689	052006 162716 000010	SUB #10, (SP) ; MOVE SP TO USER'S BRANCH
10690	052012 000240	NOP
10691	052014	14\$:
10692		
10693		; REPORT AN ERROR IF "RMR" IS SET
10694	052014 032737 000004 001344	BIT #RMR, RMER11 ; WAS RMR SET??
10695	052022 001424	BEQ 15\$; NO!!
10696	052024 013737 001344 001140	MOV RMER11, \$GDDAT ; EXPECTED STATUS
10697	052032 042737 000004 001140	BIC #RMR, \$GDDAT
10698	052040 013737 001344 001142	MOV RMER11, \$BDDAT ; RECEIVED STATUS
10699	052046 062716 000004	ADD #4, (SP) ; MOVE SP TO USER'S ERROR CALL
10700	052052 112776 000106 000000	MOVB #106, 2(SP) ; WRITE ERROR NUMBER IN USER'S CALL
10701	052060 162716 000002	SUB #2, (SP) ; MOVE SP TO RETURN FOR ERROR
10702	052064 004736	JSR PC, 2(SP) +
10703	052066 162716 000010	SUB #10, (SP) ; MOVE SP TO USER'S BRANCH
10704	052072 000240	NOP
10705	052074	15\$:
10706		
10707		; REPORT AN ERROR IF "UNS" IS SET AND "DVC" IS 0
10708	052074 032737 040000 001344	BIT #UNS, RMER11 ; WAS UNSAFE ON??
10709	052102 001430	BEQ 16\$; NO!!
10710	052104 032737 000200 001372	BIT #DVC, RMER21 ; WAS THERE A DEVICE CHECK??
10711	052112 001024	BNE 16\$; YES - DON'T REPORT UNSAFE
10712	052114 013737 001344 001140	MOV RMER11, \$GDDAT ; EXPECTED STATUS
10713	052122 042737 040000 001140	BIC #UNS, \$GDDAT
10714	052130 013737 001344 001142	MOV RMER11, \$BDDAT ; RECEIVED STATUS
10715	052136 062716 000004	ADD #4, (SP) ; MOVE SP TO USER'S ERROR CALL
10716	052142 112776 000107 000000	MOVB #107, 2(SP) ; WRITE ERROR NUMBER
10717	052150 162716 000002	SUB #2, (SP) ; MOVE SP TO RETURN FOR ERROR
10718	052154 004736	JSR PC, 2(SP) +
10719	052156 162716 000010	SUB #10, (SP) ; MOVE SP BACK TO USER'S BRANCH
10720	052162 000240	NOP
10721	052164	16\$:
10722		
10723		; AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
10724	052164 062716 000004	ADD #4, (SP) ; MOVE SP TO USER'S ERROR CALL

MO1

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 219
RECALIBRATE STATUS CHECK SUBROUTINE

SEQ 0219

10725	052170	105776	000000
10726	052174	001403	
10727	052176	062716	000004
10728	052202	000402	
10729	052204	162716	000004
10730	052210	000240	
10731	052212	000207	
10732			

	TSTB	2(SP)	; WAS AN ERROR REPORTED??
	BEG	17\$; NO!!
	ADD	#4, (SP)	; YES - AUGMENT SP RETURN
	BR	18\$	
17\$:	SUB	#4, (SP)	; NO ERROR - RETURN SP TO BRANCH
18\$:	NOP		
	RTS	PC	; STATUS CECK IS COMPLETE

```

10733 .SBTTL DRIVE CLEAR STATUS CHECK SUBROUTINE
10734 BR ??? RETURN HERE IF NO ERROR
10735 NOP RETURN HERE TO REPORT AN ERROR
10736 ERROR ERROR NUMBER DEFINED BY SUB
10737 JSR PC,2(SP)+ GO BACK TO SUB FOR MORE ERROR CHECKS
10738 ??? RETURN HERE IF NO MORE ERRORS
10739
10740 052214 DRVSTS:
10741
10742 ;CLEAR USER'S ERROR CALL
10743 052214 062716 000004 ADD #4,(SP) ;MOVE SP TO ERROR CALL
10744 052220 105076 000000 CLRB 2(SP) ;CLEAR ERROR CALL
10745 052224 162716 000004 SUB #4,(SP) ;MOVE SP TO USER'S BRANCH
10746
10747 052230 013737 001330 001142 ;REPORT ERROR IF RMCS1 NOT INITIALIZED
10748 052236 042737 173700 001142 4$: MOV RMCS1,$BDDAT ;CHECK RMCS1
10749 052244 012737 004010 001140 BIC #1C<DVA!FNCMSK>,$BDDAT ;CLEAR DONT CARES
10750 052252 023737 001140 001142 MOV #DVA!DRVCLR,$GDDAT ;EXPECT DVA
10751 052260 001443 BEQ 6$ ;COMPARE EXPECTED & RECEIVED
10752 052262 062716 000004 ADD #4,(SP) ;BRANCH IF EQUAL
10753 052266 112776 000141 000000 MOV #141,2(SP) ;MOVE SP TO ERROR CALL
10754 052274 162716 000002 SUB #2,(SP) ;WRITE NUMBER OF ERROR IN CALL
10755 052300 004736 JSR PC,2(SP)+ ;MOVE SP TO RETURN FOR ERROR
10756 052302 162716 000010 SUB #10,(SP) ;REPORT THE ERROR VIA USER
10757 052306 000240 NOP ;MOVE SP TO NO ERROR RETURN
10758
10759 052310 013737 001342 001142 ;REPORT ERROR IF RMOS NOT INITIALIZED
10760 052316 042737 021101 001142 5$: MOV RMOS,$BDDAT ;CHECK RMOS
10761 052324 012737 010600 001140 BIC #PCM!OM!VV.PIP,$BDDAT ;CLEAR DONT CARES
10762 052332 023737 001140 001142 MOV #MOL!DPR!DRY,$GDDAT ;EXPECT DRY & DPR
10763 052340 001413 BEQ 6$ ;COMPARE EXPECTED & RECEIVED
10764 052342 062716 000004 ADD #4,(SP) ;BRANCH IF EQUAL
10765 052346 112776 000142 000000 MOV #142,2(SP) ;MOVE SP TO ERROR CALL
10766 052354 162716 000002 SUB #2,(SP) ;WRITE NUMBER OF ERROR IN CALL
10767 052360 004736 JSR PC,2(SP)+ ;MOVE SP TO RETURN FOR ERROR
10768 052362 162716 000010 SUB #10,(SP) ;REPORT THE ERROR VIA USER
10769 052366 000240 NOP ;MOVE SP TO NO ERROR RETURN
10770
10771 052370 005037 001140 001142 ;REPORT ERROR IF RMER1 NOT INITIALIZED
10772 052374 013737 001344 001142 6$: CLR $GDDAT ;EXPECT 0'S
10773 052402 001413 BEQ 8$ ;CHECK RMER1
10774 052404 062716 000004 ADD #4,(SP) ;BRANCH IF EQUAL
10775 052410 112776 000143 000000 MOV #143,2(SP) ;MOVE SP TO ERROR CALL
10776 052416 162716 000002 SUB #2,(SP) ;WRITE NUMBER OF ERROR IN CALL
10777 052422 004736 JSR PC,2(SP)+ ;MOVE SP TO RETURN FOR ERROR
10778 052424 162716 000010 SUB #10,(SP) ;REPORT THE ERROR VIA USER
10779 052430 000240 NOP ;MOVE SP TO NO ERROR RETURN
10780
10781 052432 013737 001346 001142 ;REPORT ERROR IF ATA NOT INITIALIZED
10782 052440 010146 001142 8$: MOV RMASI,$BDDAT ;CHECK ATTENTION BIT
10783 052442 010246 MOV R1,-(SP) ;PUSH R1 ON STACK
10784 052444 013701 MOV R2,-(SP) ;PUSH R2 ON STACK
10785 052450 116102 000001 MOV TSTQUE,R1
10786 052454 042702 177400 MOV #1(R1),R2
10787 052460 005102 COM R2
10788 052462 040237 001142 BIC R2,$BDDAT

```

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 221
DRIVE CLEAR STATUS CHECK SUBROUTINE

SEQ 0221

10789	052466	012602			MOV	(SP)+,R2	::POP STACK INTO R2	
10790	052470	012601			MOV	(SP)+,R1	::POP STACK INTO R1	
10791	052472	005737	001142		TST	\$BDDAT	::IS ATTENTION CLEARED??	
10792	052476	001413			BEQ	9\$::BRANCH IF ATTENTION CLEARED	
10793	052500	062716	000004		ADD	#4,(SP)	::MOVE SP TO ERROR CALL	
10794	052504	112776	000144	000000	MOVB	#144,2(SP)	::WRITE NUMBER OF ERROR IN CALL	
10795	052512	162716	000002		SUB	#2,(SP)	::MOVE SP TO RETURN FOR ERROR	
10796	052516	004736			JSR	PC,2(SP)+	::REPORT THE ERROR VIA USER	
10797	052520	162716	000010		SUB	#10,(SP)	::MOVE SP TO NO ERROR RETURN	
10798	052524	000240			NOP			
10799								
10800	052526	013737	001354	001142	9\$:	MOV	RMMR1,\$BDDAT	::CHECK RMMR1
10801	052534	042737	000046	001142	BIC	#WC!LS!LST,\$BDDAT	::CLEAR DONT CARES	
10802	052542	012737	000010	001140	MOV	#MLO,\$GDDAT	::EXPECT WRITE DATA ON	
10803	052550	023737	001140	001142	CMP	\$GDDAT,\$BDDAT	::COMPARE EXPECTED AND RECEIVED	
10804	052556	001413			BEQ	11\$::BRANCH IF ZERO	
10805	052560	062716	000004		ADD	#4,(SP)	::MOVE SP TO ERROR CALL	
10806	052564	112776	000145	000000	MOVB	#145,2(SP)	::WRITE NUMBER OF ERROR IN CALL	
10807	052572	162716	000002		SUB	#2,(SP)	::MOVE SP TO RETURN FOR ERROR	
10808	052576	004736			JSR	PC,2(SP)+	::REPORT THE ERROR VIA USER	
10809	052600	162716	000010		SUB	#10,(SP)	::MOVE SP TO NO ERROR RETURN	
10810	052604	000240			NOP			
10811								
10812	052606	013737	001370	001142	11\$:	MOV	RMMR2,\$BDDAT	::CHECK RMMR2
10813	052614	042737	140000	001142	BIC	#RQA!RQB,\$BDDAT	::CLEAR RQA, RQB	
10814	052622	012737	011777	001140	MOV	#TST!1777,\$GDDAT	::EXPECT TEST BIT ON	
10815	052630	023737	001140	001142	CMP	\$GDDAT,\$BDDAT	::COMPARE EXPECTED & RECEIVED	
10816	052636	001413			BEQ	15\$::BRANCH IF EQUAL	
10817	052640	062716	000004		ADD	#4,(SP)	::MOVE SP TO ERROR CALL	
10818	052644	112776	000146	000000	MOVB	#146,2(SP)	::WRITE NUMBER OF ERROR IN CALL	
10819	052652	162716	000002		SUB	#2,(SP)	::MOVE SP TO RETURN FOR ERROR	
10820	052656	004736			JSR	PC,2(SP)+	::REPORT THE ERROR VIA USER	
10821	052660	162716	000010		SUB	#10,(SP)	::MOVE SP TO NO ERROR RETURN	
10822	052664	000240			NOP			
10823	052666	005037	001140		15\$:	CLR	\$GDDAT	::EXPECT ZEROS
10824								
10825	052672	013737	001376	001142	17\$:	MOV	RMEC2,\$BDDAT	::CHECK RMEC2
10826	052700	001413			BEQ	17\$::BRANCH IF 0	
10827	052702	062716	000004		ADD	#4,(SP)	::MOVE SP TO ERROR CALL	
10828	052706	112776	000150	000000	MOVB	#150,2(SP)	::WRITE NUMBER OF ERROR IN CALL	
10829	052714	162716	000002		SUB	#2,(SP)	::MOVE SP TO RETURN FOR ERROR	
10830	052720	004736			JSR	PC,2(SP)+	::REPORT THE ERROR VIA USER	
10831	052722	162716	000010		SUB	#10,(SP)	::MOVE SP TO NO ERROR RETURN	
10832	052726	000240			NOP			
10833								
10834	052730	013737	001372	001142	17\$:	MOV	RMER2,\$BDDAT	::CHECK RMER2
10835	052736	001413			BEQ	18\$::BRANCH IF NO ERROR	
10836	052740	062716	000004		ADD	#4,(SP)	::MOVE SP TO ERROR CALL	
10837	052744	112776	000147	000000	MOVB	#147,2(SP)	::WRITE NUMBER OF ERROR IN CALL	
10838	052752	162716	000002		SUB	#2,(SP)	::MOVE SP TO RETURN FOR ERROR	
10839	052756	004736			JSR	PC,2(SP)+	::REPORT THE ERROR VIA USER	
10840	052760	162716	000010		SUB	#10,(SP)	::MOVE SP TO NO ERROR RETURN	
10841	052764	000240			NOP			
10842	052766				18\$:			
10843								
10844	052766				19\$:			

CO2

CZRMO80 RM03/2 FCTNL TST 2
CZRMO8.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 222
DRIVE CLEAR STATUS CHECK SUBROUTINE

SEQ 0222

ADDRESS	HEX	ASCII	OPERATION	COMMENT
10845				
10846				; AUGMENT RETURN ADDRESS IF ANY ERROR WAS FOUND
10847	052766	062716	000004	ADD #4 (SP) ; MOVE SP TO ERROR CALL
10848	052772	105776	000000	TSTB 0(SP) ; WAS AN ERROR DETECTED??
10849	052776	001403		BEG 21\$; NO!!
10850	053000	062716	000004	ADD #4 (SP) ; YES - MOVE SP TO ERROR RETURN
10851	053004	000402		BR 23\$
10852	053006	162716	000004	21\$: SUB #4, (SP) ; MOVE SP BACK TO NO ERROR RETURN
10853	053012	000240		23\$: NOP
10854	053014	000207		RTS PC ; RETURN TO USER

CZRM030 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 223
DATA TRANSFER COMMAND STATUS CHECK SUBROUTINE

SEQ 0223

```

10855 .SBTTL DATA TRANSFER COMMAND STATUS CHECK SUBROUTINE
10856
10857 ;THIS SUBROUTINE VERIFIES THE RESULTS OF ALL DATA TRANSFER COMMANDS
10858 ;USING STATUS STORED IN THE GET BUFFER AND TEST PARAMETERS
10859 ;STORED IN THE PUT BUFFER. ERRORS ARE REPORTED BY WRITING
10860 ;THE ERROR NUMBER IN THE USER'S ERROR CALL.
10861
10862 ;USER'S SUBROUTINE CALL:
10863 ;(1) JSR PC,DTASTS
10864 ;(2) BR ??
10865 ;(3) NOP
10866 ;(4) ERROR
10867 ;(5) JSR PC,2(SP)+
10868 ;(6) ??
10869
10870
10871 053016 DTASTS:
10872
10873 ;CLEAR USER'S ERROR CALL AND ERROR FLAGS
10874 053016 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR
10875 053022 105076 000000 CLRB 2(SP) ;CLEAR LOW ORDER BYTE OF TRAP
10876 053026 162716 000004 SUB #4,(SP) ;RESTORE SP TO NO ERROR
10877 053032 005037 056412 CLR 500$ ;CLEAR ERROR FLAGS
10878
10879 ;REPORT ANY CONTROL BUS PARITY ERROR WHILE READING REMOTE REGISTERS. I.E.. MCPE = 1
10880 053036 032737 020000 001330 BIT #MCPE,RMCS11 ;WAS THERE A PARITY ERROR??
10881 053044 001422 BEQ 10$ ;NO!!
10882 053046 013737 001330 001140 MOV RMCS11,$GDDAT ;EXPECTED STATUS
10883 053054 042737 020000 001140 BIC #MCPE,$GDDAT
10884 053062 013737 001330 001142 MOV RMCS11,$BDDAT ;RECEIVED STATUS
10885 053070 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
10886 053074 112773 000013 000000 MOVB #13,2(SP) ;WRITE ERROR NUMBER
10887 053102 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN IF ERROR
10888 053106 004736 JSR PC,2(SP)+ ;REPORT ERROR AND RETURN
10889 053110 000466 BR 30$
10890
10891 10$:
10892 ;REPORT ANY CONTROL BUS PARITY ERROR WHILE WRING REMOTE REGISTERS. I.E.,
10893 ;PAR=1 AND DPE = 0
10894 053112 032737 000010 001344 BIT #PAR,RMER11 ;WAS THERE A PARITY ERROR??
10895 053120 001435 BEQ 20$ ;NO!!
10896 053122 032737 000010 001372 BIT #DPE,RMER21 ;WAS IT DUE TO CONTROL BUS??
10897 053130 001031 BNE 20$ ;NO!!
10898 053132 013737 001344 001140 MOV RMER11,$GDDAT ;EXPECTED STATUS
10899 053140 042737 000010 001140 BIC #PAR,$GDDAT
10900 053146 013737 001344 001142 MOV RMER11,$BDDAT ;RECEIVED STATUS
10901 053154 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR
10902 053160 112776 000050 000000 MOVB #50,2(SP) ;WRITE ERROR NUMBER
10903 053166 032737 001000 001330 BIT #MXF,RMCS11 ;DID MXF GET SET??
10904 053174 001003 BNE 15$ ;YES!!
10905 053176 112776 000274 000000 MOVB #274,2(SP) ;NO - CHANGE ERROR NUMBER
10906 053204 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN IF ERROR
10907 053210 004736 JSR PC,2(SP)+ ;REPORT ERROR AND RETURN
10908 053212 000425 BR 30$
10909
10910 053214 20$:

```

```

10911
10912 ;LOOK FOR ANY ERRORS WHICH MAY HAVE OCCURRED DURING COMMAND INITIATION OR
10913 ;MECHANICAL POSITIONING
10914
10915 ;FIRST TEST MXF WHICH WOULD INDICATE COMPOSITE ERROR SET WHEN FUNCTION
10916 ;CODE AND GO BIT WERE LOADED
10917 053214 032737 001000 001340      BIT      #MXF,RMCS2I      ;WAS "MISSED TRANSFER" SET??
10918 053222 001425                      BEQ      40$          ;NO!!
10919 053224 013737 001340 001140      MOV      RMCS2I,$GDDAT    ;EXPECTED STATUS
10920 053232 042737 001000 001140      BIC      #MXF,$GDDAT
10921 053240 013737 001340 001142      MOV      RMCS2I,$BDDAT    ;RECEIVED STATUS
10922 053246 062716 000004                      ADD      #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
10923 053252 112776 000275 000000      MOVVB   #275,2(SP)      ;WRITE ERROR NUMBER
10924 053260 162716 000002                      SUB      #2,(SP)      ;MOVE SP TO RETURN IF ERROR
10925 053264 004736                      JSR      PC,2(SP)+      ;REPORT ERROR AND RETURN
10926 053266
10927 30$:
10928 ;RESTORE SP TO NO ERROR RETURN AND BYPASS FURTHER STATUS CHECKING
10929 053266 162716 000010                      SUB      #10,(SP)      ;MOVE SP TO NO ERROR
10930 053272 000137 056364                      JMP      380$          ;SKIP TO END OF SUB
10931
10932 053276
10933 40$:
10934 ;REPORT AN ERROR IF "OPI" ERROR OCCURRED DUE TO "MOL" = 0, OR IF "OPI"
10935 ;AND "MOL" ARE SET, BUT "VV" IS RESET, INDICATING AN INTERMITTENT
10936 ;"MOL"
10937 053276 032737 020000 001344      BIT      #OPI,RMER1I    ;IS "OPI" SET??
10938 053304 001447                      BEQ      60$          ;NO!!
10939 053306 013737 001344 001140      MOV      RMER1I,$GDDAT    ;EXPECTED STATUS
10940 053314 042737 020000 001140      BIC      #OPI,$GDDAT
10941 053322 013737 001344 001142      MOV      RMER1I,$BDDAT    ;RECEIVED STATUS
10942 053330 032737 010000 001342      BIT      #MOL,RMDSI    ;WAS MEDIUM OFF LINE??
10943 053336 001404                      BEQ      45$          ;YES!!
10944 053340 032737 000100 001342      BIT      #VV,RMDSI    ;WAS "MOL" INTERMITTENT??
10945 053346 001013                      BNE      50$          ;NO!!
10946 053350 062716 000004                      ADD      #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
10947 053354 112776 000276 000000      MOVVB   #276,2(SP)      ;WRITE ERROR NUMBER IN CALL
10948 053362 162716 000002                      SUB      #2,(SP)      ;MOVE SP TO RETURN IF ERROR
10949 053366 004736                      JSR      PC,2(SP)+      ;REPORT ERROR AND RETURN
10950 053370 162716 000010                      SUB      #10,(SP)      ;RESTORE SP TO NO ERROR
10951 053374 000413                      BR       60$
10952
10953 50$:
10954 ;REPORT "OPI" ERROR, WHICH IS DUE TO "ON CYLINDER" NOT DROPPING OF
10955 ;"RUN" TIMEOUT (20 MS) OR SEARCH TIMEOUT (50 MS)
10956 053376 062716 000004                      ADD      #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
10957 053402 112776 000277 000000      MOVVB   #277,2(SP)      ;WRITE ERROR NUMBER IN CALL
10958 053410 162716 000002                      SUB      #2,(SP)      ;MOVE SP TO RETURN IF ERROR
10959 053414 004736                      JSR      PC,2(SP)+      ;REPORT ERROR AND RETURN
10960 053416 162716 000010                      SUB      #10,(SP)      ;RESTORE SP TO NO ERROR
10961 053422 000240                      NOP
10962 053424
10963
10964 60$:
10965 ;LOOK FOR "IVC" ERROR DURING COMMAND INITIATION
10966 053424 032737 010000 001372      BIT      #IVC,RMER2I    ;WAS THERE AN "IVC" ERROR??
10966 053432 001432                      BEQ      70$          ;NO!!

```

10967					;REPORT "IVC" ERROR DUE TO "VV" = 0 OR REPORT ERRONEOUS "IVC" ERROR
10968	053434	013737	001372	001140	MOV RMER2I,\$GDDAT ;EXPECTED STATUS
10969	053442	042737	010000	001140	BIC #IVC,\$GDDAT
10970	053450	013737	001372	001142	MOV RMER2I,\$BDDAT ;RECEIVED STATUS
10971	053456	062716	000004		ADD #4,(SP) ;MOVE SP TO USER'S ERROR
10972	053462	112776	000300	000000	MOVB #300,2(SP) ;WRITE ERROR NUMBER IN CALL
10973	053470	032737	000100	001342	BIT #VV,RMSI ;WAS VOLUME VALID??
10974	053476	001403			BEQ 65\$;NO!!
10975	053500	112776	000301	000000	MOVB #301,2(SP) ;CHANGE ERROR NUMBER
10976	053506	162716	000002		SUB #2,(SP) ;MOVE SP TO RETURN IF ERROR
10977	053512	004736			JSR PC,2(SP)+ ;REPORT "IVC" ERROR AND RETURN
10978	053514	162716	000010		SUB #10,(SP) ;RESTORE SP TO NO ERROR
10979	053520				
10980					
10981					;SEE IF "ILF" OR "RMR" IS SET
10982	053520	032737	000007	001344	BIT #ILF,ILF!RMR,RMER1I
10983	053526	001510			BEQ 100\$;NO ERRORS DETECTED
10984					;REPORT AN ERROR IF "ILF" IS SET
10985	053530	032737	000002	001344	BIT #ILF,RMER1I ;WAS "ILF" DETECTED??
10986	053536	001424			BEQ 80\$;NO!!
10987	053540	013737	001344	001140	MOV RMER1I,\$GDDAT ;EXPECTED STATUS
10988	053546	042737	000002	001140	BIC #ILF,\$GDDAT
10989	053554	013737	001344	001142	MOV RMER1I,\$BDDAT ;RECEIVED STATUS
10990	053562	062716	000004		ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
10991	053566	112776	000302	000000	MOVB #302,2(SP) ;WRITE ERROR NUMBER IN CALL
10992	053574	162716	000002		SUB #2,(SP) ;MOVE SP TO RETURN IF ERROR
10993	053600	004736			JSR PC,2(SP)+ ;REPORT ERROR AND RETURN
10994	053602	162716	000010		SUB #10,(SP) ;RESTORE SP TO NO ERROR
10995	053606	000240			NOP
10996	053610				80\$:
10997					
10998					;REPORT AN ERROR IF "ILF" IS SET
10999	053610	032737	000001	001344	BIT #ILF,RMER1I ;WAS "ILF" DETECTED??
11000	053616	001424			BEQ 90\$;NO!!
11001	053620	013737	001344	001140	MOV RMER1I,\$GDDAT ;EXPECTED STATUS
11002	053626	042737	000001	001140	BIC #ILF,\$GDDAT
11003	053634	013737	001344	001142	MOV RMER1I,\$BDDAT ;RECEIVED STATUS
11004	053642	062716	000004		ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11005	053646	112776	000303	000000	MOVB #303,2(SP) ;WRITE ERROR NUMBER IN CALL
11006	053654	162716	000002		SUB #2,(SP) ;MOVE SP TO RETURN IF ERROR
11007	053660	004736			JSR PC,2(SP)+ ;REPORT ERROR AND RETURN
11008	053662	162716	000010		SUB #10,(SP) ;RESTORE SP TO NO ERROR
11009	053666	000240			NOP
11010	053670				90\$:
11011					
11012					;REPORT AN ERROR IF "RMR" IS SET
11013	053670	032737	000004	001344	BIT #RMR,RMER1I ;WAS "RMR" DETECTED??
11014	053676	001424			BEQ 100\$;NO!!
11015	053700	013737	001344	001140	MOV RMER1I,\$GDDAT ;EXPECTED STATUS
11016	053706	042737	000004	001140	BIC #RMR,\$GDDAT
11017	053714	013737	001344	001142	MOV RMER1I,\$BDDAT ;RECEIVED STATUS
11018	053722	062716	000004		ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11019	053726	112776	000304	000000	MOVB #304,2(SP) ;WRITE ERROR NUMBER IN CALL
11020	053734	162716	000002		SUB #2,(SP) ;MOVE SP TO RETURN IF ERROR
11021	053740	004736			JSR PC,2(SP)+ ;REPORT ERROR AND RETURN
11022	053742	162716	000010		SUB #10,(SP) ;RESTORE SP TO NO ERROR

11023	053746	000240			NOP		
11024	053750				100\$:		
11025					; DETERMINE WHETHER OR NOT "IAE"	SHOULD BE SET AND CHECK FOR ERROR	
11026	053750	012737	002000	001140	MOV	#IAE,\$GDDAT ; SETUP FOR "IAE" = 1	
11027	053756	052737	040000	056412	BIS	#SKI,500\$; SET SKI FLAG	
11028	053764	022737	001466	001434	CMP	#822,,\$RMDCO ; IS CYLINDER > 822??	
11029	053772	103425			BLO	110\$; YES!!	
11030	053774	042737	040000	056412	BIC	#SKI,500\$; RESET SKI FLAG	
11031	054002	122737	000004	001407	CMPB	#4,\$RMDAO+1 ; IS TRACK > 4??	
11032	054010	103416			BLO	110\$; YES!!	
11033	054012	122737	000037	001406	CMPB	#31,,\$RMDAO ; IS SECTOR > 31??	
11034	054020	103412			BLO	110\$; YES!!	
11035	054022	122737	000035	001406	CMPB	#29,,\$RMDAO ; IS SECTOR > 29??	
11036	054030	103004			BHIS	105\$; NO - IAE SHOULD BE ZERO	
11037	054032	032737	010000	001432	BIT	#FMT16,\$RMOFO ; 16 BIT FORMAT??	
11038	054040	001402			BEQ	110\$; YES!!	
11039	054042	005037	001140		CLR	\$GDDAT ; IAE SHOULD BE ZERO	
11040	054046	013737	001344	001142	110\$:	MOV	#RMR11,\$BDDAT ; GET RECEIVED STATUS
11041	054054	042737	175777	001142	BIC	#ICIAE,\$BDDAT	
11042	054062	023737	001140	001142	CMP	\$GDDAT,\$BDDAT ; IS "IAE" STATUS OK??	
11043	054070	001004			BNE	115\$; NO!!	
11044	054072	042737	040000	056412	BIC	#SKI,500\$; IAE OK - SKI SHOULD BE 0	
11045	054100	000412			BR	120\$	
11046	054102	062716	000004		115\$:	ADD	#4,(SP) ; MOVE SP TO USER'S ERROR CALL
11047	054106	112776	000305	000000	MOVB	#305,2(SP) ; WRITE ERROR NUMBER	
11048	054114	162716	000002		SUB	#2,(SP) ; MOVE SP TO RETURN IF ERROR	
11049	054120	004736			JSR	PC,2(SP)+ ; REPORT ERROR AND RETURN	
11050	054122	162716	000010		SUB	#10,(SP) ; MOVE SP TO NO ERROR	
11051	054126				120\$:		
11052							
11053					; REPORT AN ERROR IF "SKI" IS SET AND "IAE" STATUS WAS OK		
11054	054126	013737	001372	001142	MOV	#RMR21,\$BDDAT ; RECEIVED STATUS	
11055	054134	042737	137777	001142	BIC	#ICSKI,\$BDDAT	
11056	054142	013737	056412	001140	MOV	500\$,\$GDDAT ; EXPECTED STATUS	
11057	054150	042737	137777	001140	BIC	#ICSKI,\$GDDAT	
11058	054156	032737	040000	001372	BIT	#SKI,\$RMR21 ; WAS "SKI" SET??	
11059	054164	001417			BEQ	140\$; NO!!	
11060	054166	032737	040000	056412	BIT	#SKI,500\$; WAS SKI CAUSED BY IAE = 0??	
11061	054174	001032			BNE	150\$; YES - DON'T REPORT SKI	
11062	054176	062716	000004		ADD	#4,(SP) ; MOVE SP TO USER'S ERROR CALL	
11063	054202	112776	000306	000000	MOVB	#306,2(SP) ; WRITE ERROR NUMBER	
11064	054210	162716	000002		SUB	#2,(SP) ; MOVE SP TO RETURN IF ERROR	
11065	054214	004736			JSR	PC,2(SP)+ ; REPORT ERROR AND RETURN	
11066	054216	162716	000010		SUB	#10,(SP) ; MOVE SP TO NO ERROR	
11067	054222	000417			BR	150\$	
11068							
11069	054224				140\$:		
11070							
11071					; REPORT AN ERROR IF SKI = 0 AND IAE WAS NOT DETECTED		
11072	054224	032737	040000	056412	BIT	#SKI,500\$; SHOULD SKI BE SET??	
11073	054232	001413			BEQ	150\$; NO!!	
11074	054234	062716	000004		ADD	#4,(SP) ; MOVE SP TO USER'S ERROR CALL	
11075	054240	112776	000307	000000	MOVB	#307,2(SP) ; WRITE ERROR NUMBER IN CALL	
11076	054246	162716	000002		SUB	#2,(SP) ; MOVE SP TO RETURN IF ERROR	
11077	054252	004736			JSR	PC,2(SP)+ ; REPORT ERROR AND RETURN	
11078	054254	162716	000010		SUB	#10,(SP) ; RESTORE SP TO NO ERROR	

H02

CZRMDB0 RM03/2 FCTNL TST 2
CZRMDB.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 227
DATA TRANSFER COMMAND STATUS CHECK SUBROUTINE

SEQ 0227

```

11079 054260 000240      NOP
11080 054262      150$:
11081
11082      ;LOOK FOR "LSC" OR "LBC" OR "DVC" IN ERROR REGISTER #2
11083 054262 032737 006200 001372      BIT      #LSC!LBC!DVC,RMER2I
11084 054270 001512      BEQ      180$      ;NO ERRORS SET
11085
11086      ;REPORT ANY DEVICE FAULT, I.E., "DVC" = 1
11087 054272 032737 000200 001372      BIT      #DVC,RMER2I      ;IS "DVC" = 1??
11088 054300 001424      BEQ      160$      ;NO!!
11089 054302 013737 001372 001140      MOV      RMER2I,$GDDAT      ;EXPECTED STATUS
11090 054310 042737 000200 001140      BIC      #DVC,$GDDAT
11091 054316 013737 001372 001142      MOV      RMER2I,$BDDAT      ;RECEIVED STATUS
11092 054324 062716 000004      ADD      #4,(SP)      ;MOVE SP TO USERS ERROR
11093 054330 112776 000310 000000      MOV      #310,2(SP)      ;WRITE ERROR NUMBER IN CALL
11094 054336 162716 000002      SUB      #2,(SP)      ;MOVE SP TO RETURN IF ERROR
11095 054342 004736      JSR      PC,2(SP)+      ;REPORT ERROR AND RETURN
11096 054344 162716 000010      SUB      #10,(SP)      ;RESTORE SP TO NO ERROR
11097 054350 000240      NOP
11098 054352      160$:
11099
11100      ;REPORT LOSS OF BIT CLOCK, I.E., "LBC" = 1, IF "MOL" = 1
11101 054352 032737 002000 001372      BIT      #LBC,RMER2I      ;IS "LBC" SET??
11102 054360 001430      BEQ      170$      ;NO!!
11103 054362 032737 010000 001342      BIT      #MOL,RMDSI      ;WAS LBC ERROR BY MOL = 0
11104 054370 001424      BEQ      170$      ;YES!!
11105 054372 013737 001372 001140      MOV      RMER2I,$GDDAT      ;EXPECTED STATUS
11106 054400 042737 002000 001140      BIC      #LBC,$GDDAT
11107 054406 013737 001372 001142      MOV      RMER2I,$BDDAT      ;RECEIVED STATUS
11108 054414 062716 000004      ADD      #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
11109 054420 112776 000311 000000      MOV      #311,2(SP)      ;WRITE ERROR NUMBER IN CALL
11110 054426 162716 000002      SUB      #2,(SP)      ;MOVE SP TO RETURN IF ERROR
11111 054432 004736      JSR      PC,2(SP)+      ;REPORT ERROR AND RETURN
11112 054434 162716 000010      SUB      #10,(SP)      ;RESTORE SP TO NO ERROR
11113 054440 000240      NOP
11114 054442      170$:
11115
11116      ;REPORT LOS OF SYSTEM CLOCK, I.E., "LSC" = 1
11117 054442 032737 004000 001372      BIT      #LSC,RMER2I      ;IS "LSC" = 1??
11118 054450 001422      BEQ      180$      ;NO!!
11119 054452 013737 001372 001140      MOV      RMER2I,$GDDAT      ;EXPECTED STATUS
11120 054460 042737 004000 001140      BIC      #LSC,$GDDAT
11121 054466 013737 001372 001142      MOV      RMER2I,$BDDAT      ;RECEIVED STATUS
11122 054474 062716 000004      ADD      #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
11123 054500 112776 000312 000000      MOV      #312,2(SP)      ;WRITE ERROR NUMBER
11124 054506 004736      JSR      PC,2(SP)+      ;REPORT ERROR AND RETURN
11125 054510 162716 000010      SUB      #10,(SP)      ;RESTORE SP TO NO ERROR
11126 054514 000240      NOP
11127 054516      180$:
11128
11129      ;LOOK FOR "UNS" OR "DTE" OR "WLE" IN ERROR REGISTER #1
11130 054516 032737 054000 001344      BIT      #UNS!DTE!WLE,RMER1I
11131 054524 001527      BEQ      220$      ;NO BITS SET
11132
11132      ;REPORT "UNS" ERROR IF "DVC" = 0
11133 054526 032737 040000 001344      BIT      #UNS,RMER1I      ;IS "UNS" SET??
11134 054534 001427      BEQ      190$      ;NO!!

```

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 228
DATA TRANSFER COMMAND STATUS CHECK SUBROUTINE

SEQ 0228

```

11135 054536 032737 000200 001372      BIT      #DVC,RMER2I      ;WAS "UNS" CAUSED BY "DVC"?
11136 054544 001023                BNE      190$          ;YES!!
11137 054546 013737 001344 001140      MOV      RMER1I,$GDDAT      ;EXPECTED STATUS
11138 054554 042737 040000 001140      BIC      #UNS,$GDDAT
11139 054562 013737 001344 001142      MOV      RMER1I,$BDDAT      ;RECEIVED STATUS
11140 054570 062716 000004                ADD      #4,(SP)          ;MOVE SP TO USERS ERROR CALL
11141 054574 112776 000313 000000      MOV      #313,2(SP)        ;WRITE ERROR NUMBER
11142 054602 162716 000002                SUB      #2,(SP)          ;MOVE SP TO RETURN IF ERROR
11143 054606 004736                JSR      ,C,2(SP)+      ;REPORT ERROR AND RETURN
11144 054610 162716 000010                SUB      #10,(SP)        ;RESTORE SP TO NO ERROR
11145 054614                190$:
11146
11147      ;REPORT ANY DRIVE TIMING ERROR, I.E. "DTE" = 1
11148 054614 032737 010000 001344      BIT      #DTE,RMER1I      ;IS DTE SET?
11149 054622 001423                BEQ      200$          ;NO!!
11150 054624 013737 001344 001140      MOV      RMER1I,$GDDAT      ;EXPECTED STATUS
11151 054632 042737 010000 001140      BIC      #DTE,$GDDAT
11152 054640 013737 001344 001142      MOV      RMER1I,$BDDAT      ;RECEIVED STATUS
11153 054646 062716 000004                ADD      #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
11154 054652 112776 000314 000000      MOV      #314,2(SP)        ;WRITE ERROR NUMBER IN CALL
11155 054660 162716 000002                SUB      #2,(SP)          ;MOVE SP TO RETURN IF ERROR
11156 054664 004736                JSR      PC,2(SP)+      ;REPORT ERROR AND RETURN
11157 054666 162716 000010                SUB      #10,(SP)        ;MOVE SP TO NO ERROR
11158 054672                200$:
11159
11160      ;REPORT AN ERROR IF WRITE LOCK ERROR IS SET. SEE IF DRIVE IS NOT
11161      ;WRITE PROTECTED, OR IF FUNCTION WAS NOT A WRITE
11162 054672 032737 004000 001344      BIT      #WLE,RMER1I      ;WAS "WLE" SET?
11163 054700 001441                BEQ      220$          ;NO!!
11164 054702 013737 001344 001142      MOV      RMER1I,$BDDAT      ;RECEIVED STATUS
11165 054710 013737 001344 001140      MOV      RMER1I,$GDDAT      ;EXPECTED STATUS
11166 054716 052737 004000 001140      BIS      #WLE,$GDDAT
11167 054724 062716 000004                ADD      #4,(SP)          ;MOVE SP TO USERS ERROR CALL
11168 054730 112776 000315 000000      MOV      #315,2(SP)        ;WRITE ERROR NUMBER IN CALL
11169 054736 032737 004000 001342      BIT      #WRL,RMDSI      ;WAS DRIVE WRITE PROTECTED?
11170 054744 001404                BEQ      205$          ;NO!!
11171 054746 032737 000010 001400      BIT      #BIT3,RMCS10      ;WAS COMMAND A WRITE?
11172 054754 001406                BEQ      210$          ;YES!!
11173 054756 112776 000316 000000      MOV      #316,2(SP)        ;CHANGE ERROR NUMBER
11174 054764 042737 004000 001140      BIC      #WLE,$GDDAT
11175 054772 162716 000002                SUB      #2,(SP)          ;MOVE SP TO RETURN IF ERROR
11176 054776 004736                JSR      PC,2(SP)+      ;REPORT ERROR AND RETURN
11177 055000 162716 000010                SUB      #10,(SP)        ;MOVE SP TO NO ERROR
11178
11179 055004                220$:
11180
11181      ;OMIT DATA ERROR CHECKS IF ANY PREVIOUS ERRORS HAVE BEEN DETECTED
11182 055004 062716 000004                ADD      #4,(SP)          ;MOVE SP TO USER'S ERROR
11183 055010 105776 000000                TSTB      2(SP)          ;WAS ERROR DETECTED?
11184 055014 001404                BEQ      225$          ;NO - DO DATA CHECKS
11185 055016 162716 000004                SUB      #4,(SP)          ;RESTORE SP
11186 055022 000137 056024                JMP      340$          ;SKIP DATA CHECKS
11187 055026 162716 000004                SUB      #4,(SP)          ;RESTORE SP
11188
11189      ;CHECK HEADER ERRORS IF FUNCTION WAS NOT WRITE HEADER AND DATA, AND
11190      ;IF HEADER COMPARE IS NOT INHIBITED

```

CZRMD80 RM03/2 FCTNL TST 2
CZRMOB.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 229
DATA TRANSFER COMMAND STATUS CHECK SUBROUTINE

SEQ 0229

```

11191 055032 013737 001400 056414      MOV      RMCS10,510$      ;STRIP AND STORE FUNCTION CODE
11192 055040 042737 177700 056414      BIC      #CFNCMSK,510$
11193 055046 022737 000063 056414      CMP      #WH!GO,510$      ;WAS FUNCTION WRITE HEADER & DATA??
11194 055054 001512          BEQ      250$      ;YES - SKIP HEADER CHECKS
11195 055056 032737 002000 001362      BIT      #HCI,RMOFI      ;WAS HCI SET??
11196 055064 001105          BNC      250$      ;YES - SKIP HEADER CHECKS
11197
11198          ;SEE IF ANY HEADER ERRORS ARE SET, I.E., "FER" OR "HCRC" OR "HCE"
11199 055066 032737 000620 001344      BIT      #HCRC!FER!HCE,RMER11
11200 055074 001533          BEQ      270$      ;NO ERRORS SET
11201
11202          ;REPORT HEADER CRC ERROR IF SET
11203 055076 032737 000400 001344      BIT      #HCRC,RMER11      ;WAS HCRC SET??
11204 055104 001422          BEQ      230$      ;NO!!
11205 055106 013737 001344 001140      MOV      RMER11,$GDDAT      ;EXPECTED STATUS
11206 055114 042737 000400 001140      BIC      #HCRC,$GDDAT
11207 055122 013737 001344 001142      MOV      RMER11,$BDDAT      ;RECEIVED STATUS
11208 055130 062716 000004          ADD      #4,(SP)      ;MOVE SP TO USERS ERROR
11209 055134 112776 000317 000000      MOV      #317,2(SP)      ;WRITE ERROR NUMBER
11210 055142 162716 000002      SUB      #2,(SP)      ;MOVE SP TO RETURN IF ERROR
11211 055146 004736          JSR      PC,2(SP)+      ;REPORT ERROR AND RETURN
11212 055150 000501          BR      260$
11213 055152          230$:
11214
11215          ;REPORT FORMAT ERROR IF SET
11216 055152 032737 000020 001344      BIT      #FER,RMER11      ;WAS "FER" SET??
11217 055160 001422          BEQ      240$      ;NO!!
11218 055162 013737 001344 001140      MOV      RMER11,$GDDAT      ;EXPECTED STATUS
11219 055170 042737 000020 001140      BIC      #FER,$GDDAT
11220 055176 013737 001344 001142      MOV      RMER11,$BDDAT      ;RECEIVED STATUS
11221 055204 062716 000004          ADD      #4,(SP)      ;MOVE SP TO USERS ERROR
11222 055210 112776 000320 000000      MOV      #320,2(SP)      ;WRITE ERROR NUMBER
11223 055216 162716 000002      SUB      #2,(SP)      ;MOVE SP TO RETURN IF ERROR
11224 055222 004736          JSR      PC,2(SP)+      ;REPORT ERROR AND RETURN
11225 055224 000453          BR      260$
11226 055226          240$:
11227
11228          ;REPORT HEADER COMPARE ERROR IF SET
11229 055226 032737 000200 001344      BIT      #HCE,RMER11      ;WAS "HCE" SET??
11230 055234 001453          BEQ      270$      ;NO!!
11231 055236 013737 001344 001140      MOV      RMER11,$GDDAT      ;EXPECTED STATUS
11232 055244 042737 000200 001140      BIC      #HCE,$GDDAT
11233 055252 013737 001344 001142      MOV      RMER11,$BDDAT      ;RECEIVED STATUS
11234 055260 062716 000004          ADD      #4,(SP)      ;MOVE SP TO USER'S ERROR
11235 055264 112776 000321 000000      MOV      #321,2(SP)      ;WRITE ERROR NUMBER
11236 055272 162716 000002      SUB      #2,(SP)      ;MOVE SP TO RETURN IF ERROR
11237 055276 004736          JSR      PC,2(SP)+      ;REPORT ERROR AND RETURN
11238 055300 000425          BR      260$
11239
11240          ;THERE SHOULD BE NO HEADER ERRORS BECAUSE
11241          ;.COMMAND WAS WRITE HEADER AND DATA, OR
11242          ;.HEADER COMPARE INHIBIT WAS SET
11243 055302 032737 000620 001344      BIT      #HCE!FER!HCRC,RMER11
11244 055310 001425          BEQ      250$      ;NO ERRORS WERE SET
11245 055312 013737 001344 001140      MOV      RMER11,$GDDAT      ;EXPECTED STATUS
11246 055320 042737 000620 001140      BIC      #HCE!FER!HCRC,$GDDAT
11247 055326 013737 001344 001142      MOV      RMER11,$BDDAT      ;RECEIVED STATUS

```

K02

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19MACY11 30(1046) 23-NOV-77 12:38 PAGE 230
DATA TRANSFER COMMAND STATUS CHECK SUBROUTINE

SEQ 0230

```

11247 055334 062716 000004      ADD    #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
11248 055340 112776 000322 000000      MOV    #322,2(SP)    ;WRITE ERROR NUMBER
11249 055346 162716 000002      SUB    #2,(SP)      ;MOVE SP TO RETURN IF ERROR
11250 055352 004736      JSR    PC,2(SP)+    ;REPORT ERROR AND RETURN
11251 055354 162716 000010 260$:      SUB    #10,(SP)    ;MOVE SP TO NO ERROR
11252 055360 000137 056024      JMP    340$        ;OMIT FURTHER DATA CHECKS
11253
11254 055364      270$:
11255
11256 ;IF COMMAND WAS A WRITE COMMAND, GO DO WRITE ERROR CHECKS, OTHERWISE
11257 ;DO READ ERROR CHECKS
11258 055364 032737 000010 056414      BIT    #BIT3,510$    ;WAS THIS A WRITE COMMAND?
11259 055372 001002      BNE    275$        ;NO!!
11260 055374 000137 055612      JMP    310$        ;GO DO WRITE STATUS CHECK
11261 055400
11262
11263 ;REPORT DATA CHECK IF SET
11264 055400 032737 100000 001344      BIT    #DCK,RMER11    ;DATA CHECK ERROR??
11265 055406 001450      BEQ    290$        ;NO!!
11266 055410 013737 001344 001140      MOV    RMER11,$GDDAT  ;EXPECTED STATUS
11267 055416 042737 100000 001140      BIC    #DCK,$GDDAT
11268 055424 013737 001344 001142      MOV    RMER11,$BDDAT  ;RECEIVED STATUS
11269 055432 062716 000004      ADD    #4,(SP)      ;MOVE SP TO USER'S ERROR
11270 055436 112776 000323 000000      MOV    #323,2(SP)    ;WRITE ERROR NUMBER
11271 055444 032737 004000 001362      BIT    #EC1,RMOFI    ;WAS ECC CORRECTION DISABLED??
11272 055452 001021      BNE    280$        ;YES!!
11273 055454 112776 000324 000000      MOV    #324,2(SP)    ;CHANGE TO RECOVERABLE ERROR
11274 055462 032737 000100 001344      BIT    #ECH,RMER11    ;IS ERROR RECOVERABLE??
11275 055470 001007      BNE    276$        ;NO!!
11276
11277 ;DO NOT REPORT RECOVERABLE ERROR IF READ COMMAND
11278 055472 032737 000020 056414      BIT    #BIT4,510$    ;WAS THIS A READ COMMAND ??
11279 055500 001406      BEQ    280$        ;NO!!
11280 055502 162716 000004      SUB    #4,(SP)      ;RESTORE SP
11281 055506 000410 000000 276$:      BR     290$        ;SKIP ERROR - DATA WILL BE CORRECTED
11282 055510 112776 000325 000000 280$:      MOV    #325,2(SP)    ;CHANGE TO NON RECOVERABLE
11283 055516 162716 000002      SUB    #2,(SP)      ;MOVE SP TO RETURN IF ERROR
11284 055522 004736      JSR    PC,2(SP)+    ;REPORT ERROR AND RETURN
11285 055524 162716 000010      SUB    #10,(SP)    ;RESTOR SP TO NO ERROR
11286 055530
11287
11288 ;REPORT DATA BUS PARITY ERROR IF SET, I.E. MOPE = 1
11289 055530 032737 000400 001340      BIT    #MOPE,RMCS2I    ;PARITY ERROR SET??
11290 055536 001423      BEQ    300$        ;NO!!
11291 055540 013737 001340 001140      MOV    RMCS2I,$GDDAT  ;EXPECTED STATUS
11292 055546 042737 000400 001140      BIC    #MOPE,$GDDAT
11293 055554 013737 001340 001142      MOV    RMCS2I,$BDDAT  ;RECEIVED STATUS
11294 055562 062716 000004      ADD    #4,(SP)      ;MOVE SP TO USER'S ERROR
11295 055566 112776 000326 000000      MOV    #326,2(SP)    ;WRITE ERROR NUMBER
11296 055574 162716 000002      SUB    #2,(SP)      ;MOVE SP TO RETURN IF ERROR
11297 055600 004736      JSR    PC,2(SP)+    ;REPORT ERROR AND RETURN
11298 055602 162716 000010      SUB    #10,(SP)    ;MOVE SP TO NO ERROR
11299 055606 000137 056024 300$:      JMP    340$        ;SKIP WRITE STATUS CHECK
11300
11301 055612      310$:
11302

```



```

11303 ;TEST TO SEE THAT OFFSET MODE WAS RESET; REPORT ERROR IF "OM" = 1
11304 055612 032737 000001 001342 BIT #OM,RMSI ;IS OFFSET ON??
11305 055620 001423 BEQ 320$ ;NO
11306 055622 013737 001342 001140 MOV RMSI,$GDDAT ;EXPECTED STATUS
11307 055630 042737 000001 001140 BIC #OM,$GDDAT
11308 055636 013737 001342 001142 MOV RMSI,$BDDAT ;RECEIVED STATUS
11309 055644 062716 000001 000000 ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11310 055650 112776 000327 000000 MOVB #327,2(SP) ;WRITE ERROR NUMBER IN CALL
11311 055656 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN IF ERROR
11312 055662 004736 JSR PC,2(SP)+ ;REPORT ERROR AND RETURN
11313 055664 162716 000010 SUB #10,(SP) ;MOVE SP TO NO ERROR
11314 055670 320$:
11315 ;TEST FOR DATA BUS PARITY ERROR; REPORT ERROR IF "DPE" = 1
11316 055670 032737 000010 001372 BIT #DPE,RMER2I ;DATA PARITY ERROR??
11317 055676 001423 BEQ 330$ ;NO!!
11318 055700 013737 001372 001140 MOV RMER2I,$GDDAT ;EXPECTED STATUS
11319 055706 042737 000010 001140 BIC #DPE,$GDDAT
11320 055714 013737 001372 001142 MOV RMER2I,$BDDAT ;RECEIVED STATUS
11321 055722 062716 000004 000000 ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11322 055726 112776 000330 000000 MOVB #330,2(SP) ;WRITE ERROR NUMBER
11323 055734 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN IF ERROR
11324 055740 004736 JSR PC,2(SP)+ ;REPORT ERROR AND RETURN
11325 055742 162716 000010 SUB #10,(SP) ;MOVE SP TO NO ERROR
11326 055746 330$:
11327 ;TEST FOR WRITE CLOCK FAILURE; REPORT ERROR IF "WCF" = 1
11328 055746 032737 000040 001344 BIT #WCF,RMER1I ;IS "WCF" SET??
11329 055754 001423 BEQ 340$ ;NO!!
11330 055756 013737 001344 001140 MOV RMER1I,$GDDAT ;EXPECTED STATUS
11331 055764 042737 000040 001140 BIC #WCF,$GDDAT
11332 055772 013737 001344 001142 MOV RMER1I,$BDDAT ;RECEIVED STATUS
11333 056000 062716 000004 000000 ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11334 056004 112776 000331 000000 MOVB #331,2(SP) ;WRITE ERROR NUMBER
11335 056012 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN IF ERROR
11336 056016 004736 JSR PC,2(SP)+ ;REPORT ERROR AND RETURN
11337 056020 162716 000010 SUB #10,(SP) ;MOVE SP TO NO ERROR
11338 056024 340$:
11339 ;REPORT "DATA LATE" ERROR IF "DLT" = 1
11340 056024 032737 100000 001340 BIT #DLT,RMCS2I ;IS "DLT" SET??
11341 056032 001423 BEQ 350$ ;NO!!
11342 056034 013737 001340 001140 MOV RMCS2I,$GDDAT ;EXPECTED STATUS
11343 056042 042737 100000 001140 BIC #DLT,$GDDAT
11344 056050 013737 001340 001142 MOV RMCS2I,$BDDAT ;RECEIVED STATUS
11345 056056 062716 000004 000000 ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11346 056062 112776 000332 000000 MOVB #332,2(SP) ;WRITE ERROR NUMBER
11347 056070 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN IF ERROR
11348 056074 004736 JSR PC,2(SP)+ ;REPORT ERROR AND RETURN
11349 056076 162716 000010 SUB #10,(SP) ;MOVE SP TO NO ERROR
11350 056102 350$:
11351 ;LOOK FOR UNEXPECTED CHANGES IN DRIVE STATUS
11352 056102 013746 001342 001342 MOV RMSI,-(SP) ;STACK DRIVE STATUS
11353 056106 042716 147E77 BIC #1C<PIP!MOL!VV>,(SP) ;CLEAR DONT CARES
11354 056112 022726 010100 CMP #MOL!VV,(SP)+ ;IS DRIVE STATUS OK??
11355 056116 001522 BEQ 380$ ;YES!!

```

```

11359
11360 ;REPORT ERROR IF POSITIONING IN PROGRESS AND NO SEEK INCOMPLETE ERROR,
11361 ;I.E, PIP = 1 AND SKI = 0
11362 056120 032737 020000 001342 BIT #PIP,RMDSI ;IS "PIP" SET??
11363 056126 001430 BEQ 360$ ;NO!!
11364 056130 032737 040000 001372 BIT #SKI,RMER2I ;WAS "SKI" ERROR REPORTED??
11365 056136 001024 BNE 360$ ;YES-DONT REPORT PIP
11366 056140 013737 001342 001140 MOV RMDSI,$GDDAT ;EXPECTED STATUS
11367 056146 042737 020000 001140 BIC #PIP,$GDDAT
11368 056154 013737 001342 001142 MOV RMDSI,$BDDAT ;RECEIVED STATUS
11369 056162 062716 000004 ADD #4,(SP) ;MOVE SP TO USERS ERROR CALL
11370 056166 112776 000333 000000 MOVB #333,2(SP) ;WRITE ERROR NUMBER
11371 056174 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN IF ERROR
11372 056200 004736 JSR PC,2(SP)+ ;REPORT ERROR AND RETURN
11373 056202 162716 000010 SUB #10,(SP) ;MOVE SP TO NO ERROR
11374 056206 000240 NOP
11375 056210
11376
11377 ;REPORT ERROR IF MEDIUM IS NOT ON LINE AND OPI ERROR WAS NOT
11378 ;REPORTED I.E., MOL = OPI = 0
11379 056210 032737 010000 001342 BIT #MOL,RMDSI ;IS MEDIUM ON LINE??
11380 056216 001027 BNE 370$ ;YES!!
11381 056220 032737 020000 001344 BIT #OPI,RMER1I ;WAS OPI ERROR REPORTED??
11382 056226 001023 BNE 370$ ;YES!!
11383 056230 013737 001342 001140 MOV RMDSI,$GDDAT ;EXPECTED STATUS
11384 056236 052737 010000 001140 BIS #MOL,$GDDAT
11385 056244 013737 001342 001142 MOV RMDSI,$BDDAT ;RECEIVED STATUS
11386 056252 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR
11387 056256 112776 000334 000000 MOVB #334,2(SP) ;WRITE ERROR NUMBER
11388 056264 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN IF ERROR
11389 056270 004736 JSR PC,2(SP)+ ;REPORT ERROR AND RETURN
11390 056272 162716 000010 SUB #10,(SP) ;MOVE SP TO NO ERROR
11391 056276
11392
11393 ;REPORT ERROR IF VOLUME IS NOT VALID AND "IVC" ERROR WAS NOT
11394 ;REPORTED I.E., VV = IVC = 0
11395 056276 032737 000100 001342 BIT #VV,RMDSI ;IS VOLUME VALID??
11396 056304 001027 BNE 380$ ;YES!!
11397 056306 032737 010000 001372 BIT #IVC,RMER2I ;WAS IVC ERROR REPORTED??
11398 056314 001033 BNE 390$ ;YES!!
11399 056316 013737 001342 001140 MOV RMDSI,$GDDAT ;EXPECTED STATUS
11400 056324 052737 000100 001140 BIS #VV,$GDDAT
11401 056332 013737 001342 001142 MOV RMDSI,$BDDAT ;RECEIVED STATUS
11402 056340 062716 000004 ADD #4,(SP) ;MOVE SP TO USERS ERROR CALL
11403 056344 112776 000335 000000 MOVB #335,2(SP) ;WRITE ERROR NUMBER
11404 056352 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN IF ERROR
11405 056356 004736 JSR PC,2(SP)+ ;REPORT ERROR AND RETURN
11406 056360 162716 000010 SUB #10,(SP) ;MOVE SP TO NO ERROR
11407 056364
11408
11409 ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS FOUND
11410 056364 062716 000004 ADD #4,(SP) ;MOVE SP TO ERROR CALL
11411 056370 105776 000000 TSTB 2(SP) ;ANY ERROR??
11412 056374 001403 BEQ 390$ ;NO!!
11413 056376 062716 000004 ADD #4,(SP) ;YES - MOVE SP TO ERROR RETURN
11414 056402 000402 BR 400$

```

N02

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 233
DATA TRANSFER COMMAND STATUS CHECK SUBROUTINE

SEQ 0233

11415	056404	162716	000004	390\$:	SUB	#4,(SP)	;MOVE SP TO NO ERROR RETURN
11416							
11417	056410	000207		400\$:	RTS	PC	;RETURN TO USER
11418							
11419	056412	000000		500\$:	.WORD		;ERROR FLAGS
11420	056414	000000		510\$:	.WORD		;TEMPORARY STORAGE

```

11421 .SBTTL  STATIC DRIVE STATUS CHECK SUBROUTINE
11422
11423 ;THIS SUBROUTINE LOOKS FOR UNEXPECTED CHANGES IN DRIVE
11424 ;STATUS, CH AS THE DRIVE LOSING VOLUME VALID. THE SUBROUTINE
11425 ;CAN BE USED BY HOUSEKEEPING AND OTHER COMMANDS DURING WHICH THERE
11426 ;SHOULD NO. BE ANY DRIVE ERRORS OR CHANGES IN STATE.
11427
11428 ;THE FOLLOWING CONDITIONS ARE TESTED AND REPORTED AS ERRORS
11429 ;IF TRUE:
11430
11431 ;.MOL = 0, INDICATES DRIVE WENT OFFLINE, NOTE
11432 ;THAT MOL IS ASSUMED TO HAVE BEEN SET
11433 ;.VV = 0, INDICATES THE DRIVE LOST VOLUME VALID
11434 ;.PIP = 1, INDICATES THAT THE DRIVE IS OFF CYLINDER
11435 ;.SKI = 1, INDICATES THE DRIVE HAS AN UNEXPECTED SKI ERROR
11436 ;.DVC = 1, INDICATES AN UNEXPECTED DEVICE FAULT
11437
11438 ;THE SUBROUTINE IS CALLED AFTER STORING STATUS IN THE GET BUFFER.
11439
11440 ;(1) JSR PC,STCDRVSTS
11441 BR ??? RETURN HERE IF NO ERROR
11442 NOP RETURN HERE TO REPORT AN ERROR
11443 ERROR ERROR NUMBER DEFINED BY SUB
11444 JSR PC,2(SP)+ GO BACK TO SUB FOR MORE ERROR CHECKS
11445 ??? RETURN HERE IF NO MORE ERRORS
11446
11447 056416 STCDRVSTS:
11448
11449 ;CLEAR USER'S ERROR CALL
11450 ADD #4(SP) ;MOVE SP TO USER'S ERROR CALL
11451 CLR B 2(SP) ;CLEAR ERROR NUMBER
11452 SUB #4(SP) ;MOVE SP BACK TO NO ERROR RETURN
11453
11454 ;SEE IF "MOL" = "VV" = 1 AND "PIP" = 0
11455 MOV RMO SI, -(SP) ;PUT DRIVE STATUS ON STACK
11456 BIC #1C(PIP!MOL!VV), (SP)
11457 CMP #MOL!VV, (SP)+ ;ARE MOL, VV AND PIP O.K.??
11458 BEQ 30$ ;YES!!
11459
11460 ;REPORT AN ERROR IF MOL = 0 AND "OPI" = 0
11461 BIT #MOL, RMO SI ;IS MOL ON ??
11462 BNE 10$ ;YES!!
11463 BIT #OPI, RMER11 ;WAS "OPI" SET??
11464 BNE 10$ ;YES-DONT REPORT "MOL" = 0
11465 MOV RMO SI, $GDDAT ;EXPECTED STATUS
11466 BIS #MOL, $GDDAT
11467 MOV RMO SI, $BDDAT ;RECEIVED STATUS
11468 ADD #4(SP) ;MOVE SP TO USER'S ERROR CALL
11469 MOVB #207, 2(SP) ;WRITE ERROR NUMBER IN CALL
11470 SUB #2, (SP) ;MOVE SP TO RETURN FOR ERROR
11471 JSR PC, 2(SP)+ ;REPORT ERROR VIA USER
11472 SUB #10, (SP) ;MOVE SP BACK TO NO ERROR RETURN

```

```

11472 056536 000240      NOP
11473 056540      10$:
11474
11475      ;REPORT AN ERROR IF VOLUME VALID IS NOW ZERO AND "IVC" = 0
11476 056540 032737 000100 001342      BIT      #VV,RMSI      ;IS "VV" = 0??
11477 056546 001030      BNE      20$      ;NO!!
11478 056550 032737 010000 001372      BIT      #IVC,RMER2I    ;WAS "IVC" SET??
11479 056556 001024      BNE      20$      ;YES-DONT REPORT "VV" = 0
11480 056560 013737 001342 001140      MOV      RMSI,$GDDAT    ;EXPECTED STATUS
11481 056566 052737 000100 001342      BIS      #VV,RMSI
11482 056574 013737 001342 001142      MOV      RMSI,$BDDAT    ;RECEIVED STATUS
11483 056602 062716 000004      ADD      #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
11484 056606 112776 000210 000000      MOVB     #210,2(SP)    ;WRITE ERROR NUMBER IN CALL
11485 056614 162716 000002      SUB      #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
11486 056620 004736      JSR      PC,2(SP)+    ;REPORT ERROR VIA USER
11487 056622 162716 000010      SUB      #10,(SP)     ;MOVE SP BACK TO NO ERROR
11488 056626 000240      NOP
11489 056630      20$:
11490
11491      ;REPORT AN ERROR IF DRIVE IS OFF CYLINDER AND "SKI" = 0
11492 056630 032737 020000 001342      BIT      #PIP,RMSI      ;IS DRIVE OFF CYLINDER??
11493 056636 001430      BEQ      30$      ;NO!!
11494 056640 032737 040000 001372      BIT      #SKI,RMER2I    ;WAS "SKI" SET??
11495 056646 001024      BNE      30$      ;YES-DONT REPORT "PIP" = 1
11496 056650 013737 001342 001140      MOV      RMSI,$GDDAT    ;EXPECTED STATUS
11497 056656 042737 020000 001140      BIC      #PIP,$GDDAT
11498 056664 013737 001342 001142      MOV      RMSI,$BDDAT    ;RECEIVED STATUS
11499 056672 062716 000004      ADD      #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
11500 056676 112776 000211 000000      MOVB     #211,2(SP)    ;WRITE ERROR NUMBER IN USER'S CALL
11501 056704 162716 000002      SUB      #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
11502 056710 004736      JSR      PC,2(SP)+    ;REPORT ERROR VIA USER
11503 056712 162716 000010      SUB      #10,(SP)     ;MOVE SP TO NO ERROR RETURN
11504 056716 000240      NOP
11505 056720      30$:
11506
11507      ;SEE IF "SKI" = "DVC" = 0
11508 056720 013746 001372      MOV      RMER2I,-(SP)    ;PUT ERROR REG 2 ON STACK
11509 056724 042726 137577      BIC      #1C(SKI!DVC),(SP)+
11510 056730 001460      BEQ      60$      ;BRANCH IF NO ERROR
11511 056732      40$:
11512
11513      ;REPORT AN ERROR IF THERE IS A DEVICE FAULT
11514 056732 032737 000200 001372      BIT      #DVC,RMER2I    ;ANY DEVICE FAULT??
11515 056740 001424      BEQ      50$      ;NO!!
11516 056742 013737 001372 001140      MOV      RMER2I,$GDDAT    ;EXPECTED STATUS
11517 056750 042737 000200 001140      BIC      #DVC,$GDDAT
11518 056756 013737 001372 001142      MOV      RMER2I,$BDDAT    ;RECEIVED STATUS
11519 056764 062716 000004      ADD      #4,(SP)      ;MOVE SP TO USER'S CALL
11520 056770 112776 000212 000000      MOVB     #212,2(SP)    ;WRITE NUMBER OF ERROR IN CALL
11521 056776 162716 000002      SUB      #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
11522 057002 004736      JSR      PC,2(SP)+    ;REPORT ERROR VIA USER
11523 057004 162716 000010      SUB      #10,(SP)     ;MOVE SP BACK TO NO ERROR
11524 057010 000240      NOP
11525 057012      50$:
11526
11527      ;REPORT AN ERROR IF "SKI" = 1

```

Address	Hex	Hex	Hex	Hex	Assembly	Comment
11528	057012	032737	040000	001372	BIT	#SKI,RMER2I ; IS THERE A SEEK INCOMPLETE ERROR
11529	057020	001424			BEQ	60\$; NO!!
11530	057022	013737	001372	001140	MOV	RMER2I,\$GDDAT ; EXPECTED STATUS
11531	057030	042737	040000	001140	BIC	#SKI,\$GDDAT
11532	057036	013737	001372	001142	MOV	RMER2I,\$BDDAT ; RECEIVED STATUS
11533	057044	062716	000004		ADD	#4,(SP) ; MOVE SP TO USER'S ERROR CALL
11534	057050	112776	000213	000000	MOVB	#213,2(SP) ; WRITE ERROR NUMBER IN USER'S ERROR CALL
11535	057056	162716	000002		SUB	#2,(SP) ; MOVE SP TO RETURN FOR ERROR
11536	057062	004736			JSR	PC,2(SP)+ ; REPORT ERROR VIA USER
11537	057064	162716	000010		SUB	#10,(SP) ; MOVE SP BACK TO NO ERROR
11538	057070	000240			NOP	
11539	057072					60\$:
11540						
11541						; AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
11542	057072	062716	000004		ADD	#4,(SP) ; MOVE SP TO USER'S ERROR CALL
11543	057076	105776	000000		TSTB	2(SP) ; WAS AN ERROR DETECTED??
11544	057102	001403			BEQ	70\$; NO!!
11545	057104	062716	000004		ADD	#4,(SP) ; YES - MOVE SP TO USER'S ERROR RETURN
11546	057110	000402			BR	80\$
11547	057112	162716	000004		SUB	#4,(SP) ; NO - MOVE SP TO NO ERROR RETURN
11548	057116	000240			NOP	
11549	057120	000207			RTS	PC ; RETURN TO USER

```

11550      .SBTTL  COMPOSITE ERROR CHECK SUBROUTINE
11551
11552      ; THIS SUBROUTINE CHECKS THE STORED CONTENTS OF RMER1 AND
11553      ; RMER2 AFTER MASKING EACH REGISTER WORD WITH THE USER'S STATUS
11554      ; MASKS AND REPORTS AN ERROR IF ANY BITS ARE LEFT ON AFTER
11555      ; THE MASKS ARE APPLIED.
11556
11557      ; CALL:
11558      ; (1)      JSR      PC, CMPERRSTS
11559      ;           .WORD    MASK FOR ERROR REGISTER 1
11560      ;           .WORD    MASK FOR ERROR REGISTER 2
11561      ;           BR      ???
11562      ;           NOP
11563      ;           ERPOR
11564      ;           JSR      PC, @ (SP)+
11565      ;           ???
11566      ; RETURN HERE IF NO ERROR
11567      ; RETURN HERE TO REPORT AN ERROR
11568      ; ERROR NUMBER DEFINED BY SUB
11569      ; GO BACK TO SUB FOR MORE ERROR CHECKS
11570      ; RETURN HERE IF NO MORE ERRORS
11571
11572      ; NOTE:  BITS TO BE MASKED SHOULD BE ONE; BITS TO BE TESTED SHOULD
11573      ; BE ZERO
11574
11575      CMPERRSTS:
11576      ; MASK AND STORE THE CONTENTS OF RMER1 AND RMER2
11577      MOV      RMER1, $TMP1      ; STORE RMER1 AT TEMP STORAGE
11578      BIC      @ (SP), $TMP1     ; MASK RMER1
11579      ADD      #2, (SP)          ; MOVE SP TO NEXT MASK
11580      MOV      RMER2, $TMP2     ; STORE RMER2 AT TEMP STORAGE
11581      BIC      @ (SP), $TMP2     ; MASK RMER2
11582
11583      ; CLEAR USER'S ERROR CALL
11584      ADD      #6, (SP)          ; MOVE SP TO USER'S ERROR CALL
11585      CLRB     @ (SP)           ; CLEAR ERROR NUMBER
11586      SUB      #4, (SP)         ; LEAVE SP AT NO ERROR RETURN
11587
11588      ; SEE IF THERE WERE ANY ERRORS IN RMER1, I.E. $TMP1
11589      TST      $TMP1           ; ANY ERRORS TO REPORT??
11590      BEQ      SS              ; NO !!
11591      MOV      $TMP1, $BDDAT    ; RECEIVED STATUS FOR TYPEOUT
11592      CLR      $GDDAT          ; EXPECTED STATUS FOR TYPEOUT
11593      ADD      #4, (SP)         ; MOVE SP TO USER'S ERROR CALL
11594      MOVB     #66, @ (SP)      ; CORRECTABLE DATA CHECK ERROR #
11595      SUB      #2, (SP)         ; MOVE SP TO RETURN FOR ERROR
11596      JSR      PC, @ (SP)+      ; REPORT ERROR VIA USER
11597      SUB      #10, (SP)        ; MOVE SP BACK TO BRANCH
11598      NOP
11599      SS:
11600
11601      ; SEE IF THERE ARE ANY ERRORS TO REPORT IN RMER2 ($TMP2)
11602      TST      $TMP2           ; ANY ERRORS IN RMER2?
11603      BEQ      10$            ; NO !!
11604      MOV      $TMP2, $BDDAT    ; RECEIVED STATUS FOR TYPEOUT
11605      CLR      $GDDAT          ; EXPECTED STATUS FOR TYPEOUT
11606      ADD      #4, (SP)         ; MOVE SP TO USER'S ERROR CALL

```

CZRMDB0 RM03/2 FCTNL TST 2
CZRMDB.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 238
COMPOSITE ERROR CHECK SUBROUTINE

SEQ 0238

11606	057264	112776	000067	000000	MOV8	#67, 2(SP)	;WRITE ERROR NUMBER IN USER'S CALL
11607	057272	162716	000002		SUB	#2, (SP)	;MOVE SP TO RETURN FOR ERROR
11608	057276	004736			JSR	PC, 2(SP)+	;REPORT ERROR VIA USER
11609	057300	162716	000010		SUB	#10, (SP)	;MOVE SP TO NO ERROR RETURN
11610	057304	000240			NOP		
11611	057306						
11612							
11613							
11614	057306	062716	000004		ADD	#4, (SP)	;MOVE SP TO USER'S ERROR CALL
11615	057312	105776	000000		TSTB	2(SP)	;WAS THERE AN ERROR CALLED??
11616	057316	001403			BEQ	20\$;NO!!
11617	057320	062716	000004		ADD	#4, (SP)	;YES - MOVE SP TO ERROR RETURN
11618	057324	000402			BR	30\$	
11619	057326	162716	000004	20\$:	SUB	#4, (SP)	;MOVE SP TO NO ERROR RETURN
11620	057332	000207		30\$:	RTS	PC	;RETURN TO USER
11621							
11622							

CZRMDBO RMO3/2 FCTNL TST 2
CZRMDB.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 239
STOP AND SHUTDOWN SUBROUTINES

SEQ 0239

```

11623 .SBTTL STOP AND SHUTDOWN SUBROUTINES
11624
11625 057334 STOP:
11626
11627 ;DROP PRIORITY TO ALLOW CONSOLE INTERRUPT
11628 057334 012746 000140      MOV      #PR3,-(SP)      ;;PUT NEW PS ON STACK
11629 057340 012746 057346      MOV      #64$,-(SP)      ;;PUT NEW PC ON STACK
11630 057344 000002      RTI              ;;POP NEW PC AND PS
11631 057346
11632 057346 000240      64$:      NOP
11633 ;RAISE PRIORITY TO INHIBIT INTERRUPT
11634 057350 012746 000300      MOV      #PR6,-(SP)      ;;PUT NEW PS ON STACK
11635 057354 012746 057362      MOV      #65$,-(SP)      ;;PUT NEW PC ON STACK
11636 057360 000002      RTI              ;;POP NEW PC AND PS
11637 057362
11638
11639 057362 000207      65$:
11640
11641      10$:      RTS      PC      ;CONTINUE
11642
11643 057364      SHUT:
11644 057364 005737 001326      TST      @CTLFG      ;HALT ?
11645 057370 001002      BNE      .+6      ;BRANCH IF YES
11646 057372 000137 007124      JMP      READY      ;CONTINUE
11647 057376 104401 057420      TYPE      ,100$      ;TYPE THE HALT MESSAGE
11648 057402 005737 000042      TST      42      ;RUNNING STANDALONE ??
11649 057406 001402      BEQ      10$      ;YES !!
11650 057410 000137 032766      JMP      $EOP      ;NO - GO TO END OF PASS
11651 057414
11652 057420 000137 005324      10$:      JMP      START      ;GO TO START
11653 057426 042524 052123 053440 100$:      .ASCIZ  @TEST WAS HALTED@<CR><LF><LF>
11654 057434 042524 006504 046101
11655 057442 000
11656 057444      .EVEN

```

C2RM080 RM03/2 FCTNL TST 2
C2RM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 240
STOP AND SHUTDOWN SUBROUTINES

SEQ 0240

```

11657 057444 012737 177777 001326 SHUT2: MOV      # -1, CTFLG      ; SET THE CONTROL-C FLAG
11658 057452 000002                      RTI                      ; EXIT FROM INTERRUPT
11659                      .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
11660
11661                      ; *****
11662                      ; *SAVE R0-R5
11663                      ; *CALL:
11664                      ; *      SAVREG
11665                      ; *UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
11666                      ; *
11667                      ; *TOP---(+16)
11668                      ; * +2---(+18)
11669                      ; * +4---R5
11670                      ; * +6---R4
11671                      ; * +8---R3
11672                      ; * +10---R2
11673                      ; * +12---R1
11674                      ; * +14---R0
11675
11676 057454                      $SAVREG:
11677 057454 010046                      MOV      R0, -(SP)          ;: PUSH R0 ON STACK
11678 057456 010146                      MOV      R1, -(SP)          ;: PUSH R1 ON STACK
11679 057460 010246                      MOV      R2, -(SP)          ;: PUSH R2 ON STACK
11680 057462 010346                      MOV      R3, -(SP)          ;: PUSH R3 ON STACK
11681 057464 010446                      MOV      R4, -(SP)          ;: PUSH R4 ON STACK
11682 057466 010546                      MOV      R5, -(SP)          ;: PUSH R5 ON STACK
11683 057470 016646 000022              MOV      22(SP), -(SP)      ;: SAVE PS OF MAIN FLOW
11684 057474 016646 000022              MOV      22(SP), -(SP)      ;: SAVE PC OF MAIN FLOW
11685 057500 016646 000022              MOV      22(SP), -(SP)      ;: SAVE PS OF CALL
11686 057504 016646 000022              MOV      22(SP), -(SP)      ;: SAVE PC OF CALL
11687 057510 000002                      RTI
11688
11689                      ; *RESTORE R0-R5
11690                      ; *CALL:
11691                      ; *      RESREG
11692 057512                      $RESREG:
11693 057512 012666 000022              MOV      (SP)+, 22(SP)      ;: RESTORE PC OF CALL
11694 057516 012666 000022              MOV      (SP)+, 22(SP)      ;: RESTORE PS OF CALL
11695 057522 012666 000022              MOV      (SP)+, 22(SP)      ;: RESTORE PC OF MAIN FLOW
11696 057526 012666 000022              MOV      (SP)+, 22(SP)      ;: RESTORE PS OF MAIN FLOW
11697 057532 012605                      MOV      (SP)+, R5          ;: POP STACK INTO R5
11698 057534 012604                      MOV      (SP)+, R4          ;: POP STACK INTO R4
11699 057536 012603                      MOV      (SP)+, R3          ;: POP STACK INTO R3
11700 057540 012602                      MOV      (SP)+, R2          ;: POP STACK INTO R2
11701 057542 012601                      MOV      (SP)+, R1          ;: POP STACK INTO R1
11702 057544 012600                      MOV      (SP)+, R0          ;: POP STACK INTO R0
11703 057546 000002                      RTI
11704                      .SBTTL  BINARY TO ASCII AND TYPE ROUTINE
11705
11706                      ; *****
11707                      ; *THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 16-BIT
11708                      ; *BINARY-ASCII NUMBER AND TYPE IT.
11709                      ; *CALL:
11710                      ; *      MOV      NUMBER, -(SP)      ;: NUMBER TO BE TYPED
11711                      ; *      TYPBN                      ;: TYPE IT
11712

```

```

11713 057550 010146          $TYPBN: MOV R1, -(SP)          ;; SAVE R1 ON THE STACK
11714 057552 016601 000006    MOV 6(SP), R1          ;; GET THE INPUT NUMBER
11715 057556 000261          SEC                      ;; SET "C" SO CAN KEEP TRACK OF THE NUMBER OF BITS
11716 057560 112737 000060 057622 1$: MOVB #'0, $BIN    ;; SET CHARACTER TO AN ASCII "0".
11717 057566 006101          ROL R1                  ;; GET THIS BIT
11718 057570 001406          BEQ 2$                  ;; DONE?
11719 057572 105537 057622    ADCB $BIN              ;; NO--SET THE CHARACTER EQUAL TO THIS BIT
11720 057576 104401 057622    TYPE , $BIN            ;; GO TYPE THIS BIT
11721 057602 000241          CLC                      ;; CLEAR "C" SO CAN KEEP TRACK OF BITS
11722 057604 000765          BR 1$                    ;; GO DO THE NEXT BIT
11723 057606 012601          MOV (SP)+, R1            ;; POP THE STACK INTO R1
11724 057610 016666 000002 000004 MOV 2(SP), 4(SP)    ;; ADJUST THE STACK
11725 057616 012616          MOV (SP)+, (SP)
11726 057620 000002          RTI                      ;; RETURN TO USER
11727 057622 000      000    $BIN: .BYTE 0, 0          ;; STORAGE FOR ASCII CHAR. AND TERMINATOR
11728          .SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
11729
11730          ;*****
11731          ;THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
11732          ;SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
11733          ;NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
11734          ;BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
11735          ;REPLACED WITH SPACES.
11736          ;CALL:
11737          ;      MOV NUM, -(SP)          ;; PUT THE BINARY NUMBER ON THE STACK
11738          ;      TYPDS                  ;; GO TO THE ROUTINE
11739
11740          $TYPDS: MOV R0, -(SP)          ;; PUSH R0 ON STACK
11741 057624 010046          MOV R1, -(SP)          ;; PUSH R1 ON STACK
11742 057626 010146          MOV R2, -(SP)          ;; PUSH R2 ON STACK
11743 057630 010246          MOV R3, -(SP)          ;; PUSH R3 ON STACK
11744 057632 010346          MOV R5, -(SP)          ;; PUSH R5 ON STACK
11745 057634 010546          MOV #20200, -(SP)      ;; SET BLANK SWITCH AND SIGN
11746 057636 012746 020200    MOV 20(SP), R5        ;; GET THE INPUT NUMBER
11747 057642 016605 000020    BPL 1$                ;; BR IF INPUT IS POS.
11748 057646 100004          NEG R5                  ;; MAKE THE BINARY NUMBER POS.
11749 057650 005405          MOVB #'-', 1(SP)        ;; MAKE THE ASCII NUMBER NEG.
11750 057652 112766 000055 000001 1$: CLR R0        ;; ZERO THE CONSTANTS INDEX
11751 057660 005000          MOV #5DBLK, R3        ;; SETUP THE OUTPUT POINTER
11752 057662 012703 060040    MOVB #' ', (R3)+      ;; SET THE FIRST CHARACTER TO A BLANK
11753 057666 112723 000040    CLR R2                ;; CLEAR THE BCD NUMBER
11754 057672 005002          MOV $OTBL(R0), R1      ;; GET THE CONSTANT
11755 057674 016001 060030    SUB R1, R5            ;; FORM THIS BCD DIGIT
11756 057700 160105          BLT 4$                  ;; BR IF DONE
11757 057702 002402          INC R2                 ;; INCREASE THE BCD DIGIT BY 1
11758 057704 005202          BR 3$
11759 057706 000774          ADD R1, R5              ;; ADD BACK THE CONSTANT
11760 057710 060105          TST R2                  ;; CHECK IF BCD DIGIT=0
11761 057712 005702          BNE 5$                  ;; FALL THROUGH IF 0
11762 057714 001002          TSTB (SP)              ;; STILL DOING LEADING 0'S?
11763 057716 105716          BMI 7$                  ;; BR IF YES
11764 057720 100407          ASLB (SP)              ;; MSD?
11765 057722 106316          BCC 6$                  ;; BR IF NO
11766 057724 103003          MOVB 1(SP), -1(R3)     ;; YES--SET THE SIGN
11767 057726 116363 000001 177777 6$: MOVB #'0, R2  ;; MAKE THE BCD DIGIT ASCII
11768 057734 052702 000060

```

```

11769 057740 052702 000040      7$: BIS      #' R2      ;; MAKE IT A SPACE IF NOT ALREADY A DIGIT
11770 057744 110223              MOV      R2, (R3)+    ;; PUT THIS CHARACTER IN THE OUTPUT BUFFER
11771 057746 005720              TST      (R0)+        ;; JUST INCREMENTING
11772 057750 020027 000010      CMP      R0, #10      ;; CHECK THE TABLE INDEX
11773 057754 002746              BLT      2$          ;; GO DO THE NEXT DIGIT
11774 057756 003002              BGT      8$          ;; GO TO EXIT
11775 057760 010502              MOV      R5, R2        ;; GET THE LSD
11776 057762 000764              BR       6$          ;; GO CHANGE TO ASCII
11777 057764 105726      8$: TST      (SP)+        ;; WAS THE LSD THE FIRST NON-ZERO?
11778 057766 100003              BPL      9$          ;; BR IF NO
11779 057770 116663 177777 177776 9$: MOV      -1(SP), -2(R3)  ;; YES--SET THE SIGN FOR TYPING
11780 057776 105013              CLRB      (R3)        ;; SET THE TERMINATOR
11781 060000 012605              MOV      (SP)+, R5    ;; POP STACK INTO R5
11782 060002 012603              MOV      (SP)+, R3    ;; POP STACK INTO R3
11783 060004 012602              MOV      (SP)+, R2    ;; POP STACK INTO R2
11784 060006 012601              MOV      (SP)+, R1    ;; POP STACK INTO R1
11785 060010 012600              MOV      (SP)+, R0    ;; POP STACK INTO R0
11786 060012 104401 060040      TYPE      $DBLK      ;; NOW TYPE THE NUMBER
11787 060016 016666 000002 000004  MOV      2(SP), 4(SP)  ;; ADJUST THE STACK
11788 060024 012616              MOV      (SP)+, (SP)
11789 060026 000002              RTI
11790 060030 023420      $DTBL: 10000.
11791 060032 001750              1000.
11792 060034 000144              100.
11793 060036 000012              10.
11794 060040 000004      $DBLK: .BLKW      4
11795                      .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
11796
11797                      ;*****
11798                      ;THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
11799                      ;OCTAL (ASCII) NUMBER AND TYPE IT.
11800                      ;$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
11801                      ;CALL:
11802                      ;      MOV      NUM, -(SP)      ;; NUMBER TO BE TYPED
11803                      ;      TYPOS      ;; CALL FOR TYPEOUT
11804                      ;      .BYTE      N            ;; N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
11805                      ;      .BYTE      M            ;; M=1 OR 0
11806                      ;                      ;; 1=TYPE LEADING ZEROS
11807                      ;                      ;; 0=SUPPRESS LEADING ZEROS
11808
11809                      ;$STYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
11810                      ;$TYPOS OR $TYPOC
11811                      ;CALL:
11812                      ;      MOV      NUM, -(SP)      ;; NUMBER TO BE TYPED
11813                      ;      TYPON      ;; CALL FOR TYPEOUT
11814
11815                      ;$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
11816                      ;CALL:
11817                      ;      MOV      NUM, -(SP)      ;; NUMBER TO BE TYPED
11818                      ;      TYPOC      ;; CALL FOR TYPEOUT
11819
11820 060050 017646 000000 060273 $TYPOS: MOV      2(SP), -(SP)  ;; PICKUP THE MODE
11821 060054 116637 000001      MOV      1(SP), $OFILL  ;; LOAD ZERO FILL SWITCH
11822 060062 112637 060275      MOV      (SP)+, $OMODE+1  ;; NUMBER OF DIGITS TO TYPE
11823 060066 062716 000002      ADD      #2, (SP)      ;; ADJUST RETURN ADDRESS
11824 060072 000406      BR       $STYPON

```

```

11825 060074 112737 000001 060273 $TYPOC: MOVB #1,$OFILL ;; SET THE ZERO FILL SWITCH
11826 060102 112737 000006 060275 MOVB #6,$OMODE+1 ;; SET FOR SIX(6) DIGITS
11827 060110 112737 000005 060272 $TYPON: MOVB #5,$OCNT ;; SET THE ITERATION COUNT
11828 060116 010346 MOV R3,-(SP) ;; SAVE R3
11829 060120 010446 MOV R4,-(SP) ;; SAVE R4
11830 060122 010546 MOV R5,-(SP) ;; SAVE R5
11831 060124 113704 060275 MOVB $OMODE+1,R4 ;; GET THE NUMBER OF DIGITS TO TYPE
11832 060130 005404 NEG R4 ;; SUBTRACT IT FOR MAX. ALLOWED
11833 060132 062704 000006 ADD #6,R4 ;; SAVE IT FOR USE
11834 060136 110437 060274 MOVB R4,$OMODE ;; GET THE ZERO FILL SWITCH
11835 060142 113704 060273 MOVB $OFILL,R4 ;; PICKUP THE INPUT NUMBER
11836 060146 016605 000012 MOV 12(SP),R5 ;; CLEAR THE OUTPUT WORD
11837 060152 005003 CLR R3 ;; ROTATE MSB INTO "C"
11838 060154 006105 1S: ROL R5 ;; GO DO MSB
11839 060156 000404 BR 3S ;; FORM THIS DIGIT
11840 060160 006105 2S: ROL R5
11841 060162 006105 ROL R5
11842 060164 006105 ROL R5
11843 060166 010503 MOV R5,R3
11844 060170 006103 3S: ROL R3 ;; GET LSB OF THIS DIGIT
11845 060172 105337 060274 DECB $OMODE ;; TYPE THIS DIGIT?
11846 060176 100016 BPL 7S ;; BR IF NO
11847 060200 042703 177770 BIC #177770,R3 ;; GET RID OF JUNK
11848 060204 001002 BNE 4S ;; TEST FOR 0
11849 060206 005704 TST R4 ;; SUPPRESS THIS 0?
11850 060210 001403 BEQ 5S ;; BR IF YES
11851 060212 005204 4S: INC R4 ;; DON'T SUPPRESS ANYMORE 0'S
11852 060214 052703 000060 BIS #0,R3 ;; MAKE THIS DIGIT ASCII
11853 060220 052703 000040 5S: BIS #' ,R3 ;; MAKE ASCII IF NOT ALREADY
11854 060224 110337 060270 MOVB R3,$S ;; SAVE FOR TYPING
11855 060230 104401 060270 TYPE #S ;; GO TYPE THIS DIGIT
11856 060234 105337 060272 7S: DECB $OCNT ;; COUNT BY 1
11857 060240 003347 BGT 2S ;; BR IF MORE TO DO
11858 060242 002402 BLT 6S ;; BR IF DONE
11859 060244 005204 INC R4 ;; INSURE LAST DIGIT ISN'T A BLANK
11860 060246 000744 BR 2S ;; GO DO THE LAST DIGIT
11861 060250 012605 6S: MOV (SP)+,R5 ;; RESTORE R5
11862 060252 012604 MOV (SP)+,R4 ;; RESTORE R4
11863 060254 012603 MOV (SP)+,R3 ;; RESTORE R3
11864 060256 016666 000002 000004 MOV 2(SP),4(SP) ;; SET THE STACK FOR RETURNING
11865 060264 012616 MOV (SP)+,(SP)
11866 060266 000002 RTI ;; RETURN
11867 060270 000 8S: .BYTE 0 ;; STORAGE FOR ASCII DIGIT
11868 060271 000 .BYTE 0 ;; TERMINATOR FOR TYPE ROUTINE
11869 060272 000 $OCNT: .BYTE 0 ;; OCTAL DIGIT COUNTER
11870 060273 000 $OFILL: .BYTE 0 ;; ZERO FILL SWITCH
11871 060274 000000 $OMODE: .WORD 0 ;; NUMBER OF DIGITS TO TYPE
11872 .SBTTL TYPE ROUTINE

```

```

*****
*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.
*
```

```

11881      ;*CALL:
11882      ;*1) USING A TRAP INSTRUCTION
11883      ;*      TYPE      ,MESADR      ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
11884      ;*OR
11885      ;*      TYPE
11886      ;*      MESADR
11887      ;*
11888
11889      060276 105737 001173      $TYPE:  TSTB      $TPFLG      ;; IS THERE A TERMINAL?
11890      060302 100002      BPL      1$      ;; BR IF YES
11891      060304 000000      HALT      ;; HALT HERE IF NO TERMINAL
11892      060306 000430      BR      3$      ;; LEAVE
11893      060310 010046      1$:  MOV      RD, -(SP)      ;; SAVE RD
11894      060312 017600 000002      MOV      22(SP), RD      ;; GET ADDRESS OF ASCIZ STRING
11895      060316 122737 000001 001242      CMPB      #APTENV, $ENV      ;; RUNNING IN APT MODE
11896      060324 001011      BNE      62$      ;; NO GO CHECK FOR APT CONSOLE
11897      060326 132737 000100 001243      BITB      #APTSPOOL, $ENVM      ;; SPOOL MESSAGE TO APT
11898      060334 001405      BEQ      62$      ;; NO GO CHECK FOR CONSOLE
11899      060336 010037 060346      MOV      RD, 61$      ;; SETUP MESSAGE ADDRESS FOR APT
11900      060342 004737 063222      JSR      PC, $ATY3      ;; SPOOL MESSAGE TO APT
11901      060346 000000      61$:  .WORD      0      ;; MESSAGE ADDRESS
11902      060350 132737 000040 001243      62$:  BITB      #APTCSUP, $ENVM      ;; APT CONSOLE SUPPRESSED
11903      060356 001003      BNE      60$      ;; YES, SKIP TYPE OUT
11904      060360 112046      2$:  MOVB      (RD)+, -(SP)      ;; PUSH CHARACTER TO BE TYPED ONTO STACK
11905      060362 001005      BNE      4$      ;; BR IF IT ISN'T THE TERMINATOR
11906      060364 005726      TST      (SP)+      ;; IF TERMINATOR POP IT OFF THE STACK
11907      060366 012600      60$:  MOV      (SP)+, RD      ;; RESTORE RD
11908      060370 062716 000002      3$:  ADD      #2, (SP)      ;; ADJUST RETURN PC
11909      060374 000002      RTI      ;; RETURN
11910      060376 122716 000011      4$:  CMPB      #HT, (SP)      ;; BRANCH IF <HT>
11911      060402 001430      BEQ      8$      ;;
11912      060404 122716 000200      CMPB      #CRLF, (SP)      ;; BRANCH IF NOT <CRLF>
11913      060410 001006      BNE      5$      ;;
11914      060412 005726      TST      (SP)+      ;; POP <CR><LF> EQUIV
11915      060414 104401      TYPE      ;; TYPE A CR AND LF
11916      060416 001217      $CRLF
11917      060420 105037 060554      CLRB      $CHARCNT      ;; CLEAR CHARACTER COUNT
11918      060424 000755      BR      2$      ;; GET NEXT CHARACTER
11919      060426 004737 060510      5$:  JSR      PC, $TYPEC      ;; GO TYPE THIS CHARACTER
11920      060432 123726 001172      6$:  CMPB      $FILLC, (SP)+      ;; IS IT TIME FOR FILLER CHARS.?
11921      060436 001350      BNE      2$      ;; IF NO GO GET NEXT CHAR.
11922      060440 013746 001170      MOV      $NULL, -(SP)      ;; GET # OF FILLER CHARS. NEEDED
11923      060444 105366 000001      7$:  DECB      1(SP)      ;; AND THE NULL CHAR.
11924      060450 002770      BLT      6$      ;; DOES A NULL NEED TO BE TYPED?
11925      060452 004737 060510      JSR      PC, $TYPEC      ;; BR IF NO--GO POP THE NULL OFF OF STACK
11926      060456 105337 060554      DECB      $CHARCNT      ;; GO TYPE A NULL
11927      060462 000770      BR      7$      ;; DO NOT COUNT AS A COUNT
11928      060462 000770      BR      7$      ;; LOOP
11929
11930      ;HORIZONTAL TAB PROCESSOR
11931
11932      060464 112716 000040      8$:  MOVB      #' , (SP)      ;; REPLACE TAB WITH SPACE
11933      060470 004737 060510      9$:  JSR      PC, $TYPEC      ;; TYPE A SPACE
11934      060474 132737 000007 060554      BITB      #7, $CHARCNT      ;; BRANCH IF NOT AT
11935      060502 001372      BNE      9$      ;; TAB STOP
11936      060504 005726      TST      (SP)+      ;; POP SPACE OFF STACK

```

```

11937 060506 000724          BR      2$          ;; GET NEXT CHARACTER
11938 060510 105777 120450  $TYPEC: TSTB      2$TPS          ;; WAIT UNTIL PRINTER IS READY
11939 060514 100375          BPL      $TYPEC
11940 060516 116677 000002 120442  MOVB      2(SP), 2$TPB          ;; LOAD CHAR TO BE TYPED INTO DATA REG.
11941 060524 122766 000015 000002  CMPB      #CR, 2(SP)          ;; IS CHARACTER A CARRIAGE RETURN?
11942 060532 001003          BNE      1$          ;; BRANCH IF NO
11943 060534 105037 060554          CLRB      $CHARCNT          ;; YES--CLEAR CHARACTER COUNT
11944 060540 000406          BR      $TYPEX          ;; EXIT
11945 060542 122766 000012 000002 1$: CMPB      #LF, 2(SP)          ;; IS CHARACTER A LINE FEED?
11946 060550 001402          BEQ      $TYPEX          ;; BRANCH IF YES
11947 060552 105227          INCB      (PC)+          ;; COUNT THE CHARACTER
11948 060554 000000          $CHARCNT: .WORD 0          ;; CHARACTER COUNT STORAGE
11949 060556 000207          $TYPEX: RTS      PC
11950
11951          .SBTTL  SCOPE HANDLER ROUTINE
11952
11953          ;*****
11954          ;THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
11955          ;AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>.)
11956          ;AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
11957          ;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
11958          ;$SW14=1  LOOP ON TEST
11959          ;$SW11=1  INHIBIT ITERATIONS
11960          ;$SW09=1  LOOP ON ERROR
11961          ;$SW08=1  LOOP ON TEST IN SWR<7:0>
11962          ;CALL
11963          ;*      SCOPE          ;; SCOPE=10T
11964
11965 060560          $SCOPE:
11966 060560 104410          CKSWR          ;; TEST FOR CHANGE IN SOFT-SWR
11967 060562 004737 057334          JSR      PC, STOP
11968 060566 032777 040000 120360 1$: BIT      #BIT14, 2$SWR          ;; LOOP ON PRESENT TEST?
11969 060574 001131          BNE      $OVER          ;; YES IF SW14=1
11970          ;*****START OF CODE FOR THE XOR TESTER*****
11971 060576 000416          $XTSTR: BR      6$          ;; IF RUNNING ON THE "XOR" TESTER CHANGE
11972          ;THIS INSTRUCTION TO A "NOP" (NOP=24C)
11973 060600 013746 000004          MOV      2$ERRVEC, -(SP)          ;; SAVE THE CONTENTS OF THE ERROR VECTOR
11974 060604 012737 060624 000004          MOV      2$ERRVEC, 2$ERRVEC          ;; SET FOR TIMEOUT
11975 060612 005737 177060          TST      2$177060          ;; TIME OUT ON XOR?
11976 060616 012637 000004          MOV      -(SP)+, 2$ERRVEC          ;; RESTORE THE ERROR VECTOR
11977 060622 000500          BR      $SVLAD          ;; GO TO THE NEXT TEST
11978 060624 022626          SS:  CMP      (SP)+, (SP)+          ;; CLEAR THE STACK AFTER A TIME OUT
11979 060626 012637 000004          MOV      (SP)+, 2$ERRVEC          ;; RESTORE THE ERROR VECTOR
11980 060632 000440          BR      7$          ;; LOOP ON THE PRESENT TEST
11981 060634          6$: ;*****END OF CODE FOR THE XOR TESTER*****
11982 060634 032777 000400 120312  BIT      #BIT08, 2$SWR          ;; LOOP ON SPEC. TEST?
11983 060642 001421          BEQ      2$          ;; BR IF NO
11984 060644 005046          CLR      -(SP)          ;; CLEAR A TEMP. LOCATION
11985 060646 117716 120302          MOVB      2$SWR, (SP)          ;; PICKUP THE DESIRED TEST NUMBER
11986 060652 001414          BEQ      8$          ;; BRANCH IF BAD TEST NUMBER IN SWR
11987 060654 022716 000035          CMP      #35, (SP)          ;; CHECK THE NUMBER IN THE SWR
11988 060660 002411          BLT      8$          ;; BRANCH IF TEST NUMBER IS OUT OF RANGE
11989 060662 011637 001116          MOV      (SP), $TSTNM          ;; UPDATE THE TEST NUMBER
11990 060666 005316          DEC      (SP)          ;; BACKUP BY ONE
11991 060670 006316          ASL      (SP)          ;; SCALE THE TEST NUMBER AS AN INDEX
11992 060672 062716 061076          ADD      #SW08TBL, (SP)          ;; FORM THE ADDRESS OF TEST POINTER

```

11993	060676	013637	001122		MOV	2(SP)+, \$LPADR	SET LOOP ADDRESS TO DESIRED TEST
11994	060702	000466			BR	\$OVER	GO LOOP ON THE TEST
11995	060704	005726		8\$:	TST	(SP)+	CLEAN THE BAD TEST NUMBER OFF OF THE STACK
11996	060706	105737	001117	2\$:	TSTB	\$ERFLG	HAS AN ERROR OCCURRED?
11997	060712	001421			BEQ	3\$	BR IF NO
11998	060714	123737	001131	001117	CMPB	\$ERMAX, \$ERFLG	MAX. ERRORS FOR THIS TEST OCCURRED?
11999	060722	101015			BHI	3\$	BR IF NO
12000	060724	032777	001000	120222	BIT	#BIT09, \$SWR	LOOP ON ERROR?
12001	060732	001404			BEQ	4\$	BR IF NO
12002	060734	013737	001124	001122	7\$:	MOV	\$LPERR, \$LPADR
12003	060742	000446			BR	\$OVER	SET LOOP ADDRESS TO LAST SCOPE
12004	060744	105037	001117		4\$:	CLRB	\$ERFLG
12005	060750	005037	001206		CLR	\$TIMES	ZERO THE ERROR FLAG
12006	060754	000415			BR	1\$	CLEAR THE NUMBER OF ITERATIONS TO MAKE
12007	060756	032777	004000	120170	3\$:	BIT	#BIT11, \$SWR
12008	060764	001011			1\$	BNE	INHIBIT ITERATIONS?
12009	060766	005737	001230		TST	\$PASS	BR IF YES
12010	060772	001406			BEQ	1\$	IF FIRST PASS OF PROGRAM
12011	060774	005237	001120		INC	\$ICNT	INHIBIT ITERATIONS
12012	061000	013737	001206	001120	CMP	\$TIMES, \$ICNT	INCREMENT ITERATION COUNT
12013	061006	002024			BGE	\$OVER	CHECK THE NUMBER OF ITERATIONS MADE
12014	061010	012737	000001	001120	1\$:	MOV	#1, \$ICNT
12015	061016	013737	061074	001206	MOV	\$MXCNT, \$TIMES	BR IF MORE ITERATION REQUIRED
12016	061024	105237	001116		\$SVLAD:	INCB	REINITIALIZE THE ITERATION COUNTER
12017	061030	113737	001116	001226	MOV	\$STNM, \$TESTN	SET NUMBER OF ITERATIONS TO DO
12018	061036	011637	001122		MOV	(SP), \$LPADR	COUNT TEST NUMBERS
12019	061042	011637	001124		MOV	(SP), \$LPERR	SET TEST NUMBER IN APT MAILBOX
12020	061046	005037	001210		CLR	\$ESCAPE	SAVE SCOPE LOOP ADDRESS
12021	061052	112737	000001	001131	MOV	#1, \$ERMAX	SAVE ERROR LOOP ADDRESS
12022	061060	013777	001116	120070	\$OVER:	MOV	\$STNM, \$DISPLAY
12023	061066	013716	001122		MOV	\$LPADR, (SP)	CLEAR THE ESCAPE FROM ERROR ADDRESS
12024	061072	000002			RTI		ONLY ALLOW ONE(1) ERROR ON NEXT TEST
12025	061074	000012			\$MXCNT:	10.	DISPLAY TEST NUMBER
12026	061076				\$SWOBTBL:		FUDGE RETURN ADDRESS
12027	061076	00754			.WORD	TST1+2	FIXES PS
12028	061100	007362			.WORD	TST2+2	MAX. NUMBER OF ITERATIONS
12029	061102	007546			.WORD	TST3+2	STARTING ADDRESS OF TEST 1
12030	061104	007730			.WORD	TST4+2	STARTING ADDRESS OF TEST 2
12031	061106	010530			.WORD	TST5+2	STARTING ADDRESS OF TEST 3
12032	061110	011330			.WORD	TST6+2	STARTING ADDRESS OF TEST 4
12033	061112	012144			.WORD	TST7+2	STARTING ADDRESS OF TEST 5
12034	061114	012720			.WORD	TST10+2	STARTING ADDRESS OF TEST 6
12035	061116	013640			.WORD	TST11+2	STARTING ADDRESS OF TEST 7
12036	061120	014440			.WORD	TST12+2	STARTING ADDRESS OF TEST 10
12037	061122	015214			.WORD	TST13+2	STARTING ADDRESS OF TEST 11
12038	061124	016132			.WORD	TST14+2	STARTING ADDRESS OF TEST 12
12039	061126	016706			.WORD	TST15+2	STARTING ADDRESS OF TEST 13
12040	061130	017462			.WORD	TST16+2	STARTING ADDRESS OF TEST 14
12041	061132	020236			.WORD	TST17+2	STARTING ADDRESS OF TEST 15
12042	061134	021040			.WORD	TST20+2	STARTING ADDRESS OF TEST 16
12043	061136	021564			.WORD	TST21+2	STARTING ADDRESS OF TEST 17
12044	061140	022310			.WORD	TST22+2	STARTING ADDRESS OF TEST 20
12045	061142	023032			.WORD	TST23+2	STARTING ADDRESS OF TEST 21
12046	061144	023330			.WORD	TST24+2	STARTING ADDRESS OF TEST 22
12047	061146	023626			.WORD	TST25+2	STARTING ADDRESS OF TEST 23
12048	061150	024354			.WORD	TST26+2	STARTING ADDRESS OF TEST 24
							STARTING ADDRESS OF TEST 25
							STARTING ADDRESS OF TEST 26

12049	061152	025102			.WORD	TST27+2	;; STARTING ADDRESS OF TEST 27
12050	061154	025630			.WORD	TST30+2	;; STARTING ADDRESS OF TEST 30
12051	061156	026522			.WORD	TST31+2	;; STARTING ADDRESS OF TEST 31
12052	061160	027330			.WORD	TST32+2	;; STARTING ADDRESS OF TEST 32
12053	061162	030052			.WORD	TST33+2	;; STARTING ADDRESS OF TEST 33
12054	061164	030574			.WORD	TST34+2	;; STARTING ADDRESS OF TEST 34
12055	061166	031752			.WORD	TST35+2	;; STARTING ADDRESS OF TEST 35
12056					.SBTTL ERROR HANDLER ROUTINE		
12057					*****		
12058					; THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,		
12059					; SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL		
12060					; AND GO TO ERRTP ON ERROR		
12061					; THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:		
12062					; SW15=1 HALT ON ERROR		
12063					; SW13=1 INHIBIT ERROR TYPEOUTS		
12064					; SW10=1 BELL ON ERROR		
12065					; SW09=1 LOOP ON ERROR		
12066					; CALL		
12067					; * ERROR N ;; ERROR=EMT AND N=ERROR ITEM NUMBER		
12068					; * ERROR N ;; ERROR=EMT AND N=ERROR ITEM NUMBER		
12069					; * ERROR N ;; ERROR=EMT AND N=ERROR ITEM NUMBER		
12070	061170				; ERROR:		
12071	061170	104410			7\$:	CKSWR	;; TEST FOR CHANGE IN SOFT-SWR
12072	061172	105237	001117			INCB	;; SET THE ERROR FLAG
12073	061176	001775				BEQ	;; DON'T LET THE FLAG GO TO ZERO
12074	061200	013777	001116	117750		MOV	;; DISPLAY TEST NUMBER AND ERROR FLAG
12075	061206	032777	002000	117740		BIT	;; BELL ON ERROR?
12076	061214	001402				BEQ	;; NO - SKIP
12077	061216	104401	001212			TYPE	;; RING BELL
12078	061222	005237	001126		1\$:	INC	;; COUNT THE NUMBER OF ERRORS
12079	061226	011637	001132			MOV	;; GET ADDRESS OF ERROR INSTRUCTION
12080	061232	162737	000002	001132		SUB	;; STRIP AND SAVE THE ERROR ITEM CODE
12081	061240	117737	117666	001130		MOVB	;; SKIP TYPEOUT IF SET
12082	061246	032777	020000	117700		BIT	;; SKIP TYPEOUTS
12083	061254	001004				BNE	;; GO TO USER ERROR ROUTINE
12084	061256	004737	033176			JSR	;; GO TO USER ERROR ROUTINE
12085	061262	104401	001217			TYPE	;; GO TO USER ERROR ROUTINE
12086	061266				20\$:		
12087	061266	122737	000001	001242		CMPB	;; RUNNING IN APT MODE
12088	061274	001007				BNE	;; NO SKIP APT ERROR REPORT
12089	061276	113737	001130	061310		MOVB	;; SET ITEM NUMBER AS ERROR NUMBER
12090	061304	004737	063232			JSR	;; REPORT FATAL ERROR TO APT
12091	061310	000			21\$:	.BYTE	
12092	061311	000				.BYTE	
12093	061312	000777			22\$:	BR	;; APT ERROR LOOP
12094	061314	005777	117634		2\$:	TST	;; HALT ON ERROR
12095	061320	100002				BPL	;; SKIP IF CONTINUE
12096	061322	000000				HALT	;; HALT ON ERROR!
12097	061324	104410				CKSWR	;; TEST FOR CHANGE IN SOFT-SWR
12098	061326	032777	001000	117620	3\$:	BIT	;; LOOP ON ERROR SWITCH SET?
12099	061334	001402				BEQ	;; BR IF NO
12100	061336	013716	001124			MOV	;; FUDGE RETURN FOR LOOPING
12101	061342	005737	001210		4\$:	TST	;; CHECK FOR AN ESCAPE ADDRESS
12102	061346	001402				BEQ	;; BR IF NONE
12103	061350	013716	001210			MOV	;; FUDGE RETURN ADDRESS FOR ESCAPE
12104	061354				5\$:		

```

12105 061354 022737 033156 000042      CMP      #SENDAD,2#42      ;;ACT-11 AUTO-ACCEPT?
12106 061362 001001                      BNE      6$              ;;BRANCH IF NO
12107 061364 000000                      HALT                      ;;YES
12108 061366 000000                      6$:                          RTI              ;;RETURN
12109 061366 000002                      .SBTTL   TTY INPUT ROUTINE

12110
12111
12112
12113
12114 061370 000000      .ENABL   LSB
12115 061372 000000      $TKCNT: .WORD   0              ;;NUMBER OF ITEMS IN QUEUE
12116 061374 000000      $TKQIN: .WORD   0              ;;INPUT POINTER
12117 061376 000001      $TKQOUT: .WORD  0              ;;OUTPUT POINTER
12118 061377 000001      $TKQSRV: .BLKB  1              ;;TTY KEYBOARD QUEUE
12119 061400 000000      $TKQEND: .
12120
12121
12122
12123
12124
12125
12126
12127
12128
12129 061400 005037 061370 061372      *TK INITIALIZE ROUTINE
12130 061404 012737 061376 061372      *THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
12131 061412 013737 061372 061374      *SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
12132 061420 012737 061450 000060      *CALL:
12133 061426 012737 000200 000062      *      JSR      PC,$TKINT
12134 061434 005777 117522
12135 061440 012777 000100 117512      *      RETURN
12136 061446 000207
12137
12138
12139
12140
12141
12142
12143
12144
12145 061450 117746 117506      $TKINT: CLR      $TKCNT      ;;CLEAR COUNT OF ITEMS IN QUEUE
12146 061454 042716 177600      MOV      $TKQSRV,$TKQIN  ;;MOVE THE STARTING ADDRESS OF THE
12147 061460 021627 000003      MOV      $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
12148 061464 001007          MOV      $TKSRV,$TKVEC      ;;INITIALIZE THE KEYBOARD VECTOR
12149 061466 104401 062564      MOV      #200,$TKVEC+2  ;;"BA" LEVEL 4
12150 061472 004737 061400      TST      $TKB          ;;CLEAR DONE FLAG
12151 061476 005726          MOV      #100,$TKS          ;;ENABLE TTY KEYBOARD INTERRUPT
12152 061500 000137 057444      RTS      PC          ;;RETURN TO CALLER
12153 061504 021627 000007
12154 061510 001004
12155 061512 022737 000176 001154
12156 061520 001500
12157
12158 061522
12159 061522 022737 000001 061370      *TK SERVICE ROUTINE
12160 061530 001004      *THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
12161
12162
12163
12164
12165
12166
12167
12168
12169
12170
12171
12172
12173
12174
12175
12176
12177
12178
12179
12180
12181
12182
12183
12184
12185
12186
12187
12188
12189
12190
12191
12192
12193
12194
12195
12196
12197
12198
12199
12200
12201
12202
12203
12204
12205
12206
12207
12208
12209
12210
12211
12212
12213
12214
12215
12216
12217
12218
12219
12220
12221
12222
12223
12224
12225
12226
12227
12228
12229
12230
12231
12232
12233
12234
12235
12236
12237
12238
12239
12240
12241
12242
12243
12244
12245
12246
12247
12248
12249
12250
12251
12252
12253
12254
12255
12256
12257
12258
12259
12260
12261
12262
12263
12264
12265
12266
12267
12268
12269
12270
12271
12272
12273
12274
12275
12276
12277
12278
12279
12280
12281
12282
12283
12284
12285
12286
12287
12288
12289
12290
12291
12292
12293
12294
12295
12296
12297
12298
12299
12300
12301
12302
12303
12304
12305
12306
12307
12308
12309
12310
12311
12312
12313
12314
12315
12316
12317
12318
12319
12320
12321
12322
12323
12324
12325
12326
12327
12328
12329
12330
12331
12332
12333
12334
12335
12336
12337
12338
12339
12340
12341
12342
12343
12344
12345
12346
12347
12348
12349
12350
12351
12352
12353
12354
12355
12356
12357
12358
12359
12360
12361
12362
12363
12364
12365
12366
12367
12368
12369
12370
12371
12372
12373
12374
12375
12376
12377
12378
12379
12380
12381
12382
12383
12384
12385
12386
12387
12388
12389
12390
12391
12392
12393
12394
12395
12396
12397
12398
12399
12400
12401
12402
12403
12404
12405
12406
12407
12408
12409
12410
12411
12412
12413
12414
12415
12416
12417
12418
12419
12420
12421
12422
12423
12424
12425
12426
12427
12428
12429
12430
12431
12432
12433
12434
12435
12436
12437
12438
12439
12440
12441
12442
12443
12444
12445
12446
12447
12448
12449
12450
12451
12452
12453
12454
12455
12456
12457
12458
12459
12460
12461
12462
12463
12464
12465
12466
12467
12468
12469
12470
12471
12472
12473
12474
12475
12476
12477
12478
12479
12480
12481
12482
12483
12484
12485
12486
12487
12488
12489
12490
12491
12492
12493
12494
12495
12496
12497
12498
12499
12500
12501
12502
12503
12504
12505
12506
12507
12508
12509
12510
12511
12512
12513
12514
12515
12516
12517
12518
12519
12520
12521
12522
12523
12524
12525
12526
12527
12528
12529
12530
12531
12532
12533
12534
12535
12536
12537
12538
12539
12540
12541
12542
12543
12544
12545
12546
12547
12548
12549
12550
12551
12552
12553
12554
12555
12556
12557
12558
12559
12560
12561
12562
12563
12564
12565
12566
12567
12568
12569
12570
12571
12572
12573
12574
12575
12576
12577
12578
12579
12580
12581
12582
12583
12584
12585
12586
12587
12588
12589
12590
12591
12592
12593
12594
12595
12596
12597
12598
12599
12600
12601
12602
12603
12604
12605
12606
12607
12608
12609
12610
12611
12612
12613
12614
12615
12616
12617
12618
12619
12620
12621
12622
12623
12624
12625
12626
12627
12628
12629
12630
12631
12632
12633
12634
12635
12636
12637
12638
12639
12640
12641
12642
12643
12644
12645
12646
12647
12648
12649
12650
12651
12652
12653
12654
12655
12656
12657
12658
12659
12660
12661
12662
12663
12664
12665
12666
12667
12668
12669
12670
12671
12672
12673
12674
12675
12676
12677
12678
12679
12680
12681
12682
12683
12684
12685
12686
12687
12688
12689
12690
12691
12692
12693
12694
12695
12696
12697
12698
12699
12700
12701
12702
12703
12704
12705
12706
12707
12708
12709
12710
12711
12712
12713
12714
12715
12716
12717
12718
12719
12720
12721
12722
12723
12724
12725
12726
12727
12728
12729
12730
12731
12732
12733
12734
12735
12736
12737
12738
12739
12740
12741
12742
12743
12744
12745
12746
12747
12748
12749
12750
12751
12752
12753
12754
12755
12756
12757
12758
12759
12760
12761
12762
12763
12764
12765
12766
12767
12768
12769
12770
12771
12772
12773
12774
12775
12776
12777
12778
12779
12780
12781
12782
12783
12784
12785
12786
12787
12788
12789
12790
12791
12792
12793
12794
12795
12796
12797
12798
12799
12800
12801
12802
12803
12804
12805
12806
12807
12808
12809
12810
12811
12812
12813
12814
12815
12816
12817
12818
12819
12820
12821
12822
12823
12824
12825
12826
12827
12828
12829
12830
12831
12832
12833
12834
12835
12836
12837
12838
12839
12840
12841
12842
12843
12844
12845
12846
12847
12848
12849
12850
12851
12852
12853
12854
12855
12856
12857
12858
12859
12860
12861
12862
12863
12864
12865
12866
12867
12868
12869
12870
12871
12872
12873
12874
12875
12876
12877
12878
12879
12880
12881
12882
12883
12884
12885
12886
12887
12888
12889
12890
12891
12892
12893
12894
12895
12896
12897
12898
12899
12900
12901
12902
12903
12904
12905
12906
12907
12908
12909
12910
12911
12912
12913
12914
12915
12916
12917
12918
12919
12920
12921
12922
12923
12924
12925
12926
12927
12928
12929
12930
12931
12932
12933
12934
12935
12936
12937
12938
12939
12940
12941
12942
12943
12944
12945
12946
12947
12948
12949
12950
12951
12952
12953
12954
12955
12956
12957
12958
12959
12960
12961
12962
12963
12964
12965
12966
12967
12968
12969
12970
12971
12972
12973
12974
12975
12976
12977
12978
12979
12980
12981
12982
12983
12984
12985
12986
12987
12988
12989
12990
12991
12992
12993
12994
12995
12996
12997
12998
12999
13000

```

```

12161 061532 104401 001212      TYPE      $BELL      ;; RING THE TTY BELL
12162 061536 005726      TST      (SP)+      ;; CLEAN CHARACTER OFF OF STACK
12163 061540 000451      BR      5$      ;; EXIT
12164 061542 021627 000023      3$: CMP      (SP), #23      ;; IS IT A CONTROL-S?
12165 061546 001021      BNE      32$      ;; BRANCH IF NO
12166 061550 005077 117404      CLR      2$TKS      ;; DISABLE TTY KEYBOARD INTERRUPTS
12167 061554 005726      TST      (SP)+      ;; CLEAN CHAR OFF STACK
12168 061556 105777 117376      31$: TSTB     2$TKS      ;; WAIT FOR A CHAR
12169 061562 100375      BPL      31$      ;; LOOP UNTIL ITS THERE
12170 061564 117746 117372      MOV      2$TKB, -(SP)      ;; GET THE CHARACTER
12171 061570 042716 177600      BIC      #1C177, (SP)      ;; MAKE IT 7-BIT ASCII
12172 061574 022627 000021      CMP      (SP)+, #21      ;; IS IT A CONTROL-Q?
12173 061600 001366      BNE      31$      ;; BRANCH IF NO
12174 061602 012777 000100 117350      MOV      #100, 2$TKS      ;; REENABLE TTY KEYBOARD INTERRUPTS
12175 061610 000002      RTI      ;; RETURN
12176 061612 005237 061370      32$: INC      $TKCNT      ;; COUNT THIS CHARACTER
12177 061616 021627 000140      CMP      (SP), #140      ;; IS IT UPPER CASE?
12178 061622 002405      BLT      4$      ;; BRANCH IF YES
12179 061624 021627 000175      CMP      (SP), #175      ;; IS IT A SPECIAL CHAR?
12180 061630 003002      BGT      4$      ;; BRANCH IF YES
12181 061632 042716 000040      BIC      #40, (SP)      ;; MAKE IT UPPER CASE
12182 061636 112677 177530      4$: MOV      (SP)+, 2$TKQIN      ;; AND PUT IT IN QUEUE
12183 061642 005237 061372      INC      $TKQIN      ;; UPDATE THE POINTER
12184 061646 023727 061372 061377      CMP      $TKQIN, $STKGEND      ;; GO OFF THE END?
12185 061654 001003      BNE      5$      ;; BRANCH IF NO
12186 061656 012737 061376 061372      MOV      $STKQSR, $TKQIN      ;; RESET THE POINTER
12187 061664 000002      5$: RTI      ;; RETURN
12188
12189      ;; *****
12190      ;; SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
12191      ;; ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
12192      ;; SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
12193      ;; CALL WHEN OPERATING IN TTY INTERRUPT MODE.
12194 061666 022737 000176 001154      $CKSWR: CMP      $SWREG, SWR      ;; IS THE SOFT-SWR SELECTED
12195 061674 001124      BNE      15$      ;; EXIT IF NOT
12196 061676 105777 117256      TSTB     2$TKS      ;; IS A CHAR WAITING?
12197 061702 100121      BPL      15$      ;; IF NOT, EXIT
12198 061704 117746 117252      MOV      2$TKB, -(SP)      ;; YES
12199 061710 042716 177600      BIC      #1C177, (SP)      ;; MAKE IT 7-BIT ASCII
12200 061714 021627 000007      CMP      (SP), #7      ;; IS IT A CONTROL-G?
12201 061720 001300      BNE      2$      ;; IF NOT, PUT IT IN THE TTY QUEUE
12202      ;; AND EXIT
12203
12204      ;; *****
12205      ;; CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
12206      ;; ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
12207      ;; CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
12208 061722 123727 001150 000001      6$: CMPB     $AUTOB, #1      ;; ARE WE RUNNING IN AUTO-MODE?
12209 061730 001674      BEQ      2$      ;; BRANCH IF YES
12210 061732 005726      TST      (SP)+      ;; CLEAR CONTROL-G OFF STACK
12211 061734 004737 061400      JSR      PC, $TKINT      ;; FLUSH THE TTY INPUT QUEUE
12212 061740 005077 117214      CLR      2$TKS      ;; DISABLE TTY KEYBOARD INTERRUPTS
12213 061744 112737 000001 001151      MOV      #1, $INTAG      ;; SET INTERRUPT MODE INDICATOR
12214
12215 061752 104401 062576      TYPE      $CNTLG      ;; ECHO THE CONTROL-G (1G)
12216 061756 104401 062603      $GTSWR: TYPE      $MSWR      ;; TYPE CURRENT CONTENTS

```

12217	061762	013746	000176		MOV	SWREG, -(SP)	:: SAVE SWREG FOR TYPEOUT
12218	061766	104402			TYP0C		:: GO TYPE--OCTAL ASCII(ALL DIGITS)
12219	061770	104401	062614		TYPE	, \$MNEW	:: PROMPT FOR NEW SWR
12220	061774	005046		19\$:	CLR	-(SP)	:: CLEAR COUNTER
12221	061776	005046			CLR	-(SP)	:: THE NEW SWR
12222	062000	105777	117154	7\$:	TSTB	2\$TKS	:: CHAR THERE?
12223	062004	100375			BPL	7\$:: IF NOT TRY AGAIN
12224							
12225	062006	117746	117150		MOVB	2\$TKB, -(SP)	:: PICK UP CHAR
12226	062012	042716	177600		BIC	#1C177, (SP)	:: MAKE IT 7-BIT ASCII
12227							
12228	062016	021627	000003		CMP	(SP), #3	:: IS IT A CONTROL-C?
12229	062022	001015			BNE	9\$:: BRANCH IF NOT
12230	062024	104401	062564		TYPE	, \$CNTLC	:: YES, ECHO CONTROL-C (↑C)
12231	062030	062706	000006		ADD	#6, SP	:: CLEAN UP STACK
12232	062034	123727	001151	000001	CMPB	\$INTAG, #1	:: REENABLE TTY KEYBOARD INTERRUPTS?
12233	062042	001003			BNE	8\$:: BRANCH IF NO
12234	062044	012777	000100	117106	MOV	#100, 2\$TKS	:: ALLOW TTY KEYBOARD INTERRUPTS
12235	062052	000137	057444	8\$:	JMP	SHUT2	:: CONTROL-C RESTART
12236							
12237							
12238	062056	021627	000025	9\$:	CMP	(SP), #25	:: IS IT A CONTROL-U?
12239	062062	001005			BNE	10\$:: BRANCH IF NOT
12240	062064	104401	062571		TYPE	, \$CNTLU	:: YES, ECHO CONTROL-U (↑U)
12241	062070	062706	000006	20\$:	ADD	#6, SP	:: IGNORE PREVIOUS INPUT
12242	062074	000737			BR	19\$:: LET'S TRY IT AGAIN
12243							
12244							
12245	062076	021627	000015	10\$:	CMP	(SP), #15	:: IS IT A <CR>?
12246	062102	001022			BNE	16\$:: BRANCH IF NO
12247	062104	005766	000004		TST	4(SP)	:: YES, IS IT THE FIRST CHAR?
12248	062110	001403			BEQ	11\$:: BRANCH IF YES
12249	062112	016677	000002	117034	MOV	2(SP), 2\$SWR	:: SAVE NEW SWR
12250	062120	062706	000006	11\$:	ADD	#6, SP	:: CLEAN UP STACK
12251	062124	104401	001217	14\$:	TYPE	, \$CRLF	:: ECHO <CR> AND <LF>
12252	062130	123727	001151	000001	CMPB	\$INTAG, #1	:: RE-ENABLE TTY KBD INTERRUPTS
12253	062136	001003			BNE	15\$:: BRANCH IF NOT
12254	062140	012777	000100	117012	MOV	#100, 2\$TKS	:: RE-ENABLE TTY KBD INTERRUPTS
12255	062146	000002		15\$:	RTI		:: RETURN
12256	062150	004737	060510	16\$:	JSR	PC, \$TYPEC	:: ECHO CHAR
12257	062154	021627	000060		CMP	(SP), #60	:: CHAR < 0?
12258	062160	002420			BLT	18\$:: BRANCH IF YES
12259	062162	021627	000067		CMP	(SP), #67	:: CHAR > 7?
12260	062166	003015			BGT	18\$:: BRANCH IF YES
12261	062170	042726	000060		BIC	#60, (SP)+	:: STRIP-OFF ASCII
12262	062174	005766	000002		TST	2(SP)	:: IS THIS THE FIRST CHAR
12263	062200	001403			BEQ	17\$:: BRANCH IF YES
12264	062202	006316			ASL	(SP)	:: NO, SHIFT PRESENT
12265	062204	006316			ASL	(SP)	:: CHAR OVER TO MAKE
12266	062206	006316			ASL	(SP)	:: ROOM FOR NEW ONE.
12267	062210	005266	000002	17\$:	INC	2(SP)	:: KEEP COUNT OF CHAR
12268	062214	056616	177776		BIS	-2(SP), (SP)	:: SET IN NEW CHAR
12269	062220	000667			BR	7\$:: GET THE NEXT ONE
12270	062222	104401	001216	18\$:	TYPE	, \$QUES	:: TYPE ?<CR><LF>
12271	062226	000720			BR	20\$:: SIMULATE CONTROL-U
12272				.DSABL	LSB		

```

12273
12274
12275
12276
12277
12278
12279
12280
12281
12282
12283 062230 011646
12284 062232 016666 000004 000002
12285 062240 005066 000004
12286 062244 005046
12287 062246 012746 062254
12288 062252 000002
12289 062254
12290 062254 005737 061370
12291 062260 001775
12292 062262 005337 061370
12293 062266 117766 177102 000004
12294 062274 005237 061374
12295 062300 023727 061374 061377
12296 062306 001003
12297 062310 012737 061376 061374
12298 062316 000002
12299
12300
12301
12302
12303
12304
12305
12306 062320 010346
12307 062322 005046
12308 062324 012703 062554
12309 062330 022703 062564
12310 062334 101456
12311 062336 104411
12312 062340 112613
12313 062342 122713 000177
12314 062346 001022
12315 062350 005716
12316 062352 001007
12317 062354 112737 000134 062552
12318 062362 104401 062552
12319 062366 012716 177777
12320 062372 005303
12321 062374 020327 062554
12322 062400 103434
12323 062402 111337 062552
12324 062406 104401 062552
12325 062412 000746
12326 062414 005716
12327 062416 001406
12328 062420 112737 000134 062552

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

$RDCHR: MOV (SP), -(SP)
: PUSH DOWN THE PC AND
MOV 4(SP), 2(SP)
: THE PS
CLR 4(SP)
: GET READY FOR A CHARACTER
CLR -(SP)
: PUT NEW PS ON STACK
MOV #64$, -(SP)
: PUT NEW PC ON STACK
RTI
: POP NEW PC AND PS

64$:
1$: TST $TKCNT
: WAIT ON A CHARACTER
BEQ 1$
DEC $TKCNT
: DECREMENT THE COUNTER
MOVB $TKQOUT, 4(SP)
: GET ONE CHARACTER
INC $TKQOUT
: UPDATE THE POINTER
CMP $TKQOUT, $TKQEND
: DID IT GO OFF OF THE END?
BNE 2$
: BRANCH IF NO
MOV $TKQSRT, $TKQOUT
: RESET THE POINTER
RTI
: RETURN

2$:
:*****
:THIS ROUTINE WILL INPUT A STRING FROM THE TTY
:CALL:
:RDLIN
:RETURN HERE
:INPUT A STRING FROM THE TTY
:ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
:TERMINATOR WILL BE A BYTE OF ALL 0'S

$RDLIN: MOV R3, -(SP)
: SAVE R3
CLR -(SP)
: CLEAR THE RUBOUT KEY
1$: MOV $TTYIN, R3
: GET ADDRESS
2$: CMP $TTYIN+8., R3
: BUFFER FULL?
BLOS 4$
: BR IF YES
RDCHR
: GO READ ONE CHARACTER FROM THE TTY
MOVB (SP)+, (R3)
: GET CHARACTER
10$: CMPB #177, (R3)
: IS IT A RUBOUT
BNE 5$
: BR IF NO
TST (SP)
: IS THIS THE FIRST RUBOUT?
BNE 6$
: BR IF NO
MOVB #' \, 9$
: TYPE A BACK SLASH
TYPE 9$
: SET THE RUBOUT KEY
MOV 1-1, (SP)
: BACKUP BY ONE
6$: DEC R3
: STACK EMPTY?
CMP R3, $TTYIN
: BR IF YES
BLO 4$
: SETUP TO TYPEOUT THE DELETED CHAR.
MOVB (R3), 9$
: GO TYPE
TYPE 9$
: GO READ ANOTHER CHAR.
BR 2$
: RUBOUT KEY SET?
5$: TST (SP)
: BR IF NO
BEQ 7$
: TYPE A BACK SLASH
MOVB #' \, 9$

```

12329	062426	104401	062552		TYPE	9\$		
12330	062432	005016			CLR	(SP)		;; CLEAR THE RUBOUT KEY
12331	062434	122713	000025	7\$:	CMPB	#25,(R3)		;; IS CHARACTER A CTRL U?
12332	062440	001003			BNE	8\$;; BR IF NO
12333	062442	104401	062571		TYPE	\$CNTLU		;; TYPE A CONTROL "U"
12334	062446	000726			BR	1\$;; GO START OVER
12335	062450	122713	000022	8\$:	CMPB	#22,(R3)		;; IS CHARACTER A "R"?
12336	062454	001011			BNE	3\$;; BRANCH IF NO
12337	062456	105013			CLRB	(R3)		;; CLEAR THE CHARACTER
12338	062460	104401	001217		TYPE	\$CRLF		;; TYPE A "CR" & "LF"
12339	062464	104401	062554		TYPE	\$TTYIN		;; TYPE THE INPUT STRING
12340	062470	000717			BR	2\$;; GO PICKUP ANOTHER CHACTER
12341	062472	104401	001216	4\$:	TYPE	\$QUES		;; TYPE A '?'
12342	062476	000712			BR	1\$;; CLEAR THE BUFFER AND LOOP
12343	062500	111337	062552	3\$:	MOVB	(R3),9\$;; ECHO THE CHARACTER
12344	062504	104401	062552		TYPE	9\$		
12345	062510	122723	000015		CMPB	#15,(R3)+		;; CHECK FOR RETURN
12346	062514	001305			BNE	2\$;; LOOP IF NOT RETURN
12347	062516	105063	177777		CLRB	-1(R3)		;; CLEAR RETURN (THE 15)
12348	062522	104401	001220		TYPE	\$LF		;; TYPE A LINE FEED
12349	062526	005726			TST	(SP)+		;; CLEAN RUBOUT KEY FROM THE STACK
12350	062530	012603			MOV	(SP)+,R3		;; RESTORE R3
12351	062532	011646			MOV	(SP)-,(SP)		;; ADJUST THE STACK AND PUT ADDRESS OF THE
12352	062534	016666	000004	000002	MOV	4(SP),2(SP)		;; FIRST ASCII CHARACTER ON IT
12353	062542	012766	062554	000004	MOV	#TTYIN,4(SP)		
12354	062550	000002			RTI			;; RETURN
12355	062552	000		9\$:	.BYTE	0		;; STORAGE FOR ASCII CHAR. TO TYPE
12356	062553	000			.BYTE	0		;; TERMINATOR
12357	062554	000010		\$TTYIN:	.BLKB	8		;; RESERVE 8 BYTES FOR TTY INPUT
12358	062564	041536	005015	000	\$CNTLC:	.ASCIZ /1C/15/12/		;; CONTROL "C"
12359	062571	136	006525	000012	\$CNTLU:	.ASCIZ /1U/15/12/		;; CONTROL "U"
12360	062576	043536	005015	000	\$CNTLG:	.ASCIZ /1G/15/12/		;; CONTROL "G"
12361	062603	015	051412	051127	\$MSWR:	.ASCIZ 15/12/SWR = /		
12362	062610	036440	000040					
12363	062614	020040	042516	020127	\$MNEW:	.ASCIZ / NEW = /		
12364	062622	020075	000					
12365	062626				.EVEN			
12366					.SBTTL	READ AN OCTAL NUMBER FROM THE TTY		
12367								
12368								
12369								
12370								
12371								
12372								
12373								
12374								
12375								
12376	062626	011646			\$RDOCT:	MOV (SP)-,(SP)		;; PROVIDE SPACE FOR THE
12377	062630	016666	000004	000002	MOV	4(SP),2(SP)		;; INPUT NUMBER
12378	062636	010046			MOV	RO,-(SP)		;; PUSH RO ON STACK
12379	062640	010146			MOV	R1,-(SP)		;; PUSH R1 ON STACK
12380	062642	010246			MOV	R2,-(SP)		;; PUSH R2 ON STACK
12381	062644	104412			1\$:	RDLIN		;; READ AN ASCII LINE
12382	062646	012600			MOV	(SP)+,RO		;; GET ADDRESS OF 1ST CHARACTER
12383	062650	005001			CLR	R1		;; CLEAR DATA WORD
12384	062652	005002			CLR	R2		

CZRM08U RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 253
READ AN OCTAL NUMBER FROM THE TTY

SEQ 0253

12385 062654 112046
12386 062656 001412
12387 062660 006301
12388 062662 006102
12389 062664 006301
12390 062666 006102
12391 062670 006301
12392 062672 006102
12393 062674 042716 177770
12394 062700 062601
12395 062702 000764
12396 062704 005726
12397 062706 010166 000012
12398 062712 010237 062726
12399 062716 012602
12400 062720 012601
12401 062722 012600
12402 062724 000002
12403 062726 000000
12404
12405
12406
12407
12408
12409
12410
12411
12412 062730 016646 000002
12413 062734 042716 000020
12414 062740 012746 062746
12415 062744 000002
12416 062746 010046
12417 062750 016600 000002
12418 062754 005740
12419 062756 111000
12420 062760 006300
12421 062762 016000 063002
12422 062766 000200
12423
12424
12425
12426
12427 062770 011646
12428 062772 016666 000004 000002
12429 063000 000002
12430
12431
12432
12433
12434
12435
12436
12437
12438 063002 062770
12439 063004 060276
12440 063006 060074

```
2$:  MOVB  (RO)+, -(SP)  ;; PICKUP THIS CHARACTER
     BEQ    3$           ;; IF ZERO GET OUT
     ASL    R1           ;; *2
     ROL    R2
     ASL    R1           ;; *4
     ROL    R2
     ASL    R1           ;; *8
     ROL    R2
     BIC    #C7, (SP)    ;; STRIP THE ASCII JUNK
     ADD    (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  J
.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
        .STYPOC ;; CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
```

12441	063010	060050			\$TYP05	::CALL=TYP05	TRAP+3(104403)	TYPE OCTAL NUMBER (NO LEADING ZEROS)
12442	063012	060110			\$TYP0N	::CALL=TYP0N	TRAP+4(104404)	TYPE OCTAL NUMBER (AS PER LAST CALL)
12443	063014	057624			\$TYP0S	::CALL=TYP0S	TRAP+5(104405)	TYPE DECIMAL NUMBER (WITH SIGN)
12444	063016	057550			\$TYPBN	::CALL=TYPBN	TRAP+6(104406)	TYPE BINARY (ASCII) NUMBER
12445								
12446	063020	061756			\$GTSWR	::CALL=GTSWR	TRAP+7(104407)	GET SOFT-SWR SETTING
12447								
12448	063022	061666			\$CKSWR	::CALL=CKSWR	TRAP+10(104410)	TEST FOR CHANGE IN SOFT-SWR
12449	063024	062230			\$ROCHR	::CALL=ROCHR	TRAP+11(104411)	TTY TYPEIN CHARACTER ROUTINE
12450	063026	062320			\$ROLIN	::CALL=ROLIN	TRAP+12(104412)	TTY TYPEIN STRING ROUTINE
12451	063030	062626			\$RDOCT	::CALL=RDOCT	TRAP+13(104413)	READ AN OCTAL NUMBER FROM TTY
12452	063032	057454			\$SAVREG	::CALL=SAVREG	TRAP+14(104414)	SAVE R0-R5 ROUTINE
12453	063034	057512			\$RESREG	::CALL=RESREG	TRAP+15(104415)	RESTORE R0-R5 ROUTINE
12454								
12455					.SBTTL	POWER DOWN AND UP ROUTINES		
12456								
12457					::*****			
12458	063036	012737	063176	000024	\$PWRDN:	MOV \$SILLUP, 2#PWRVEC	::SET FOR FAST UP	
12459	063044	012737	000340	000026		MOV #340, 2#PWRVEC+2	::PRIO:7	
12460	063052	010046				MOV R0, -(SP)	::PUSH R0 ON STACK	
12461	063054	010146				MOV R1, -(SP)	::PUSH R1 ON STACK	
12462	063056	010246				MOV R2, -(SP)	::PUSH R2 ON STACK	
12463	063060	010346				MOV R3, -(SP)	::PUSH R3 ON STACK	
12464	063062	010446				MOV R4, -(SP)	::PUSH R4 ON STACK	
12465	063064	010546				MOV R5, -(SP)	::PUSH R5 ON STACK	
12466	063066	017746	116062			MOV 2SWR, -(SP)	::PUSH 2SWR ON STACK	
12467	063072	010637	063202			MOV SP, \$SAVR6	::SAVE SP	
12468	063076	012737	063110	000024		MOV \$PWRUP, 2#PWRVEC	::SET UP VECTOR	
12469	063104	000000				HALT		
12470	063106	000776				BR -2	::HANG UP	
12471								
12472					::*****			
12473					\$PWRUP:	MOV \$SILLUP, 2#PWRVEC	::SET FOR FAST DOWN	
12474	063110	012737	063176	000024		MOV \$SAVR6, SP	::GET SP	
12475	063116	013706	063202			CLR \$SAVR6	::WAIT LOOP FOR THE TTY	
12476	063122	005037	063202		1\$:	INC \$SAVR6	::WAIT FOR THE INC	
12477	063126	005237	063202			BNE 1\$::OF WORD	
12478	063132	001375				MOV (SP)+, 2SWR	::POP STACK INTO 2SWR	
12479	063134	012677	116014			MOV (SP)+, R5	::POP STACK INTO R5	
12480	063140	012605				MOV (SP)+, R4	::POP STACK INTO R4	
12481	063142	012604				MOV (SP)+, R3	::POP STACK INTO R3	
12482	063144	012603				MOV (SP)+, R2	::POP STACK INTO R2	
12483	063146	012602				MOV (SP)+, R1	::POP STACK INTO R1	
12484	063150	012601				MOV (SP)+, R0	::POP STACK INTO R0	
12485	063152	012600				MOV \$PWRDN, 2#PWRVEC	::SET UP THE POWER DOWN VECTOR	
12486	063154	012737	063036	000024		MOV #340, 2#PWRVEC+2	::PRIO:7	
12487	063162	012737	000340	000026		TYPE	::REPORT THE POWER FAILURE	
12488	063170	104401			\$PWRMG:	.WORD \$POWER	::POWER FAIL MESSAGE POINTER	
12489	063172	063204				RTI		
12490	063174	000002			\$SILLUP:	HALT	::THE POWER UP SEQUENCE WAS STARTED	
12491	063176	000000				BR -2	::BEFORE THE POWER DOWN WAS COMPLETE	
12492	063200	000776			\$SAVR6:	0	::PUT THE SP HERE	
12493	063202	000000			\$POWER:	.ASCIZ <15><12>"POWER"		
12494	063204	005015	047520	042527				
12495	063212	000122						
12496						.EVEN		


```

12497      .SBTTL  APT COMMUNICATIONS ROUTINE
12498
12499      ;*****
12500 063214 112737 000001 063460 $ATY1:  MOV  #1,$FFLG      ;; TO REPORT FATAL ERROR
12501 063222 112737 000001 063456 $ATY3:  MOV  #1,$MFLG      ;; TO TYPE A MESSAGE
12502 063230 000403          $ATYC          ;;
12503 063232 112737 000001 063460 $ATY4:  MOV  #1,$FFLG      ;; TO ONLY REPORT FATAL ERROR
12504 063240          $ATYC:
12505 063240          MOV  R0,-(SP)      ;; PUSH R0 ON STACK
12506 063242          MOV  R1,-(SP)      ;; PUSH R1 ON STACK
12507 063244 105737 063456          TSTB  $MFLG      ;; SHOULD TYPE A MESSAGE?
12508 063250 001450          BEQ  5$      ;; IF NOT: BR
12509 063252 122737 000001 001242      CMPB  #APTENV,$ENV      ;; OPERATING UNDER APT?
12510 063260 001031          BNE  3$      ;; IF NOT: BR
12511 063262 132737 000100 001243      BITB  #APTPOOL,$ENVM      ;; SHOULD SPOOL MESSAGES?
12512 063270 001425          BEQ  3$      ;; IF NOT: BR
12513 063272 017600 000004          MOV  24(SP),R0      ;; GET MESSAGE ADDR.
12514 063276 062766 000002 000004      ADD  #2,4(SP)      ;; BUMP RETURN ADDR.
12515 063304 005737 001222          1$:  TST  $MSGTYPE      ;; SEE IF DONE W/ LAST XMISSION?
12516 063310 001375          BNE  1$      ;; IF NOT: WAIT
12517 063312 010037 001236          MOV  R0,$MSGAD      ;; PUT ADDR IN MAILBOX
12518 063316 105720          2$:  TSTB  (R0)+      ;; FIND END OF MESSAGE
12519 063320 001376          BNE  2$
12520 063322 163700 001236          SUB  $MSGAD,R0      ;; SUB START OF MESSAGE
12521 063326 006200          ASR  R0      ;; GET MESSAGE LGTH IN WORDS
12522 063330 010037 001240          MOV  R0,$MSGGLT      ;; PUT LENGTH IN MAILBOX
12523 063334 012737 000004 001222      MOV  #4,$MSGTYPE      ;; TELL APT TO TAKE MSG.
12524 063342 000413          BR  5$
12525 063344 017637 000004 063370 3$:  MOV  24(SP),4$      ;; PUT MSG ADDR IN JSR LINKAGE
12526 063352 062766 000002 000004      ADD  #2,4(SP)      ;; BUMP RETURN ADDRESS
12527 063360 013746 177776          MOV  177776,-(SP)      ;; PUSH 177776 ON STACK
12528 063364 004737 060276          JSR  PC,$TYPE      ;; CALL TYPE MACRO
12529 063370 000000          4$:  .WORD  0
12530 063372          5$:
12531 063372 105737 063460          10$: TSTB  $FFLG      ;; SHOULD REPORT FATAL ERROR?
12532 063376 001416          BEQ  12$      ;; IF NOT: BR
12533 063400 005737 001242          TST  $ENV      ;; RUNNING UNDER APT?
12534 063404 001413          BEQ  12$      ;; IF NOT: BR
12535 063406 005737 001222          11$: TST  $MSGTYPE      ;; FINISHED LAST MESSAGE?
12536 063412 001375          BNE  11$      ;; IF NOT: WAIT
12537 063414 017637 000004 001221      MOV  24(SP),$FATAL      ;; GET ERROR #
12538 063422 062766 000002 000004      ADD  #2,4(SP)      ;; BUMP RETURN ADDR.
12539 063430 005237 001222          INC  $MSGTYPE      ;; TELL APT TO TAKE ERROR
12540 063434 105037 063460          12$: CLRB  $FFLG      ;; CLEAR FATAL FLAG
12541 063440 105037 063457          CLRB  $LFLG      ;; CLEAR LOG FLAG
12542 063444 105037 063456          CLRB  $MFLG      ;; CLEAR MESSAGE FLAG
12543 063450 012601          MOV  (SP)+,R1      ;; POP STACK INTO R1
12544 063452 012600          MOV  (SP)+,R0      ;; POP STACK INTO R0
12545 063454 000207          RTS  PC      ;; RETURN
12546 063456          000          $MFLG: .BYTE  0      ;; MESSG. FLAG
12547 063457          000          $LFLG: .BYTE  0      ;; LOG FLAG
12548 063460          000          $FFLG: .BYTE  0      ;; FATAL FLAG
12549          063462          .EVEN
12550          000200      APTSIZE=200
12551          000001      APTENV=001
12552          000100      APTPOOL=100

```

K04

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 256
APT COMMUNICATIONS ROUTINE

SEQ 0256

12553 000040
12554
12555

APTCSUP=040
.NLIST BEX

.SBTTL CONSOLE MESSAGES

```

063462 SCTMSG:
063462 005015 040503 047116 .ASCII <CR><LF>CANNOT RECOVER THE BAD SECTOR FILES FROM LAST <CR><LF>
063544 051124 041501 020113 .ASCII <CR><LF>CANNOT RECOVER THE BAD SECTOR FILES FROM LAST <CR><LF>
063572 051 .ASCII <CR><LF>CANNOT RECOVER THE BAD SECTOR FILES FROM LAST <CR><LF>
063573 075 000 .ASCII <CR><LF>CANNOT RECOVER THE BAD SECTOR FILES FROM LAST <CR><LF>
063575 015 025012 000 .ASCII <CR><LF>CANNOT RECOVER THE BAD SECTOR FILES FROM LAST <CR><LF>
063601 077 000 .ASCII <CR><LF>CANNOT RECOVER THE BAD SECTOR FILES FROM LAST <CR><LF>
063603 .ASCII <CR><LF>CANNOT RECOVER THE BAD SECTOR FILES FROM LAST <CR><LF>
063603 015 006412 052012 .ASCII <CR><LF>CANNOT RECOVER THE BAD SECTOR FILES FROM LAST <CR><LF>
063660 005015 044124 020105 .ASCII <CR><LF>CANNOT RECOVER THE BAD SECTOR FILES FROM LAST <CR><LF>
063736 005015 054524 042520 .ASCII <CR><LF>CANNOT RECOVER THE BAD SECTOR FILES FROM LAST <CR><LF>
063772 .ASCII <CR><LF>CANNOT RECOVER THE BAD SECTOR FILES FROM LAST <CR><LF>
063772 005015 044103 047101 .ASCII <CR><LF>CANNOT RECOVER THE BAD SECTOR FILES FROM LAST <CR><LF>
064071 015 052412 042523 .ASCII <CR><LF>CANNOT RECOVER THE BAD SECTOR FILES FROM LAST <CR><LF>
064130 005015 046522 031460 .ASCII <CR><LF>CANNOT RECOVER THE BAD SECTOR FILES FROM LAST <CR><LF>
064155 015 042412 052116 .ASCII <CR><LF>CANNOT RECOVER THE BAD SECTOR FILES FROM LAST <CR><LF>
064204 005015 042101 051104 .ASCII <CR><LF>CANNOT RECOVER THE BAD SECTOR FILES FROM LAST <CR><LF>
064236 005015 046522 031460 .ASCII <CR><LF>CANNOT RECOVER THE BAD SECTOR FILES FROM LAST <CR><LF>
064266 005015 047105 051124 .ASCII <CR><LF>CANNOT RECOVER THE BAD SECTOR FILES FROM LAST <CR><LF>
064312 005015 042101 051104 .ASCII <CR><LF>CANNOT RECOVER THE BAD SECTOR FILES FROM LAST <CR><LF>
064342 005015 046522 031460 .ASCII <CR><LF>CANNOT RECOVER THE BAD SECTOR FILES FROM LAST <CR><LF>
064376 005015 047105 051124 .ASCII <CR><LF>CANNOT RECOVER THE BAD SECTOR FILES FROM LAST <CR><LF>
064423 .ASCII <CR><LF>CANNOT RECOVER THE BAD SECTOR FILES FROM LAST <CR><LF>
064423 015 052012 050131 .ASCII <CR><LF>CANNOT RECOVER THE BAD SECTOR FILES FROM LAST <CR><LF>
064501 040 052516 041115 .ASCII <CR><LF>CANNOT RECOVER THE BAD SECTOR FILES FROM LAST <CR><LF>
064513 015 052012 051105 .ASCII <CR><LF>CANNOT RECOVER THE BAD SECTOR FILES FROM LAST <CR><LF>
064562 005015 044103 047101 .ASCII <CR><LF>CANNOT RECOVER THE BAD SECTOR FILES FROM LAST <CR><LF>
064621 015 051012 051505 .ASCII <CR><LF>CANNOT RECOVER THE BAD SECTOR FILES FROM LAST <CR><LF>
064647 015 047012 052117 .ASCII <CR><LF>CANNOT RECOVER THE BAD SECTOR FILES FROM LAST <CR><LF>

```

06-674

.EVEN

.SBTTL FUNCTION CODE TABLE

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

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

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

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

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

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

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

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

; BIT 08 IS NOT USED.

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

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

; BIT 05 IS NOT USED.

; BIT 04 IS NOT USED.

; BIT 03 IS NOT USED.

; BIT 02 IS NOT USED.

; BIT 01 IS NOT USED.

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

064674

FNCDTB:

;FUNCTION CODE TABLE

064674	020000	.WORD	OPI	:NOP
064676	130001	.WORD	OPI:ATA:ILF:IVC	:ILLEGAL FUNCTION (2)
064700	132000	.WORD	ATA:OPI:IVC:IAE	:SEEK
064702	130000	.WORD	ATA:OPI:IVC	:RECALIBRATE
064704	020000	.WORD	OPI	:DRIVE CLEAR
064706	030000	.WORD	OPI:IVC	:RELEASE
064710	130000	.WORD	OPI:ATA:IVC	:OFFSET
064712	130000	.WORD	OPI:ATA:IVC	:RETURN TO CENTERLINE
064714	020000	.WORD	OPI	:READ IN PRESET
064716	020000	.WORD	OPI	:PACK ACKNOWLEDGE
064720	130001	.WORD	OPI:ATA:ILF:IVC	:ILLEGAL FUNCTION (24)
064722	130001	.WORD	OPI:ATA:ILF:IVC	:ILLEGAL FUNCTION (26)
064724	132000	.WORD	ATA:OPI:IVC:IAE	:SEARCH
064726	130001	.WORD	OPI:ATA:ILF:IVC	:ILLEGAL FUNCTION (32)
064730	130001	.WORD	OPI:ATA:ILF:IVC	:ILLEGAL FUNCTION (34)
064732	130001	.WORD	OPI:ATA:ILF:IVC	:ILLEGAL FUNCTION (36)
064734	130001	.WORD	OPI:ATA:ILF:IVC	:ILLEGAL FUNCTION (40)
064736	130001	.WORD	OPI:ATA:ILF:IVC	:ILLEGAL FUNCTION (42)
064740	130001	.WORD	OPI:ATA:ILF:IVC	:ILLEGAL FUNCTION (44)
064742	130001	.WORD	OPI:ATA:ILF:IVC	:ILLEGAL FUNCTION (46)
064744	073300	.WORD	WCE:OPI:IVC:IAE:AOE:HCE:ECH	:WRITE CHECK DATA
064746	073300	.WORD	WCE:OPI:IVC:IAE:AOE:HCE:ECH	:WRITE CHECK HEADER AND DATA
064750	130001	.WORD	OPI:ATA:ILF:IVC	:ILLEGAL FUNCTION (54)
064752	130001	.WORD	OPI:ATA:ILF:IVC	:ILLEGAL FUNCTION (56)
064754	037200	.WORD	OPI:IVC:WLE:IAE:AOE:HCE	:WRITE DATA
064756	037000	.WORD	OPI:IVC:WLE:IAE:AOE	:WRITE HEADER AND DATA
064760	130001	.WORD	OPI:ATA:ILF:IVC	:ILLEGAL FUNCTION (64)
064762	130001	.WORD	OPI:ATA:ILF:IVC	:ILLEGAL FUNCTION (66)
064764	033300	.WORD	OPI:IVC:IAE:AOE:HCE:ECH	:READ DATA
064766	033300	.WORD	OPI:IVC:IAE:AOE:HCE:ECH	:READ HEADER AND DATA
064770	130001	.WORD	OPI:ATA:ILF:IVC	:ILLEGAL FUNCTION (74)
064772	130001	.WORD	OPI:ATA:ILF:IVC	:ILLEGAL FUNCTION (76)

CZRMD80 RM03/2 FCTNL TST 2
CZRMD8.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 260
ATTENTION (ATA) TABLE

SEQ 02E0

.SBTTL ATTENTION (ATA) TABLE

064774	001	ATNTBL: .BYTE	1.
064775	002	.BYTE	2.
064776	004	.BYTE	4.
064777	010	.BYTE	8.
065000	020	.BYTE	16.
065001	040	.BYTE	32.
065002	100	.BYTE	64.
065003	200	.BYTE	128.

.SBTTL DATA PATTERN TABLE

065004		RGDTPT:	
065004		MIXED:	
065004	000000	.WORD	0.
065006	000001	.WORD	1.
065010	000003	.WORD	3.
065012	000007	.WORD	7.
065014	000017	.WORD	15.
065016	000037	.WORD	31.
065020	000077	.WORD	63.
065022	000177	.WORD	127.
065024	000377	.WORD	255.
065026	000777	.WORD	511.
065030	001777	.WORD	1023.
065032	003777	.WORD	2047.
065034	007777	.WORD	4095.
065036	017777	.WORD	8191.
065040	037777	.WORD	16383.
065042	077777	.WORD	32767.
065044	177777	ONES: .WORD	65535.
065046	177777	.WORD	65535.
065050	077777	.WORD	32767.
065052	037777	.WORD	16383.
065054	017777	.WORD	8191.
065056	007777	.WORD	4095.
065060	003777	.WORD	2047.
065062	001777	.WORD	1023.
065064	000777	.WORD	511.
065066	000377	.WORD	255.
065070	000177	.WORD	127.
065072	000077	.WORD	63.
065074	000037	.WORD	31.
065076	000017	.WORD	15.
065100	000007	.WORD	7.
065102	000003	.WORD	3.
065104	000001	.WORD	1.
065106	000000	ZEROS: .WORD	0.
065110	000000	.WORD	0.
065112	000001	.WORD	1.
065114	000002	.WORD	2.
065116	000004	.WORD	4.
065120	000010	.WORD	8.
065122	000020	.WORD	16.
065124	000040	.WORD	32.
065126	000100	.WORD	64.
065130	000200	.WORD	128.
065132	000400	.WORD	256.
065134	001000	.WORD	512.
065136	002000	.WORD	1024.
065140	004000	.WORD	2048.
065142	010000	.WORD	4096.
065144	020000	.WORD	8192.
065146	040000	.WORD	16384.
065150	100000	.WORD	32768.
065152	100000	.WORD	32768.

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 262
DATA PATTERN TABLE

SEQ 0262

065154	040000	.WORD	16384.
065156	020000	.WORD	8192.
065160	010000	.WORD	4096.
065162	004000	.WORD	2048.
065164	002000	.WORD	1024.
065166	001000	.WORD	512.
065170	000400	.WORD	256.
065172	000200	.WORD	128.
065174	000100	.WORD	64.
065176	000040	.WORD	32.
065200	000020	.WORD	16.
065202	000010	.WORD	8.
065204	000004	.WORD	4.
065206	000002	.WORD	2.
065210	000001	.WORD	1.
065212	000000	.WORD	0.
065214	177777	.WORD	65535.
065216	177776	.WORD	65534.
065220	177774	.WORD	65532.
065222	177770	.WORD	65528.
065224	177760	.WORD	65520.
065226	177740	.WORD	65504.
065230	177700	.WORD	65472.
065232	177600	.WORD	65408.
065234	177400	.WORD	65280.
065236	177000	.WORD	65024.
065240	176000	.WORD	64512.
065242	174000	.WORD	63488.
065244	170000	.WORD	61440.
065246	160000	.WORD	57344.
065250	140000	.WORD	49152.
065252	100000	.WORD	32768.
065254	000000	.WORD	0.
065256	000000	.WORD	0.
065260	100000	.WORD	32768.
065262	140000	.WORD	49152.
065264	160000	.WORD	57344.
065266	170000	.WORD	61440.
065270	174000	.WORD	63488.
065272	176000	.WORD	64512.
065274	177000	.WORD	65024.
065276	177400	.WORD	65280.
065300	177600	.WORD	65408.
065302	177700	.WORD	65472.
065304	177740	.WORD	65504.
065306	177760	.WORD	65520.
065310	177770	.WORD	65528.
065312	177774	.WORD	65532.
065314	177776	.WORD	65534.
065316	177777	.WORD	65535.
065320	125252	.WORD	43690.
065322	152525	.WORD	43690./2
065324	125252	.WORD	43690.
065326	177777	.WORD	65535.
065330	177776	.WORD	65534.
065332	177775	.WORD	65533.

EARLY:

E05

CZRMDB0 RM03/2 FCTNL TST 2
CZRMDB.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 263
DATA PATTERN TABLE

SEQ 0263

065334	177773	.WORD	65531.
065336	177767	.WORD	65527.
065340	177757	.WORD	65519.
065342	177737	.WORD	65503.
065344	177677	.WORD	65471.
065346	177577	.WORD	65407.
065350	177377	.WORD	65279.
065352	176777	.WORD	65023.
065354	175777	.WORD	64511.
065356	173777	.WORD	63487.
065360	167777	.WORD	61439.
065362	157777	.WORD	57343.
065364	137777	.WORD	49151.
065366	077777	.WORD	32767.
065370	077777	.WORD	32767.
065372	137777	.WORD	49151.
065374	157777	.WORD	57343.
065376	167777	.WORD	61439.
065400	173777	.WORD	63487.
065402	175777	.WORD	64511.
065404	176777	.WORD	65023.
065406	177377	.WORD	65279.
065410	177577	.WORD	65407.
065412	177677	.WORD	65471.
065414	177737	.WORD	65503.
065416	177757	.WORD	65519.
065420	177767	.WORD	65527.
065422	177773	.WORD	65531.
065424	177775	.WORD	65533.
065426	177776	.WORD	65534.
065430	177777	.WORD	65535.
065432			

ENRGDT:

.SBTTL ERROR MESSAGE TABLE

065432	072026	000000	000000	EMT1:	.WORD	EMS1,0
065436	072075	072112	000000	EMT2:	.WORD	EMS2,0
065444	072075	072155	000000	EMT3:	.WORD	EMS3,0
065452	072220	072250	000000	EMT4:	.WORD	EMS4,0
065460	072220	072362	000000	EMT5:	.WORD	EMS5,0
065466	077055	074243	000000	EMT6:	.WORD	EMS6,0
065474	075011	077102	000000	EMT7:	.WORD	EMS7,0
065502	072315	000000	000000	EMT10:	.WORD	EMS10,0
065506	072362	000000	000000	EMT11:	.WORD	EMS11,0
065512	072424	072435	000000	EMT12:	.WORD	EMS12,0
065520	072476	072507	072520	EMT13:	.WORD	EMS13,0
065532	072572	074243	000000	EMT14:	.WORD	EMS14,0
065540	072424	072655	000000	EMT15:	.WORD	EMS15,EMS16,0
065546	072424	072700	073027	EMT16:	.WORD	EMS16,0
065556	072424	072714	073040	EMT17:	.WORD	EMS17,0
065566	072424	072742	073040	EMT18:	.WORD	EMS18,0
065576	072424	072771	073027	EMT19:	.WORD	EMS19,0
065606	072424	073006	073027	EMT20:	.WORD	EMS20,0
065616	072424	073050	073040	EMT21:	.WORD	EMS21,0
065626	072424	073077	073040	EMT22:	.WORD	EMS22,0
065636	072424	073126	073040	EMT23:	.WORD	EMS23,0
065646	072424	073154	073040	EMT24:	.WORD	EMS24,0
065656	072424	073225	073040	EMT25:	.WORD	EMS25,0
065666	072424	073254	073040	EMT26:	.WORD	EMS26,0
065676	072424	073303	073040	EMT27:	.WORD	EMS27,0
065706	072424	073331	073040	EMT28:	.WORD	EMS28,0
065716	072424	073360	073040	EMT29:	.WORD	EMS29,0
065726	072424	073406	073040	EMT30:	.WORD	EMS30,0
065736	072424	073435	073040	EMT31:	.WORD	EMS31,0
065746	072424	073464	073040	EMT32:	.WORD	EMS32,0
065756	072424	073537	073040	EMT33:	.WORD	EMS33,0
065766	074327	072632	000000	EMT34:	.WORD	EMS34,0
065774	074515	076223	074523	EMT35:	.WORD	EMS35,0
066004	076314	076324	074451	EMT36:	.WORD	EMS36,0
066016	073631	073747	074523	EMT37:	.WORD	EMS37,0
066026	074554	073747	074523	EMT38:	.WORD	EMS38,0
066036	074602	073747	074523	EMT39:	.WORD	EMS39,0
066046	074630	073747	074523	EMT40:	.WORD	EMS40,0
066056	074261	074243	000000	EMT41:	.WORD	EMS41,0
066064	072476	072520	074215	EMT42:	.WORD	EMS42,0
066074	072424	073602	073040	EMT43:	.WORD	EMS43,0
066106	073631	073747	074377	EMT44:	.WORD	EMS44,0
066124	073631	073747	074377	EMT45:	.WORD	EMS45,0
066142	073660	073747	074377	EMT46:	.WORD	EMS46,0
066152	076251	076266	073747	EMT47:	.WORD	EMS47,0
066164	073707	074451	074377	EMT48:	.WORD	EMS48,0
066202	077055	074461	074377	EMT49:	.WORD	EMS49,0
066212	074032	074377	075211	EMT50:	.WORD	EMS50,0
066230	074436	074032	074377	EMT51:	.WORD	EMS51,0
066250	076202	074377	075211	EMT52:	.WORD	EMS52,0
066264	000000			EMT53:	.WORD	EMS53,0
066266	076407	076452	074424	EMT54:	.WORD	EMS54,0
066306	073735	077502	077663	EMT55:	.WORD	EMS55,0
066326	077014	074705	074451	EMT56:	.WORD	EMS56,0

066340	077014	074705	074451	EMT67:	.WORD	EMS165	EMS103	EMS72	EMS171,0
066352	073602	072632	076707	EMT70:	.WORD	EMS46	EMS20	EMS163,0	
066362	074436	074630	076707	EMT71:	.WORD	EMS71	EMS101	EMS163,0	
066372	073631	074451	076707	EMT72:	.WORD	EMS47	EMS72	EMS163,0	EMS115,EMS140,EMS141,0
066410	073631	074451	076707	EMT73:	.WORD	EMS47	EMS72	EMS163,0	EMS115,EMS141,EMS72,0
066426	074032	073747	076707	EMT74:	.WORD	EMS56	EMS53	EMS163,0	
066436	074436	074032	076707	EMT75:	.WORD	EMS71	EMS53	EMS163,0	EMS115,EMS150,EMS152,EMS72,0
066456	073660	073747	076707	EMT76:	.WORD	EMS50	EMS53	EMS163,0	
066466	076251	076266	073747	EMT77:	.WORD	EMS50	EMS53	EMS163,0	
066500	074515	076223	076707	EMT100:	.WORD	EMS71	EMS53	EMS163,0	EMS115,EMS47,EMS70,0
066516	074515	076407	076707	EMT101:	.WORD	EMS71	EMS53	EMS163,0	EMS115,EMS56,EMS73,0
066534	077055	074461	076707	EMT102:	.WORD	EMS47	EMS53	EMS163,0	
066544	076372	076426	074476	EMT103:	.WORD	EMS47	EMS53	EMS163,0	
066556	073707	074476	076707	EMT104:	.WORD	EMS47	EMS53	EMS163,0	EMS115,EMS50,EMS70,0
066574	077014	074002	076707	EMT105:	.WORD	EMS47	EMS53	EMS163,0	
066604	077014	074554	076707	EMT106:	.WORD	EMS47	EMS53	EMS163,0	
066614	076314	076324	073747	EMT107:	.WORD	EMS47	EMS53	EMS163,0	EMS115,EMS143,EMS70,0
066634	075011	075051	075216	EMT110:	.WORD	EMS47	EMS53	EMS163,0	EMS116,EMS111,0
066646	075135	072155	000000	EMT111:	.WORD	EMS47	EMS53	EMS163,0	
066654	075135	072112	000000	EMT112:	.WORD	EMS47	EMS53	EMS163,0	
066662	075011	075211	075216	EMT113:	.WORD	EMS47	EMS53	EMS163,0	EMS116,EMS117,EMS114,0
066676	075135	075270	000000	EMT114:	.WORD	EMS47	EMS53	EMS163,0	
066704	075305	075745	075771	EMT115:	.WORD	EMS47	EMS53	EMS163,0	
066714	075350	075745	075771	EMT116:	.WORD	EMS47	EMS53	EMS163,0	
066724	075405	075745	075771	EMT117:	.WORD	EMS47	EMS53	EMS163,0	
066734	075450	075745	075771	EMT120:	.WORD	EMS47	EMS53	EMS163,0	
066744	075502	075745	075771	EMT121:	.WORD	EMS47	EMS53	EMS163,0	
066754	075545	075745	075771	EMT122:	.WORD	EMS47	EMS53	EMS163,0	
066764	076145	075745	075771	EMT123:	.WORD	EMS47	EMS53	EMS163,0	
066774	075647	075745	075771	EMT124:	.WORD	EMS47	EMS53	EMS163,0	
067004	075705	075745	075771	EMT125:	.WORD	EMS47	EMS53	EMS163,0	
067014	075305	075745	076014	EMT126:	.WORD	EMS47	EMS53	EMS163,0	
067026	075350	075745	076014	EMT127:	.WORD	EMS47	EMS53	EMS163,0	
067040	075405	075745	076014	EMT130:	.WORD	EMS47	EMS53	EMS163,0	
067052	075450	075745	076014	EMT131:	.WORD	EMS47	EMS53	EMS163,0	
067064	075502	075745	076014	EMT132:	.WORD	EMS47	EMS53	EMS163,0	
067076	075545	075745	076014	EMT133:	.WORD	EMS47	EMS53	EMS163,0	
067110	076145	075745	076014	EMT134:	.WORD	EMS47	EMS53	EMS163,0	
067122	075647	075745	076014	EMT135:	.WORD	EMS47	EMS53	EMS163,0	
067134	075705	075745	076014	EMT136:	.WORD	EMS47	EMS53	EMS163,0	
067146	075305	075745	076056	EMT137:	.WORD	EMS47	EMS53	EMS163,0	
067156	075405	075745	076056	EMT140:	.WORD	EMS47	EMS53	EMS163,0	
067166	075305	075745	076120	EMT141:	.WORD	EMS47	EMS53	EMS163,0	
067176	076145	075745	076120	EMT142:	.WORD	EMS47	EMS53	EMS163,0	
067206	075450	075745	076120	EMT143:	.WORD	EMS47	EMS53	EMS163,0	
067216	075502	075745	076120	EMT144:	.WORD	EMS47	EMS53	EMS163,0	
067226	075545	075745	076120	EMT145:	.WORD	EMS47	EMS53	EMS163,0	
067236	075705	075745	076120	EMT146:	.WORD	EMS47	EMS53	EMS163,0	
067246	076654	075745	076120	EMT147:	.WORD	EMS47	EMS53	EMS163,0	
067256	075647	075745	076120	EMT150:	.WORD	EMS47	EMS53	EMS163,0	
067266	076202	075211	076223	EMT151:	.WORD	EMS47	EMS53	EMS163,0	EMS70,0
067300	076251	075211	076266	EMT152:	.WORD	EMS47	EMS53	EMS163,0	EMS72,0
067312	076314	076324	076353	EMT153:	.WORD	EMS47	EMS53	EMS163,0	EMS115,EMS143,EMS72,0
067330	076314	076324	072632	EMT154:	.WORD	EMS47	EMS53	EMS163,0	EMS115,EMS143,EMS70,0
067346	076407	076452	076520	EMT155:	.WORD	EMS47	EMS53	EMS163,0	
067360	076372	076426	076520	EMT156:	.WORD	EMS47	EMS53	EMS163,0	

067372	076372	076426	076554	ENT157:	.WORD	EMS147,EMS151,EMS156,EMS157,0
067404	076624	076554	076607	ENT160:	.WORD	EMS161,EMS156,EMS160,0
067414	072771	073027	076554	ENT161:	.WORD	EMS125,EMS27,EMS156,EMS160,0
067426	073006	073027	076554	ENT162:	.WORD	EMS126,EMS27,EMS156,EMS160,0
067440	073631	076707	076202	ENT163:	.WORD	EMS47,EMS163,EMS140,0
067450	073631	072632	000000	ENT164:	.WORD	EMS47,EMS20,0
067456	074032	072632	000000	ENT165:	.WORD	EMS56,EMS20,0
067464	073602	072632	000000	ENT166:	.WORD	EMS46,EMS20,0
067472	100203	072632	000000	ENT167:	.WORD	EMS224,EMS20,0
067500	077055	076520	077030	ENT170:	.WORD	EMS167,EMS154,EMS166,0
067510	077055	076520	076572	ENT171:	.WORD	EMS167,EMS154,EMS157,0
067520	077055	076520	076574	ENT172:	.WORD	EMS167,EMS154,EMS155,0
067530	077131	075745	075771	ENT173:	.WORD	EMS171,EMS132,EMS133,0
067540	077131	075745	076014	ENT174:	.WORD	EMS171,EMS132,EMS134,EMS123,0
067552	075135	077304	000000	ENT175:	.WORD	EMS113,EMS177,0
067560	077326	077343	000000	ENT176:	.WORD	EMS200,EMS201,0
067566	077441	076223	073040	ENT177:	.WORD	EMS203,EMS141,EMS30,EMS202,0
067600	077441	073707	073040	ENT180:	.WORD	EMS203,EMS51,EMS30,EMS202,0
067612	077441	077454	073040	ENT181:	.WORD	EMS203,EMS204,EMS30,EMS202,0
067624	077441	073660	073040	ENT182:	.WORD	EMS203,EMS50,EMS30,EMS202,0
067636	077441	076266	073040	ENT203:	.WORD	EMS203,EMS143,EMS30,EMS202,0
067650	076407	074476	077411	ENT204:	.WORD	EMS150,EMS74,EMS202,EMS115,EMS152,EMS72,0
067666	073660	074705	074461	ENT205:	.WORD	EMS50,EMS103,EMS72,0
067676	074714	074461	077411	ENT206:	.WORD	EMS104,EMS73,EMS202,0
067706	074515	076223	075211	ENT207:	.WORD	EMS75,EMS141,EMS115,EMS140,0
067720	074515	076407	000000	ENT210:	.WORD	EMS75,EMS150,0
067726	073707	074451	075211	ENT211:	.WORD	EMS51,EMS72,EMS115,EMS50,EMS70,0
067742	076251	073747	075211	ENT212:	.WORD	EMS142,EMS53,EMS115,EMS143,EMS72,0
067756	073660	074705	073747	ENT213:	.WORD	EMS50,EMS103,EMS53,0
067766	073735	077502	077030	ENT214:	.WORD	EMS52,EMS205,EMS166,EMS206,EMS115,EMS51,EMS72,0
070006	073735	075246	074032	ENT215:	.WORD	EMS52,EMS117,EMS56,EMS163,0
070020	074032	073040	074243	ENT216:	.WORD	EMS56,EMS30,EMS64,0
070030	073602	073040	074243	ENT217:	.WORD	EMS46,EMS30,EMS64,0
070040	073050	073040	074243	ENT220:	.WORD	EMS31,EMS30,EMS64,0
070050	073631	073040	074243	ENT221:	.WORD	EMS47,EMS30,EMS64,0
070060	073735	075246	074554	ENT222:	.WORD	EMS52,EMS117,EMS77,0
070070	073735	075246	074215	ENT223:	.WORD	EMS52,EMS117,EMS63,0
070100	000000			ENT224:	.WORD	
070102	000000			ENT225:	.WORD	
070104	000000			ENT226:	.WORD	
070106	000000			ENT227:	.WORD	
070110	000000			ENT230:	.WORD	
070112	000000			ENT231:	.WORD	
070114	000000			ENT232:	.WORD	
070116	000000			ENT233:	.WORD	
070120	000000			ENT234:	.WORD	
070122	000000			ENT235:	.WORD	
070124	000000			ENT236:	.WORD	
070126	000000			ENT237:	.WORD	
070130	000000			ENT240:	.WORD	
070132	000000			ENT241:	.WORD	
070134	000000			ENT242:	.WORD	
070136	000000			ENT243:	.WORD	
070140	000000			ENT244:	.WORD	
070142	000000			ENT245:	.WORD	
070144	077055	075745	077541	ENT246:	.WORD	EMS167,EMS132,EMS207,0

070154	077055	075745	077566	EMT247:	.WORD	EMS167,EMS132,EMS210,EMS125,0
070166	077055	077577	077566	EMT250:	.WORD	EMS167,EMS211,EMS210,EMS207,EMS206,0
070202	077615	077640	000000	EMT251:	.WORD	EMS212,EMS213,0
070210	077014	077615	000000	EMT252:	.WORD	EMS165,EMS212,0
070216	076372	076426	076554	EMT253:	.WORD	EMS147,EMS151,EMS156,EMS210,EMS26,EMS27,0
070234	077441	074630	073040	EMT254:	.WORD	EMS203,EMS101,EMS30,0
070244	077441	077055	073040	EMT255:	.WORD	EMS203,EMS167,EMS30,0
070254	077441	074602	073040	EMT256:	.WORD	EMS203,EMS100,EMS30,0
070264	072424	073602	073040	EMT257:	.WORD	EMS11,EMS46,EMS30,EMS102,0
070276	073735	075246	074032	EMT260:	.WORD	EMS117,EMS56,EMS102,0
070310	073735	077502	100008	EMT261:	.WORD	EMS206,EMS202,EMS206,EMS115,EMS51,EMS72,0
070330	074714	074461	077411	EMT262:	.WORD	EMS104,EMS373,EMS302,0
070340	073660	073747	074666	EMT263:	.WORD	EMS50,EMS33,EMS102,0
070350	074032	073747	074666	EMT264:	.WORD	EMS50,EMS33,EMS102,0
070370	074436	074032	074666	EMT265:	.WORD	EMS71,EMS56,EMS102,EMS115,EMS150,EMS152,EMS70,0
070410	076251	076266	073747	EMT266:	.WORD	EMS71,EMS56,EMS102,EMS115,EMS150,EMS152,EMS72,0
070422	073660	076353	073747	EMT267:	.WORD	EMS142,EMS143,EMS53,EMS102,0
070440	073631	073747	074107	EMT270:	.WORD	EMS40,EMS146,EMS115,EMS52,EMS117,EMS46,0
070454	073631	073747	074107	EMT271:	.WORD	EMS47,EMS53,EMS102,EMS115,EMS140,0
070472	074515	076223	074666	EMT272:	.WORD	EMS47,EMS53,EMS102,EMS115,EMS141,EMS72,0
070510	073707	074476	074666	EMT273:	.WORD	EMS51,EMS74,EMS102,EMS115,EMS47,EMS73,0
070526	074215	073747	074107	EMT274:	.WORD	EMS51,EMS74,EMS102,EMS115,EMS50,EMS70,0
070544	075216	073747	073660	EMT275:	.WORD	EMS53,EMS57,EMS115,EMS41,EMS146,0
070556	073631	073747	074107	EMT276:	.WORD	EMS47,EMS53,EMS57,EMS41,EMS57,0
070572	073631	073747	074107	EMT277:	.WORD	EMS47,EMS53,EMS57,EMS115,EMS140,0
070610	074032	073747	074107	EMT300:	.WORD	EMS47,EMS53,EMS57,EMS115,EMS141,EMS72,0
070630	074436	074032	073747	EMT301:	.WORD	EMS56,EMS57,EMS115,EMS150,EMS152,EMS70,0
070652	077014	074602	074705	EMT302:	.WORD	EMS71,EMS56,EMS53,EMS57,EMS115,EMS150,EMS152,EMS72,0
070664	077014	074630	074705	EMT303:	.WORD	EMS165,EMS100,EMS103,EMS57,0
070676	077014	074554	074705	EMT304:	.WORD	EMS165,EMS101,EMS103,EMS57,0
070710	073602	073040	074243	EMT305:	.WORD	EMS46,EMS30,EMS64,EMS57,0
070722	073660	073747	074107	EMT306:	.WORD	EMS50,EMS53,EMS57,0
070732	073660	076353	075211	EMT307:	.WORD	EMS50,EMS146,EMS115,EMS52,EMS117,EMS46,EMS57,0
070752	076251	076266	073747	EMT310:	.WORD	EMS142,EMS143,EMS53,EMS57,0
070764	074714	074705	073747	EMT311:	.WORD	EMS104,EMS103,EMS53,EMS57,0
070776	074743	074705	073747	EMT312:	.WORD	EMS105,EMS103,EMS53,EMS57,0
071010	076314	076324	074705	EMT313:	.WORD	EMS144,EMS145,EMS103,EMS57,EMS115,EMS143,EMS70,0
071030	073406	074705	073747	EMT314:	.WORD	EMS42,EMS103,EMS53,EMS57,0
071042	073050	074705	073747	EMT315:	.WORD	EMS31,EMS103,EMS53,EMS57,0
071054	074436	073050	074705	EMT316:	.WORD	EMS71,EMS31,EMS103,EMS57,0
071066	073435	074705	074107	EMT317:	.WORD	EMS43,EMS103,EMS57,0
071076	073537	074705	074107	EMT320:	.WORD	EMS45,EMS103,EMS57,0
071106	073464	074705	074107	EMT321:	.WORD	EMS44,EMS103,EMS57,0
071116	074772	072632	000000	EMT322:	.WORD	EMS106,EMS20,0
071124	073254	074705	074107	EMT323:	.WORD	EMS36,EMS103,EMS57,0
071134	077204	073254	074705	EMT324:	.WORD	EMS173,EMS56,EMS103,EMS57,0
071146	077164	073254	074705	EMT325:	.WORD	EMS172,EMS56,EMS103,EMS57,0
071160	072476	077221	072520	EMT326:	.WORD	EMS13,EMS174,EMS15,EMS35,EMS53,EMS175,0
071176	076372	076426	074476	EMT327:	.WORD	EMS147,EMS151,EMS74,EMS175,0
071210	074327	073747	077227	EMT330:	.WORD	EMS66,EMS53,EMS175,0
071220	073126	074705	073747	EMT331:	.WORD	EMS33,EMS103,EMS53,EMS175,0
071232	073331	074705	073747	EMT332:	.WORD	EMS40,EMS103,EMS53,EMS57,0
071244	073307	074476	074107	EMT333:	.WORD	EMS51,EMS74,EMS57,EMS115,EMS50,EMS70,0
071262	074515	076223	074107	EMT334:	.WORD	EMS75,EMS141,EMS57,EMS115,EMS47,EMS73,0
071300	074515	076407	076452	EMT335:	.WORD	EMS75,EMS150,EMS152,EMS57,EMS115,EMS56,EMS73,0
071320	074135	074150	074177	EMT336:	.WORD	EMS60,EMS61,EMS62,0

J05

CZRMDBO RM03/2 FCTNL TST 2
CZRMDB.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 268
ERROR MESSAGE TABLE

SEQ 0268

071330	075216	075246	073154	EMT337: .WORD	EMS116,EMS117,EMS34,0
071340	073154	073747	073761	EMT340: .WORD	EMS34,EMS53,EMS54,EMS111,0
071352	074003	073154	000000	EMT341: .WORD	EMS55,EMS34,0
071360	073735	075246	074032	EMT342: .WORD	EMS52,EMS117,EMS56,EMS57,0
071372	073735	075246	073537	EMT343: .WORD	EMS52,EMS117,EMS45,EMS57,0
071404	073735	075246	073464	EMT344: .WORD	EMS52,EMS117,EMS44,EMS57,0
071416	073735	075246	100025	EMT345: .WORD	EMS52,EMS117,EMS221,0
071426	074772	072632	075211	EMT346: .WORD	EMS106,EMS20,EMS115,EMS223,EMS72,0
071442	073735	077502	100075	EMT347: .WORD	EMS52,EMS205,EMS222,EMS206,0
071454	074515	076407	074656	EMT350: .WORD	EMS75,EMS150,EMS102,EMS115,EMS56,EMS73,0
071472	077055	074461	074656	EMT351: .WORD	EMS167,EMS73,EMS102,0
071502	077701	000000		EMT352: .WORD	EMS215,0
071506	077752	077441	077722	EMT353: .WORD	EMS217,EMS203,EMS216,0
071516	100025	072632	000000	EMT354: .WORD	EMS221,EMS20,0

K05

CZRMD80 RM03/2 FCTNL TST 2
CZRMD8.P11 23-NOV-77 12:19MACY11 30(1046) 23-NOV-77 12:38 PAGE 269
ERROR MESSAGE TABLE

SEQ 0269

071524	100255	101071	101150	EHT1:	.WORD	EH1,STSH1,STSH2,STSH4,0
071536	101071	101150	101301	EHT2:	.WORD	STSH1,STSH2,STSH4,0
071546	100274	000000		EHT110:	.WORD	EH110,0
071552	100303	000000		EHT111:	.WORD	EH111,0
071556	100322	000000		EHT114:	.WORD	EH114,0
071562	100351	101071	101150	EHT223:	.WORD	EH223,STSH1,STSH2,STSH4,0
071574	100377	101071	101150	EHT256:	.WORD	EH256,STSH1,STSH2,STSH4,0
071606	100454	101071	101150	EHT336:	.WORD	EH336,STSH1,STSH2,STSH4,0
071620	100513	101071	101150	EHT337:	.WORD	EH337,STSH1,STSH2,STSH4,0
071632	100652	101071	101150	EHT344:	.WORD	EH344,STSH1,STSH2,STSH4,0
071644	101012	000000		EHT353:	.WORD	EH353,0

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 270
ERROR MESSAGE TABLE

SEQ 0270

071650	101342	101436	101454	EDT1:	.WORD	ED1,STSD1,STSD2,STSD4
071660	101436	101454	101506	EDT2:	.WORD	STSD1,STSD2,STSD4
071666	101350			EDT110:	.WORD	ED110
071670	101354			EDT111:	.WORD	ED111
071672	101362			EDT114:	.WORD	ED114
071674	101372	101436	101454	EDT223:	.WORD	ED223,STSD1,STSD2,STSD4
071704	101402	101436	101454	EDT336:	.WORD	ED336,STSD1,STSD2,STSD4
071714	101414	101436	101454	EDT337:	.WORD	ED337,STSD1,STSD2,STSD4
071724	101414	101436	101454	EDT344:	.WORD	ED337,STSD1,STSD2,STSD4,0
071736	101426			EDT353:	.WORD	ED353

M05

CZRM080 RM03/2 FCTNL TST 2 MACY11 30(1046) 23-NOV-77 12:38 PAGE 271
CZRM08.P11 23-NOV-77 12:19 ERROR MESSAGE TABLE

SEQ 0271

071740	101521	101537	101537	EFT1:	.WORD	EF111,STSF,STSF,STSF
071750	101537	101537	101537	EFT2:	.WORD	STSF,STSF,STSF
071756	101520			EFT110:	.WORD	EF110
071760	101521			EFT111:	.WORD	EF111
071762	101523			EFT114:	.WORD	EF114
071764	101523	101537	101537	EFT223:	.WORD	EF114,STSF,STSF,STSF
071774	101526	101537	101537	EFT336:	.WORD	EF336,STSF,STSF,STSF
072004	101526	101537	101537	EFT337:	.WORD	EF336,STSF,STSF,STSF
072014	101526	101537	101537	EFT344:	.WORD	EF336,STSF,STSF,STSF
072024	101523			EFT353:	.WORD	EF114

.SBTTL ERROR MESSAGE STRINGS

072026	051127	047117	020107	EMS1:	.ASC12	WRONG UNIT SELECTED (RMCS2, BITS 0-2) 2
072075	104	053105	041511	EMS2:	.ASC12	DEVICE WENT 2
072112	047125	053101	044501	EMS3:	.ASC12	UNAVAILABLE "DVA" (RMCS1, BIT 11) 2
072155	116	047117	054105	EMS4:	.ASC12	NONEXISTENT "MED" (RMCS2, BIT 12) 2
072220	047503	046515	047101	EMS5:	.ASC12	COMMAND NOT COMPLETED 2
072250	047503	052116	047522	EMS6:	.ASC12	CONTROLLER NOT READY (RMCS1, BIT 7) 2
072315	104	044522	042526	EMS7:	.ASC12	DRIVE NOT READY "DRY" (RMDS, BIT 7) 2
072362	047507	047040	052117	EMS10:	.ASC12	GO NOT RESET "GO" (RMCS1, BIT 0) 2
072424	047111	040526	044514	EMS11:	.ASC12	INVALID 2
072435	106	047125	052103	EMS13:	.ASC12	FUNCTION CODE (RMCS1, BITS 1-5) 2
072476	040515	051523	052103	EMS13:	.ASC12	MASSBUS 2
072507	103	047117	051124	EMS14:	.ASC12	CONTROL 2
072520	052502	020123	040520	EMS15:	.ASC12	BUS PARITY ERROR 2
072542	046442	050103	021106	EMS16:	.ASC12	"MCPE" (RMCS1, BIT 13) 2
072572	051124	047101	043123	EMS17:	.ASC12	TRANSFER ERROR (RMCS1, BIT 14) 2
072632	044123	052517	020114	EMS20:	.ASC12	SHOULD NOT BE SET 2
072655	127	051117	020104	EMS21:	.ASC12	WORD COUNT (RMWC) 2
072700	052502	020123	051050	EMS22:	.ASC12	BUS (RMB) 2
072714	046042	052103	020042	EMS23:	.ASC12	"LBT" (RMDS, BIT 10) 2
072742	040442	042517	020042	EMS24:	.ASC12	"AOE" (RMER1, BIT 09) 2
072771	104	051511	020111	EMS25:	.ASC12	"DISK" (RMDA) 2
073006	054503	044514	042116	EMS26:	.ASC12	CYLINDER (RMOC) 2
073027	101	042104	042527	EMS27:	.ASC12	ADDRESS 2
073040	052123	052101	051522	EMS30:	.ASC12	STATUS 2
073050	053442	042514	020042	EMS31:	.ASC12	"MLE" (RMER1, BIT 11) 2
073077	042	050125	021105	EMS32:	.ASC12	"LPE" (RMCS2, BIT 13) 2
073126	053442	043103	020042	EMS33:	.ASC12	"WCF" (RMER1, BIT 5) 2
073154	051127	052111	020105	EMS34:	.ASC12	WRITE CHECK ERROR "WCE" (RMCS2, BIT 14) 2
073225	042	042115	042520	EMS35:	.ASC12	"MOPE" (RMCS2, BIT 8) 2
073254	042042	045503	020042	EMS36:	.ASC12	"DCK" (RMER1, BIT 15) 2
073303	042	041505	021110	EMS37:	.ASC12	"ECH" (RMER1, BIT 6) 2
073331	042	046104	021124	EMS40:	.ASC12	"DLT" (RMCS2, BIT 15) 2
073360	046442	043130	020042	EMS41:	.ASC12	"MXF" (RMCS2, BIT 9) 2
073406	042042	042524	020042	EMS42:	.ASC12	"DTE" (RMER1, BIT 12) 2
073435	042	041510	041522	EMS43:	.ASC12	"HCRC" (RMER1, BIT 8) 2
073464	042510	042101	051105	EMS44:	.ASC12	HEADER COMPARE ERROR "HCE" (RMER1, BIT 7) 2
073537	106	051117	040515	EMS45:	.ASC12	FORMAT ERROR "FER" (RMER1, BIT 4) 2
073602	044442	042501	020042	EMS46:	.ASC12	"IAE" (RMER1, BIT 10) 2
073631	042	050117	021111	EMS47:	.ASC12	"OPT" (RMER1, BIT 13) 2
073660	051442	044513	020042	EMS50:	.ASC12	"SKT" (RMER2, BIT 14) 2
073707	042	044520	021120	EMS55:	.ASC12	"PIP" (RMDS, BIT 13) 2
073735	124	042510	051040	EMS56:	.ASC12	THE RM03 2
073747	104	052105	041505	EMS53:	.ASC12	DETECTED 2
073761	101	020124	047101	EMS54:	.ASC12	AT AN UNEXPECTED 2
074003	111	041516	051117	EMS55:	.ASC12	INCORRECT DATA DURING 2
074032	047111	040526	044514	EMS56:	.ASC12	INVALID COMMAND ERROR "IVC" (RMER2, BIT 12) 2
074107	104	051125	047111	EMS57:	.ASC12	DURING DATA TRANSFER 2
074135	104	052101	020101	EMS60:	.ASC12	DATA READ 2
074150	047504	051505	047040	EMS61:	.ASC12	DOES NOT COMPARE WITH 2
074177	104	052101	020101	EMS62:	.ASC12	DATA WRITTEN 2
074215	042	040520	021122	EMS63:	.ASC12	"PAR" (RMER1, BIT 3) 2
074243	111	020123	047111	EMS64:	.ASC12	IS INCORRECT 2
074261	103	046517	047520	EMS65:	.ASC12	COMPOSITE ERROR "ERR" (RMDS, BIT 14) 2
074327	104	052101	020101	EMS66:	.ASC12	DATA PARITY ERROR "DPE" (RMER2, BIT 3) 2

074377	104	051125	047111	EMS67:	.ASCIZ	2DURING SEEK COMMAND 2
074424	051511	051040	051505	EMS70:	.ASCIZ	2IS RESET 2
074436	051105	047522	047516	EMS71:	.ASCIZ	2ERRONEOUS 2
074451	111	020123	047523	EMS72:	.ASCIZ	2IS SET 2
074461	104	042111	047540	EMS73:	.ASCIZ	2DID NOT SET 2
074476	044504	020104	047516	EMS74:	.ASCIZ	2DID NOT RESET 2
074515	114	051517	020124	EMS75:	.ASCIZ	2LOST 2
074523	104	051125	047511	EMS76:	.ASCIZ	2DURING PACK ACK COMMAND 2
074554	051042	051115	020042	EMS77:	.ASCIZ	2"RA" (RMR1, BIT 2) 2
074602	044442	051114	020042	EMS78:	.ASCIZ	2"IL" (RMR1, BIT 1) 2
074630	044442	043114	020042	EMS79:	.ASCIZ	2"LF" (RMR1, BIT 0) 2
074656	052504	044522	047516	EMS80:	.ASCIZ	2DURING SEARCH COMMAND 2
074705	105	051122	051117	EMS81:	.ASCIZ	2ERROR 2
074714	046042	041502	020042	EMS82:	.ASCIZ	2"LC" (RMR2, BIT 10) 2
074743	042	051514	051103	EMS83:	.ASCIZ	2"SC" (RMR2, BIT 11) 2
074772	042510	042101	051103	EMS84:	.ASCIZ	2HEADER ERRORS 2
075011	102	051523	052040	EMS85:	.ASCIZ	2BUS TIMEOUT (04 TRAP) 2
075040	042101	051104	051505	EMS86:	.ASCIZ	2ADDRESS 2
075051	127	042510	020116	EMS87:	.ASCIZ	2WHEN READING/Writing RM REGISTERS 2
075113	101	020124	047516	EMS88:	.ASCIZ	2NOT THE FOLLOWING 2
075135	124	042510	051440	EMS89:	.ASCIZ	2THE SELECTED DEVICE IS 2
075165	116	047117	051440	EMS90:	.ASCIZ	2NONEXISTENT DEVICE 2
075211	040	006455	051440	EMS91:	.ASCIZ	2" -2(CR)<LF> 2
075216	044124	020105	047516	EMS92:	.ASCIZ	2THE MASSBUS CONTROLLER 2
075246	040506	046111	047516	EMS93:	.ASCIZ	2FAILED TO DETECT 2
075270	047516	020124	051440	EMS94:	.ASCIZ	2NOT AN RM03 2
075305	103	047117	051440	EMS95:	.ASCIZ	2CONTROL STATUS REGISTER 1, RMCS1, 2
075350	052502	020123	051440	EMS96:	.ASCIZ	2BUS ADDRESS REGISTER, RMAA, 2
075405	103	047117	051440	EMS97:	.ASCIZ	2CONTROL STATUS REGISTER 2, RMCS2, 2
075450	051105	047522	051440	EMS98:	.ASCIZ	2ERROR REGISTER 1, RMR1, 2
075502	052101	042524	051440	EMS99:	.ASCIZ	2ATTENTION SUMMARY REGISTER, RMAS, 2
075545	115	044501	051440	EMS100:	.ASCIZ	2MAINTENANCE REGISTER #1, RMR #1, 2
075610	041505	020103	051440	EMS101:	.ASCIZ	2ECC POSITION REGISTER, RMEC1, 2
075647	105	041503	051440	EMS102:	.ASCIZ	2ECC PATTERN REGISTER, RMEC2, 2
075705	115	044501	051440	EMS103:	.ASCIZ	2MAINTENANCE REGISTER 2, RMR2, 2
075745	116	052117	051440	EMS104:	.ASCIZ	2NOT INITIALIZED BY 2
075771	125	044516	051440	EMS105:	.ASCIZ	2UNIBUS INITIALIZE 2
076014	047503	052116	047522	EMS106:	.ASCIZ	2CONTROLLER CLEAR, I.E. BIT 5 OF 2
076056	044122	030461	047522	EMS107:	.ASCIZ	2"HI" ERROR CLEAR (RMCS1, BIT 14) 2
076120	051104	053111	020103	EMS108:	.ASCIZ	2DRIVE CLEAR COMMAND 2
076145	104	044522	047522	EMS109:	.ASCIZ	2DRIVE STATUS REGISTER, RMDS 2
076202	042515	044504	047522	EMS110:	.ASCIZ	2MEDIUM OFF LINE 2
076223	042	047515	021114	EMS111:	.ASCIZ	2"ML" (RMDS, BIT 12) 2
076251	104	044522	021114	EMS112:	.ASCIZ	2DRIVE FAULT 2
076266	042042	041526	020042	EMS113:	.ASCIZ	2"DVC" (RMR2, BIT 7) 2
076314	047125	040523	047522	EMS114:	.ASCIZ	2UNSAFE 2
076324	052442	051516	020042	EMS115:	.ASCIZ	2"UNS" (RMR1, BIT 14) 2
076353	123	047510	047522	EMS116:	.ASCIZ	2SHOULD BE SET 2
076372	043117	051506	052105	EMS117:	.ASCIZ	2OFFSET MODE 2
076407	040	047526	052105	EMS118:	.ASCIZ	2VOLUME VALID 2
076426	047442	021115	024040	EMS119:	.ASCIZ	2"OM" (RMDS, BIT 0) 2
076452	053042	021126	024040	EMS120:	.ASCIZ	2"VV" (RMDS, BIT 6) 2
076476	040520	045503	047522	EMS121:	.ASCIZ	2PACK ACK COMMAND 2
076520	047516	020124	047522	EMS122:	.ASCIZ	2NOT SET BY 2
076534	043117	051506	052105	EMS123:	.ASCIZ	2OFFSET COMMAND 2
076554	047516	020124	042522	EMS124:	.ASCIZ	2NOT RESET BY 2

076572	052122	020103	047503	EMS157:	.ASCIZ	2ATC COMMAND 2
076607	122	050111	041440	EMS160:	.ASCIZ	2RIP (2
076624	043117	051506	052105	EMS161:	.ASCIZ	2OFFSE1 2STER (RMOF) 2
076654	051105	047522	020122	EMS162:	.ASCIZ	2ROR R 2ER 22, RMER2, 2
076707	104	051125	047111	EMS163:	.ASCIZ	2DURING 2PRATE 2
076733	111	020123	047111	EMS164:	.ASCIZ	2IS INTERMITT 2R DRIVE DIDNT DROP ON 2
077002	054503	044514	042116		.ASCIZ	2CYLINDER 2
077014	047125	054105	042520	EMS165:	.ASCIZ	2UNEXPECTED 2
077030	042522	040503	044514	EMS166:	.ASCIZ	2CALIBRATE COMMAND 2
077055	042	052101	021101	EMS167:	.ASCIZ	2ATA" (RMOS, BIT15) 2
077102	044127	047106	051104	EMS170:	.ASCIZ	2OPEN READING REGISTER 2
077131	105	051122	051117	EMS171:	.ASCIZ	2ROR REGISTER 22, RMER2, 2
077164	047516	051116	041505	EMS172:	.ASCIZ	2UNRECOVERABLE 2
077204	042522	047503	042526	EMS173:	.ASCIZ	2RECOVERABLE 2
077221	104	052101	020101	EMS174:	.ASCIZ	2DATA 2
077227	104	051125	047111	EMS175:	.ASCIZ	2DURING WRITE COMMAND 2
077255	042	050117	021105	EMS176:	.ASCIZ	2OPE" (RMER2, BIT 13) 2
077304	047111	053440	044522	EMS177:	.ASCIZ	2IN WRITE PROTECT 2
077326	040503	020116	047516	EMS200:	.ASCIZ	2CAN NOT SET 2
077343	104	040511	047107	EMS201:	.ASCIZ	2DIAGNOSTIC MODE "DMD" (RMMR1, BIT 0) 2
077411	104	051125	047111	EMS202:	.ASCIZ	2DURING DIAGNOSTIC MODE 2
077441	111	041516	051117	EMS203:	.ASCIZ	2INCORRECT 2
077454	053442	046122	020042	EMS204:	.ASCIZ	2HRL" (RMOS, BIT 11) 2
077502	054105	041505	052125	EMS205:	.ASCIZ	2EXECUTED 2
077514	044527	044124	041440	EMS206:	.ASCIZ	2WITH COMP ERROR SET 2
077541	042	047507	020042	EMS207:	.ASCIZ	2GO" (RMCS1, BIT 0) 2
077566	051127	052111	047111	EMS210:	.ASCIZ	2WAITING 2
077577	127	051501	051040	EMS211:	.ASCIZ	2WAS RESET BY 2
077615	120	047522	051107	EMS212:	.ASCIZ	2PROGRAM INTERRUPT 2
077640	040527	020123	047516	EMS213:	.ASCIZ	2WAS NOT GENERATED 2
077663	123	042505	020113	EMS214:	.ASCIZ	2SEEK COMMAND 2
077701	120	047522	051107	EMS215:	.ASCIZ	2PROGRAM TIMEOUT 2
077722	052504	044522	042516	EMS216:	.ASCIZ	2DURING LOOK AHEAD TEST 2
077752	047514	045517	040440	EMS217:	.ASCIZ	2LOOK AHEAD REGISTER,RMLA, 2
100005	123	040505	041522	EMS220:	.ASCIZ	2SEARCH COMMAND 2
100025	102	042101	051440	EMS221:	.ASCIZ	2BAD SECTOR ERROR "BSE" (RMER2, BIT 15) 2
100075	101	042040	052101	EMS222:	.ASCIZ	2DATA TRANSFER COMMAND 2
100126	042510	042101	051105	EMS223:	.ASCIZ	2HEADER COMPARE INHIBIT "HCI" (RMOF, BIT 10) 2
100203	116	047117	051105	EMS224:	.ASCIZ	2NONEXISTENT MEMORY "NEM" (RMCS2, BIT 11) 2

100255	105	050130	052103	EH1:	.ASCIZ	RECEVD			
100274	052502	040523	051104	EH110:	.ASCIZ	RECEVD			
100303	040	046522	051503	EH111:	.ASCIZ	RMCS1	RMCS1		
100322	042522	042503	042126	EH114:	.ASCIZ	RECEVD	SNGPRT	DULPRT	
100351	105	050130	052103	EH223:	.ASCIZ	RECEVD	DATA		
100377	105	050130	052103	EH256:	.ASCIZ	RECEVD	RGSTR	(CR)(LF)	
100426	052123	052101	051525		.ASCIZ	STATUS	STATUS	INDEX	
100454	042107	042101	051522	EH336:	.ASCIZ	GDADRS	GDADRS	BDADRS	BDADRS
100513	122	041515	031123	EH337:	.ASCIZ	STATUS	STATUS	FAILING	DATA(CR)(LF)
100553	137	057537	057537		.ASCIZ	*****	*****	*****	*****
100613	105	050130	052103		.ASCIZ	RECEVD	RECEVD	BIT	ADRESS
100652	046522	051105	020061	EH344:	.ASCIZ	RMER1	STATUS	HEADER	FAILING(CR)(LF)
100713	137	057537	057537		.ASCIZ	*****	*****	WORD	BIT(CR)(LF)
100752	054105	041520	042124		.ASCIZ	RECEVD	RECEVD	NUMBER	POSITION
101012	054105	041520	042124	EH353:	.ASCIZ	RECEVD	RECEVD	(CR)(LF)	
101032	041474	037122	046074		.ASCIZ	(CR)(LF)	RMLA	RMLA	RMOF
101071	040	046522	051503	STSH1:	.ASCIZ	RMCS1	RMCS2	RMD5	RMER1
101136	020040	051040	040515		.ASCIZ	RMAS	(CR)(LF)		
101150	051040	053515	020103	STSH2:	.ASCIZ	RMWC	RMBA	RMDA	RMOF
101215	040	020040	051040		.ASCIZ	RMEC1	RMEC2	(CR)(LF)	
101241	040	046522	040504	STSH3:	.ASCIZ	RMDA	RMDC	RMOF	RMLA(CR)(LF)
101301	040	046522	051115	STSH4:	.ASCIZ	RMMR1	RMMR2	RMDT	RMSN(CR)(LF)

E06

CZRMDB0 RM03/2 FCTNL TST 2
CZRMDB.F11 23-NOV-77 12:19MACY11 30(1046) 23-NOV-77 12:38 PAGE 276
ERRCR MESSAGE STRINGS

SEQ 0276

	101342			.EVEN		
101342	001140	001142	000000	ED1:	.WORD	\$GDDAT,\$BDDAT,0
101350	001276	000000		ED110:	.WORD	\$BASE,0
101354	001174	001176	000000	ED111:	.WORD	\$TMP0,\$TMP1,0
101362	001356	001176	001200	ED114:	.WORD	RMDTI,\$TMP1,\$TMP2,0
101372	001140	001142	001174	ED223:	.WORD	\$GDDAT,\$BDDAT,\$TMP0,0
101402	001134	001140	001136	ED336:	.WORD	\$GDADR,\$GDDAT,\$BDADR,\$BDDAT,0
101414	001140	001142	001174	ED337:	.WORD	\$GDDAT,\$BDDAT,\$TMP0,\$TMP1,0
101426	001140	001142	001432	ED353:	.WORD	\$GDDAT,\$BDDAT,\$MOFO,0
101436	001330	001340	001342	STSD1:	.WORD	RMCS1I,RMCS2I,RMSI,RMER1I,RMER2I,RMASI,0
101454	001332	001334	001336	STSD2:	.WORD	RMWCI,RMBAI,RMDAI,RMOFI,RMOCI,RMECI
101470	001376	000000			.WORD	RMEC2I,0
101474	001336	001364	001362	STSD3:	.WORD	RMDAI,RMDCI,RMOFI,RMLAI,0
101506	001354	001370	001356	STSD4:	.WORD	RMMR1I,RMMR2I,RMDTI,RMSNI,0

CZRMDB0 RMO3/2 FCTNL TST 2
CZRMDB.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 277
ERROR MESSAGE STRINGS

F06

SEQ 0277

101520	000			EF110:	.BYTE	0
101521	000	000		EF111:	.BYTE	0,0
101523	000	000	000	EF114:	.BYTE	0,0,0
101526	000	000	000	EF336:	.BYTE	0,0,0,0
101532	000	000	000	EF337:	.BYTE	0,0,0,0,0
101537	000	000	000	STSF:	.BYTE	0,0,0,0,0,0,0

G06

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 278
ERROR MESSAGE STRINGS

SEQ 0278

101546 000402
102552 177777
102554 177777
102556 000402
103562 177777
103564 177777

.EVEN
MFGFIL: .BLKW 258.
 .WORD -1
 .WORD -1
USRFIL: .BLKW 258.
 .WORD -1
 .WORD -1

.EVEN

C2RMO80 RM03/2 FCTNL TST 2
C2RMO8.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 279
ERROR MESSAGE STRINGS

SEQ 0279

103566
103566 000402
104572 000402

BUFFER:
BUFONE: .BLKW 258.
BUFTWO: .BLKW 258.

103566 103566
103566 005015

= BUFFER

103570 046011 051511 020124
103610 057411 057537 057537
103630 030524 041411 047117
103663 124 004462 042504
103715 124 004463 051104
103741 124 004464 047506
103767 124 004465 047506
104015 124 004466 042532
104040 033524 043011 051117
104074 030524 004460 047506
104141 124 030461 043011
104167 124 031061 043011
104223 124 031461 043011
104270 030524 004464 047506
104325 124 032461 043011
104363 124 033061 043011
104424 030524 004467 047506
104460 031124 004460 047506
1044520 031124 004461 047506
104557 124 031062 043011
104613 124 031462 043011
104643 124 032063 043011
104674 031124 004465 047506
104737 124 033062 043011
105001 124 033463 043011
105046 031524 004460 047506
105074 031524 004461 053111
105121 124 031063 043011
105155 124 031463 043011
105211 124 032063 043011
105254 031524 004465 047506
105320 005015
105322 047411 042520 040522
105360 057411 057537 057537
105416 005015
105420 053523 052111 044103
105436 026455 026455 026455
105474 020040 032461 004411
105521 040 030440 004464
105545 040 030440 004463
105603 040 030440 004462
105613 040 030440 004461
105645 040 030440 004460
105672 020040 034440 004411
105717 040 020040 004470

```

HELP:
ASCII (CR)(LF)
ASCII CAUTION: PARTS 2 AND 3 OF THE FUNCTIONAL TEST LEAVE 2(CR)(LF)
ASCII 2BAD HEADERS ON THE PACK SO BE SURE TO FORMAT THE PACKS2(CR)(LF)
ASCII 2WHEN DONE 2(CR)(LF)
ASCII 2
ASCII 2LIST OF TESTS2(CR)(LF)
ASCII 2*****2(CR)(LF)
ASCII 2T1 CONTROLLER ACCESS TEST2(CR)(LF)
ASCII 2T2 DEVICE AVAILABLE TEST2(CR)(LF)
ASCII 2T3 DRIVE TYPE TEST2(CR)(LF)
ASCII 2T4 FORMAT ZEROS - 182(CR)(LF)
ASCII 2T5 FORMAT ZEROS - 162(CR)(LF)
ASCII 2T6 ZERO FILL TEST2(CR)(LF)
ASCII 2T7 FORMAT CHECK ZEROS - 162(CR)(LF)
ASCII 2T10 FORMAT CHECK ZEROS W/ HCE ERRORS2(CR)(LF)
ASCII 2T11 FORMAT ONES - 162(CR)(LF)
ASCII 2T12 FORMAT CHECK ONES - 162(CR)(LF)
ASCII 2T13 FORMAT CHECK ONES W/ HCE ERRORS2(CR)(LF)
ASCII 2T14 FORMAT MULTIPLE SECTORS2(CR)(LF)
ASCII 2T15 FORMAT W/ HEAD SWITCHING2(CR)(LF)
ASCII 2T16 FORMAT W/ MID TRANSFER SEEK2(CR)(LF)
ASCII 2T17 FORMAT W/ IMPLIED SEEK2(CR)(LF)
ASCII 2T20 FORMAT EACH SECTOR ADDRESS2(CR)(LF)
ASCII 2T21 FORMAT EACH TRACK ADDRESS2(CR)(LF)
ASCII 2T22 FORMAT PRIME CYLINDERS2(CR)(LF)
ASCII 2T23 FORMAT LAST SECTOR2(CR)(LF)
ASCII 2T24 FORMAT W/ AOE ERROR2(CR)(LF)
ASCII 2T25 FORMAT INVALID SECTOR ADDRESS2(CR)(LF)
ASCII 2T26 FORMAT INVALID TRACK ADDRESS2(CR)(LF)
ASCII 2T27 FORMAT INVALID CYLINDER ADDRESS2(CR)(LF)
ASCII 2T30 FORMAT AT OFFSET2(CR)(LF)
ASCII 2T31 IVC FORMAT TEST2(CR)(LF)
ASCII 2T32 FORMAT ERROR TEST - 182(CR)(LF)
ASCII 2T33 FORMAT ERROR TEST - 162(CR)(LF)
ASCII 2T34 FORMAT HCE, FIRST HEADER WORD2(CR)(LF)
ASCII 2T35 FORMAT HCE, SECOND HEADER WORD2(CR)(LF)
ASCII (CR)(LF)
ASCII 2 OPERATIONAL SWITCH SETTINGS2(CR)(LF)
ASCII 2*****2(CR)(LF)
ASCII (CR)(LF)
ASCII 2SWITCH
ASCII 2-----2(CR)(LF)
ASCII 215
ASCII 214 HALT ON ERROR2(CR)(LF)
ASCII 213 LOOP ON TEST2(CR)(LF)
ASCII 212 INHIBIT ERROR TYPEOUTS2(CR)(LF)
ASCII 211 2(CR)(LF)
ASCII 210 INHIBIT ITERATIONS2(CR)(LF)
ASCII 29 BELL ON ERROR2(CR)(LF)
ASCII 28 LOOP ON ERROR2(CR)(LF)
ASCII 28 LOOP ON TEST IN SWR(7:0)2(CR)(LF)

```

CZRMDB0 RM03/2 FCTNL TST 2
CZRMDB.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 280
ERROR MESSAGE STRINGS

SEQ 0280

105757	040	020040	004467	.ASCII	2	7	TN1282<CR><LF>
105774	020040	033040	004411	.ASCII	2	6	TN642<CR><LF>
106010	020040	032440	004411	.ASCII	2	5	TN322<CR><LF>
106024	020040	032040	004411	.ASCII	2	4	TN162<CR><LF>
106040	020040	031440	004411	.ASCII	2	3	TN82<CR><LF>
106053	040	020040	004462	.ASCII	2	2	TN42<CR><LF>
106066	020040	030440	004411	.ASCII	2	1	TN22<CR><LF>
106101	040	020040	004460	.ASCII	2	0	TN12<CR><LF>
106114	005015			.ASCII			<CR><LF>
106116	005015	000		.ASCII2			<CR><LF>
	000001			.END			

MACY11 30(1046) 23-NOV-77 12:38 PAGE 282
CROSS REFERENCE TABLE -- USER SYMBOLS

[illegible]

K06

CZRMDB0 RM03/2 FCTNL TST 2
CZRMDB.P11 23-NOV-77 12:19MACY11 30(1046) 23-NOV-77 12:38 PAGE 283
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0282

ARGS = 000004

ASND A 001500
ASND C 001476
ASWREG= 000000
ATA = 100000
ATESTN= 000000
ATNMSK= 000377
ATNTBL 064774
AUNIT = 000000
AUSWR = 000000
AVECT1= 120254
AVECT2= 000000
A16 = 000400
A17 = 001000
BACK = 000000

3669#	3706#	3738#	3745#	3752#	3781#	3788#	3795#	3804#	3837#	3874#	3906#	3913#
3920#	3949#	3956#	3963#	3972#	4005#	4042#	4075#	4082#	4089#	4119#	4126#	4133#
4142#	4175#	4212#	4244#	4251#	4258#	4286#	4293#	4300#	4335#	4372#	4404#	4411#
4418#	4449#	4460#	4523#	4560#	4592#	4599#	4606#	4635#	4642#	4649#	4658#	4691#
4728#	4760#	4767#	4774#	4802#	4809#	4816#	4851#	4888#	4920#	4927#	4934#	4965#
4976#	5039#	5076#	5108#	5115#	5122#	5150#	5157#	5164#	5198#	5235#	5267#	5274#
5281#	5309#	5316#	5323#	5357#	5394#	5426#	5433#	5440#	5468#	5475#	5482#	5516#
5555#	5588#	5595#	5602#	5630#	5637#	5644#	5681#	5719#	5726#	5733#	5762#	5769#
5776#	5785#	5826#	5864#	5871#	5878#	5907#	5914#	5921#	5930#	5971#	6009#	6016#
6023#	6052#	6059#	6066#	6075#	6104#	6152#	6159#	6166#	6188#	6234#	6241#	6248#
6283#	6321#	6328#	6335#	6344#	6375#	6382#	6389#	6430#	6468#	6475#	6482#	6491#
6522#	6529#	6536#	6577#	6615#	6622#	6629#	6638#	6669#	6676#	6683#	6723#	6773#
6780#	6787#	6835#	6842#	6849#	6858#	6898#	6949#	6963#	6972#	7003#	7017#	7051#
7089#	7096#	7103#	7133#	7144#	7155#	7189#	7227#	7234#	7241#	7271#	7282#	7293#
7328#	7375#	7382#	7389#	7418#	7433#	7448#	7468#	7486#	7513#	7564#	7611#	7618#
7625#	7654#	7665#	7683#	7708#								
1484#	8461#	8490#	8536#	8537#	8539#	8540#	8541#	8543#	8588#			
1483#	8460#	8489#	8544#	8545#	8547#	8587#						
1360#	1373#											
1012#	9323#	9324#	9326#	10058#	10059#	10094#	10097#	10589#	10590#	10629#	10632#	12555#
1360#	1364#											
1049#	10786#											
3466#	3484#	12555#										
1360#	1367#											
1360#	1374#											
1253#	1360#	1399#										
1360#	1400#											
1202#												
1201#												
3669#	3674#	3706#	3710#	3738#	3742#	3745#	3749#	3752#	3756#	3781#	3785#	3788#
3792#	3795#	3799#	3804#	3810#	3837#	3842#	3874#	3878#	3906#	3910#	3913#	3917#
3920#	3924#	3949#	3953#	3956#	3960#	3963#	3967#	3972#	3978#	4005#	4010#	4042#
4046#	4075#	4079#	4082#	4086#	4089#	4093#	4119#	4123#	4126#	4130#	4133#	4137#
4142#	4148#	4175#	4180#	4212#	4216#	4244#	4248#	4251#	4255#	4258#	4262#	4286#
4290#	4293#	4297#	4300#	4304#	4335#	4340#	4372#	4376#	4404#	4408#	4411#	4415#
4418#	4422#	4449#	4453#	4460#	4464#	4523#	4528#	4560#	4564#	4568#	4596#	4599#
4603#	4606#	4610#	4635#	4639#	4642#	4646#	4649#	4653#	4658#	4664#	4691#	4696#
4728#	4732#	4760#	4764#	4767#	4771#	4774#	4778#	4802#	4806#	4809#	4813#	4816#
4820#	4851#	4856#	4888#	4892#	4920#	4924#	4927#	4931#	4934#	4938#	4965#	4969#
4976#	4980#	5039#	5044#	5076#	5080#	5108#	5112#	5115#	5119#	5122#	5126#	5150#
5154#	5157#	5161#	5164#	5168#	5198#	5203#	5235#	5239#	5267#	5271#	5274#	5278#
5281#	5285#	5309#	5313#	5316#	5320#	5323#	5327#	5357#	5362#	5394#	5398#	5426#
5430#	5433#	5437#	5440#	5444#	5468#	5472#	5475#	5479#	5482#	5486#	5516#	5521#
5555#	5559#	5588#	5592#	5595#	5599#	5602#	5606#	5630#	5634#	5637#	5641#	5644#
5648#	5681#	5686#	5719#	5723#	5726#	5730#	5733#	5737#	5762#	5766#	5769#	5773#
5776#	5780#	5785#	5791#	5826#	5831#	5864#	5868#	5871#	5875#	5878#	5882#	5907#
5911#	5914#	5918#	5921#	5925#	5930#	5936#	5971#	5976#	6009#	6013#	6016#	6020#
6023#	6027#	6052#	6056#	6059#	6063#	6066#	6070#	6075#	6081#	6104#	6109#	6152#
6156#	6159#	6163#	6166#	6170#	6188#	6193#	6234#	6238#	6241#	6245#	6248#	6252#
6283#	6288#	6321#	6325#	6328#	6332#	6335#	6339#	6344#	6349#	6375#	6379#	6382#
6386#	6389#	6393#	6430#	6435#	6468#	6472#	6475#	6479#	6482#	6486#	6491#	6496#
6522#	6526#	6529#	6533#	6536#	6540#	6577#	6582#	6615#	6619#	6622#	6626#	6629#
6633#	6638#	6643#	6669#	6673#	6676#	6680#	6683#	6687#	6723#	6728#	6773#	6777#
6780#	6784#	6787#	6791#	6835#	6839#	6842#	6846#	6849#	6853#	6858#	6864#	6898#
6903#	6949#	6953#	6963#	6967#	6972#	6977#	7003#	7007#	7017#	7021#	7051#	7056#
7089#	7093#	7096#	7100#	7103#	7107#	7133#	7137#	7144#	7148#	7155#	7159#	7189#

		7194	7227	7231	7234	7238	7241	7245	7271	7275	7282	7286	7293	7297
		7328	7333	7375	7379	7382	7386	7389	7393	7418	7422	7433	7437	7448
		7452	7468	7472	7486	7490	7513	7518	7564	7569	7611	7615	7618	7622
		7625	7629	7654	7658	7665	7669	7683	7687	7708	7713			
BADSCY	034714	8177												
BAI	= 000010	1221	9712											
B800	= 000001	1139												
B801	= 000002	1138												
B802	= 000004	1137												
B803	= 000010	1136												
B804	= 000020	1135												
B805	= 000040	1134												
B806	= 000100	1133												
B807	= 000200	1132												
B808	= 000400	1131												
B809	= 001000	1130												
BIT0	= 000001	928	3313	3634	3636	8999								
BIT00	= 000001	918	928	954	1003	1022	1041	1079	1098	1139	1225	1241		
BIT01	= 000002	917	927	953	1002	1040	1078	1097	1138	1224	1240			
BIT02	= 000004	916	926	952	1001	1039	1077	1096	1137	1223	1239			
BIT03	= 000010	915	925	951	1000	1038	1076	1095	1136	1151	1221	1238		
BIT04	= 000020	914	924	950	999	1037	1094	1135	1220					
BIT05	= 000040	913	923	949	1036	1075	1093	1134	1219					
BIT06	= 000100	912	922	1021	1035	1057	1074	1092	1133	1204	1218	1237		
BIT07	= 000200	911	921	1020	1034	1056	1073	1091	1115	1132	1150	1203	1217	
BIT08	= 000400	910	920	998	1019	1033	1055	1072	1090	1131	1202	1215	11982	
BIT09	= 001000	909	919	997	1018	1032	1054	1071	1089	1130	1201	1214	12000	12098
BIT1	= 000002	927	8467	8625	9729									
BIT10	= 002000	908	996	1017	1031	1053	1070	1088	1114	1129	1149	1200	1213	1236
		12075												
BIT11	= 004000	907	948	1016	1030	1069	1087	1105	1113	1128	1148	1212	1235	8095
		12007												
BIT12	= 010000	906	1015	1029	1068	1086	1112	1127	1147	1211	1234	8045		
BIT13	= 020000	905	1014	1028	1067	1085	1104	1126	1146	1198	1210	1233	12082	
BIT14	= 040000	904	1013	1027	1066	1084	1103	1125	1145	1156	1197	1209	1232	8010
		11968												
BIT15	= 100000	903	1012	1026	1065	1083	1102	1124	1144	1155	1196	1208	1231	7997
		8710	8764											
BIT2	= 000004	926	8999											
BIT3	= 000010	925	8123	11171	11258									
BIT4	= 000020	924	8073	11277										
BIT5	= 000040	923												
BIT6	= 000100	922	8022											
BIT7	= 000200	921	3308	9005										
BIT8	= 000400	920												
BIT9	= 001000	919												
BOTADR	033736	7861	7879	7882	7897	7942								
BOTFLG	033740	7847	7889	7892	7895	7943								
BPTVEC	= 000014	935												
BSE	= 100000	1144	7446	7457	8373	9616								
BUFFER	103566	8192	8441	12555										
BUFOHE	103566	3661	3805	3829	3973	3997	4143	4167	4327	4515	4659	4683	4843	5031
		5190	5349	5508	5673	5786	5818	5931	5963	6076	6275	6422	6569	6715
		6959	6869	6890	7043	7181	7320	7337	7339	7341	7458	7477	7509	7556
		7573	7575	7577	7674	7705	12555							
BUFTWC	104572	3763	3806	3931	3974	4101	4144	4427	4474	4487	4617	4660	4943	4989

```

CC      = 004000
CH      = 002000
CHRCNT 033741
CKSWR   = 104410
CLEAN   = ***** U
CLKADR  001502
CLKVCT  001504
CLF     = 000040
CLRSTS  046776
CLSPRN  063572
CMNSTA  006772
CMPBUF  037064
CMPERR  057122
CNSLOO  064071
CNSLO1  064130
CNSLO2  064155
CNSLO3  064236
CNSLO4  064266
CNSLO5  064342
CNSLO6  064376
CNSLO7  064423
CNTCLR  046660
CNT     = 000100
CR      = 000015
CRLF   = 000200
CTLFG  001326
CYLMSK = 001777
DBCK    = 100000
DBEN    = 040000
DBL     = 002000
DCK     = 100000
DOISP   = 177570
DEBL    = 020000
DEVSEL  045206
DISPLA  001156
DISPRE  000174
DLT     = 100000
DMO     = 000001
DPE     = 000010
DPEHI   = 040000
DPELO  020000
DPR     = 000403
DRQ     = 004000
DRVCLR  = 000010
DRVSTS  052214
DRY     = 000200
DSWR    = 177570
DTASTS  053016

```

DTE = 010000
CTG = 010000
DULPRY = 024024

5002	5744	5787	5889	5932	6034	6077	6117	6201	6354	6501	6648	6812
6860	6982	7115	7253	7400	7459	7478	7636	7675	12555#			
1128#												
1129#												
7848#	7866*	7872*	7873	7876*	7880	7885*	7896*	7944#				
11966	12071	12097	12448#									
7462	12555											
1485#	8961*	8963	8967*	8969	8994							
1486#	8962*	8968*										
1219#	3315	10155										
8032	10191#											
3386	3402	3423	12555#									
3338	3473#											
3804	3972	4142	4658	5785	5930	6075	6858	8690#				
11570#												
3348	12555#											
3383	12555#											
3392	12555#											
3397	12555#											
3408	12555#											
3418	12555#											
3429	12555#											
3442	12555#											
3573	3617	8012	10147#									
1092#												
843#	3456	7864	11652	11941	11751	12555						
844#	3292	11912	11951									
1415#	3512*	11643	11657*									
1119#	8628											
1065#												
1066#												
1236#												
1026#	1043	8366	8698	9654	9674	11264	11267					
850#	1336	3256										
1067#												
7899	9857#											
1336#	3256*	3264*	12022*	12074*								
1263#	3264											
1208#	9498	11343	11346									
1079#	1098#	6907										
1151#	8367	8373	9160	9440	9936	10475	10896	11317	11320			
1232#												
1233#												
1019#	10097	10294	10295	10761								
1105#												
963#	8220	10749										
8238	10740#											
1020#	9127	9223	9226	9227	10097	10294	10295	10761				
849#	1335	3255										
3745	3788	3913	3956	4082	4126	4251	4293	4411	4460	4599	4642	4767
4809	4927	4976	5115	5157	5274	5316	5433	5475	5595	5637	5726	5769
5871	5914	6016	6059	6159	6241	6328	6382	6475	6529	6622	6676	6780
6842	7096	7144										

N06

CZRMDB0 RM03/2 FCTNL TST 2
CZRMDB.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 286
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0285

DVA = 004000	948#	3321	3592	8850	8923	9088	9091	9882	10200	10748	10749		
DVC = 000200	1150#	10029	10284	10536	10574	10577	10710	11083	11087	11090	11135	11509	11514
EARLY = 065320	11517												
EBL = 020000	12555#												
ECH = 000100	1085#												
ECI = 004000	1035#	1043	8366	8700	9637	9652	9670	9676	11274	12555			
ECRC = 001000	1113#	8702	9672	11271									
EDT1 = 071650	1084#												
	1520	1527	1534	1541	1548	1555	1569	1576	1583	1590	1597	1604	1611
	1618	1625	1632	1639	1646	1653	1660	1667	1674	1681	1688	1695	1702
	1709	1716	1723	1730	1737	1744	1751	1758	1765	1772	1779	1786	1793
	1800	1807	1815	1822	1829	1836	1843	1851	1859	1867	1881	1888	1895
	1902	1909	1916	1923	1931	1938	1945	1952	1959	1966	1973	1980	1987
	1994	2001	2008	2015	2057	2064	2071	2078	2085	2092	2099	2106	2113
	2120	2127	2134	2141	2148	2155	2162	2169	2176	2183	2190	2197	2204
	2211	2218	2225	2232	2239	2246	2253	2260	2267	2274	2281	2288	2295
	2302	2309	2316	2323	2330	2337	2344	2351	2358	2365	2372	2379	2386
	2408	2415	2422	2429	2436	2443	2450	2457	2464	2471	2478	2485	2492
	2513	2520	2527	2534	2541	2548	2555	2562	2569	2576	2583	2590	2597
	2758	2765	2772	2779	2786	2793	2800	2807	2814	2821	2828	2835	2842
	2854	2861	2868	2875	2882	2889	2896	2903	2910	2917	2924	2931	2938
	2949	2956	2963	2970	2977	2984	2991	2998	3005	3012	3019	3026	3033
	3040	3047	3054	3061	3068	3075	3082	3089	3096	3103	3110	3117	3124
	3180	12555#											
EDT110 071666	2022	12555#											
EDT111 071670	2029	2036	12555#										
EDT114 071672	2050	12555#											
EDT2 071660	2499	2506	2702	2709	12555#								
EDT223 071674	2548	2737	12555#										
EDT336 071704	2324	3082	3096	3103	12555#								
EDT337 071714	3089	12555#											
EDT344 071724	3124	12555#											
EDT353 071736	3173	12555#											
ED1 101342	12555#												
ED110 101350	12555#												
ED111 101354	12555#												
ED114 101362	12555#												
ED223 101372	12555#												
ED336 101402	12555#												
ED337 101414	12555#												
ED353 101426	12555#												
EECC = 000020	1094#												
EFT1 071740	1521	1528	1535	1542	1549	1556	1570	1577	1584	1591	1598	1605	1612
	1619	1626	1633	1640	1647	1654	1661	1668	1675	1682	1689	1696	1703
	1710	1717	1724	1731	1738	1745	1752	1759	1766	1773	1780	1787	1794
	1801	1808	1815	1823	1830	1837	1844	1852	1860	1868	1882	1889	1896
	1903	1910	1917	1924	1932	1939	1946	1953	1960	1967	1974	1981	1988
	1995	2002	2009	2016	2058	2065	2072	2079	2086	2093	2100	2107	2114
	2121	2128	2135	2142	2149	2156	2163	2170	2177	2184	2191	2198	2205
	2212	2219	2226	2233	2240	2247	2254	2261	2268	2275	2282	2289	2296
	2303	2310	2317	2324	2331	2338	2345	2352	2359	2366	2373	2380	2387
	2409	2416	2423	2430	2437	2444	2451	2458	2465	2472	2479	2486	2493
	2514	2521	2528	2535	2542	2549	2556	2563	2570	2577	2584	2591	2598
	2759	2766	2773	2780	2787	2794	2801	2808	2815	2822	2829	2836	2843
	2855	2862	2869	2876	2883	2890	2897	2904	2911	2918	2925	2932	2939
	2950	2957	2964	2971	2978	2985	2992	2999	3006	3013	3020	3027	3034

		3041	3048	3055	3062	3069	3076	3111	3118	3132	3139	3146	3153	3160
		3181	12555#											
EFT110	071756	2023	12555#											
EFT111	071760	2030	2037	12555#										
EFT114	071762	2051	12555#											
EFT2	071750	2500	2507	2703	2710	12555#								
EFT223	071764	2549	2738	12555#										
EFT336	071774	2325	3083	3097	3104	12555#								
EFT337	072004	3090	12555#											
EFT344	072014	3125	12555#											
EFT353	072024	3174	12555#											
EF110	101520	12555#												
EF111	101521	12555#												
EF114	101523	12555#												
EF336	101526	12555#												
EF337	101532	12555#												
EHT1	071524	1519	1526	1533	1540	1547	1554	1568	1575	1582	1589	1596	1603	1610
		1617	1624	1631	1638	1645	1652	1659	1666	1673	1680	1687	1694	1701
		1708	1715	1722	1729	1736	1743	1750	1757	1764	1771	1778	1785	1792
		1799	1806	1814	1821	1828	1835	1842	1850	1858	1866	1880	1887	1894
		1901	1908	1915	1922	1930	1937	1944	1951	1958	1965	1972	1979	1986
		1993	2000	2007	2014	2056	2063	2070	2077	2084	2091	2098	2105	2112
		2119	2126	2133	2140	2147	2154	2161	2168	2175	2182	2189	2196	2203
		2210	2217	2224	2231	2238	2245	2252	2259	2266	2273	2280	2287	2294
		2301	2308	2315	2330	2337	2344	2351	2365	2372	2379	2386	2393	2400
		2407	2414	2421	2428	2435	2442	2449	2456	2463	2470	2477	2484	2491
		2512	2519	2526	2533	2540	2580	2687	2694	2715	2722	2729	2743	2750
		2557	2564	2571	2578	2585	2592	2800	2807	2815	2822	2829	2837	2845
		2853	2862	2870	2878	2885	2892	2899	2906	2913	2920	2927	2934	2941
		2948	2955	2962	2969	2976	2983	2990	2997	3004	3011	3018	3025	3032
		3039	3046	3053	3060	3067	3074	3109	3116	3130	3137	3144	3151	3158
		3179	12555#											
EHT110	071546	2021	12555#											
EHT111	071552	2028	2035	12555#										
EHT114	071556	2049	12555#											
EHT2	071536	2498	2505	2701	2708	12555#								
EHT223	071562	2547	12555#											
EHT256	071574	2736	12555#											
EHT336	071606	2323	3081	3095	3102	12555#								

C07

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19MACY11 30(1046) 23-NOV-77 12:38 PAGE 288
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0287

EMS102	074656	12555#
EMS103	074705	12555#
EMS104	074714	12555#
EMS105	074743	12555#
EMS106	074772	12555#
EMS111	072424	12555#
EMS110	075011	12555#
EMS111	075040	12555#
EMS112	075051	12555#
EMS113	075135	12555#
EMS114	075165	12555#
EMS115	075211	12555#
EMS116	075216	12555#
EMS117	075246	12555#
EMS12	072435	12555#
EMS120	075270	12555#
EMS121	075305	12555#
EMS122	075350	12555#
EMS123	075405	12555#
EMS124	075450	12555#
EMS125	075502	12555#
EMS126	075545	12555#
EMS127	075610	12555#
EMS13	072476	12555#
EMS130	075647	12555#
EMS131	075705	12555#
EMS132	075745	12555#
EMS133	075771	12555#
EMS134	076014	12555#
EMS135	076056	12555#
EMS136	076120	12555#
EMS137	076145	12555#
EMS14	072507	12555#
EMS140	076202	12555#
EMS141	076223	12555#
EMS142	076251	12555#
EMS143	076266	12555#
EMS144	076314	12555#
EMS145	076324	12555#
EMS146	076353	12555#
EMS147	076372	12555#
EMS15	072520	12555#
EMS150	076407	12555#
EMS151	076426	12555#
EMS152	076452	12555#
EMS153	076476	12555#
EMS154	076520	12555#
EMS155	076534	12555#
EMS156	076554	12555#
EMS157	076572	12555#
EMS16	072542	12555#
EMS160	076607	12555#
EMS161	076624	12555#
EMS162	076654	12555#
EMS163	076727	12555#
EMS164	076753	12555#

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 289
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0288

EMS165	077014	125555#
EMS166	077030	125555#
EMS167	077055	125555#
EMS17	072572	125555#
EMS170	077102	125555#
EMS171	077131	125555#
EMS172	077164	125555#
EMS173	077204	125555#
EMS174	077221	125555#
EMS175	077227	125555#
EMS176	077255	125555#
EMS177	077304	125555#
EMS2	072075	125555#
EMS20	072632	125555#
EMS200	077326	125555#
EMS201	077343	125555#
EMS202	077411	125555#
EMS203	077441	125555#
EMS204	077454	125555#
EMS205	077502	125555#
EMS206	077514	125555#
EMS207	077541	125555#
EMS21	072655	125555#
EMS210	077566	125555#
EMS211	077577	125555#
EMS212	077615	125555#
EMS213	077640	125555#
EMS214	077663	125555#
EMS215	077701	125555#
EMS216	077722	125555#
EMS217	077752	125555#
EMS22	072700	125555#
EMS220	100005	125555#
EMS221	100025	125555#
EMS222	100075	125555#
EMS223	100126	125555#
EMS224	100203	125555#
EMS23	072714	125555#
EMS24	072742	125555#
EMS25	072771	125555#
EMS26	073006	125555#
EMS27	073027	125555#
EMS3	072112	125555#
EMS30	073040	125555#
EMS31	073050	125555#
EMS32	073077	125555#
EMS33	073126	125555#
EMS34	073154	125555#
EMS35	073225	125555#
EMS36	073254	125555#
EMS37	073303	125555#
EMS4	072155	125555#
EMS40	073331	125555#
EMS41	073360	125555#
EMS42	073406	125555#
EMS43	073435	125555#

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 290
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0289

EMS44	073464	12555#	
EMS45	073537	12555#	
EMS46	073602	12555#	
EMS47	073631	12555#	
EMS5	072220	12555#	
EMS50	073660	12555#	
EMS51	073707	12555#	
EMS52	073735	12555#	
EMS53	073747	12555#	
EMS54	073761	12555#	
EMS55	074003	12555#	
EMS56	074032	12555#	
EMS57	074107	12555#	
EMS6	072250	12555#	
EMS60	074135	12555#	
EMS61	074150	12555#	
EMS62	074177	12555#	
EMS63	074215	12555#	
EMS64	074243	12555#	
EMS65	074261	12555#	
EMS66	074327	12555#	
EMS67	074377	12555#	
EMS7	072315	12555#	
EMS70	074424	12555#	
EMS71	074436	12555#	
EMS72	074451	12555#	
EMS73	074461	12555#	
EMS74	074476	12555#	
EMS75	074515	12555#	
EMS76	074523	12555#	
EMS77	074554	12555#	
EMTVEC =	000030	938#	3239*
EMT1	065432	1518	12555#
EMT10	065502	1567	12555#
EMT100	066500	1964	12555#
EMT101	066516	1971	12555#
EMT102	066534	1978	12555#
EMT103	066544	1985	12555#
EMT104	066556	1992	12555#
EMT105	066574	1999	12555#
EMT106	066604	2006	12555#
EMT107	066614	2013	12555#
EMT11	065506	1574	12555#
EMT110	066634	2020	12555#
EMT111	066646	2027	12555#
EMT112	066654	2034	12555#
EMT113	066662	2041	12555#
EMT114	066676	2048	12555#
EMT115	066704	2055	12555#
EMT116	066714	2062	12555#
EMT117	066724	2069	12555#
EMT12	065512	1581	12555#
EMT120	066734	2076	12555#
EMT121	066744	2083	12555#
EMT122	066754	2090	12555#
EMT123	066764	2097	12555#

3240*

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 291
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0290

ENT124	066774	2104	12555#
ENT125	067004	2111	12555#
ENT126	067014	2118	12555#
ENT127	067026	2125	12555#
ENT13	065520	1588	12555#
ENT130	067040	2132	12555#
ENT131	067052	2139	12555#
ENT132	067064	2146	12555#
ENT133	067076	2153	12555#
ENT134	067110	2160	12555#
ENT135	067122	2167	12555#
ENT136	067134	2174	12555#
ENT137	067146	2181	12555#
ENT14	065532	1595	12555#
ENT140	067156	2188	12555#
ENT141	067166	2195	12555#
ENT142	067176	2202	12555#
ENT143	067206	2209	12555#
ENT144	067216	2216	12555#
ENT145	067226	2223	12555#
ENT146	067236	2230	12555#
ENT147	067246	2237	12555#
ENT15	065540	1602	12555#
ENT150	067256	2244	12555#
ENT151	067266	2251	12555#
ENT152	067300	2258	12555#
ENT153	067312	2265	12555#
ENT154	067330	2272	12555#
ENT155	067346	2279	12555#
ENT156	067360	2286	12555#
ENT157	067372	2293	12555#
ENT16	065546	1609	12555#
ENT160	067404	2300	12555#
ENT161	067414	2307	12555#
ENT162	067426	2314	12555#
ENT163	067440	12555#	
ENT164	067450	2329	12555#
ENT165	067456	2336	12555#
ENT166	067464	2343	12555#
ENT167	067472	2350	12555#
ENT17	065556	1616	12555#
ENT170	067500	12555#	
ENT171	067510	2364	12555#
ENT172	067520	2371	12555#
ENT173	067530	2378	12555#
ENT174	067540	2385	12555#
ENT175	067552	2392	12555#
ENT176	067560	2399	12555#
ENT177	067566	2406	12555#
ENT2	065436	1525	12555#
ENT20	065566	1623	12555#
ENT200	067600	2413	12555#
ENT201	067612	2420	12555#
ENT202	067624	2427	12555#
ENT203	067636	2434	12555#
ENT204	067650	2441	12555#

G07

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19MACY11 30(1046) 23-NOV-77 12:38 PAGE 292
CROSS REFERENCE TABLE -- USER SYMBOLS

SEG 0291

EMT205	067666	2448	12555#
EMT206	067676	2455	12555#
EMT207	067706	2462	12555#
EMT21	065576	1630	12555#
EMT210	067720	2469	12555#
EMT211	067726	2476	12555#
EMT212	067742	2483	12555#
EMT213	067756	2490	12555#
EMT214	067766	2497	12555#
EMT215	070006	2504	12555#
EMT216	070020	2511	12555#
EMT217	070030	2518	12555#
EMT22	065606	1637	12555#
EMT220	070040	2525	12555#
EMT221	070050	2532	12555#
EMT222	070060	2539	12555#
EMT223	070070	2546	12555#
EMT224	070100	12555#	
EMT225	070102	12555#	
EMT226	070104	12555#	
EMT227	070106	12555#	
EMT23	065616	1644	12555#
EMT230	070110	12555#	
EMT231	070112	12555#	
EMT232	070114	12555#	
EMT233	070116	12555#	
EMT234	070120	12555#	
EMT235	070122	12555#	
EMT236	070124	12555#	
EMT237	070126	12555#	
EMT24	065626	1651	12555#
EMT240	070130	12555#	
EMT241	070132	12555#	
EMT242	070134	12555#	
EMT243	070136	12555#	
EMT244	070140	12555#	
EMT245	070142	12555#	
EMT246	070144	2679	12555#
EMT247	070154	2686	12555#
EMT25	065636	1658	12555#
EMT250	070166	2693	12555#
EMT251	070202	2700	12555#
EMT252	070210	2707	12555#
EMT253	070216	2714	12555#
EMT254	070234	2721	12555#
EMT255	070244	2728	12555#
EMT256	070254	2735	12555#
EMT257	070264	2742	12555#
EMT26	065646	1665	12555#
EMT260	070276	2749	12555#
EMT261	070310	2756	12555#
EMT262	070330	2763	12555#
EMT263	070340	2770	12555#
EMT264	070350	2777	12555#
EMT265	070370	2784	12555#
EMT266	070410	2791	12555#

H07

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 293
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0292

ENT267	070422	2799	12555#
ENT27	065656	1672	12555#
ENT270	070440	2806	12555#
ENT271	070454	2814	12555#
ENT272	070472	2821	12555#
ENT273	070510	2828	12555#
ENT274	070526	2836	12555#
ENT275	070544	2844	12555#
ENT276	070556	2852	12555#
ENT277	070572	2861	12555#
ENT3	065444	1532	12555#
ENT30	065666	1679	12555#
ENT300	070610	2869	12555#
ENT301	070630	2877	12555#
ENT302	070652	2884	12555#
ENT303	070664	2891	12555#
ENT304	070676	2898	12555#
ENT305	070710	2905	12555#
ENT306	070722	2912	12555#
ENT307	070732	2919	12555#
ENT31	065676	1686	12555#
ENT310	070752	2926	12555#
ENT311	070764	2933	12555#
ENT312	070776	2940	12555#
ENT313	071010	2947	12555#
ENT314	071030	2954	12555#
ENT315	071042	2961	12555#
ENT316	071054	2968	12555#
ENT317	071066	2975	12555#
ENT32	065706	1693	12555#
ENT320	071076	2982	12555#
ENT321	071106	2989	12555#
ENT322	071116	2996	12555#
ENT323	071124	3003	12555#
ENT324	071134	3010	12555#
ENT325	071146	3017	12555#
ENT326	071160	3024	12555#
ENT327	071176	3031	12555#
ENT33	065716	1700	12555#
ENT330	071210	3038	12555#
ENT331	071220	3045	12555#
ENT332	071232	3052	12555#
ENT333	071244	3059	12555#
ENT334	071262	3066	12555#
ENT335	071300	3073	12555#
ENT336	071320	2322	12555#
ENT337	071330	3087	12555#
ENT34	065726	1707	12555#
ENT340	071340	3094	12555#
ENT341	071352	3101	12555#
ENT342	071360	3108	12555#
ENT343	071372	3115	12555#
ENT344	071404	3122	12555#
ENT345	071416	3129	12555#
ENT346	071426	3136	12555#
ENT347	071442	3143	12555#

12555#

CZRMDBO RM03/2 FCTNL TST 2
CZRMDB.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 294
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0293

ENT35	065736	1714	12555#
ENT350	071454	3150	12555#
ENT351	071472	3157	12555#
ENT352	071502	3164	12555#
ENT353	071506	3171	12555#
ENT354	071516	3178	12555#
ENT36	065746	1721	12555#
ENT37	065756	1728	12555#
ENT4	065452	1539	12555#
ENT40	065766	1735	12555#
ENT41	065774	1742	12555#
ENT42	066004	1749	12555#
ENT43	066016	1756	12555#
ENT44	066026	1763	12555#
ENT45	066036	1770	12555#
ENT46	066046	1777	12555#
ENT47	066056	1784	12555#
ENT5	065460	1546	12555#
ENT50	066064	1791	12555#
ENT51	066074	1798	12555#
ENT52	066106	1805	12555#
ENT53	066124	1813	12555#
ENT54	066142	1820	12555#
ENT55	066152	1827	12555#
ENT56	066164	1834	12555#
ENT57	066202	1841	12555#
ENT6	065466	1553	12555#
ENT60	066212	1849	12555#
ENT61	066230	1857	12555#
ENT62	066250	1865	12555#
ENT63	066264	12555#	
ENT64	066266	1879	12555#
ENT65	066306	1886	12555#
ENT66	066326	1893	12555#
ENT67	066340	1900	12555#
ENT7	065474	1560	12555#
ENT70	066352	1907	12555#
ENT71	066362	1914	12555#
ENT72	066372	1921	12555#
ENT73	066410	1929	12555#
ENT74	066426	1936	12555#
ENT75	066436	1943	12555#
ENT76	066456	1950	12555#
ENT77	066466	1957	12555#
ENRGDT	065432	12555#	
EQUALS	063573	12555#	
ERR =	040000	1013#	9269
ERRMB	033734	7828#	9271
ERRTYP	033176	7807#	9287
ERRVEC=	000004	931#	9321
		3580	7849
		8914	7941#
		8989	
		10151	
		7814	
		7822	

9269	9271	9287	9321										
7829#	7832	7844	7849	7941#									
12084	3254#	3265#	3533	3534	3535#	3536#	3547#	3548#	3552#	3553#	3579		
3253	3581#	3582#	3586#	3587#	3598#	3599#	8841	8842	8846#	8847#	8878#	8879#	
8915	8919#	8920#	8945#	8946#	8957	8958	8959#	8960#	8966#	8974#	8975#		
8990	8991#	8992#	9016#	9017#	9864	9865	9868#	9869#	9902#	9903#	10150		
10151	10152#	10153#	10164#	10165#	11973	11974#	11976#	11979#					
7814	7946#												
7822	7947#												

ERTY03 033742
ERTY01 033750

J07

CZRMDB0 RM03/2 FCTNL TST 2
CZRMDB.P11 23-NOV-77 12:19MACY11 30(1046) 23-NOV-77 12:38 PAGE 295
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0294

ERTY02 033760
ERTY03 033767
ERTY04 033775
ESRC = 004000
FER = 000020

FIND = 000001

FMT16 = 010000

FNCOTB 064674
FNCMSK= 000077
F0 = 000002
F1 = 000004
F2 = 000010
F3 = 000020
F4 = 000040
GENBUF 036620

GET 037516

7831	7949#												
7837	7951#												
7933	7953#												
1087#													
1037#	1043	7142	7152	7280	7290	7431	7441	8366	9601	11199	11216	11219	
11242	11245												
3669#	3671#	3706#	3707	3738#	3739	3745#	3746	3752#	3753	3781#	3782	3788#	
3789	3795#	3796	3804#	3807	3837#	3839#	3874#	3875	3906#	3907	3913#	3914	
3920#	3921	3949#	3950	3956#	3957	3963#	3964	3972#	3975	4005#	4007#	4042#	
4043	4075#	4076	4082#	4083	4089#	4090	4119#	4120	4126#	4127	4133#	4134	
4142#	4145	4175#	4177#	4212#	4213	4244#	4245	4251#	4252	4258#	4259	4286#	
4287	4293#	4294	4300#	4301	4335#	4337#	4372#	4373	4404#	4405	4411#	4412	
4418#	4419	4449#	4450	4460#	4461	4523#	4525#	4560#	4561	4592#	4593	4599#	
4600	4606#	4607	4635#	4636	4642#	4643	4649#	4650	4658#	4661	4691#	4693#	
4728#	4729	4760#	4761	4767#	4768	4774#	4775	4802#	4803	4809#	4810	4816#	
4817	4851#	4853#	4888#	4889	4920#	4921	4927#	4928	4934#	4935	4965#	4966	
4976#	4977	5039#	5041#	5076#	5077	5108#	5109	5115#	5116	5122#	5123	5150#	
5151	5157#	5158	5164#	5165	5198#	5200#	5235#	5236	5267#	5268	5274#	5275	
5281#	5282	5309#	5310	5316#	5317	5323#	5324	5357#	5359#	5394#	5395	5426#	
5427	5433#	5434	5440#	5441	5468#	5469	5475#	5476	5482#	5483	5516#	5518#	
5555#	5556	5588#	5589	5595#	5596	5602#	5603	5630#	5631	5637#	5638	5644#	
5645	5681#	5683#	5719#	5720	5726#	5727	5733#	5734	5762#	5763	5769#	5770	
5776#	5777	5785#	5788	5826#	5828#	5864#	5865	5871#	5872	5878#	5879	5907#	
5908	5914#	5915	5921#	5922	5930#	5933	5971#	5973#	6009#	6010	6016#	6017	
6023#	6024	6052#	6053	6059#	6060	6066#	6067	6075#	6078	6104#	6106#	6152#	
6153	6159#	6160	6166#	6167	6188#	6190	6234#	6235	6241#	6242	6248#	6249	
6283#	6285#	6321#	6322	6328#	6329	6335#	6336	6344#	6346	6375#	6376	6382#	
6383	6389#	6390	6430#	6432#	6468#	6469	6475#	6476	6482#	6483	6491#	6493#	
6522#	6523	6529#	6530	6536#	6537	6577#	6579#	6615#	6616	6622#	6623	6629#	
6630	6638#	6640#	6669#	6670	6676#	6677	6683#	6684	6723#	6725#	6773#	6774	
6780#	6781	6787#	6788	6835#	6836	6842#	6843	6849#	6850	6858#	6861	6898#	
6900#	6949#	6950	6963#	6964	6972#	6974#	7003#	7004	7017#	7018	7051#	7053#	
7089#	7090	7096#	7097	7103#	7104	7133#	7134	7144#	7145	7155#	7156	7189#	
7191#	7227#	7228	7234#	7235	7241#	7242	7271#	7272	7282#	7283	7293#	7294	
7328#	7330#	7375#	7376	7382#	7383	7389#	7390	7418#	7419	7433#	7434	7448#	
7449	7468#	7469	7486#	7487	7513#	7515#	7564#	7566#	7611#	7612	7618#	7619	
7625#	7626	7654#	7655	7665#	7666	7683#	7684	7708#	7710#				
1112#	3827	3995	4165	4325	4513	4681	4841	5029	5188	5347	5506	5671	
5816	5961	6115	6199	6420	6567	6713	6888	7113	7179	7318	7427	7507	
7554	7703	8202	8631	8633	8704	9961	11037						
9315	12555#												
955#	10748	11192											
953#	9214												
952#	9214												
951#	9214	9788											
950#	9214												
949#	9214												
3665	3833	4001	4171	4331	4519	4687	4847	5035	5194	5353	5512	5677	
5822	5967	6279	6426	6573	6719	6894	7047	7185	7324	7511	7560	7707	
8615#													
3626	3698	3730	3773	3866	3898	3941	4034	4067	4111	4204	4236	4278	
4364	4396	4441	4481	4552	4584	4627	4720	4752	4794	4880	4912	4957	
4996	5068	5100	5142	5227	5259	5301	5386	5418	5460	5547	5580	5622	
5711	5754	5856	5899	6001	6044	6144	6226	6313	6367	6460	6514	6607	
6661	6765	6827	6941	6995	7081	7125	7219	7263	7367	7410	7603	7646	
8025	8049	8076	8099	8125	8229	8262	8298	8332	8832#				

SEQ 0295

GETBUF		001330	1420#	8844										
GETINX		001506	1491#	3623*	3624*	4479*	4480*	4994*	4995*	8047*	8048*	8097*	8098*	8790
GETSTS		037432	3693	3861	4029	4199	4359	4547	4715	4875	5063	5222	5381	5542
			5852	5997	6141	6222	6309	6363	6456	6510	6603	6657	6761	6937
			7077	7215	7363	7599	8024	8075	8118	8216	8786#			6991
GMS	=	***** U	1262	3231	7766	7773	12439	12440	12441	12442	12443	12444	12446	12448
			12450	12451	12452	12453								12449
GO	=	000001	954#	3678	3715	3762	3846	3883	3930	4014	4052	4100	4184	4221
			4344	4381	4431	4532	4569	4616	4700	4737	4784	4860	4897	4947
			5085	5132	5207	5244	5291	5366	5403	5450	5527	5565	5612	5690
			5835	5888	5980	6033	6118	6202	6292	6353	6439	6500	6586	6647
			6732	6747	6795	6811	6918	6981	7060	7114	7198	7252	7346	7399
			7582	7635	7714	8059	8108	8220	8253	8287	8321	9002	9113	9125
			9146	9236	9239	11193								9130
GTSWR	=	104407	3286	3507	12446#									
HCE	=	000200	1034#	1043	7466	7476	7663	7673	8366	9564	9569	9584	9586	9599
			11199	11229	11232	11242	11245	12555						9614
HCI	=	002000	1114#	9562	11195									
HCRC	=	000400	1033#	1043	8366	9571	11199	11203	11206	11242	11245			
HELP		103566	3368	12555#										
HELPOS		063603	3362	12555#										
HT	=	000011	841#	7870	11910	11951								
IAE	=	002000	1031#	9468	9765	9780	9806	9952	9969	10468	10522	10525	11026	11041
IDXMSK	=	000077	1190#	8858	8861	8931	8934							12555
IE	=	000100	1204#	1237#										
ILF	=	000001	1041#	9340	9342	10362	10422	10426	10468	10489	10492	10982	10999	11002
ILF02	=	000002	960#											12555
ILF24	=	000024	970#											
ILF26	=	000026	971#											
ILF30	=	000030	973#											
ILF32	=	000032	974#											
ILF34	=	000034	975#											
ILF36	=	000036	976#											
ILF40	=	000040	977#											
ILF42	=	000042	978#											
ILF44	=	000044	979#											
ILF46	=	000046	980#											
ILF54	=	000054	983#											
ILF56	=	000056	984#											
ILF64	=	000064	987#											

[illegible]

PGM	=	001000	1018#	10760											
PHA	=	000200	1091#												
PIP	=	020000	1014#	8106	8284	10058	10079	10084	10589	10660	10665	10760	11356	11362	11367
			11455#	11492	11497										
PIRQ	=	177772	848#												
PIRQVE	=	000240	942#												
PLFS	=	002000	1088#												
PRIERR	=	040512	3738#	3781	3976	3949	4075	4119	4244	4286	4404	4449	4592	4635	4760
			4802#	4920	465	5108	5150	5267	5309	5426	5468	5588	5630	5719	5762
			5864#	5907	6009	6052	6152	6234	6321	6375	6468	6522	6615	6669	6773
			6835#	6949	7003	7089	7133	7227	7271	7375	7418	7611	7654	9061#	
			3443#	3452	12555#										
PROMPT	=	063575	865#												
PRO	=	000000	866#												
PR1	=	000040	867#												
PR2	=	000100	868#	11628											
PR3	=	000140	859#												
PR4	=	000200	870#												
PR5	=	000240	871#	3224	3514	3536	3582	8847	8920	8960	8992	9869	10153	11634	
PR6	=	000300	872#												
PR7	=	000340	845#	846											
FS	=	177776	1200#												
PSEL	=	002000	846#												
PSW	=	177776	3685#	3721	3764	3853	3889	3932	4021	4058	4102	4191	4227	4269	4351
PUT	=	037766	4387#	4432	4539	4575	4618	4707	4743	4785	4867	4903	4948	5055	5091
			5133#	5214	5250	5292	5373	5409	5451	5534	5571	5613	5699	5745	5844
			5890#	5989	6035	6133	6214	6301	6355	6448	6502	6595	6649	6739	6753
			6802#	6818	6910	6929	6983	7069	7116	7207	7254	7355	7401	7528	7591
			7637#	7723	8062	8111	8221	8254	8288	8322	8908#				
PUTBUF	=	001400	1447#	8192	8441#	8917									
PUTINX	=	001535	1496#	3679	3716	3847	3884	4015	4053	4185	4222	4345	4382	4533	4570
			4701#	4738	4861	4898	5049	5086	5208	5245	5367	5404	5528	5566	5691
			5836#	5981	6120	6203	6293	6440	6587	6733	6748	6796	6813	6908#	6909#
			6920#	7061	7199	7347	7520	7583	7715	8060#	8061#	8109#	8110#	8205	8918
			9165#												
PMRVEC	=	000024	937#	3243#	3244#	12458#	12459#	12468#	12474#	12486#	12487#				
QSTMRK	=	063601	3357#	3462	12555#										
RCLSTS	=	050452	8132#	830											

SEQ 0298

RMASO RMAA	= 001416 000004	1457# 1246# 5406 7066	3541# 5568 7204	3542 5696 7352	3718 5841 7525	3886 5986 7588	4055 6124 7719	4224 6208 8210	4384 6298	4572 6445	4740 6592	4900 6750	5088 6815	5247 6926
RMAAE RMAAI RMAAO	= 000050 001334 001404	1249# 1425# 1452# 5031# 6354# 7253#	4475 3661# 5190# 6422# 7320#	4990 3763# 5349# 6501# 7400#	9715 3829# 5508# 6569# 7509#	9717 3931# 5673# 6648# 7556#	10211 3997# 5744# 6715# 7636#	12555 4101# 5818# 6812# 7705#	4167# 4327# 5889# 6869# 8203#	4327# 4515# 5963# 6890# 8621	4515# 4617# 6034# 6982# 8708	4617# 4683# 6117# 7043# 9711	4683# 4843# 6201# 7115# 9714	4843# 6275# 7181#
RMCC RMCCI RMCCO RMCSI	= 000036 001366 001436 000000	1171# 1438# 1465# 1160# 4349 5407 6800 8922	1244# 4385 5532 6816 9001	3320 4537 5569 6927 9875	3538# 4573 5697 7067	3584 4705 5842 7205	3683 4741 5987 7353	3719 4865 6126 7526	3851 4901 6209 7589	3887 5053 6299 7721	4019 5089 6446 8060	4056 5212 6593 8109	4189 5248 6737 8211	4225 5371 6751 8849
RMCS1I RMCS1O	001330 001400	1423# 9236 1450# 4168# 4700# 5244# 5819# 6500# 7060# 7706#	9088 9238 3662# 4184# 4737# 5291# 5835# 6570# 7114# 7714#	9090 9249 3678# 4221# 4784# 5350# 5818# 6516# 7182# 8059#	9092 9292 3715# 4268# 4844# 5366# 5964# 6647# 7198# 8108#	9109 9698 3762# 4328# 4860# 5403# 5980# 6716# 7252# 8220#	9111 10198 3830# 4344# 4897# 5450# 6033# 6732# 7321# 8253#	9114 10747 3846# 4381# 4947# 5509# 6118# 6747# 7346# 8287#	9125 10880 3883# 4431# 5032# 5527# 6202# 6795# 7399# 8321#	9129 10882 3930# 4516# 5048# 5565# 6276# 6811# 7510# 8467	9131 10884 3998# 4532# 5085# 5612# 6292# 6891# 7519# 8625	9143 10903 4014# 4569# 5132# 5674# 6353# 6918# 7557# 9213	9145 12555 4052# 4616# 5191# 5690# 6423# 6981# 7582# 9729	9147 4100# 4684# 5207# 5743# 6439# 7044# 7635# 11171
RMCS2 RMCS2I	= 000010 001340	11191 1247# 1427# 9456 11343	3315# 4458 9499 11345	3316# 4466 9514 11347	3317 4468 9529 12555	3543# 4974 9640	3544 4982 9712	3585 4984 10220	8848 9070 10917	8921 9095 10919	9874# 9097 10921	9876 9098 11289	10155# 9284 11291	10157# 9355 11293
RMCS2O RMCS3 RMDA	001410 = 000052 = 000006	1454# 1250# 1161# 5692 7348	3680 5837 7521 9811	3848 5982 7584 12555	4016 6121 7716	4186 6204 8206	4346 6294	4534 6441	4702 6588	4862 6734	5050 6797	5209 6922	5368 7062	5529 7200
RMDAI RMDAO	001336 001406	1426# 1453# 5670# 6399# 7317# 9957	9811 3658# 5675 6400 7553# 9959	12555 3826# 5795# 6419# 8199# 9963	3994# 4164# 5796 6424 8380# 11031	4164# 4324# 5815# 6546# 8381 11033	4324# 4512# 5820 6547 8383 11035	4512# 4680# 5940# 6566# 8401 8797	4680# 4840# 5941 6712# 8409#	4840# 5028# 5960# 6717 8461	5028# 5187# 6114# 6887# 8588#	5187# 5346# 6198# 6892 8624	5346# 5505# 6272# 7040# 9737	5505# 6277 7178# 9738
RMD8 RMD8I RMD8O RMDC	= 000022 001352 001422 = 000034	1248# 1432# 1459# 1170# 5693 7349	3545# 4486 3681 5838 7522	3546 5001 3849 5983 7585	4017 6122 7717	4187 6205 8207	4347 6295	4535 6442	4703 6589	4863 6735	5051 6798	5210 6923	5369 7063	5530 7201
RMDCI RMDCO	001364 001434	1437# 1464# 5525 6565# 8623 1162#	9823 3657# 5526# 6571 9739	12555 3826# 5564# 6692# 9954	3993# 4163# 5669# 6693 11028	4163# 4323# 5814# 6711#	4323# 4511# 5959# 6886#	4511# 4679# 5965 7039#	4679# 4839# 6085# 7177#	4839# 5027# 6086 7316#	5027# 5186# 6113# 7552#	5186# 5345# 6197# 8200#	5345# 5504# 6271# 8460	5504# 6418# 8587#
RMDS	= 000012	1162# 8047	8097											

189*

527
923#

832

CZRM080 RM03/2 FCTNL TST 2
CZRM08.P11 23-NOV-77 12:19

MACY11 30(1046) 23-NOV-77 12:38 PAGE 302
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0301

STSH3	101241	12555#												
STSH4	101301	12555#												
SWR	001154	1335#	3233	3255*	3257	3263*	3270*	3284	3505	7809	11968	11982	11985	12000
		12007	12075	12082	12094	12098	12155	12194	12249*	12466	12479*			
SWREG	000176	1264#	3263	3284	3505	12155	12194	12217						
SW0	= 000001	900#												
SW00	= 000001	890#	900											
SW01	= 000002	889#	899											
SW02	= 000004	888#	898											
SW03	= 000010	887#	897											
SW04	= 000020	886#	896											
SW05	= 000040	885#	895											
SW06	= 000100	884#	894											
SW07	= 000200	883#	893											
SW08	= 000400	882#	892											
SW09	= 001000	881#	891											
SW1	= 000002	899#												
SW10	= 002000	880#												
SW11	= 004000	879#												
SW12	= 010000	878#												
SW13	= 020000	877#	7809											
SW14	= 040000	876#												
SW15	= 100000	875#												
SW2	= 000004	898#												
SW3	= 000010	897#												
SW4	= 000020	896#												
SW5	= 000040	895#												
SW6	= 000100	894#												
SW7	= 000200	893#												
SW8	= 000400	892#												
SW9	= 001000	891#												
TADMSK	= 003400	1007#												
TAG	= 020000	1126#												
TAGADR	= 001114	1307#	1314											
TAP	= 040000	1103#												
TA1	= 000400	998#												
TA2	= 001000	997#												
TA4	= 002000	996#												
TBITVE	= 000014	933#												
TIMOUT	040326	3694	3729	3772	3862	3897	3940	4030	4066	4110	4200	4235	4277	4360
		4395	4440	4548	4583	4626	4716	4751	4793	4876	4911	4956	5064	5099
		5141	5223	5258	5300	5382	5417	5459	5543	5579	5621	5710	5753	5855
		5898	6000	6043	6143	6225	6312	6366	6459	6513	6606	6660	6764	6826
		6940	6994	7080	7124	7218	7262	7366	7409	7534	7602	7645	7729	8119
		8297	8331	8985#										
TKVEC	= 000060	940#	12132*	12133*										
TPVEC	= 000064	941#												
TRAPVE	= 000034	939#	3241*	3242*										
TRE	= 040000	1197#	9113	9130	9146	9291	9293	9698						
TRIVEC	= 000014	934#												
TST	= 010000	1127#	10272	10814										
TSTNMB	033732	7820*	7821*	7823	7940#									
TSTPRP	034000	3669	3837	4005	4175	4335	4523	4691	4851	5039	5198	5357	5516	5681
		5826	5971	6104	6188	6283	6344	6430	6491	6577	6638	6723	6898	6972
		7051	7189	7328	7513	7564	7708	7988#						
TSTQUE	001450	1475#	3477	3478*	3518	3530	3570	3614	3652	3820	3988	4158	4318	4506

		4674	4834	5022	5181	5340	5499	5664	5809	5954	6099	6184	6266	6413
		6560	6706	6881	7034	7172	7310	7546	7736	7738*	7742*	9871	10156	10223
		10784												
TST1	007162	3523#	12027											
TST10	012716	4313#	12034											
TST11	013636	4501#	12035											
TST12	014436	4669#	12036											
TST13	015212	4829#	12037											
TST14	016130	5017#	12038											
TST15	016704	5176#	12039											
TST16	017460	5335#	12040											
TST17	020234	5494#	12041											
TST2	007360	3565#	12028											
TST20	021036	5659#	12042											
TST21	021562	5804#	12043											
TST22	022306	5949#	12044											
TST23	023030	6094#	12045											
TST24	023326	6179#	12046											
TST25	023624	6261#	12047											
TST26	024352	6408#	12048											
TST27	025100	6555#	12049											
TST3	007544	3609#	12029											
TST30	025626	6701#	12050											
TST31	026520	6876#	12051											
TST32	027326	7029#	12052											
TST33	030050	7167#	12053											
TST34	030572	7305#	12054											
TST35	031750	7541#	12055											
TST4	007726	3647#	12030											
TST5	010526	3815#	12031											
TST6	011326	3983#	12032											
TST7	012142	4153#	12033											
TYPBN =	104406	7930	12444#											
TYPOS =	104405	7770	7777	7924	12443#									
TYPE =	104401	3278	3298	3326	3348	3351	3357	3362	3365	3368	3372	3375	3383	3386
		3392	3397	3402	3408	3418	3423	3429	3442	3443	3446	3452	3455	3462
		7764	7771	7778	7813	7814	7822	7831	7837	7846	7884	7890	7894	7907
		7913	7915	7933	7935	11646	11720	11786	11855	11915	12077	12085	12149	12161
		12215	12216	12219	12230	12240	12251	12270	12318	12324	12329	12333	12338	12339
		12341	12344	12348	12439#	12488								
TYPOC =	104402	3385	7927	12218	12440#									
TYPON =	104404	12442#												
TYPOS =	104403	3328	3399	3420	7817	7825	7834	7840	12441#					

CZRMO80 RM03/2 FCTNL TST 2 MACY11 30(1046) 23-NOV-77 12:38 PAGE 304
CZRMO8.P11 23-NOV-77 12:19 CROSS REFERENCE TABLE -- USER SYMBOLS

```
WCD      = 000050
WCE      = 040000
WCEHI    = 010000
WCELO    = 004000
WCF      = 000040
WCH      = 000052
WD        = 000060
WH        = 000062
```

981#													
1209#	4458	4467	4974	4983	9354	12555							
1234#													
1235#													
1036#	1043	9425	11330	11333									
982#	4268	4431	4784	4947	5132	5291	5450	5612					
985#													
986#	3662	3715	3830	3883	3998	4052	4168	4221	4328	4381	4516	4569	
4684	4737	4844	4897	5032	5085	5191	5244	5350	5403	5509	5565	5674	
5690	5819	5835	5964	5980	6276	6292	6423	6439	6570	6586	6716	6747	
6891	6918	7044	7060	7182	7198	7321	7346	7510	7519	7557	7582	7706	

```
WLE      = 004000
WRL      = 004000
XDPMG    064562
XSIZ     006014
ZEROS    065106
```

7714	11193											
1030#	1043	9406	9410	9423	9438	9453	11130	11162	11166	11174	12555	
1016#	9408	11169										
3298	12555#											
3308#	3449											
3663	3831	3999	4169	4329	5033	5192	5351	5510	7045	7183	7322	7558

```
SAPTHD      001100
$ASTAT=     ***** U
$ATYC       063240
$ATY1       063214
$ATY3       063222
$ATY4       063232
$AUTOR      001150
$BASE       001276
```

12555#	1294	1300#										
12531	12546											
12502	12504#											
12500#												
11900	12501#											
12090	12503#											
1332#	3288#	3509#	12208	12365								
1401#	3314	3384	3394#	3529	3569	3613	3651	3819	3987	4157	4317	4505
4673	4833	5021	5190	5339	5498	5663	5808	5953	6098	6183	6265	6412
6559	6705	6880	7023	7171	7309	7545	8843	8916	8993	9870	10154	12555
1327#	4475#	4476#	4489	4990#	4991#	5004	8758#	12555				
1329#	4468#	4486#	4493	4984#	5001#	5009	6958#	7012#	7150#	7288#	7439#	7455#
7474#	7671#	8756#	9070#	9071#	9074	9092#	9098#	9114#	9131#	9147#	9175#	9225#
9226#	9238#	9239#	9249#	9250#	9252	9270#	9271#	9272	9292#	9293#	9294	9325#
9326#	9327	9341#	9342#	9343	9355#	9356#	9368#	9369#	9385#	9386#	9411#	9412#
9426#	9427#	9441#	9442#	9456#	9457#	9469#	9470#	9499#	9500#	9514#	9515#	9529#
9530#	9546#	9547#	9572#	9573#	9587#	9588#	9602#	9603#	9617#	9618#	9640#	9641#
9655#	9656#	9677#	9678#	9696#	9717#	9768#	9769#	9770	9795#	9796#	9797	9811#
9812	9823#	9824	9941#	9968#	9969#	9970	9990#	10001#	10002#	10008	10019	10032#
10045#	10070#	10084#	10085#	10098#	10114#	10198#	10199#	10201	10211#	10220#	10228	10238#
10239#	10248#	10249#	10251	10261#	10270#	10271#	10273	10283#	10284#	10293#	10294#	10296
10338#	10352#	10368#	10382#	10396#	10410#	10424#	10479#	10493#	10509#	10526#	10547#	10564#
10578#	10603#	10618#	10633#	10649#	10666#	10684#	10698#	10714#	10747#	10748#	10750	10759#
10760#	10762	10772#	10781#	10788#	10791	10800#	10801#	10803	10812#	10813#	10815	10825#
10834#	10884#	10900#	10921#	10941#	10970#	10989#	11003#	11017#	11040#	11041#	11042	11054#
11055#	11091#	11107#	11121#	11139#	11152#	11164#	11207#	11220#	11233#	11246#	11268#	11293#
11308#	11321#	11334#	11347#	11368#	11385#	11401#	11466#	11482#	11498#	11518#	11532#	11588#

```
$BELL      001212
$BIN       057622
$CDW1      001302
$CDW2      001304
$CHARC     060554
$CKSWR     061666
$CMTAG     001114
$CM3       = 000000
$CM4       = 000005
$CNTLC     062564
```

11603*	11603*	11603*	11603*	11603*	11603*	11603*
11603*	12555					
11603*	12077	12110	12161	12358		
11716*	11719*	11720	11727*			
1403*						
1404*						
11917*	11927*	11934	11943*	11948*		
12194*	12448					
1315*	3230	3231	3239	3245	3246	3247
1345*						
1345*	1346*	1347*	1348*	1349*	1350*	
12149	12230	12358*				

CZRMDB0 RM03/2 FCTNL TST 2
CZRMDB.P11 23-NOV-77 12:19MACY11 30(1046) 23-NOV-77 12:38 PAGE 305
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0304

\$CNTLG	062576	12215	12360*											
\$CNTLU	062571	12240	12333	12359*										
\$CPUOP	001250	1375*												
\$CRLF	001217	1354*			7846	7884	7894	7907	7915	7935	11916	11951	12085	12110
		12251	12338	12358										
		11752	11786	11794*										
\$DBLK	060040	1405*												
\$DDW0	001306	1406*												
\$DDW1	001310	1407*												
\$DDW2	001312	1408*												
\$DDW3	001314	1409*												
\$DDW4	001316	1410*												
\$DDW5	001320	1411*												
\$DDW6	001322	1412*												
\$DDW7	001324	1366*												
\$DEVCT	001232	1402*	3310*	3311*	3323*	3441*	3450*	3468*	3469	3476				
\$DEVN	001300	7760	7781	7787*										
\$DOAGN	033166	11755	11790*											
\$DTBL	060030	1273	3276	7783*	12105									
\$ENDAO	033156	3245	7762*											
\$ENDCT	033022	7790*												
\$ENULL	033172	1371*	3282	3303	3503	11895	12087	12509	12533					
\$ENV	001242	1372*	3268	3308	11897	11902	12511							
\$ENVN	001243	3558	7752*	11649										
\$EOP	032766	3245*	7759*	7763										
\$EOPCT	033014	3601	3641	7734*										
\$EOSP	032732	1318*	11956	11996	11998	12004*	12026	12072*	12110					
\$ERFLG	001117	1324*	3248*	11998	11021*	12026								
\$ERMAX	001131	3239	12070*											
\$ERROR	061170	1325*	7838	12079*	12080*	12081	12110							
\$ERRPC	001132	1514*	7853											
\$ERRTB	001564	1322*	7775	7779*	12078*	12110								
\$ERTTL	001126	1351*	3247*	12020*	12101	12103	12110							
\$ESCAP	001210	1370*												
\$ETABL	001242	1306	1413*											
\$ETEND	001326	1363*	12537*											
\$FATAL	001224	12500*	12503*	12531	12540*	12548*								
\$FFLG	063460	1343*	11920	11951										
\$FILLC	001172	1342*	11951											
\$FILLS	001171	1326*	4474*	4489	4489*	5004	8757*	12555						
\$GDADR	001134	1328*	4466*	4467*	4487*	4488*	4493	4982*	4983*	5002*	5003*	5009	6959*	6960*
\$GDDAT	001140	7013*	7014*	7151*	7152*	7289*	7290*	7440*	7441*	7456*	7457*	7475*	7476*	7672*
		7673*	8755*	9072*	9073*	9074	9090*	9091*	9097*	9099*	9111*	9112*	9113*	9129*
		9130*	9145*	9146*	9173*	9174*	9227*	9240*	9251*	9252	9264*	9269*	9272	9283*
		9291*	9294	9320*	9323*	9324*	9327	9339*	9340*	9343	9350*	9702*	9708*	9709*
		9710*	9711*	9714*	9715	9762*	9767*	9770	9779*	9792*	9797	9808*	9809*	9810*
		9812	9821*	9822*	9824	9940*	9952*	9965*	9970	9989*	10003*	10004*	10031*	10044*
		10068*	10069*	10083*	10096*	10097*	10112*	10113*	10200*	10201	10210*	10226*	10228	10237*
		10250*	10251	10260*	10272*	10273	10282*	10295*	10296	10336*	10337*	10350*	10351*	10369*
		10370*	10383*	10384*	10397*	10398*	10411*	10412*	10425*	10426*	10477*	10478*	10491*	10492*
		10507*	10508*	10524*	10525*	10545*	10546*	10562*	10563*	10576*	10577*	10601*	10602*	10617*
		10619*	10631*	10632*	10647*	10648*	10664*	10665*	10682*	10683*	10696*	10697*	10712*	10713*
		10749*	10750	10761*	10762	10771*	10802*	10803	10814*	10815	10823*	10882*	10883*	10898*
		10899*	10919*	10920*	10939*	10940*	10968*	10969*	10987*	10988*	11001*	11002*	11015*	11016*
		11026*	11039*	11042	11056*	11057*	11089*	11090*	11105*	11106*	11119*	11120*	11137*	11138*
		11150*	11151*	11165*	11166*	11174*	11205*	11206*	11218*	11219*	11231*	11232*	11244*	11245*

SEQ 0305

[illegible]

\$RDOCT	062626	12376#	12451											
\$RDSZ =	000010	12299#												
\$RESRE	057512	11692#	12453											
\$RTNAD	033170	7789#												
\$R2A =	***** U	12454												
\$SAVRE	057454	11676#	12452											
\$SAVR6	063202	12467#	12475	12476*	12477*	12493#								
\$SCOPE	060560	3237	11965#											
\$SETUP=	000137	1255#	3236	3237	3239	3241	3243	3245	3246	3247	3249	3276	3279	3500
		7754	11966	12071	12097	12105	12153	12158	12159	12189	12365			
\$STUP =	177777	1255#												
\$SVLAD	061024	11977	12016#											
\$SVPC =	000200	1271#	1276											
\$SWR =	167400	801#	812	817	818	819	820	821	822	823	824	1350	1351	1352
		3246	3247	3249	3250	3531	3571	3615	3653	3821	3989	4159	4319	4507
		4675	4835	5023	5182	5341	5500	5665	5810	5955	6100	6185	6267	6414
		6561	6707	6882	7035	7173	7311	7547	7749	7755	7782	7788	7790	11957
		11958	11959	11960	11961	11968	11980	11982	11983	11996	11997	11998	12005	12006
		12007	12019	12022	12025	12062	12063	12064	12065	12066	12075	12082	12094	12098
		12110	12490											
\$SWREG	001244	1373#	3270											
\$SWRMK=	000000	824	825	11961	11962	11986								
\$SWOBT	061076	11992	12026#											
\$TESTN	001226	1364#	3531*	3571*	3615*	3653*	3821*	3989*	4159*	4319*	4507*	4675*	4835*	5023*
		5182*	5341*	5500*	5665*	5810*	5955*	6100*	6185*	6267*	6414*	6561*	6707*	6882*
		7035#	7173*	7311*	7547*	7821	12017*							
\$TIMES	001206	1350#	3246*	7755*	12005*	12012	12015*	12025						
\$TKB	001162	1338#	12113	12134	12145	12170	12198	12225						
\$TKCNT	061370	12114#	12129*	12159	12176*	12290	12292*							
\$TKINT	061400	3340	3513	12129#	12150	12211								
\$TKQEN=	061377	12118#	12184	12295										
\$TKQIN	061372	12115#	12130*	12131	12182*	12183*	12184	12186*						
\$TKQOU	061374	12116#	12131*	12293	12294*	12295	12297*							
\$TKQSR	061376	12117#	12130	12186	12297									
\$TKS	001160	1337#	12113	12135*	12166*	12168	12174*	12196	12212*	12222	12234*	12254*		
\$TKSRV	061450	12132	12145#											
\$TMP0	001174	1345#	3585*	3588	3663*	3831*	3993*	4169*	4329*	4517*	4685*	4845*	5033*	5192*
		5351*	5510*	5675*	5820*	5965*	6277*	6424*	6571*	6717*	6892*	7045*	7183*	7

[illegible]

CALCLR	801#	3573	3617												
CALGET	801#	3698	3730	3773	3866	3898	3941	4034	4067	4111	4204	4236	4278	4364	4396
	4441	4481	4552	4584	4627	4720	4752	4794	4880	4912	4957	4996	5068	5100	5142
	5227	5259	5301	5386	5418	5460	5547	5580	5622	5711	5754	5856	5899	6001	6044
	6144	6226	6313	6367	6460	6514	6607	6661	6765	6827	6941	6995	7081	7125	7219
	7263	7367	7410	7603	7646										
CALPUT	801#	3685	3721	3764	3853	3889	3932	4021	4058	4102	4191	4227	4269	4351	4387
	4432	4539	4575	4618	4707	4743	4785	4867	4903	4948	5055	5091	5133	5214	5250
	5292	5373	5409	5451	5534	5571	5613	5699	5745	5844	5890	5989	6035	6133	6214
	6301	6355	6448	6502	6595	6649	6739	6753	6802	6818	6910	6929	6983	7069	7116
	7207	7254	7355	7401	7528	7591	7637	7723							
CALSJB	801#	3669	3706	3738	3745	3752	3781	3788	3795	3804	3837	3874	3906	3913	3920
	3949	3956	3963	3972	4005	4042	4075	4082	4089	4119	4126	4133	4142	4175	4212
	4244	4251	4258	4286	4293	4300	4335	4372	4404	4411	4418	4449	4460	4523	4560
	4592	4599	4606	4635	4642	4649	4658	4691	4728	4760	4767	4774	4802	4816	4816
	4851	4888	4920	4927	4934	4965	4976	5039	5076	5108	5115	5122	5150	5157	5164
	5198	5235	5267	5274	5281	5309	5316	5323	5357	5394	5426	5433	5440	5468	5475
	5482	5516	5555	5588	5595	5602	5630	5637	5644	5681	5719	5726	5733	5762	5769
	5776	5785	5826	5864	5871	5878	5907	5914	5921	5930	5971	6009	6016	6023	6052
	6059	6066	6075	6104	6152	6159	6166	6188	6234	6241	6248	6283	6321	6328	6335
	6344	6375	6382	6389	6430	6468	6475	6482	6491	6522	6529	6536	6577	6615	6622
	6629	6638	6669	6676	6683	6723	6773	6780	6787	6835	6842	6849	6858	6898	6949
	6963	6972	7003	7017	7051	7089	7096	7103	7133	7144	7155	7189	7227	7234	7241
	7271	7282	7293	7328	7375	7382	7389	7418	7433	7448	7468	7486	7513	7564	7611
	7618	7625	7654	7665	7683	7708									
CNACMS	3520#	3522													
COMMEN	943#														
DVAVMS	3561#	3563													
DVTYMS	3605#	3607													
ENDCOM	943#														
ERROR	837#														
	3733	3741	3748	3755	3767	3776	3784	3791	3798	3809	3841	3856	3869	3877	3892

GETDB	801#														
GETDS	801#														
GETDT	801#	3623													
GETMR1	801#														
GETPRI	943#														
GETSWR	801#	943#	3279#	3500											
MULT	943#														
NEWTST	943#	3520	3561	3605	3644	3812	3980	4150	4310	4498	4666	4826	5014	5173	5332
	5491	5656	5801	5946	6091	6176	6258	6405	6552	6698	6873	7026	7164	7302	7538
NWTST	801#	3523	3565	3609	3647	3815	3983	4153	4313	4501	4669	4829	5017	5176	5335
	5494	5659	5804	5949	6094	6179	6261	6408	6555	6701	6876	7029	7167	7305	7541
POP	943#	3547	3548	3552	3553	3586	3587	3598	3599	8213	8432	8433	8446	8589	8590
	8591	8663	8665	8666	8667	8668	8769	8771	8772	8773	8802	8803	8804	8877	8879
	8880	8881	8882	8883	8884	8944	8946	8947	8948	8949	8950	8951	8973	8975	9015
	9017	9018	9019	9020	9168	9172	9316	9900	9901	9902	9903	10163	10165	10166	10167
	10227	10789	10790	11697	11781	12399	12479	12480	12543	12544					
PUSH	943#	3533	3534	3578	3580	8190	8415	8416	8439	8457	8458	8459	8615	8617	8618
	8619	8620	8691	8692	8693	8694	8786	8788	8789	8836	8837	8838	8839	8840	8841
	8842	8909	8910	8911	8912	8913	8914	8915	8957	8958	8986	8987	8988	8989	8990
	9164	9313	9864	9865	9866	9867	10147	10149	10150	10151	10221	10782	10783	11677	11740
	12378	12460	12466	12504	12506	12527									
PUTCS1	801#														
PUTDC	801#														
PUTER1	801#														
PUTMR1	801#														
REPORT	943#														
RGBFMC	801#	1421	1448												
SCOPE	808#	3566	3610	3648	3816	3984	4154	4314	4502	4670	4830	5018	5177	5336	5495
	5660	5805	5950	6095	6180	6262	6409	6556	6702	6877	7030	7168	7306	7542	
SETPRI	943#	3224	3514	11628	11634	12286									
SETTRA	12431#	12440	12441	12442	12443	12444	12446	12448	12449	12450	12451	12452	12453		
SETUP	943#	3229													
SKIP	943#														
SLASH	943#														
SPACE	943#														
STARS	943#	1269	1287	1289	1296	1310	1356	1359	3520	3522	3561	3564	3605	3608	3644
	3646	3812	3814	3980	3982	4150	4152	4310	4312	4498	4500	4666	4668	4826	4828
	5014	5016	5173	5175	5332	5334	5491	5493	5656	5658	5801	5803	5946	5948	6091
	6093	6176	6178	6258	6260	6405	6407	6552	6554	6698	6700	6873	6875	7026	7028
	7164	7166	7302	7304	7538	7540	7745	7794	7995	8008	8020	8042	8071	8093	8121
	8185	8451	9211	9219	9304	9687	11661	11706	11730	11797	11874	11953	12058	12112	12189
	12204	12275	12299	12368	12406	12456	12472	12499							
SWPSU	943#	3251#													
TACS	801#	1415													
TRMTRP	12431#														
TYPBIN	943#														
TYPDEC	943#	7768	7775												
TYPNAM	801#	943#	3272												
TYPNUM	943#														
TYPOCS	943#	3398	3419	7815	7823	7832	7838								
TYPOCT	943#	3384	12217												
TYPTXT	943#	7764	7771												
XPER	801#	1518	1525	1532	1539	1546	1553	1560	1567	1574	1581	1588	1595	1602	1609
	1616	1623	1630	1637	1644	1651	1658	1665	1672	1679	1686	1693	1700	1707	1714
	1721	1728	1735	1742	1749	1756	1763	1770	1777	1784	1791	1798	1805	1813	1820
	1827	1834	1841	1849	1857	1865	1872	1879	1886	1893	1900	1907	1914	1921	1929

	1936	1943	1950	1957	1964	1971	1978	1985	1992	1999	2006	2013	2020	2027	2034
	2041	2048	2055	2062	2069	2076	2083	2090	2097	2104	2111	2118	2125	2132	2139
	2146	2153	2160	2167	2174	2181	2188	2195	2202	2209	2216	2223	2230	2237	2244
	2251	2258	2265	2272	2279	2286	2293	2300	2307	2314	2322	2329	2336	2343	2350
	2357	2364	2371	2378	2385	2392	2399	2406	2413	2420	2427	2434	2441	2448	2455
	2462	2469	2476	2483	2490	2497	2504	2511	2518	2525	2532	2539	2546	2553	2560
	2567	2574	2581	2588	2595	2602	2609	2616	2623	2630	2637	2644	2651	2658	2665
	2672	2679	2686	2693	2700	2707	2714	2721	2728	2735	2742	2749	2756	2763	2770
	2777	2784	2791	2799	2806	2814	2821	2828	2836	2844	2852	2861	2869	2877	2884
	2891	2898	2905	2912	2919	2926	2933	2940	2947	2954	2961	2968	2975	2982	2989
	2996	3003	3010	3017	3024	3031	3038	3045	3052	3059	3066	3073	3080	3087	3094
	3101	3108	3115	3122	3129	3136	3143	3150	3157	3164	3171	3178			
\$\$CHRE	1308#														
\$\$CHTH	1308#	1345	1346	1347	1348	1349									
\$\$ESCA	943#														
\$\$NEWT	943#	3520	3561	3605	3644	3812	3980	4150	4310	4498	4666	4826	5014	5173	5332
\$\$SET	5491	5656	5801	5946	6091	6176	6258	6405	6552	6698	6873	7026	7164	7302	7538
\$\$SETH	12431#	12440	12441	12442	12443	12444	12446	12448	12449	12450	12451	12452	12453		
\$\$SKIP	3267#														
\$.EQUAT	801#	833													
\$.HEAOE	801#	802													
\$.SETUP	801#	1255													
\$.SWAMI	801#	813													
\$.SWALO	801#	825#													
\$.SACTI	801#	1267#													
\$.SAPTB	801#	1357#													
\$.SAPTH	801#	1285													
\$.SAPTY	801#	12497													
\$.SCATC	801#	1256													
\$.SCMTA	801#	1308													
\$.SDIV	801#														
\$.SEOP	801#	7743													
\$.SERRO	801#	12056													
\$.SERRT	801#														
\$.SMULT	801#														
\$.SPOUE	801#	12454													
\$.SRAND	801#														
\$.SRODE	801#														
\$.SROOC	801#	12366													
\$.SREAO	801#	12110													
\$.SSAVE	801#	11659													
\$.SSCOP	801#	11951													
\$.SSIZE	801#														
\$.STRAP	801#	12404													
\$.STYPB	801#	11704													
\$.STYPO	801#	11728													
\$.STYPE	801#	11872													
\$.STYPO	801#	11795													

. ABS. 106121 000

ERRORS DETECTED: 0

N08

CZRMDB0 RM03/2 FCTNL TST 2 MACY11 30(1046) 23-NOV-77 12:38 PAGE 313
CZRMDB.P11 23-NOV-77 12:19 CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0311

RM03:CZRMDB.BIN, RM03:CZRMDB.SEG/DOC/NL:TOC/SOL/CRF=RM03:CZRMDB.P11
RUN-TIME: 65 61 4 SECONDS
RUN-TIME RATIO: 599/130=4.6
CORE USED: 33K (65 PAGES)

DOCUMENT PAGES: 311