

RM02/03/05

RM05/3/2 FCTNL TST 3
CZRM0B0

AH-F928B-MC
FICHE 1 OF 2

AUG 1981
COPYRIGHT © 80-81
MADE IN USA



RM02/03/05

RM05/3/2 FCTNL TST 3
CZRM0B0

AH-F928B-MC
FICHE 2 OF 2

AUG 1981
COPYRIGHT © 80-81
MADE IN USA



.REM \

IDENTIFICATION

PRODUCT CODE: AC-F927B-MC
PRODUCT NAME: CZRM0B0 RM05/3/2 FUNCTIONAL TEST, PT 3
PRODUCT DATE: APRIL 1981
MAINTAINER: CX DIAGNOSTIC GROUP
AUTHOR: MIKE LEAVITT

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

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

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

COPYRIGHT (C) 1980,1981 DIGITAL EQUIPMENT CORPORATION

CONTENTS

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

1.0 INTRODUCTION

1.1 ABSTRACT

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

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

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

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

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

1.2 UNIT UNDER TEST

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

2.0 OPERATING REQUIREMENTS

2.1 HARDWARE REQUIREMENTS

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

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

2.2 MEDIA REQUIREMENTS

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

2.3 PREREQUISITE DIAGNOSTIC PROGRAMS

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

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

3.0 OPERATING PROCEDURE

3.1 LOADING

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

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

3.2 SWITCH OPTIONS

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

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

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

3.3 STARTING

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

3.4 HALTING

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

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

3.5 RESTARTING

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

4.0 OPERATOR INTERFACE

4.1 PROGRAM ID

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

4.2 CONSOLE DIALOGUE

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

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

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

EXAMPLE 1

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

EXAMPLE 2

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

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

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

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

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

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

'DRIVE(S) TO BE TESTED, 0'

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

'DRIVE(S) TO BE TESTED, NONE'

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

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

'DRIVE 0'

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

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

4.3 PROGRESS REPORTS

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

'END OF PASS 1'

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

'TOTAL ERRORS SINCE LAST REPORT 0'

4.4 PERFORMANCE REPORT

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

4.5 PROGRAM HALTS

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

4.6 ERROR REPORTS

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

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

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

4.7 EXECUTION TIME

TIME FOR RM05:

PASS 1 OF THE PROGRAM TAKES ABOUT 15 SECONDS. PASS 2 AND

SUBSEQUENT PASSES TAKE 1 MINUTES 10 SECONDS.

TIME FOR RM02/3:

PASS 1 OF THE PROGRAM TAKES ABOUT 6 SECONDS. PASS 2 AND
 SUBSEQUENT PASSES TAKE 40 SECONDS.

5.0 ENVIRONMENTAL SUPPORT

5.1 PROCESSOR COMPATIBILITY

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

5.2 DUAL PORT CONFIGURATIONS

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

5.3 MEMORY PARITY HARDWARE

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

5.4 MEMORY MANAGEMENT HARDWARE

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

5.5 ACT11, APT11 COMPATIBILITY

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

5.6 XXDP COMPATIBILITY

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

343
344
345
346 5.7 OPERATING SYSTEM COMPATIBILITY
347348 THE PROGRAM IS NOT COMPATIBLE WITH ANY SOFTWARE OPERATING SYSTEM.
349350
351 6.0 TEST DESCRIPTION
352
353
354
355
356
357358 TEST 1 CONTROLLER ACCESS TEST
359

360 PURPOSE:

361 TO VERIFY THAT THE UNIBUS ADDRESS OF THE SUBSYSTEM IS
362 CORRECT, AS DEFINED AT LOCATION \$BASE.
363

364 PROCEDURE:

365 THE TEST TRIES TO ACCESS ALL MASSBUS CONTROLLER REGISTERS
366 USING THE \$BASE ADDRESS. REGISTER CONTENTS ARE IGNORED DURING
367 THE TEST, AND THE TEST FAILS IF A BUS TIMEOUT OCCURS FOR ANY
368 REGISTER TRANSFER.
369370 IF THE TEST FAILS AND THE PROGRAM IS RUNNING IN A STAND
371 ALONE ENVIRONMENT, I.E., LOCATION 42 IS 0, THE PROGRAM WILL JUMP
372 TO LOCATION 204 WHICH ALLOWS THE OPERATOR TO CHANGE THE \$BASE
373 ADDRESS VIA CONSOLE DIALOGUE. OTHERWISE, THE PROGRAM ESCAPES TO
374 THE END OF PASS HANDLER.
375
376
377
378
379
380381 TEST 2 - 25 WRITE/READ DATA TESTS
382

383 PURPOSE:

384 TO TEST WRITE DATA AND READ DATA FUNCTIONALITY OF THE
385 RM05/3/2 SUBSYSTEM USING A SET OF VARIABLES WHICH INCLUDE WORD
386 COUNT, HEAD MOTION, HEAD SWITCHING AND ERROR CONDITIONS.
387

388 PROCEDURE:

389 ALTHOUGH EACH TEST EXERCISES A DIFFERENT VARIABLE, THE
390 GENERAL PROCEDURE OF EACH TEST IS THE SAME. THE DRIVE IS
391 INITIALIZED AND RECALIBRATED IF 'PIP' OR 'SKI' ARE ACTIVE SO THAT
392 THERE ARE NO ERRORS WHEN A TEST BEGINS. THEN, THE TEST FORMATS
393 THE SECTOR BEING USED, WHICH MAY VARY FROM THE PROGRAM LISTING,
394 BECAUSE SECTORS ARE SUBSTITUTED DURING RUN TIME IF THE SELECTED
395 SECTOR IS LISTED IN THE BAD BLOCK TABLE OF THE LAST TRACK.
396 FOLLOWING THAT, THE TEST PERFORMS ANY EXPLICIT SEEKS REQUIRED FOR
397
398
399

THE CONDITIONS OF THE TEST. REGISTERS ARE PRESET AND THE WRITE 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 DATA PORTION OF THE TEST, VERIFYING THE SAME TYPE OF ERRORS AS IN THE WRITE COMMAND.

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

TEST 2 WRITE, READ ZEROS TEST

THE TEST WRITES AND READS AN ALL ZEROS DATA FIELD, CAUSING THE DRIVE TO USE NORMAL WRITE GATE THROUGHOUT THE WRITE PROCESS.

TEST 3 WRITE, WRITE CHECK ZEROS TEST

THE TEST WRITES AND WRITE CHECKS AN ALL ZEROS DATA FIELD, VERIFYING THAT THERE ARE NO ERRORS.

TEST 4 WRITE, WRITE CHECK ZEROS W/ WCE ERROR TEST

THE TEST WRITES AN ALL ZEROS DATA FIELD, THEN COMPLEMENTS EACH BIT IN THE LAST WORD OF THE WRITE BUFFER. A WRITE CHECK COMMAND IS EXECUTED AND THE TEST VERIFIES THAT THE CORRECT WRITE CHECK ERROR IS DETECTED.

TEST 5 WRITE, READ ONES TEST

THE TEST WRITES AND READS AN ALL ONES DATA FIELD, CAUSING THE DRIVE TO USE NORMAL WRITE GATE THROUGHOUT THE WRITE PROCESS.

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

TEST 6 WRITE, WRITE CHECK ONES TEST

THE TEST WRITES AND WRITE CHECKS AN ALL ONES DATA FIELD.

TEST 7 WRITE, WRITE CHECK ONES W/ WCE ERROR TEST

THE TEST WRITES AN ALL ONES DATA FIELD, THEN PERFORMS A WRITE CHECK DATA COMMAND AFTER COMPLEMENTING EACH BIT IN THE LAST WORD OF THE WRITE BUFFER. THE TEST VERIFIES THAT THE CORRECT WRITE CHECK ERROR IS DETECTED.

TEST 10 WRITE, WRITE CHECK MULTIPLE SECTORS TEST

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

TEST 11 WRITE, READ WITH IMPLIED SEEK TEST

THIS TEST SEEKS TO THE LAST CYLINDER PRIOR TO WRITING DATA ON CYLINDER 0, TRACK 0, SECTOR 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 DATA.

TEST 12 WRITE, WRITE CHECK WITH HEAD SWITCHING TEST

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

TEST 13 WRITE, WRITE CHECK WITH MID-TRANSFER SEEK TEST

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

TEST 14 WRITE, READ W/ HCE ERROR TEST

A SECTOR IS FORMATTED WITH AN INCORRECT HEADER, THEN THE TEST WRITES AND READS DATA FROM THE SECTOR AND VERIFIES THAT THE CORRECT ERROR IS DETECTED. EACH BIT POSITION OF BOTH HEADER WORDS IS TESTED IN THIS MANNER.

TEST 15 WRITE, READ W/ HCI TEST

A SECTOR IS FORMATTED WITH AN INCORRECT HEADER, THEN WRITTEN AND READ WITH HEADER COMPARE INHIBITED. THE TEST VERIFIES THAT NO ERROR IS DETECTED.

TEST 16 WRITE, READ W/ IVC ERROR TEST

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

TEST 17 WRITE, READ W/ ABORT TEST

THE TEST SETS AN ERROR IN THE ERROR REGISTER AND EXECUTES A WRITE DATA COMMAND VERIFYING THAT 'PIP' REMAINS INACTIVE. THE SAME PROCEDURE IS USED FOR READ DATA COMMAND.

TEST 20 WRITE, READ EACH CURRENT LEVEL TEST

THE TEST WRITES AND READS ON EACH OF THE FOLLOWING CYLINDERS IN ORDER TO WRITE AT EACH POSSIBLE CURRENT THRESHOLD.

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

CYLINDER	0	110 MA
..	128	104 MA
..	256	97 MA
..	384	91 MA
..	512	84 MA
..	640	77 MA
..	768	70 MA

TEST 21 WRITE, WRITE CHECK W/ CURRENT LEVEL SWITCHING TEST

THE TEST WRITES 2 SECTORS STARTING WITH THE LAST SECTOR OF CYLINDER 383. THE FIRST SECTOR IS WRITTEN AT ONE CURRENT LEVEL AND THE SECOND SECTOR IS WRITTEN AT ANOTHER CURRENT LEVEL. BOTH SECTORS ARE VERIFIED WITH A WRITE CHECK DATA COMMAND.

TEST 22 WRITE, READ EARLY PEAK SHIFT TEST

THIS TEST WRITES AND READS A SINGLE SECTOR USING A DATA PATTERN DESIGNED TO INTRODUCE EARLY PEAK SHIFT.

TEST 23 WRITE, READ MIXED PEAK SHIFT TEST

THIS TEST WRITES AND READS A SINGLE SECTOR USING A DATA PATTERN DESIGNED TO INTRODUCE MIXED PEAK SHIFT.

TEST 24 WRITE, READ EACH TRACK TEST

THE TEST WRITES AND READS WITH EACH HEAD USING A MIX OF EARLY, NORMAL AND LATE WRITE GATES.

TEST 25 READ, WRITES CHECK MULTIPLE SECTORS IN OFFSET MODE TEST

THE TEST EXECUTES READ DATA AND WRITE CHECK DATA COMMANDS IN OFFSET MODE IN BOTH OFFSET DIRECTIONS WITH THE TRANSFER SIZE OF TWO SECTORS AT A TIME.

628

479
480

```
;*LAST REVISION 04-APR-81
.TITLE CZRM0B0 RM05/3/2 FCTNL TST 3
;*COPYRIGHT (C) 1981
;*DIGITAL EQUIPMENT CORPORATION
;*COLORADO SPGS., CO. 80919
;*
;*PROGRAM BY MIKE LEAVITT
;*
;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
;*PACKAGE (MAINDEC-11-DZQAC-C5), 18-MAR-81
;*
```

481

```
.SBTTL OPERATIONAL SWITCH SETTINGS
;*
```

482

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

483
484

```
.SBTTL BASIC DEFINITIONS
;*
```

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

```
;*MISCELLANEOUS DEFINITIONS
000011 HT = 11 ;:CODE FOR HORIZONTAL TAB
000012 LF = 12 ;:CODE FOR LINE FEED
000015 CR = 15 ;:CODE FOR CARRIAGE RETURN
000200 CRLF = 200 ;:CODE FOR CARRIAGE RETURN-LINE FEED
177776 PS = 177776 ;:PROCESSOR STATUS WORD
177776 PSW=PS
177774 STKLM1 = 177774 ;:STACK LIMIT REGISTER
177772 PIRQ = 177772 ;:PROGRAM INTERRUPT REQUEST REGISTER
177570 DSWR = 177570 ;:HARDWARE SWITCH REGISTER
177570 DDISP = 177570 ;:HARDWARE DISPLAY REGISTER
```

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

000004	R4	=	%4	:: GENERAL REGISTER
000005	R5	=	%5	:: GENERAL REGISTER
000006	R6	=	%6	:: GENERAL REGISTER
000007	R7	=	%7	:: GENERAL REGISTER
000006	SP	=	%6	:: STACK POINTER
000007	PC	=	%7	:: PROGRAM COUNTER

;*PRIORITY LEVEL DEFINITIONS

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

;'SWITCH REGISTER' SWITCH DEFINITIONS

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

;*DATA BIT DEFINITIONS (BIT00 TO BIT15)

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

```

000020      BIT04  = 20
000010      BIT03  = 10
000004      BIT02  = 4
000002      BIT01  = 2
000001      BIT00  = 1
001000      BIT9=BIT09
000400      BIT8=BIT08
000200      BIT7=BIT07
000100      BIT6=BIT06
000040      BIT5=BIT05
000020      BIT4=BIT04
000010      BIT3=BIT03
000004      BIT2=BIT02
000002      BIT1=BIT01
000001      BIT0=BIT00
  
```

```

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

.SBTTL RM REGISTER BIT DEFINITIONS

; *RMCS1 CONTROL STATUS REGISTER

```

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

; FUNCTION CODES (BITS 01-05 OF RMCS1)

```

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

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


```
565
566      100000
567      040000
568      020000
569      010000
570      004000
571      002000
572      001000
573      000400
574      000200
575      000100
576      000040
577      000020
578      000010
579      000004
580      000002
581      000001
582
583      115760
584
585
586
587      ;*RMAS ATTENTION SUMMARY REGISTER
588
589      000377
590
591      ;*RMLA LOOK AHEAD REGISTER
592
593      002000
594      001000
595      000400
596      000200
597      000100
598
599      003700
600
601      ;*RMMR1 MAINTENANCE REGISTER #1
602
603      ;WRITE ONLY BITS
604      100000
605      040000
606      020000
607      010000
608      004000
609      002000
610      001000
611      000400
612      000200
613      000100
614      000040
615      000010
616      000004
617      000002
618      000001
619
620
621      100000
```

```
DCK      = BIT15      ;DATA CHECK ERROR
UNS      = BIT14      ;DRIVE UNSAFE
OPI      = BIT13      ;OPERATION INCOMPLETE
DTE      = BIT12      ;DRIVE TIMING ERROR
WLE      = BIT11      ;WRITE LOCK ERROR
IAE      = BIT10      ;INVALID ADDRESS ERROR
AOE      = BIT09      ;ADDRESS OVERFLOW ERROR
HCRC     = BIT08      ;HEADER CRC ERROR
HCE      = BIT07      ;HEADER COMPARE ERROR
ECH      = BIT06      ;ECC 'HARD' ERROR
WCF      = BIT05      ;WRITE CLOCK FAILURE
FER      = BIT04      ;FORMAT ERROR
PAR      = BIT03      ;PARITY ERROR
RMR      = BIT02      ;REGISTER MODIFICATION REFUSED
ILR      = BIT01      ;ILLEGAL REGISTER
ILF      = BIT00      ;ILLEGAL FUNCTION

NDTMSK   = DCK.DTE!WLE!AOE!HCRC!HCE!ECH!WCF!FER
; 'NDTMSK' IS USED TO MASK ERROR REGISTER 1 DURING NON - DATA
; COMMANDS, I.E., HOUSEKEEPING AND POSITIONING COMMANDS

;*RMAS ATTENTION SUMMARY REGISTER

ATNMSK   = 377      ;MASK FOR ATTENTION BITS

;*RMLA LOOK AHEAD REGISTER

SC4      = BIT10      ;SECTOR COUNT = 16
SC3      = BIT09      ;SECTOR COUNT = 8
SC2      = BIT08      ;SECTOR COUNT = 4
SC1      = BIT07      ;SECTOR COUNT = 2
SC0      = BIT06      ;SECTOR COUNT = 1

SCTMSK   = 003700    ;SECTOR COUNT MASK

;*RMMR1 MAINTENANCE REGISTER #1

;WRITE ONLY BITS
DBCK     = BIT15      ;DEBUG CLOCK
DBEN     = BIT14      ;DEBUG CLOCK ENABLE
DEBL     = BIT13      ;DIAGNOSTIC END OF BLOCK
DTO      = BIT12      ;DIAGNOSTIC TIMEOUT
MCLK     = BIT11      ;MAINTENANCE CLOCK
MRD      = BIT10      ;READ DATA
MUR      = BIT09      ;UNIT READY
MOC      = BIT08      ;ON CYLINDER
MSER     = BIT07      ;SEEK ERROR
MDF      = BIT06      ;DRIVE FAULT
MS       = BIT05      ;SECTOR PULSE
MWP      = BIT03      ;WRITE PROTECT
MI       = BIT02      ;INDEX PULSE
MSC      = BIT01      ;SECTOR COMPARE
DMD      = BIT00      ;DIAGNOSTIC MODE

;READ ONLY BITS
OCC      = BIT15      ;OCCUPIED
```

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

```

672      ;*RMER2 ERROR REGISTER 2
673
674      100000      BSE      = BIT15      ;BAD SECTOR ERROR
675      040000      SKI      = BIT14      ;SEEK INCOMPLETE
676      020000      OPE      = BIT13      ;OPERATOR PLUG FRROR
677      010000      IVC      = BIT12      ;INVALID COMMAND ERROR
678      004000      LSC      = BIT11      ;LOSS OF SYSTEM CLOCK
679      002000      LBC      = BIT10      ;LOSS OF BIT CLOCK
680      000200      DVC      = BIT07      ;DEVICE CHECK
681      000010      DPE      = BIT03      ;DATA PARITY ERROR
682
683      .SBTTL  PROGRAM MNEMONICS
684
685      100000      MSE      = BIT15      ;MANUFACTURING DETECTED SECTOR ERROR
686      040000      USE      = BIT14      ;USER DETECTED SECTOR ERROR
687
688      .SBTTL  RM REGISTER INDEX VALUES
689
690      000000      RMCS1    = 00      ;CONTROL STATUS REGISTER #1
691      000006      RMDA     = 06      ;DISK ADDRESS REGISTER
692      000012      RMD5     = 12      ;DRIVE STATUS REGISTER
693      000014      RMER1    = 14      ;ERROR REGISTER #1
694      000016      RMAS     = 16      ;ATTENTION SUMMARY REGISTER
695      000020      RMLA     = 20      ;LOOK AHEAD REGISTER
696      000024      RMMR1    = 24      ;MAINTENANCE REGISTER
697      000026      RMDT     = 26      ;DRIVE TYPE REGISTER
698      000030      RMSN     = 30      ;SERIAL NUMBER REGISTER
699      000032      RMOF     = 32      ;OFFSET REGISTER
700      000034      RMDC     = 34      ;DESIRED CYLINDER REGISTER
701      000036      RMHR     = 36      ;HOLDING REGISTER
702      000040      RMMR2    = 40      ;MAINTENANCE REGISTER #2
703      000042      RMER2    = 42      ;ERROR REGISTER #2
704      000044      RMEC1    = 44      ;ECC POSITION REGISTER
705      000046      RMEC2    = 46      ;ECC PATTERN REGISTER
706      000050      ILRG50   = 50      ;ILLEGAL REGISTER 50
707      000052      ILRG52   = 52      ;ILLEGAL REGISTER 52
708      000054      ILRG54   = 54      ;ILLEGAL REGISTER 54
709      000056      ILRG56   = 56      ;ILLEGAL REGISTER 56
710      000060      ILRG60   = 60      ;ILLEGAL REGISTER 60
711      000062      ILRG62   = 62      ;ILLEGAL REGISTER 62
712      000064      ILRG64   = 64      ;ILLEGAL REGISTER 64
713      000066      ILRG66   = 66      ;ILLEGAL REGISTER 66
714      000070      ILRG70   = 70      ;ILLEGAL REGISTER 70
715      000072      ILRG72   = 72      ;ILLEGAL REGISTER 72
716      000074      ILRG74   = 74      ;ILLEGAL REGISTER 74
717      000076      ILRG76   = 76      ;ILLEGAL REGISTER 76
718
719      000077      IDXMSK    = 77      ;MASK FOR REGISTER INDEX NUMBER
720
721      .SBTTL  RH CONTROLLER REGISTER BIT DEFINITIONS
722
723      ;*RMCS1 CONTROL STATUS REGISTER #1
724
725      100000      SC        = BIT15      ;SPECIAL CONDITION-READ ONLY
726      040000      TRE       = BIT14      ;TRANSFER ERROR
727      020000      MCPE      = BIT13      ;MASSBUS CONTROL BUS PARITY ERROR-READ ONLY
728      002000      PSEL      = BIT10      ;PORT B SELECT

```

720	001000	A17	= BIT09	:ADDRESS EXTENSION
721	000400	A16	= BIT08	:ADDRESS EXTENSION
722	000200	RDY	= BIT07	:READY-READ ONLY
723	000100	IE	= BIT06	:INTERRUPT ENABLE
724		; *RMCS2 RH CONTROL STATUS REGISTER #2		
725				
726				
727	100000	DLT	= BIT15	:DATA LATE-READ ONLY
728	040000	WCE	= BIT14	:WRITE CHECK ERROR-READ ONLY
729	020000	UPE	= BIT13	:UNIBUS PARITY ERROR
730	010000	NED	= BIT12	:NONEXISTANT DRIVE-READ ONLY
731	004000	NEM	= BIT11	:NONEXISTANT MEMORY-READ ONLY
732	002000	PGE	= BIT10	:PROGRAM ERROR-READ ONLY
733	001000	MXF	= BIT09	:MISSED TRANSFER
734	000400	MDPE	= BIT08	:MASSBUS DATA BUS PARITY ERROR-READ ONLY
735	000200	OR	= BIT07	:OUTPUT READY-READ ONLY
736	000100	IR	= BIT06	:INPUT READY-READ ONLY
737	000040	CLR	= BIT05	:CONTROLLER CLEAR
738	000020	PAT	= BIT04	:PARITY TEST
739	000010	BAI	= BIT03	:UNIBUS ADDRESS INCREMENT INHIBIT
742	000004	U2	= BIT02	:UNIT SELECT
	000002	U1	= BIT01	:UNIT SELECT
	000001	U0	= BIT00	:UNIT SELECT
743		;UNIT SELECT MASK		
744				
745				
746	000007	UNITMSK	= 7	:UNIT SELECT MASK
747				
748		; *RMCS3 RH70 CONTROL STATUS REGISTER #3		
749				
750	100000	APE	= BIT15	:ADDRESS PARITY ERROR
751	040000	DPEHI	= BIT14	:DATA PARITY ERROR HIGH WORD
752	020000	DPELO	= BIT13	:DATA PARITY ERROR LOW WORD
753	010000	WCEHI	= BIT12	:WRITE CHECK ERROR HIGH WORD
754	004000	WCELO	= BIT11	:WRITE CHECK ERROR LOW WORD
755	002000	DBL	= BIT10	:DOUBLE WORD TRANSFER
756	000100	IE	= BIT06	:INTERRUPT ENABLE
757	000010	IPCK3	= BIT03	:INVERT PARITY CHECK
758	000004	IFCK2	= BIT02	:INVERT PARITY CHECK
759	000002	IPCK1	= BIT01	:INVERT PARITY CHECK
760	000001	IPCK0	= BIT00	:INVERT PARITY CHECK
761		.SBTTL RH CONTROLLER REGISTER INDEX VALUES		
762				
763				
764	000000	RMCS1	= 00	:CONTROL, STATUS REGISTER #1
765	000002	RMWC	= 02	:WORD COUNT REGISTER
766	000004	RMBA	= 04	:BUS ADDRESS REGISTER
767	000010	RMCS2	= 10	:CONTROL, STATUS REGISTER #2
768	000022	RMDB	= 22	:DATA BUFFER
769	000050	RMBAE	= 50	:BUS ADDRESS EXTENSION
770	000052	RMCS3	= 52	:CONTROL, STATUS REGISTER #3
771				
772	176700	ABASE	= 176700	:UNIBUS ADDRESS
773	120254	AVECT1	= 120254	:UNIBUS VECTOR ADDRESS AND PRIORITY
774				

```
1          .SBTTL  TRAP CATCHER
          000000
          .=0
          ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A '+2,HALT'
          ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
          ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
          .=174
000174     000174
000176     000000
000176     000000
          DISPREG: .WORD 0          ;;SOFTWARE DISPLAY REGISTER
          SWREG:   .WORD 0          ;;SOFTWARE SWITCH REGISTER

          .SBTTL  STARTING ADDRESS(ES)

2          000200 000137 005432          JMP    @#START          ;;JUMP TO STARTING ADDRESS OF PROGRAM
3          000204 000137 005422          JMP    @#START1         ;CHANGE RH/RM BUS ADDRESS
4
5          .SBTTL  ACT11 HOOKS
          ;*****
          ;HOOKS REQUIRED BY ACT11
          000210     000210          $SVPC=.          ;SAVE PC
          000046     000046          .=46
          000046     037242          $ENDAD          ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
          000052     000052          .=52
          000052     000000          .WORD 0          ;;2)SET LOC.52 TO ZERO
          000210     000210          .=$SVPC          ;; RESTORE PC
6
7          001100          .-1100
8          .SBTTL  APT PARAMETER BLOCK
          ;*****
          ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
          ;*****
          001100     001100          .$X=.          ;;SAVE CURRENT LOCATION
          000024     000024          .=24          ;;SET POWER FAIL TO POINT TO START OF PROGRAM
          000024     000200          200          ;;FOR APT START UP
          000044     000044          .=44          ;;POINT TO APT INDIRECT ADDRESS PNTR.
          000044     001100          $APTHDR        ;;POINT TO APT HEADER BLOCK
          001100     001100          .=$X          ;;RESET LOCATION COUNTER
          ;*****
          ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
          ;INTERFACE SPEC.

          001100          $APTHD:
          001100     000000          $HIBTS: .WORD 0          ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
          001102     001222          $MBADR: .WORD $MAIL       ;;ADDRESS OF APT MAILBOX (BITS 0-15)
          001104     000006          $TSTM:  .WORD 6          ;;RUN TIM OF LONGEST TEST
          001106     000006          $PASTM: .WORD 6          ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
          001110     000006          $UNITM: .WORD 6          ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDED UNIT
          001112     000042          .WORD $ETEND-$MAIL/2      ;;LENGTH MAILBOX-ETABLE (WORDS)
9          001114          TAGADR=.
```

0

.SBTTL COMMON TAGS

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

001114	001114			SCMTAG: .TAGADR	::START OF COMMON TAGS
001114	000000			.WORD 0	
001116	000			\$TSTNM: .BYTE 0	::CONTAINS THE TEST NUMBER
001117	000			\$ERFLG: .BYTE 0	::CONTAINS ERROR FLAG
001120	000000			\$ICNT: .WORD 0	::CONTAINS SUBTEST ITERATION COUNT
001122	000000			\$LPADR: .WORD 0	::CONTAINS SCOPE LOOP ADDRESS
001124	000000			\$LPERR: .WORD 0	::CONTAINS SCOPE RETURN FOR ERRORS
001126	000000			\$ERTTL: .WORD 0	::CONTAINS TOTAL ERRORS DETECTED
001130	000			\$ITEMB: .BYTE 0	::CONTAINS ITEM CONTROL BYTE
001131	001			\$ERMAX: .BYTE 1	::CONTAINS MAX. ERRORS PER TEST
001132	000000			\$ERRPC: .WORD 0	::CONTAINS PC OF LAST ERROR INSTRUCTION
001134	000000			\$GDADR: .WORD 0	::CONTAINS ADDRESS OF 'GOOD' DATA
001136	000000			\$BDADR: .WORD 0	::CONTAINS ADDRESS OF 'BAD' DATA
001140	000000			\$GDDAT: .WORD 0	::CONTAINS 'GOOD' DATA
001142	000000			\$BDDAT: .WORD 0	::CONTAINS 'BAD' DATA
001144	000000			.WORD 0	::RESERVED--NOT TO BE USED
001146	000000			.WORD 0	
001150	000			\$AUTOB: .BYTE 0	::AUTOMATIC MODE INDICATOR
001151	000			\$INTAG: .BYTE 0	::INTERRUPT MODE INDICATOR
001152	000000			.WORD 0	
001154	177570			\$WR: .WORD DSWR	::ADDRESS OF SWITCH REGISTER
001156	177570			DISPLAY: .WORD DDISP	::ADDRESS OF DISPLAY REGISTER
001160	177560			\$TKS: 177560	::TTY KBD STATUS
001162	177562			\$TKB: 177562	::TTY KBD BUFFER
001164	177564			\$TPS: 177564	::TTY PRINTER STATUS REG. ADDRESS
001166	177566			\$TPB: 177566	::TTY PRINTER BUFFER REG. ADDRESS
001170	000			\$NULL: .BYTE 0	::CONTAINS NULL CHARACTER FOR FILLS
001171	002			\$FILLS: .BYTE 2	::CONTAINS # OF FILLER CHARACTERS REQUIRED
001172	012			\$FILLC: .BYTE 12	::INSERT FILL CHARS. AFTER A 'LINE FEED'
001173	000			\$TPFLG: .BYTE 0	::'TERMINAL AVAILABLE' FLAG (BIT<07>=0=YES)
001174	000000			\$TMP0: .WORD 0	::USER DEFINED
001176	000000			\$TMP1: .WORD 0	::USER DEFINED
001200	000000			\$TMP2: .WORD 0	::USER DEFINED
001202	000000			\$TMP3: .WORD 0	::USER DEFINED
001204	000000			\$TMP4: .WORD 0	::USER DEFINED
001206	000000			\$TIMES: 0	::MAX. NUMBER OF ITERATIONS
001210	000000			\$ESCAPE: 0	::ESCAPE ON ERROR ADDRESS
001212	207	377	377	\$BELL: .ASCIZ <207><377><377>	::CODE FOR BELL
001216	077			\$QUES: .ASCIZ /?/	::QUESTION MARK
001217	015			\$CRLF: .ASCIZ <15>	::CARRIAGE RETURN
001220	012	000		\$LF: .ASCIZ <12>	::LINE FEED

.SBTTL APT MAILBOX-ETABLE

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

001230	000000	\$PASS:	.WORD	APASS	::PASS COUNT
001232	000000	\$DEVCT:	.WORD	ADEVCT	::DEVICE COUNT
001234	000000	\$UNIT:	.WORD	AUNIT	::I/O UNIT NUMBER
001236	000000	\$MSGAD:	.WORD	AMSGAD	::MESSAGE ADDRESS
001240	000000	\$MSGLG:	.WORD	AMSGLG	::MESSAGE LENGTH
001242		\$ETABLE:			::APT ENVIRONMENT TABLE
001242	000	\$ENV:	.BYTE	AENV	::ENVIRONMENT BYTE
001243	000	\$ENVM:	.BYTE	AENVM	::ENVIRONMENT MODE BITS
001244	000000	\$SWREG:	.WORD	ASWREG	::APT SWITCH REGISTER
001246	000000	\$USWR:	.WORD	AUSWR	::USER SWITCHES
001250	000000	\$CPUOP:	.WORD	ACPUOP	::CPU TYPE,OPTIONS
		*			BITS 15-11=CPU TYPE
		*			11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
		*			11/70=06,PDQ=07,Q=10
		*			BIT 10=REAL TIME CLOCK
		*			BIT 9=FLOATING POINT PROCESSOR
		*			BIT 8=MEMORY MANAGEMENT
001252	000	\$MAMS1:	.BYTE	AMAMS1	::HIGH ADDRESS,M.S. BYTE
001253	000	\$MTYP1:	.BYTE	AMTYP1	::MEM. TYPE,BLK#1
		*			MEM.TYPE BYTE -- (HIGH BYTE)
		*			900 NSEC CORE=001
		*			300 NSEC BIPOLAR=002
		*			500 NSEC MOS=003
001254	000000	\$MADR1:	.WORD	AMADR1	::HIGH ADDRESS,BLK#1
		*			MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF 'TYPE' ABOVE
001256	000	\$MAMS2:	.BYTE	AMAMS2	::HIGH ADDRESS,M.S. BYTE
001257	000	\$MTYP2:	.BYTE	AMTYP2	::MEM.TYPE,BLK#2
001260	000000	\$MADR2:	.WORD	AMADR2	::MEM.LAST ADDRESS,BLK#2
001262	000	\$MAMS3:	.BYTE	AMAMS3	::HIGH ADDRESS,M.S.BYTE
001263	000	\$MTYP3:	.BYTE	AMTYP3	::MEM.TYPE,BLK#3
001264	000000	\$MADR3:	.WORD	AMADR3	::MEM.LAST ADDRESS,BLK#3
001266	000	\$MAMS4:	.BYTE	AMAMS4	::HIGH ADDRESS,M.S.BYTE
001267	000	\$MTYP4:	.BYTE	AMTYP4	::MEM.TYPE,BLK#4
001270	000000	\$MADR4:	.WORD	AMADR4	::MEM.LAST ADDRESS,BLK#4
001272	120254	\$VECT1:	.WORD	AVECT1	::INTERRUPT VECTOR#1,BUS PRIORITY#1
001274	000000	\$VECT2:	.WORD	AVECT2	::INTERRUPT VECTOR#2BUS PRIORITY#2
001276	176700	\$BASE:	.WORD	ABASE	::BASE ADDRESS OF EQUIPMENT UNDER TEST
001300	000000	\$DEVN:	.WORD	ADEVN	::DEVICE MAP
001302	000000	\$CDW1:	.WORD	ACDW1	::CONTROLLER DESCRIPTION WORD#1
001304	000000	\$CDW2:	.WORD	ACDW2	::CONTROLLER DESCRIPTION WORD#2
001306	000000	\$DDW0:	.WORD	ADDW0	::DEVICE DESCRIPTOR WORD#0
001310	000000	\$DDW1:	.WORD	ADDW1	::DEVICE DESCRIPTOR WORD#1
001312	000000	\$DDW2:	.WORD	ADDW2	::DEVICE DESCRIPTOR WORD#2
001314	000000	\$DDW3:	.WORD	ADDW3	::DEVICE DESCRIPTOR WORD#3
001316	000000	\$DDW4:	.WORD	ADDW4	::DEVICE DESCRIPTOR WORD#4
001320	000000	\$DDW5:	.WORD	ADDW5	::DEVICE DESCRIPTOR WORD#5
001322	000000	\$DDW6:	.WORD	ADDW6	::DEVICE DESCRIPTOR WORD#6
001324	000000	\$DDW7:	.WORD	ADDW7	::DEVICE DESCRIPTOR WORD#7
001326		\$ETEND:			
		.MEXIT			

.SBTTL USER DEFINED TAGS

001326 000000
001330 000000
001332 000000CTLFG: .WORD 0 ;CONTAINS CONTROL-C FLAG
CHGADR: .WORD 0 ;CHANGE RH/RM BUS ADDRESS = -1, NO CHANGE = 0
XXDP: .WORD 0 ;THE LOW BYTE CONTAINS THE DRIVE NUMBER FROM WHICH
;THE PROGRAM WAS LOADED. THE HIGH BYTE CONTAINS THE
;'XXDP' DEVICE CODE FOR THE RM05/3/2.
_STRK: .BYTE 0 ;LO BYTE = 0
;HI BYTE, CONTAINS LAST TRACK ADDRESS OF UNIT
;UNDER TEST. RM02/3 = 4., RM05 = 18.;THE REGISTER INPUT BUFFER IS USED FOR
;STORING DRIVE STATUS

001336

GETBUF:

;REGISTER INPUT BUFFER

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

001412

PUTBUF:

;REGISTER OUTPUT BUFFER

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

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

RMDT0: .WORD 0 ;DRIVE TYPE REGISTER
RMSNO: .WORD 0 ;SERIAL NUMBER REGISTER
RMOFO: .WORD 0 ;OFFSET REGISTER
RMDCO: .WORD 0 ;DESIRED CYLINDER REGISTER
RMHRO: .WORD 0 ;HOLDING REGISTER
RMMR20: .WORD 0 ;MAINTENANCE REGISTER #2
RMER20: .WORD 0 ;ERROR REGISTER #2
RMEC10: .WORD 0 ;ECC POSITION REGISTER
RMEC20: .WORD 0 ;ECC PATTERN REGISTER
RMBAE0: .WORD 0 ;BUS ADDRESS EXTENSION REGISTER
RMCS30: .WORD 0 ;CONTROL, STATUS REGISTER #3

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

001466 000000
001470 000000
001510 000000

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

001512 000000

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

001514 000000
001516 000000
001520 000000
001522 000000

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

001524

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

001553

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

;PUT TAGS HERE

.SBTTL ERROR POINTER TABLE

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

1	001602				
2					
3					
	001602	072404	EMT1		
	001604	076470	EHT1		
	001606	076614	EDT1		
	001610	076704	EFT1		
4					
5					
6					
	001612	072410	EMT2		
	001614	076470	EHT1		
	001616	076614	EDT1		
	001620	076704	EFT1		
7					
8					
9					
	001622	072416	EMT3		
	001624	076470	EHT1		
	001626	076614	EDT1		
	001630	076704	EFT1		
10					
11					
12					
	001632	072424	EMT4		
	001634	076470	EHT1		
	001636	076614	EDT1		
	001640	076704	EFT1		
13					
14					
15					
	001642	072432	EMT5		
	001644	076470	EHT1		
	001646	076614	EDT1		
	001650	076704	EFT1		
16					
17					

\$ERRTB:

:ERROR 1 WRONG UNIT SELECTED

EMT1
EHT1
EDT1
EFT1

:ERROR 2 DEVICE WENT UNAVAILABLE

EMT2
EHT1
EDT1
EFT1

:ERROR 3 DEVICE WENT NONEXISTENT

EMT3
EHT1
EDT1
EFT1

:ERROR 4 CONTROLLER NOT READY

EMT4
EHT1
EDT1
EFT1

:ERROR 5 DRIVE NOT READY AND GO NOT RESET

EMT5
EHT1
EDT1
EFT1

:ERROR 6 UNEXPECTED VALUE FOR 'ATA' STATUS

18	001652 072440	EMT6	
	001654 076470	EHT1	
	001656 076614	EDT1	
	001660 076704	EFT1	
19			
20		:ERROR 7	BUS TIMEOUT TRYING TO READ OR WRITE REGISTER
21	001662 072446	EMT7	
	001664 000000	0	
	001666 000000	0	
	001670 000000	0	
22			
23		:ERROR 10	DRIVE NOT READY BUT GO IS RESET
24	001672 072454	EMT10	
	001674 076470	EHT1	
	001676 076614	EDT1	
	001700 076704	EFT1	
25			
26		:ERROR 11	GO NOT RESET BUT DRIVE IS READY
27	001702 072460	EMT11	
	001704 076470	EHT1	
	001706 076614	EDT1	
	001710 076704	EFT1	
28			
29		:ERROR 12	INCORRECT FUNCTION CODE
30	001712 072464	EMT12	
	001714 076470	EHT1	
	001716 076614	EDT1	
	001720 076704	EFT1	
31			
32		:ERROR 13	PARITY ERROR READING REMOTE REGISTERS
33	001722 072472	EMT13	
	001724 076470	EHT1	
	001726 076614	EDT1	
	001730 076704	EFT1	
34			
35		:ERROR 14	TRANSFER ERROR IS INCORRECT
36	001732 072504	EMT14	
	001734 076470	EHT1	
	001736 076614	EDT1	
	001740 076704	EFT1	
37			
38		:ERROR 15	INCORRECT WORD COUNT
39			

001742	072512	EMT15
001744	076470	EHT1
001746	076614	EDT1
001750	076704	EFT1
40		
41		
42		:ERROR 16 INCORRECT BUS ADDRESS
001752	072520	EMT16
001754	076470	EHT1
001756	076614	EDT1
001760	076704	EFT1
43		
44		
45		:ERROR 17 INCORRECT LBT STATUS
001762	072530	EMT17
001764	076470	EHT1
001766	076614	EDT1
001770	076704	EFT1
46		
47		
48		:ERROR 20 INCORRECT AOE
001772	072540	EMT20
001774	076470	EHT1
001776	076614	EDT1
002000	076704	EFT1
49		
50		
51		:ERROR 21 INCORRECT DISK ADDRESS
002002	072550	EMT21
002004	076470	EHT1
002006	076614	EDT1
002010	076704	EFT1
52		
53		
54		:ERROR 22 INCORRECT CYLINDER ADDRESS
002012	072560	EMT22
002014	076470	EHT1
002016	076614	EDT1
002020	076704	EFT1
55		
56		
57		:ERROR 23 INCORRECT WLE STATUS
002022	072570	EMT23
002024	076470	EHT1
002026	076614	EDT1
002030	076704	EFT1
58		
59		
60		:ERROR 24 INCORRECT UPE STATUS
002032	072600	EMT24

	002034 076470	EMT1	
	002036 076614	EDT1	
	002040 076704	EFT1	
61			
62			
63		:ERROR 25	INCORRECT WCF STATUS
	002042 072610	EMT25	
	002044 076470	EHT1	
	002046 076614	EDT1	
	002050 076704	EFT1	
64			
65			
66		:ERROR 26	INCORRECT WCE STATUS
	002052 072620	EMT26	
	002054 076470	EHT1	
	002056 076614	EDT1	
	002060 076704	EFT1	
67			
68			
69		:ERROR 27	INCORRECT MDPE STATUS
	002062 072630	EMT27	
	002064 076470	EHT1	
	002066 076614	EDT1	
	002070 076704	EFT1	
70			
71			
72		:ERROR 30	INCORRECT DCK STATUS
	002072 072640	EMT30	
	002074 076470	EHT1	
	002076 076614	EDT1	
	002100 076704	EFT1	
73			
74			
75		:ERROR 31	INCORRECT ECM STATUS
	002102 072650	EMT31	
	002104 076470	EHT1	
	002106 076614	EDT1	
	002110 076704	EFT1	
76			
77			
78		:ERROR 32	DLT SHOULD NOT BE SET
	002112 072660	EMT32	
	002114 076470	EHT1	
	002116 076614	EDT1	
	002120 076704	EFT1	
79			
80			
81		:ERROR 33	MXF SHOULD NOT BE SET
	002122 072670	EMT33	
	002124 076470	EHT1	

	002126 076614	EDT1	
	002130 076704	EFT1	
82			
83			
84			
	002132 072700	EMT34	
	002134 076470	EHT1	
	002136 076614	EDT1	
	002140 076704	EFT1	
85			
86			
87			
	002142 072710	EMT35	
	002144 076470	EHT1	
	002146 076614	EDT1	
	002150 076704	EFT1	
88			
89			
90			
	002152 072720	EMT36	
	002154 076470	EHT1	
	002156 076614	EDT1	
	002160 076704	EFT1	
91			
92			
93			
	002162 072730	EMT37	
	002164 076470	EHT1	
	002166 076614	EDT1	
	002170 076704	EFT1	
94			
95			
96			
	002172 072740	EMT40	
	002174 076470	EHT1	
	002176 076614	EDT1	
	002200 076704	EFT1	
97			
98			
99			
	002202 072746	EMT41	
	002204 076470	EHT1	
	002206 076614	EDT1	
	002210 076704	EFT1	
100			
101			
102			
	002212 072756	EMT42	
	002214 076470	EHT1	
	002216 076614	EDT1	

;ERROR 34 DTE SHOULD NOT BE SET

;ERROR 35 INCORRECT HCRC STATUS

;ERROR 36 INCORRECT HCE STATUS

;ERROR 37 INCORRECT FER STATUS

;ERROR 40 DPE SHOULD NOT BE SET (NOT A DATA COMMAND)

;ERROR 41 LOST 'MOL' DURING PACK ACKNOWLEDGE

;ERROR 42 UNSAFE ERROR DURING PACK ACKNOWLEDGE

002220	076704	EFT1	
103			
104		:ERROR 43	'DPI' ERROR DURING PACK ACKNOWLEDGE
105			
002222	072770	EMT43	
002224	076470	EHT1	
002226	076614	EDT1	
002230	076704	EFT1	
106			
107		:ERROR 44	'RMR' ERROR DURING PACK ACKNOWLEDGE
108			
002232	073000	EMT44	
002234	076470	EHT1	
002236	076614	EDT1	
002240	076704	EFT1	
109			
110		:ERROR 45	'ILR' ERROR DURING PACK ACKNOWLEDGE
111			
002242	073010	EMT45	
002244	076470	EHT1	
002246	076614	EDT1	
002250	076704	EFT1	
112			
113		:ERROR 46	'ILF' ERROR DURING PACK ACKNOWLEDGE
114			
002252	073020	EMT46	
002254	076470	EHT1	
002256	076614	EDT1	
002260	076704	EFT1	
115			
116		:ERROR 47	COMPOSITE ERROR STATUS IS INCORRECT
117			
002262	073030	EMT47	
002264	076470	EHT1	
002266	076614	EDT1	
002270	076704	EFT1	
118			
119		:ERROR 50	PARITY ERROR WRITING REMOTE REGISTERS
120			
002272	073036	EMT50	
002274	076470	EHT1	
002276	076614	EDT1	
002300	076704	EFT1	
121			
122		:ERROR 51	INCORRECT IAE STATUS DURING SEEK COMMAND
123			
002302	073046	EMT51	
002304	076470	EHT1	
002306	076614	EDT1	
002310	076704	EFT1	

124			
125			
126		:ERROR 52	OPI ERROR DURING SEEK - MEDIUM IS NOT ON LINE
	002312	073060	EMT52
	002314	076470	EHT1
	002316	076614	EDT1
	002320	076704	EFT1
127			
128		:ERROR 53	OPI ERROR DURING SEEK - MEDIUM IS ON LINE, ASSUME
129		:	ON CYLINDER LATCH DIDN'T RESET
130			
	002322	073076	EMT53
	002324	076470	EHT1
	002326	076614	EDT1
	002330	076704	EFT1
131			
132		:ERROR 54	SEEK INCOMPLETE ERROR DURING SEEK COMMAND
133			
	002332	073114	EMT54
	002334	076470	EHT1
	002336	076614	EDT1
	002340	076704	EFT1
134			
135		:ERROR 55	DEVICE CHECK DURING SEEK COMMAND
136			
	002342	073124	EMT55
	002344	076470	EHT1
	002346	076614	EDT1
	002350	076704	EFT1
137			
138		:ERROR 56	PIP IS STILL SET AFTER SEEK - SKI IS RESET
139			
	002352	073136	EMT56
	002354	076470	EHT1
	002356	076614	EDT1
	002360	076704	EFT1
140			
141		:ERROR 57	ATA DID NOT SET DURING SEEK COMMAND
142			
	002362	073154	EMT57
	002364	076470	EHT1
	002366	076614	EDT1
	002370	076704	EFT1
143			
144		:ERROR 60	IVC ERROR DURING SEEK COMMAND - LOST
145		:	VOLUME VALID
146			
	002372	073164	EMT60
	002374	076470	EHT1
	002376	076614	EDT1

002400	076704	EFT1	
147			
148		:ERROR 61	ERRONEOUS IVC ERROR DURING SEEK COMMAND -
149		:	VOLUME VALID IS STILL SET
150			
002402	073202	EMT61	
002404	076470	EHT1	
002406	076614	EDT1	
002410	076704	EFT1	
151			
152		:ERROR 62	MOL IS ZERO, BUT OPI WAS NOT
153		:	REPORTED DURING SEEK COMMAND
154			
002412	073222	EMT62	
002414	076470	EHT1	
002416	076614	EDT1	
002420	076704	EFT1	
155			
156		:ERROR 63	UNUSED
157			
002422	000000	0	
002424	000000	0	
002426	000000	0	
002430	000000	0	
158			
159		:ERROR 64	DRIVE DID NOT DETECT "IVC" ERROR DURING SEEK
160			
002432	073240	EMT64	
002434	076470	EHT1	
002436	076614	EDT1	
002440	076704	EFT1	
161			
162		:ERROR 65	DRIVE EXECUTED A SEEK WITH ERROR SET
163			
002442	073260	EMT65	
002444	076470	EHT1	
002446	076614	EDT1	
002450	076704	EFT1	
164			
165		:ERROR 66	UNEXPECTED ERROR SET IN RMER1
166			
002452	073300	EMT66	
002454	076470	EHT1	
002456	076614	EDT1	
002460	076704	EFT1	
167			
168		:ERROR 67	UNEXPECTED ERROR SET IN RMER2
169			
002462	073312	EMT67	
002464	076470	EHT1	

002466	076614	EDT1	
002470	076704	EFT1	
170			
171		;ERROR	70 ERRONEOUS ''IAE'' ERROR DURING RECALIBRATE
172			
002472	073324	EMT70	
002474	076470	EHT1	
002476	076614	EDT1	
002500	076704	EFT1	
173			
174		;ERROR	71 ''ILF'' ERROR DURING RECALIBRATE
175			
002502	073334	EMT71	
002504	076470	EHT1	
002506	076614	EDT1	
002510	076704	EFT1	
176			
177		;ERROR	72 ''OPI'' ERROR DURING RECALIBRATE DUE TO 'MOL' = 0
178			
002512	073344	EMT72	
002514	076470	EHT1	
002516	076614	EDT1	
002520	076704	EFT1	
179			
180		;ERROR	73 ''OPI'' ERROR DURING RECALIBRATE BECAUSE ON
181		:	CYLINDER DIDNT DROP
182			
002522	073362	EMT73	
002524	076470	EHT1	
002526	076614	EDT1	
002530	076704	EFT1	
183			
184		;ERROR	74 ''IVC'' ERROR DURING RECALIBRATE - ''VV'' = 0
185			
002532	073400	EMT74	
002534	076470	EHT1	
002536	076614	EDT1	
002540	076704	EFT1	
186			
187		;ERROR	75 ERRONEOUS ''IVC'' ERROR DURING RECALIBRATE - ''VV'' = 1
188			
002542	073410	EMT75	
002544	076470	EHT1	
002546	076614	EDT1	
002550	076704	EFT1	
189			
190		;ERROR	76 ''SKI'' ERROR DURING RECALIBRATE
191			
002552	073430	EMT76	
002554	076470	EHT1	

	002556 076614	EDT1	
	002560 076704	EFT1	
192			
193			
194			:ERROR 77 'DVC' OCCURRED DURING RECALIBRATE
	002562 073440	EMT77	
	002564 076470	EHT1	
	002566 076614	EDT1	
	002570 076704	EFT1	
195			
196			:ERROR 100 LOST 'MOL' DURING RECALIBRATE - 'OPI' 0
197			
	002572 073452	EMT100	
	002574 076470	EHT1	
	002576 076614	EDT1	
	002600 076704	EFT1	
198			
199			:ERROR 101 LOST 'VV' DURING RECALIBRATE - 'IVC' = 0
200			
	002602 073470	EMT101	
	002604 076470	EHT1	
	002606 076614	EDT1	
	002610 076704	EFT1	
201			
202			:ERROR 102 'ATA' DID NOT SET DURING RECALIBRATE
203			
	002612 073506	EMT102	
	002614 076470	EHT1	
	002616 076614	EDT1	
	002620 076704	EFT1	
204			
205			:ERROR 103 'DM' DID NOT RESET DURING RECALIBRATE
206			
	002622 073516	EMT103	
	002624 076470	EHT1	
	002626 076614	EDT1	
	002630 076704	EFT1	
207			
208			:ERROR 104 'PIP' IS STIL SET AFTER RECALIBRATE
209			
	002632 073530	EMT104	
	002634 076470	EHT1	
	002636 076614	EDT1	
	002640 076704	EFT1	
210			
211			:ERROR 105 UNEXPECTED 'ILR' ERROR DURING RECALIBRATE
212			
	002642 073546	EMT105	
	002644 076470	EHT1	
	002646 076614	EDT1	

002650	076704	EFT1	
213			
214		:ERROR	106 UNEXPECTED 'RMR' ERROR DURING RECALIBRATE
215			
002652	073556	EMT106	
002654	076470	EHT1	
002656	076614	EDT1	
002660	076704	EFT1	
216			
217		:ERROR	107 'UNS' ERROR DURING RECALIBRATE - AC POWER IS LOW
218			
002662	073566	EMT107	
002664	076470	EHT1	
002666	076614	EDT1	
002670	076704	EFT1	
219			
220		:ERROR	110 CANNOT ACCESS MASSBUS CONTROLLER VIA UNIBUS
221			
002672	073606	EMT110	
002674	076512	EHT110	
002676	076632	EDT110	
002700	076722	EFT110	
222			
223		:ERROR	111 NONEXISTENT DEVICE
224			
002702	073620	EMT111	
002704	076516	EHT111	
002706	076634	EDT111	
002710	076724	EFT111	
225			
226		:ERROR	112 DEVICE NOT AVAILABLE
227			
002712	073626	EMT112	
002714	076516	EHT111	
002716	076634	EDT111	
002720	076724	EFT111	
228			
229		:ERROR	113 BUS TIMEOUT-NED STATUS FAILURE
230			
002722	073634	EMT113	
002724	000000	0	
002726	000000	0	
002730	000000	0	
231			
232		:ERROR	114 DEVICE NOT AN RM05/3/2
233			
002732	073650	EMT114	
002734	076522	EHT114	
002736	076636	EDT114	
002740	076726	EFT114	

234			
235			
236			
	002742	073656	
	002744	076470	
	002746	076614	
	002750	076704	
		EMT115	
		EHT1	
		EDT1	
		EFT1	
237			
238			
239			
	002752	073666	
	002754	076470	
	002756	076614	
	002760	076704	
		EMT116	
		EHT1	
		EDT1	
		EFT1	
240			
241			
242			
	002762	073676	
	002764	076470	
	002766	076614	
	002770	076704	
		EMT117	
		EHT1	
		EDT1	
		EFT1	
243			
244			
245			
	002772	073706	
	002774	076470	
	002776	076614	
	003000	076704	
		EMT120	
		EHT1	
		EDT1	
		EFT1	
246			
247			
248			
	003002	073716	
	003004	076470	
	003006	076614	
	003010	076704	
		EMT121	
		EHT1	
		EDT1	
		EFT1	
249			
250			
251			
	003012	073726	
	003014	076470	
	003016	076614	
	003020	076704	
		EMT122	
		EHT1	
		EDT1	
		EFT1	
252			
253			
254			
	003022	073736	
	003024	076470	
	003026	076614	
	003030	076704	
		EMT123	
		EHT1	
		EDT1	
		EFT1	

;ERROR 115 RMCS1 NOT INITIALIZED BY UNIBUS

;ERROR 116 RMBA NOT INITIALIZED BY UNIBUS

;ERROR 117 RMCS2 NOT INITIALIZED BY UNIBUS

;ERROR 120 RMER1 NOT INITIALIZED BY UNIBUS

;ERROR 121 RMAS NOT INITIALIZED BY UNIBUS

;ERROR 122 RMMR1 NOT INITIALIZED BY UNIBUS

;ERROR 123 RMDS NOT INITIALIZED BY UNIBUS

255			
256		:ERROR	124 RMEC2 NOT INITIALIZED BY UNIBUS
257	003032	073746	EMT124
	003034	076470	EHT1
	003036	076614	EDT1
	003040	076704	EFT1
258			
259		:ERROR	125 RMMR2 NOT INITIALIZED BY UNIBUS
260	003042	073756	EMT125
	003044	076470	EHT1
	003046	076614	EDT1
	003050	076704	EFT1
261			
262		:ERROR	126 RMCS1 NOT CLEARED BY CONTROLLER CLEAR
263	003052	073766	EMT126
	003054	076470	EHT1
	003056	076614	EDT1
	003060	076704	EFT1
264			
265		:ERROR	127 RMBA NOT CLEARED BY CONTROLLER CLEAR
266	003062	074000	EMT127
	003064	076470	EHT1
	003066	076614	EDT1
	003070	076704	EFT1
267			
268		:ERROR	130 RMCS2 NOT CLEARED BY CONTROLLER CLEAR
269	003072	074012	EMT130
	003074	076470	EHT1
	003076	076614	EDT1
	003100	076704	EFT1
270			
271		:ERROR	131 RMER1 NOT CLEARED BY CONTROLLER CLEAR
272	003102	074024	EMT131
	003104	076470	EHT1
	003106	076614	EDT1
	003110	076704	EFT1
273			
274		:ERROR	132 RMA5 NOT CLEARED BY CONTROLLER CLEAR
275	003112	074036	EMT132
	003114	076470	EHT1
	003116	076614	EDT1
	003120	076704	EFT1
276			

277			:ERROR 133	RMMR1 NOT CLEARED BY CONTROLLER CLEAR
278	003122	074050	EMT133	
	003124	076470	EHT1	
	003126	076614	EDT1	
	003130	076704	EFT1	
279				
280			:ERROR 134	RMDS NOT CLEARED BY CONTROLLER CLEAR
281	003132	074062	EMT134	
	003134	076470	EHT1	
	003136	076614	EDT1	
	003140	076704	EFT1	
282				
283			:ERROR 135	RMEC2 NOT CLEARED BY CONTROLLER CLEAR
284	003142	074074	EMT135	
	003144	076470	EHT1	
	003146	076614	EDT1	
	003150	076704	EFT1	
285				
286			:ERROR 136	RMMR2 NOT CLEARED BY CONTROLLER CLEAR
287	003152	074106	EMT136	
	003154	076470	EHT1	
	003156	076614	EDT1	
	003160	076704	EFT1	
288				
289			:ERROR 137	RMCS1 NOT CLEARED BY ERROR CLEAR
290	003162	074120	EMT137	
	003164	076470	EHT1	
	003166	076614	EDT1	
	003170	076704	EFT1	
291				
292			:ERROR 140	RMCS2 NOT CLEARED BY ERROR CLEAR
293	003172	074130	EMT140	
	003174	076470	EHT1	
	003176	076614	EDT1	
	003200	076704	EFT1	
294				
295			:ERROR 141	RMCS1 NOT CLEARED BY DRIVE CLEAR
296	003202	074140	EMT141	
	003204	076470	EHT1	
	003206	076614	EDT1	
	003210	076704	EFT1	
297				
298			:ERROR 142	RMDS NOT CLEARED BY DRIVE CLEAR

299	003212 074150	EMT142	
	003214 076470	EHT1	
	003216 076614	EDT1	
	003220 076704	EFT1	
300			
301		:ERROR 143	RMER1 NOT CLEARED BY DRIVE CLEAR
302	003222 074160	EMT143	
	003224 076470	EHT1	
	003226 076614	EDT1	
	003230 076704	EFT1	
303			
304		:ERROR 144	RMAS NOT CLEARED BY DRIVE CLEAR
305	003232 074170	EMT144	
	003234 076470	EHT1	
	003236 076614	EDT1	
	003240 076704	EFT1	
306			
307		:ERROR 145	RMMR1 NOT CLEARED BY DRIVE CLEAR
308	003242 074200	EMT145	
	003244 076470	EHT1	
	003246 076614	EDT1	
	003250 076704	EFT1	
309			
310		:ERROR 146	RMMR2 NOT CLEARED BY DRIVE CLEAR
311	003252 074210	EMT146	
	003254 076470	EHT1	
	003256 076614	EDT1	
	003260 076704	EFT1	
312			
313		:ERROR 147	RMER2 NOT CLEARED BY DRIVE CLEAR
314	003262 074220	EMT147	
	003264 076470	EHT1	
	003266 076614	EDT1	
	003270 076704	EFT1	
315			
316		:ERROR 150	RMEC2 NOT CLEARED BY DRIVE CLEAR
317	003272 074230	EMT150	
	003274 076470	EHT1	
	003276 076614	EDT1	
	003300 076704	EFT1	
318			
319		:ERROR 151	MEDIUM NOT ON LINE
320			

003302	074240	EMT151
003304	076470	EHT1
003306	076614	EDT1
003310	076704	EFT1
321		
322		
323		:ERROR 152 DRIVE FAULT
003312	074252	EMT152
003314	076470	EHT1
003316	076614	EDT1
003320	076704	EFT1
324		
325		
326		:ERROR 153 UNSAFE SHOULD BE SET BECAUSE DVC IS SET
003322	074264	EMT153
003324	076470	EHT1
003326	076614	EDT1
003330	076704	EFT1
327		
328		
329		:ERROR 154 UNSAFE SHOULD NOT BE SET, AC IS LOW
003332	074302	EMT154
003334	076470	EHT1
003336	076614	EDT1
003340	076704	EFT1
330		
331		
332		:ERROR 155 VOLUME VALID NOT SET BY PACK ACK
003342	074320	EMT155
003344	076470	EHT1
003346	076614	EDT1
003350	076704	EFT1
333		
334		
335		:ERROR 156 OFFSET MODE NOT SET BY OFFSET COMMAND
003352	074332	EMT156
003354	076470	EHT1
003356	076614	EDT1
003360	076704	EFT1
336		
337		
338		:ERROR 157 OFFSET MODE NOT RESET BY RTC COMMAND
003362	074344	EMT157
003364	076470	EHT1
003366	076614	EDT1
003370	076704	EFT1
339		
340		
341		:ERROR 160 RMOF NOT RESET BY RIP COMMAND
003372	074356	EMT160

003374	076470	EHT1	
003376	076614	EDT1	
003400	076704	EFT1	
342			
343			
344			;ERROR 161 RMDA NOT RESET BY RIP COMMAND
003402	074366	EMT161	
003404	076470	EHT1	
003406	076614	EDT1	
003410	076704	EFT1	
345			
346			
347			;ERROR 162 RMDC NOT RESET BY RIP COMMAND
003412	074400	EMT162	
003414	076470	EHT1	
003416	076614	EDT1	
003420	076704	EFT1	
348			
349			
350			;ERROR 163 DATA WAS ECC CORRECTED BUT DOES NOT COMPARE WITH
351			WRITE BUFFER
003422	076264	EMT336	
003424	076552	EHT336	
003426	076650	EDT336	
003430	076740	EFT336	
352			
353			
354			;ERROR 164 OPI SHOULD NOT BE SET
003432	074422	EMT164	
003434	076470	EHT1	
003436	076614	EDT1	
003440	076704	EFT1	
355			
356			
357			;ERROR 165 IVC SHOULD NOT BE SET
003442	074430	EMT165	
003444	076470	EHT1	
003446	076614	EDT1	
003450	076704	EFT1	
358			
359			
360			;ERROR 166 IAE SHOULD NOT BE SET
003452	074436	EMT166	
003454	076470	EHT1	
003456	076614	EDT1	
003460	076704	EFT1	
361			
362			
363			;ERROR 167 NEM SHOULD NOT BE SET
003462	074444	EMT167	

003464	076470	EMT1	
003466	076614	EDT1	
003470	076704	EFT1	
364			
365			
366			:ERROR 170 INCORRECT 'MOL' STATUS DURING DIAGNOSTIC MODE
003472	074452	EMT170	
003474	076470	EMT1	
003476	076614	EDT1	
003500	076704	EFT1	
367			
368			
369			:ERROR 171 'ATA' NOT SET DURING RETURN TO CENTERLINE
003502	074464	EMT171	
003504	076470	EMT1	
003506	076614	EDT1	
003510	076704	EFT1	
370			
371			
372			:ERROR 172 'ATA' NOT SET BY OFFSET COMMAND
003512	074474	EMT172	
003514	076470	EMT1	
003516	076614	EDT1	
003520	076704	EFT1	
373			
374			
375			:ERROR 173 RMER2 NOT INITIALIZED BY UNIBUS INIT
003522	074504	EMT173	
003524	076470	EMT1	
003526	076614	EDT1	
003530	076704	EFT1	
376			
377			
378			:ERROR 174 RMER2 NOT INITIALIZED BY CONTROLLER CLEAR
003532	074514	EMT174	
003534	076470	EMT1	
003536	076614	EDT1	
003540	076704	EFT1	
379			
380			
381			:ERROR 175 SELECTED DEVICE IS IN WRITE PROTECT
003542	074526	EMT175	
003544	076470	EMT1	
003546	076614	EDT1	
003550	076704	EFT1	
382			
383			
384			:ERROR 176 CANNOT SET DIAGNOSTIC MODE
003552	074534	EMT176	
003554	076470	EMT1	

	003556	076614	EDT1	
	003560	076704	EFT1	
385				
386				
387				
	003562	000000	0	
	003564	000000	0	
	003566	000000	0	
	003570	000000	0	
388				
389				
390				
	003572	074544	EMT200	
	003574	076470	EHT1	
	003576	076614	EDT1	
	003600	076704	EFT1	
391				
392				
393				
	003602	074556	EMT201	
	003604	076470	EHT1	
	003606	076614	EDT1	
	003610	076704	EFT1	
394				
395				
396				
	003612	074570	EMT202	
	003614	076470	EHT1	
	003616	076614	EDT1	
	003620	076704	EFT1	
397				
398				
399				
	003622	074602	EMT203	
	003624	076470	EHT1	
	003626	076614	EDT1	
	003630	076704	EFT1	
400				
401				
402				
	003632	074614	EMT204	
	003634	076470	EHT1	
	003636	076614	EDT1	
	003640	076704	EFT1	
403				
404				
405				
	003642	074632	EMT205	
	003644	076470	EHT1	
	003646	076614	EDT1	

```

;ERROR 177  --RESERVED FOR POWER MONITOR BIT FAILURE--

;ERROR 200  INCORRECT 'PIP' STATUS DURING DIAGNOSTIC MODE

;ERROR 201  INCORRECT 'WRL' STATUS DURING DIAGNOSTIC MODE

;ERROR 202  INCORRECT 'SKI' STATUS DURING DIAGNOSTIC MODE

;ERROR 203  INCORRECT 'DVC' STATUS DURING DIAGNOSTIC MODE

;ERROR 204  'VV' WAS NOT RESET BY MAINTENANCE UNIT READY

;ERROR 205  SELECTED DEVICE HAS A PERSISTENT 'SKI' ERROR
  
```


003650	076704	EFT1	
406			
407		;ERROR	206 'LBC' DID NOT SET DURING DIAGNOSTIC MODE
408			
003652	074642	EMT206	
003654	076470	EHT1	
003656	076614	EDT1	
003660	076704	EFT1	
409			
410		;ERROR	207 UNEXPECTED LOSS OF 'MOL' - MEDIUM IS OFF LINE
411			
003662	074652	EMT207	
003664	076470	EHT1	
003666	076614	EDT1	
003670	076704	EFT1	
412			
413		;ERROR	210 UNEXPECTED LOSS OF VOLUME VALID - 'VV' = 0
414			
003672	074664	EMT210	
003674	076470	EHT1	
003676	076614	EDT1	
003700	076704	EFT1	
415			
416		;ERROR	211 UNEXPECTED MECHANICAL MOTION - 'PIP' - 1
417			
003702	074672	EMT211	
003704	076470	EHT1	
003706	076614	EDT1	
003710	076704	EFT1	
418			
419		;ERROR	212 UNEXPECTED DEVICE FAULT - 'DVC' = 1
420			
003712	074706	EMT212	
003714	076470	EHT1	
003716	076614	EDT1	
003720	076704	EFT1	
421			
422		;ERROR	213 UNEXPECTED SEEK INCOMPLETE ERROR - 'SKI' = 1
423			
003722	074722	EMT213	
003724	076470	EHT1	
003726	076614	EDT1	
003730	076704	EFT1	
424			
425		;ERROR	214 DRIVE EXECUTED A RECALIBRATE WITH ERROR SET
426			
003732	074732	EMT214	
003734	076502	EHT2	
003736	076624	EDT2	
003740	076714	EFT2	

427			
428			
429		;ERROR 215	DRIVE DID NOT DETECT "IVC" ERROR DURING RECALIBRATE
	003742	074752	EMT215
	003744	076502	EHT2
	003746	076624	EDT2
	003750	076714	EFT2
430			
431		;ERROR 216	INCORRECT "IVC" STATUS
432			
	003752	074764	EMT216
	003754	076470	EHT1
	003756	076614	EDT1
	003760	076704	EFT1
433			
434		;ERROR 217	INCORRECT "IAE" STATUS
435			
	003762	074774	EMT217
	003764	076470	EHT1
	003766	076614	EDT1
	003770	076704	EFT1
436			
437		;ERROR 220	INCORRECT "WLE" STATUS
438			
	003772	075004	EMT220
	003774	076470	EHT1
	003776	076614	EDT1
	004000	076704	EFT1
439			
440		;ERROR 221	INCORRECT "OPI" STATUS
441			
	004002	075014	EMT221
	004004	076470	EHT1
	004006	076614	EDT1
	004010	076704	EFT1
442			
443		;ERROR 222	RM DID NOT DETECT RMR ERROR
444			
	004012	075024	EMT222
	004014	076470	EHT1
	004016	076614	EDT1
	004020	076704	EFT1
445			
446		;ERROR 223	RM DID NOT DETECT PARITY ERROR ON MASSBUS CONTROL BUS
447			
	004022	075034	EMT223
	004024	076526	EHT223
	004026	076640	EDT223
	004030	076730	EFT223

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

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

492	004212	000000	0	
	004214	000000	0	
	004216	000000	0	
	004220	000000	0	
493				
494			;ERROR	243 UNUSED
495	004222	000000	0	
	004224	000000	0	
	004226	000000	0	
	004230	000000	0	
496				
497			;ERROR	244 UNUSED
498	004232	000000	0	
	004234	000000	0	
	004236	000000	0	
	004240	000000	0	
499				
500			;ERROR	245 UNUSED
501	004242	000000	0	
	004244	000000	0	
	004246	000000	0	
	004250	000000	0	
502				
503			;ERROR	246 'ATA' NOT RESET BY GO WHEN 'ERR' = 0
504	004252	075110	EMT246	
	004254	076470	EHT1	
	004256	076614	EDT1	
	004260	076704	EFT1	
505				
506			;ERROR	247 'ATA' NOT RESET BY WRITING RMAS
507	004262	075120	EMT247	
	004264	076470	EHT1	
	004266	076614	EDT1	
	004270	076704	EFT1	
508				
509			;ERROR	250 'ATA' WAS RESET BY GO WHEN 'ERR' = 1
510	004272	075132	EMT250	
	004274	076470	EHT1	
	004276	076614	EDT1	
	004300	076704	EFT1	
511				
512			;ERROR	251 PROGRAM INTERRUPT WAS NOT GENERATED
513				

	004302	075146	EMT251	
	004304	076502	EHT2	
	004306	076624	EDT2	
	004310	076714	EFT2	
514				
515				
516				:ERROR 252 PROGRAM INTERRUPT SHOULD NOT HAVE BEEN GENERATED
	004312	075154	EMT252	
	004314	076502	EHT2	
	004316	076624	EDT2	
	004320	076714	EFT2	
517				
518				
519				:ERROR 253 OFFSET MODE WAS NOT RESET BY WRITING RMDC
	004322	075162	EMT253	
	004324	076470	EHT1	
	004326	076614	EDT1	
	004330	076704	EFT1	
520				
521				
522				:ERROR 254 INCORRECT 'ILF' STATUS
	004332	075200	EMT254	
	004334	076470	EHT1	
	004336	076614	EDT1	
	004340	076704	EFT1	
523				
524				
525				:ERROR 255 INCORRECT 'ATA' STATUS
	004342	075210	EMT255	
	004344	076470	EHT1	
	004346	076614	EDT1	
	004350	076704	EFT1	
526				
527				
528				:ERROR 256 INCORRECT 'ILR' STATUS
	004352	075220	EMT256	
	004354	076540	EHT256	
	004356	076640	EDT223	
	004360	076730	EFT223	
529				
530				
531				:ERROR 257 INVALID IAE STATUS DURING SEARCH COMMAND
	004362	075230	EMT257	
	004364	076470	EHT1	
	004366	076614	EDT1	
	004370	076704	EFT1	
532				
533				
534				:ERROR 260 'IVC' WAS NOT DETECTED DURING SEARCH COMMAND
	004372	075242	EMT260	

004374	076470	EMT1	
004376	076614	EDT1	
004400	076704	EFT1	
535			
536		;ERROR	261 DRIVE EXECUTED SEARCH WITH ERROR SET
537			
004402	075254	EMT261	
004404	076470	EHT1	
004406	076614	EDT1	
004410	076704	EFT1	
538			
539		;ERROR	262 'LBC' ERROR NOT SET DURING DIAGNOSTIC MODE
540			
004412	075274	EMT262	
004414	076470	EHT1	
004416	076614	EDT1	
004420	076704	EFT1	
541			
542		;ERROR	263 'SKI' ERROR DURING SEARCH COMMAND
543			
004422	075304	EMT263	
004424	076470	EHT1	
004426	076614	EDT1	
004430	076704	EFT1	
544			
545		;ERROR	264 'IVC' ERROR DURING SEARCH - LOST VOLUME VALID
546			
004432	075314	EMT264	
004434	076470	EHT1	
004436	076614	EDT1	
004440	076704	EFT1	
547			
548		;ERROR	265 ERRONEOUS IVC ERROR DURING SEARCH - VOLUME IS VALID
549			
004442	075334	EMT265	
004444	076470	EHT1	
004446	076614	EDT1	
004450	076704	EFT1	
550			
551		;ERROR	266 DEVICE FAULT (DVC) DURING SEARCH
552			
004452	075354	EMT266	
004454	076470	EHT1	
004456	076614	EDT1	
004460	076704	EFT1	
553			
554		;ERROR	267 SKI SHOULD HAVE BEEN SET BECAUSE CYLINDER
555		:	ADDRESS IS TOO LARGE
556			
004462	075366	EMT267	

004464	076470	EH11	
004466	076614	ED11	
004470	076704	EFT1	
557			
558			
559			:ERROR 270 OPI ERROR DURING SEARCH BECAUSE MOL = 0
004472	075404	EMT270	
004474	076470	EH11	
004476	076614	ED11	
004500	076704	EFT1	
560			
561			:ERROR 271 OPI ERROR DURING SEARCH BECAUSE ON CYLINDER
562			: DIDN'T DROP
563			
004502	075420	EMT271	
004504	076470	EH11	
004506	076614	ED11	
004510	076704	EFT1	
564			
565			:ERROR 272 LOST MOL DURING SEARCH, OPI IS NOT SET
566			
004512	075436	EMT272	
004514	076470	EH11	
004516	076614	ED11	
004520	076704	EFT1	
567			
568			:ERROR 273 PIP STIL SET AFTER SEARCH
569			
004522	075454	EMT273	
004524	076470	EH11	
004526	076614	ED11	
004530	076704	EFT1	
570			
571			:ERROR 274 PARITY ERROR OCCURRED WHILE WRITING REMOTE
572			: REGISTERS BUT MXF DID NOT SET
573			
004532	075472	EMT274	
004534	076470	EH11	
004536	076614	ED11	
004540	076704	EFT1	
574			
575			:ERROR 275 MXF ERROR - COMPOSITE ERROR OCCURRED BEFORE DATA
576			: COMMAND STARTED
577			
004542	075510	EMT275	
004544	076470	EH11	
004546	076614	ED11	
004550	076704	EFT1	
578			
579			:ERROR 276 'OPI' ERROR DURING DATA TRANSFER BECAUSE 'MOL' WAS

580		:	ZERO
581			
	004552	075522	EMT276
	004554	076470	EHT1
	004556	076614	EDT1
	004560	076704	EFT1
582			
583		:ERROR 277	'DPI' ERROR DURING DATA TRANSFER BECAUSE 1) ON
584		:	CYLINDER DIDN'T DROP OR 2) SEARCH TIMED OUT OR
585		:	3) RUN TIMED OUT
586			
	004562	075536	EMT277
	004564	076470	EHT1
	004566	076614	EDT1
	004570	076704	EFT1
587			
588		:ERROR 300	'IVC' ERROR DURING DATA TRANSFER BECAUSE VOLUME
589		:	WAS NOT VALID
590			
	004572	075554	EMT300
	004574	076470	EHT1
	004576	076614	EDT1
	004600	076704	EFT1
591			
592		:ERROR 301	ERRONEOUS 'IVC' ERROR DURING DATA TRANSFER - VOLUME
593		:	IS VALID
594			
	004602	075574	EMT301
	004604	076470	EHT1
	004606	076614	EDT1
	004610	076704	EFT1
595			
596		:ERROR 302	'ILR' ERROR DURING DATA TRANSFER
597			
	004612	075616	EMT302
	004614	076470	EHT1
	004616	076614	EDT1
	004620	076704	EFT1
598			
599		:ERROR 303	'ILF' ERROR DURING DATA TRANSFER
600			
	004622	075630	EMT303
	004624	076470	EHT1
	004626	076614	EDT1
	004630	076704	EFT1
601			
602		:ERROR 304	'RMR' ERROR DURING DATA TRANSFER
603			
	004632	075642	EMT304
	004634	076470	EHT1
	004636	076614	EDT1

004640	076704	EFT1	
604			
605		;ERROR 305	INCORRECT 'IAE' STATUS DURING DATA TRANSFER
606			
004642	075654	EMT305	
004644	076470	EHT1	
004646	076614	EDT1	
004650	076704	EFT1	
607			
608		;ERROR 306	'SKI' ERROR DURING DATA TRANSFER
609			
004652	075666	EMT306	
004654	076470	EHT1	
004656	076614	EDT1	
004660	076704	EFT1	
610			
611		;ERROR 307	DRIVE DID NOT DETECT SKI ERROR DUE TO CYLINDER
612			
004662	075676	EMT307	
004664	076470	EHT1	
004666	076614	EDT1	
004670	076704	EFT1	
613			
614		;ERROR 310	DEVICE FAULT DURING DATA TRANSFER
615			
004672	075716	EMT310	
004674	076470	EHT1	
004676	076614	EDT1	
004700	076704	EFT1	
616			
617		;ERROR 311	LOSS OF BIT CLOCK DURING DATA TRANSFER
618			
004702	075730	EMT311	
004704	076470	EHT1	
004706	076614	EDT1	
004710	076704	EFT1	
619			
620		;ERROR 312	LOSS OF SYSTEM CLOCK DURING DATA TRANSFER
621			
004712	075742	EMT312	
004714	076470	EHT1	
004716	076614	EDT1	
004720	076704	EFT1	
622			
623		;ERROR 313	UNSAFE ERROR DURING DATA TRANSFER (DVC - 0)
624			
004722	075754	EMT313	
004724	076470	EHT1	
004726	076614	EDT1	
004730	076704	EFT1	

625			
626			
627		:ERROR 314	DRIVE TIMING ERROR DURING DATA TRANSFER
	004732	075774	EMT314
	004734	076470	EHT1
	004736	076614	EDT1
	004740	076704	EFT1
628			
629		:ERROR 315	WRITE LOCK ERROR
630			
	004742	076006	EMT315
	004744	076470	EHT1
	004746	076614	EDT1
	004750	076704	EFT1
631			
632		:ERROR 316	ERRONEOUS WRITE LOCK ERROR
633			
	004752	076020	EMT316
	004754	076470	EHT1
	004756	076614	EDT1
	004760	076704	EFT1
634			
635		:ERROR 317	HEADER CRC ERROR DURING DATA TRANSFER
636			
	004762	076032	EMT317
	004764	076470	EHT1
	004766	076614	EDT1
	004770	076704	EFT1
637			
638		:ERROR 320	FORMAT ERROR DURING DATA TRANSFER
639			
	004772	076042	EMT320
	004774	076470	EHT1
	004776	076614	EDT1
	005000	076704	EFT1
640			
641		:ERROR 321	HEADER COMPARE ERROR DURING DATA TRANSFER
642			
	005002	076052	EMT321
	005004	076470	EHT1
	005006	076614	EDT1
	005010	076704	EFT1
643			
644		:ERROR 322	HEADER ERRORS SHOULD NOT BE SET
645			
	005012	076062	EMT322
	005014	076470	EHT1
	005016	076614	EDT1
	005020	076704	EFT1

646			
647			
648			
	005022	076070	
	005024	076470	
	005026	076614	
	005030	076704	
			EMT323
			EHT1
			EDT1
			EFT1
649			
650			
651			
	005032	076100	
	005034	076470	
	005036	076614	
	005040	076704	
			EMT324
			EHT1
			EDT1
			EFT1
652			
653			
654			
	005042	076112	
	005044	076470	
	005046	076614	
	005050	076704	
			EMT325
			EHT1
			EDT1
			EFT1
655			
656			
657			
	005052	076124	
	005054	076470	
	005056	076614	
	005060	076704	
			EMT326
			EHT1
			EDT1
			EFT1
658			
659			
660			
	005062	076142	
	005064	076470	
	005066	076614	
	005070	076704	
			EMT327
			EHT1
			EDT1
			EFT1
661			
662			
663			
	005072	076154	
	005074	076470	
	005076	076614	
	005100	076704	
			EMT330
			EHT1
			EDT1
			EFT1
664			
665			
666			
	005102	076164	
	005104	076470	
	005106	076614	
	005110	076704	
			EMT331
			EHT1
			EDT1
			EFT1

667

668		:ERROR 332	DATA LATE ERROR DURING DATA TRANSFER
669	005112 076176	EMT332	
	005114 076470	EHT1	
	005116 076614	EDT1	
	005120 076704	EFT1	
670			
671		:ERROR 333	PIP STIL SET AFTER DATA TRANSFER - SKI = 0
672	005122 076210	EMT333	
	005124 076470	EHT1	
	005126 076614	EDT1	
	005130 076704	EFT1	
673			
674		:ERROR 334	LOST MOL DURING DATA TRANSFER - OPI = 0
675	005132 076226	EMT334	
	005134 076470	EHT1	
	005136 076614	EDT1	
	005140 076704	EFT1	
676			
677		:ERROR 335	LOST VOLUME VALID DURING DATA TRANSFER - IVC - 0
678	005142 076244	EMT335	
	005144 076470	EHT1	
	005146 076614	EDT1	
	005150 076704	EFT1	
679			
680		:ERROR 336	DATA READ DOES NOT COMPARE WITH DATA WRITTEN
681	005152 076264	EMT336	
	005154 076552	EHT336	
	005156 076650	EDT336	
	005160 076740	EFT336	
682			
683		:ERROR 337	WRITE CHECK ERROR NOT DETECTED
684	005162 076274	EMT337	
	005164 076564	EHT337	
	005166 076660	EDT337	
	005170 076750	EFT337	
685			
686		:ERROR 340	WRITE CHECK ERROR AT UNEXPECTED ADDRESS
687	005172 076304	EMT340	
	005174 076552	EHT336	
	005176 076650	EDT336	
	005200 076740	EFT336	
688			
689		:ERROR 341	INCORRECT DATA DURING WRITE CHECK ERROR

690	005202 076316	EMT341	
	005204 076552	EHT336	
	005206 076650	EDT336	
	005210 076740	EFT336	
691			
692		:ERROR 342	'IVC' ERROR NOT DETECTED DURING DATA TRANSFER
693	005212 076324	EMT342	
	005214 076470	EHT1	
	005216 076614	EDT1	
	005220 076704	EFT1	
694			
695		:ERROR 343	'FER' NOT DETECTED DURING DATA TRANSFER
696	005222 076336	EMT343	
	005224 076470	EHT1	
	005226 076614	EDT1	
	005230 076704	EFT1	
697			
698		:ERROR 344	'HCE' NOT DETECTED DURING DATA TRANSFER
699	005232 076350	EMT344	
	005234 076576	EHT344	
	005236 076670	EDT344	
	005240 076760	EFT344	
700			
701		:ERROR 345	'BSE' NOT DETECTED DURING DATA TRANSFER
702	005242 076362	EMT345	
	005244 076470	EHT1	
	005246 076614	EDT1	
	005250 076704	EFT1	
703			
704		:ERROR 346	HEADER ERROR WAS DETECTED W/ HCI SET
705	005252 076372	EMT346	
	005254 076470	EHT1	
	005256 076614	EDT1	
	005260 076704	EFT1	
706			
707		:ERROR 347	DATA TRANSFER NOT ABORTED W/ COMP ERROR SET
708	005262 076406	EMT347	
	005264 076470	EHT1	
	005266 076614	EDT1	
	005270 076704	EFT1	
709			
710		:ERROR 350	LOST VOLUME VALID DURING SEARCH - 'IVC' = 0
711			

	005272 076420	EMT350	
	005274 076470	EHT1	
	005276 076614	EDT1	
	005300 076704	EFT1	
712			
713			
714			
	005302 076436	EMT351	
	005304 076470	EHT1	
	005306 076614	EDT1	
	005310 076704	EFT1	
715			
716			
717			
	005312 076446	EMT352	
	005314 000000	0	
	005316 000000	0	
	005320 000000	0	
718			
719			
720			
	005322 076452	EMT353	
	005324 076610	EHT353	
	005326 076702	EDT353	
	005330 076770	EFT353	
721			
722			
723			
	005332 076462	EMT354	
	005334 076470	EHT1	
	005336 076614	EDT1	
	005340 076704	EFT1	
724			
725			

;ERROR 351 'ATA' DID NOT SET DURING SEARCH

;ERROR 352 PROGRAM TIMEOUT WHILE TESTING RMLA

;ERROR 353 LOOK AHEAD TEST FAILS

;ERROR 354 BSE SHOULD NOT BE SET

;PUT ERROR TABLE HERE

1 .SBTTL ERROR TABLE USAGE

2
3 :THE ERROR TABLE ABOVE CONSISTS OF FOUR WORD ENTRIES FOR EACH ERROR
4 :NUMBER, I.E.,

5 :
6 : EMT - ERROR MESSAGE TABLE ADDRESS
7 : EHT - ERROR HEADER TABLE ADDRESS
8 : EDT - ERROR DATA TABLE ADDRESS
9 : EFT - ERROR FORMAT TABLE ADDRESS

10 :
11 :THE EMT ENTRY IS THE ADDRESS OF THE TABLE OF ERROR MESSAGE STRINGS
12 :FOR THE PARTICULAR ERROR. EACH ERROR MESSAGE TABLE LISTS THE ADDRESS
13 :OF ONE OR MORE ERROR MESSAGE STRINGS WHICH ARE TO BE FORMATTED AND
14 :TYPED BY THE ERROR TYPE SUBROUTINE. IF THE EMT ENTRY IS ZERO, THERE IS
15 :NO MESSAGE TO BE TYPED FOR THE ERROR.

16 :
17 :SIMILARLY, THE EHT, EDT, AND EFT ENTRIES ARE ADDRESSES OF TABLES
18 :OF HEADER, DATA AND FORMAT INFORMATION FOR A GIVEN ERROR. EACH ENTRY
19 :IN THE ERROR HEADER TABLE MAY OR MAY NOT HAVE AN ASSOCIATED LINE OF
20 :DATA, HOWEVER, EACH DATA LINE MUST HAVE AN ASSOCIATED FORMAT AND
21 :HEADER. THAT IS, A HEADER LINE MAY BE PRINTED WITHOUT ANY DATA,
22 :BUT A DATA LINE IS NOT PRINTED WITHOUT A HEADER, AND EACH DATA LINE
23 :MUST ALSO HAVE A FORMAT.

24 :
25 :IN SUMMARY,
26 : EACH NONZERO ENTRY IS THE ADDRESS OF A TABLE,
27 : EACH TABLE IS A LIST OF ADDRESSES WHICH DEFINES THE LOCATIONS
28 : OF MESSAGE STRINGS, HEADERS, DATA OR FORMAT.


```
1      ;THIS ROUTINE HANDLES UNEXPECTED TIMEOUTS
2
3 005342 011600      BADTMO: MOV      (SP),R0      ;SAVE PC WHERE THE TIME OUT OCCURED
4 005344 005740      TST      -(R0)      ;ADJUST PC -2
5 005346 022626      CMP      (SP)+,(SP)+    ;RESTORE STACK POINTER
6 005350 104401 005356 TYPE      ,65$      ;:TYPE ASCII STRING
   005354 000417      BR       64$      ;:GET OVER THE ASCIIZ
   ;:65$: .ASCIIZ <CRLF>/UNEXPECTED BUS TIMEOUT, PC=/
   64$:
7 005414 010046      MOV      R0,-(SP)      ;SETUP FOR TYPING OUT PC
8 005416 104402      TYPOC
9 005420 000240      NOP      ;PUT 'HALT(0)' INSTRUCTION HERE IF YOU WISH
   ;TO STOP ON UNEXPECTED TIMECUT.
10
11
12      .SBTTL START OF PROGRAM
13
14 005422 012737 177777 001330 START1: MOV      #-1,CHGADR      ;CHANGE RH/RM BUS ADDRESS
15 005430 000402      BR       START2
16
17 005432 005037 001330 START:   CLR      CHGADR      ;NO CHANGE IN ADDRESS
18 005436 000240      START2: NOP
19 005440 005227 000000      INC      #0      ;TTY LOOP, WAIT FOR INCREMENT
20 005444 001375      BNE      .-4      ;OF WORD
21 005446 000005      RESET     ;RESET THE WORLD
22
23      .SBTTL INITIALIZE THE COMMON TAGS
   ;:CLEAR THE COMMON TAGS ($CMTAG) AREA
   MOV      #$CMTAG,R6      ;:FIRST LOCATION TO BE CLEARED
   CLR      (R6)+      ;:CLEAR MEMORY LOCATION
   CMP      #SWR,R6      ;:DONE?
   BNE      .-6      ;:LOOP BACK IF NO
   MOV      #STACK,SP      ;:SETUP THE STACK POINTER
   ;:INITIALIZE A FEW VECTORS
   MOV      #SCOPE,@IOTVEC ;:IOT VECTOR FOR SCOPE ROUTINE
   MOV      #340,@IOTVEC+2 ;:LEVEL 7
   MOV      #ERROR,@EMTVEC ;:EMT VECTOR FOR ERROR ROUTINE
   MOV      #340,@EMTVEC+2 ;:LEVEL 7
   MOV      #TRAP,@TRAPVEC ;:TRAP VECTOR FOR TRAP CALLS
   MOV      #340,@TRAPVEC+2 ;:LEVEL 7
   MOV      #PWRDN,@PWRVEC ;:POWER FAILURE VECTOR
   MOV      #340,@PWRVEC+2 ;:LEVEL 7
   MOV      $ENDCT,$EOPCT ;:SETUP END-OF-PROGRAM COUNTER
   CLR      $TIMES      ;:INITIALIZE NUMBER OF ITERATIONS
   CLR      $ESCAPE      ;:CLEAR THE ESCAPE ON ERROR ADDRESS
   MOVB     #1,$ERMAX      ;:ALLOW ONE ERROR PER TEST
   MOV      #,$SLPADR      ;:INITIALIZE THE LOOP ADDRESS FOR SCOPE
   MOV      #,$SLPERR      ;:SETUP THE ERROR LOOP ADDRESS
   ;:SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
   ;:EQUAL TO A '-1', SETUP FOR A SOFTWARE SWITCH REGISTER.
   MOV      @ERRVEC,-(SP) ;:SAVE ERROR VECTOR
   MOV      #64$,@ERRVEC ;:SET UP ERROR VECTOR
   MOV      #DSWR,SWR      ;:SETUP FOR A HARDWARE SWICH REGISTER
   MOV      #DDISP,DISPLAY ;:AND A HARDWARE DISPLAY REGISTER
   CMP      #-1,@SWR      ;:TRY TO REFERENCE HARDWARE SWR
   BNE      66$      ;:BRANCH IF NO TIMEOUT TRAP OCCURRED
   ;:AND THE HARDWARE SWR IS NOT = -1
   BR       65$      ;:BRANCH IF NO TIMEOUT
```

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

005676 005037 001230      CLR    $PASS      ;;CLEAR PASS COUNT
005702 132737 000200 001243  BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
005710 001403      BEQ    67$      ;;YES, USE NON-APT SWITCH
005712 012737 001244 001154  MOV    #SSWREG,SWR  ;;NO, USE APT SWITCH REGISTER
005720
24      67$:
25      ;SETUP 'TIMEOUT' TRAP VECTOR FOR UNEXPECTED BUS TIMEOUTS
26      MOV    #BADTMO,ERRVEC  ;;SETUP FOR UNEXPECTED TIMEOUT
27      MOV    #PR6,ERRVEC+2  ;;LEVEL 6
28
      .SBTTL  TYPE PROGRAM NAME
      ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
005734 005227 177777      INC    #-1      ;;FIRST TIME?
005740 001035      BNE    68$      ;;BRANCH IF NO
005742 022737 037242 000042  CMP    #SENDAD,@#42  ;;ACT-11?
005750 001431      BEQ    68$      ;;BRANCH IF YES
005752 104401 005760      TYPE    ,69$      ;;TYPE ASCIZ STRING
005756 000426      BR     68$      ;;GET OVER THE ASCIZ
      ;;69$: .ASCIZ <CRLF>@CZRM080 - RM05/3/2 FUNCTIONAL TEST, PT 3@<CRLF>
      68$:
      .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
006034 005737 000042      TST    @#42      ;;ARE WE RUNNING UNDER XXDP/ACT?
006040 001012      BNE    70$      ;;BRANCH IF YES
006042 123727 001242 000001  CMPB   $ENV,#1      ;;ARE WE RUNNING UNDER APT?
006050 001406      BEQ    70$      ;;BRANCH IF YES
006052 023727 001154 000176  CMP    SWR,#SWREG  ;;SOFTWARE SWITCH REG SELECTED?
006060 001005      BNE    71$      ;;BRANCH IF NO
006062 104407      GTSWR      ;;GET SOFT-SWR SETTINGS
006064 000403      BR     71$
006066 112737 000001 001150 70$: MOVB   #1,$AUTOB  ;;SET AUTO-MODE INDICATOR
006074      71$:

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

```

47 006203 000
48 006204 104401 001217
49 006210 000517
50
51 006212 005227 177777 2$: INC #1 ; FIRST TIME THRU HERE ?
52 006216 001114 BNE 3$ ; NO
53 006220 104401 006226 TYPE 75$ ; TYPE ASCII STRING
006224 000410 BR 74$ ; GET OVER THE ASCII
; 75$: .ASCIIZ <CRLF>/TO TEST DRIVE /
74$:
54 006246 005046 CLR -(SP) ; CLEAR WORD ON STACK
55 006250 113716 001332 MOVB XXDP,(SP) ; GET DRIVE ADDRESS
56 006254 104403 TYPOS ; TYPE DRIVE ADDRESS
57 006256 001 .BYTE 1 ; ONLY 1 CHARACTER
58 006257 000 .BYTE 0 ; SUPPRESS LEADING ZEROS
59 006260 104401 006266 TYPE 77$ ; TYPE ASCII STRING
006264 000431 BR 76$ ; GET OVER THE ASCII
; 77$: .ASCIIZ /, HALT PROGRAM, REMOVE RRDPAK AND REPLACE IT/<CRLF>
76$:
60 006350 006350 104401 006356 TYPE 78$ ; TYPE ASCII STRING
006354 000435 BR 3$ ; GET OVER THE ASCII
; 78$: .ASCIIZ /WITH A WORK PAK, CLEAR LOCATION 40 AND RESTART PROGRAM./<CRLF>
3$:
64
65 006450 005737 000042 ; CHECK FOR AUTO MODE OR STANDALONE MODE
66 006454 001561 TST 242 ; RUNNING IN AUTO MODE ?
67 006456 012737 000377 001300 BEQ STANDALONE ; BR IF NO
68 MOV #377,$DEVM ; SET DEVICE MAP FOR ALL DRIVES
69
; PROGRAM IS RUNNING IN AUTO MODE - SEE IF SIZING IS ALLOWED
70 XSIZ:
71 006464 132737 000200 001243 BITB #BIT7,$ENVM ; SIZING ALLOWED ?
72 006472 001146 BNE 12$ ; NO
73
74 006474 005001 CLR R1 ; START FROM DRIVE 0
75 006476 013700 001276 MOV $BASE,R0 ; LOAD THE BASE ADDRESS
76 006502 104401 071430 TYPE ,SYSTAT ; TYPE 'UNIT STATUS:'
77
78 006506 136137 071746 001300 1$: BITB ATNTBL(R1),$DEVM ; IS DEVICE PRESENT IN MAP ?
79 006514 001531 BEQ 11$ ; BR IF NO
80 006516 104401 001217 TYPE ,CRLF ; CR-LF
81 006522 010146 MOV R1,-(SP) ; SAVE R1 FOR TYPEOUT
006524 104403 TYPOS ; GO TYPE--OCTAL ASCII
006526 002 .BYTE 2 ; TYPE 2 DIGIT(S)
006527 000 .BYTE 0 ; SUPPRESS LEADING ZEROS
82 006530 104401 071640 TYPE ,BLNKS4 ; TYPE 4 BLANKS
83
84 006534 012760 000040 000010 MOV #CLR,RMCS2(R0) ; CLEAR MASS BUS
85 006542 010160 000010 MOV R1,RMCS2(R0) ; LOAD THE DRIVE ADDRESS
86 006546 005760 000012 TST RMD5(R0) ; ACCESS DRIVE REGISTER
87 006552 032760 010000 000010 BIT #NED,RMCS2(R0) ; IS DRIVE PRESENT ?
88 006560 001051 BNE 3$ ; BR IF NO
89 006562 032760 004000 000000 BIT #DVA,RMCS1(R0) ; IS DRIVE AVAILABLE ?
90 006570 001450 BEQ 4$ ; BR IF NO
91 006572 016002 000026 MOV RMDT(R0),R2 ; SAVE DRIVE TYPE REGISTER IN R2
92 006576 012737 071446 006776 MOV #RM02,10$ ; ASSUME RM02 DEVICE
93 006604 022702 020025 CMP #20025,R2 ; SINGLE PORT RM02 ?
94 006610 001430 BEQ 2$ ; BR IF YES

```

95	006612	022702	024025		CMP	#24025,R2	:DUAL PORT RM02 ?
96	006616	001425			BEQ	2\$	:BR IF YES
97	006620	012737	071453	006776	MOV	#\$RM03,10\$	:ASSUME RM03 DEVICE
98	006626	022702	020024		CMP	#20024,R2	:SINGLE PORT RM03 ?
99	006632	001417			BEQ	2\$	:BR IF YES
100	006634	022702	024024		CMP	#24024,R2	:DUAL PORT RM03 ?
101	006640	001414			BEQ	2\$	:BR IF YES
102	006642	012737	071460	006776	MOV	#\$RM05,10\$	:ASSUME RM05 DEVICE
103	006650	022702	020027		CMP	#20027,R2	:SINGLE PORT RM05 ?
104	006654	001406			BEQ	2\$	:BR IF YES
105	006656	022702	024027		CMP	#24027,R2	:DUAL PORT RM05 ?
106	006662	001403			BEQ	2\$	:BR IF YES
107	006664	104401	071465		TYPE	,NOTRM	:DRIVE NOT AN RM05/3/2
108	006670	000412			BR	5\$	:CHECK NEXT DRIVE
109	006672	032760	010000	000012	2\$: BIT	#MOL,RMDS(R0)	:IS MEDIUM ON LINE ?
110	006700	001412			BEQ	6\$	:BR IF NO
111	006702	000417			BR	7\$	
112							
113	006704	104401	071523		3\$: TYPE	,NOTPRS	:DRIVE NOT PRESENT
114	006710	000402			BR	5\$	:CHECK NEXT DRIVE
115							
116	006712	104401	071540		4\$: TYPE	,NOTAVL	:DRIVE NOT AVAILABLE
117	006716	146137	071746	001300	5\$: BICB	ATNTBL(R1),SDEV	:CLEAR DEVICE FROM BIT MAP
118	006724	000425			BR	11\$	:CHECK NEXT DRIVE
119							
120	006726	104401	071557		6\$: TYPE	,UNTOFF	:DRIVE OFFLINE
121	006732	146137	071746	001300	BICB	ATNTBL(R1),SDEV	:CLEAR DEVICE FROM BIT MAP
122	006740	000413			BR	9\$	:PRINT DRIVE TYPE
123							
124	006742	005737	001332		7\$: TST	XXDP	:LOADED FROM RM05/3/2 ?
125	006746	001406			BEQ	8\$	:NO
126	006750	123701	001332		CMPB	XXDP,R1	:IS THIS THE DRIVE ?
127	006754	001360			BNE	5\$	:BR IF NO
128	006756	104401	071506		TYPE	,LODEV	:DRIVE IS LOAD DEVICE
129	006762	000755			BR	5\$	
130							
131	006764	104401	071570		8\$: TYPE	,UNTON	:DRIVE ONLINE
132	006770	104401	071642		9\$: TYPE	,BLNKS2	:TYPE 2 BLANKS
133	006774	104401			TYPE		:PRINT DRIVE TYPE
134	006776	000000			10\$: .WORD	0	:MESSAGE ADDRESS HERE
135							
136	007000	005201			11\$: INC	R1	:INCREMENT THE DRIVE ADDRESS
137	007002	020127	000007		CMP	R1,#7	:ALL DRIVES ARE CHECKED ?
138	007006	003637			BLE	1\$	:BRANCH IF NOT
139							
140	007010	104401	001217		12\$: TYPE	,\$CRLF	:CR-LF
141	007014	000137	007476		JMP	CMNSTART	:JUMP TO COMMON START

```

1      .SBTTL  STANDALONE INPUT ROUTINES
2
3      STANDALONE:
4      007020 004737 066310      JSR      PC,$TKINT      ;INITIALIZE CONSOLE
5
6      007024 005227 177777      INC      #-1          ;FIRST TIME THRU HERE ?
7      007030 001023              BNE      2$          ;BR IF NO
8
9      ;SEE IF OPERATOR WANTS HELP TEXT
10
11     007032 104401 070530      TYPE     ,MSHELP      ;WANT HELP ?
12     007036 104411              RDCHR     ;GET RESPONSE
13     007040 012637 001176      MOV      (SP)+,$TMP1    ;SAVE AND ECHO RESPONSE
14     007044 123727 001176 000131  CMPB    $TMP1,#'Y    ;WAS IT A YES RESPONSE ?
15     007052 001005              BNE      1$          ;NO
16     007054 104401 001176      TYPE     , $TMP1      ;TYPE 'Y'
17     007060 104401 106436      TYPE     ,HELP      ;YES - TYPE HELP TEXT
18     007064 000414              BR        3$
19     007066 104401 071634      1$:      TYPE     ,N          ;TYPE 'N'
20     007072 104401 001217      TYPE     , $CRLF      ;CR-LF
21     007076 000407              BR        3$
22
23     ;SEE IF USER WANTS TO CHANGE UNIBUS ADDRESS
24     007100              2$:
25     007100 005737 001330      TST      CHGADR      ;CHANGE RH/RM BUS ADDRESS ?
26     007104 001457              BEQ      7$          ;BR IF NO
27     007106 005037 001330      CLR      CHGADR      ;NO CHANGE NEXT TIME
28     007112 104401 001217      TYPE     , $CRLF      ;CR-LF
29
30     ;DIALOGUE TO CHANGE THE UNIBUS ADDRESS, VECTOR ADDRESS AND INTERRUPT PRIORITY
31     007116              3$:
32     007116 104401 071105      TYPE     ,CNSL01      ;TYPE CURRENT BUS ADDRESS
33     007122 013746 001276      MOV      $BASE,-(SP)    ;SAVE $BASE FOR TYPEOUT
34     007126 104402              TYPOC     ;GO TYPE--OCTAL ASCII(ALL DIGITS)
35     007130 104401 071642      TYPE     ,BLNKS2      ;TYPE 2 BLANKS
36     007134 104413              RDOCT     ;GET NEW BUS ADDRESS
37     007136 012637 001176      MOV      (SP)+,$TMP1    ;CARRIAGE RETURN ?
38     007142 001412              BEQ      5$          ;YES-SKIP TO NEXT ENTRY
39     007144 022737 160000 001176  CMP      #160000,$TMP1 ;BASE ADDRESS IN I/O PAGE ?
40     007152 101403              BLOS     4$          ;YES
41     007154 104401 071115      TYPE     ,CNSL02      ;TYPE WARNING MESSAGE
42     007160 000756              BR        3$          ;TRY AGAIN
43     007162 013737 001176 001276 4$:      MOV      $TMP1,$BASE ;STORE NEW BUS ADDRESS
44
45     007170 104401 071157      5$:      TYPE     ,CNSL03
46     007174 005046              CLR      -(SP)
47     007176 113716 001272      MOV      $VECT1,(SP) ;GET CURRENT VECTOR ADDRESS
48     007202 104402              TYPOC
49     007204 104401 071642      TYPE     ,BLNKS2      ;TYPE 2 BLANKS
50     007210 104413              RDOCT     ;GET NEW VECTOR ADDRESS
51     007212 012637 001176      MOV      (SP)+,$TMP1    ;CARRIAGE RETURN?
52     007216 001412              BEQ      7$          ;YES-SKIP TO NEXT ENTRY
53     007220 022737 0010C0 001176  CMP      #1000,$TMP1 ;VECTOR ADDRESS < 1000 ?
54     007226 101003              BHI     6$          ;YES!!
55     007230 104401 071166      TYPE     ,CNSL04      ;TYPE WARNING MESSAGE
56     007234 000755              BR        5$          ;RETRY
57     007236 113737 001176 001272 6$:      MOV      $TMP1,$VECT1 ;STORE NEW VECTOR ADDRESS

```

```

57
58      ;DIALOGUE TO INPUT DEVICE NUMBERS
59 007244 005227 177777      7$: INC      #-1      ;FIRST TIME THRU ?
60 007250 001002      BNE      8$      ;BR IF NO
61 007252 104401 071222      TYPE     ,CNSLO7      ;TYPE INPUT INSTRUCTIONS
62 007256 104401 001217      8$: TYPE     , $CRLF      ;CR-LF
63 007262 005037 001300      9$: CLR      $DEVM      ;CLEAR DEVICE MAP
64 007266 104401 071406      TYPE     ,MSDRVS      ;TYPE 'DRIVE(S): '
65 007272 104411      RDCHR
66 007274 012637 001176      MOV      (S)+,$TMP1      ;GET RESPONSE
67 007300 023727 001176 000101      CMP      $TMP1,#'A      ;IS INPUT 'A' ?
68 007306 001007      BNE      10$      ;NO
69 007310 104401 070514      TYPE     ,ALL      ;YES, TYPE 'ALL' AND GO
70 007314 012737 000377 001300      MOV      #377,$DEVM      ;SET DEVICE MAP FOR ALL DRIVES
71 007322 000137 006464      JMP      XSIZ      ;AUTO SIZE.
72
73 007326 023727 001176 000015 10$: CMP      $TMP1,#CR      ;CARRIAGE RETURN ?
74 007334 001436      BEQ      12$      ;YES
75 007336 104401 001176      TYPE     , $TMP1      ;ECHO RESPONSE
76 007342 023727 001176 000060      CMP      $TMP1,#'0      ;NUMBER < 0 ?
77 007350 002430      BLT      12$      ;YES
78 007352 023727 001176 000067      CMP      $TMP1,#'7      ;NUMBER > 7 ?
79 007360 003427      BLE      13$      ;NO
80 007362 000423      BR      12$      ;ILLEGAL INPUT
81
82 007364 104411      11$: RDCHR
83 007366 012637 001176      MOV      (SP)+,$TMP1      ;GET RESPONSE
84 007372 023727 001176 000015      CMP      $TMP1,#CR      ;CARRIAGE RETURN ?
85 007400 001432      BEQ      14$      ;YES
86 007402 104401 070525      TYPE     ,COMMA      ;TYPE ','
87 007406 104401 001176      TYPE     , $TMP1      ;ECHO RESPONSE
88 007412 023727 001176 000060      CMP      $TMP1,#'0      ;NUMBER < 0 ?
89 007420 002404      BLT      12$      ;YES
90 007422 023727 001176 000067      CMP      $TMP1,#'7      ;NUMBER > 7 ?
91 007430 003403      BLE      13$      ;NO
92 007432 104401 071364      12$: TYPE     ,CNSLO8      ;TYPE CR-LF '' ?ILLEGAL INPUT''
93 007436 000711      BR      9$      ;RETRY
94
95 007440 013701 001176      13$: MOV      $TMP1,R1      ;R1 = DRIVE NUMBER
96 007444 042701 177770      BIC      #^C7,R1
97 007450 156137 071746 001300      BISB     ATNTBL(R1),$DEVM      ;SET DEVICE IN MAP
98 007456 122737 000377 001300      CMPB     #377,$DEVM      ;DONE ?
99 007464 101337      BHI      11$      ;NO
100 007466 104401 001217      14$: TYPE     , $CRLF      ;CR-LF
101 007472 000137 006464      JMP      XSIZ      ;GO SIZE DEVICES

```



```

1
2 007476
3 007476 104401 071600
4 007502 013700 001300
5 007506 001004
6 007510 104401 070525
7 007514 104401 071627
8 007520 012701 001470
9 007524 010137 001466
10 007530 012702 000001
11 007534 005003
12 007536 030200
13 007540 001413
14 007542 104401 070525
15 007546 010311
16 007550 010346
    007552 104403
    007554 001
    007555 000
17 007556 116361 071746 000001
18 007564 062701 000002
19 007570 006302
20 007572 105702
21 007574 001402
22 007576 005203
23 007600 000756
24 007602 005011
25 007604 104401 001217
26
27
28 007610 004737 043550
29 007614 000425
30 007616 104401 007624
    007622 000413
    007652
31 007652 005737 000042
32 007656 001002
33 007660 000137 005432
34 007664 000137 037232
35 007670
36
37 007670 000240
38 007672 105737 001300
39 007676 001007
40 007700 005737 000042
41 007704 001002
42 007706 000137 005432
43 007712 000137 037232
44
45 007716 105037 001116
46 007722 005037 001206
47 007726 005037 001326
48 007732 004737 066310
49 007736 012746 000240
    007742 012746 007750
    007746 000002

;ASSEMBLE TEST QUE FROM DEVICE MAP
CMNSTART:
    TYPE ,DRIVES ;TYPE 'DRIVE(S) TO BE TESTED'
    MOV $DEVN,R0 ;R0 = DEVICE MAP
    BNE 1$ ;BR IF DRIVES TO TEST ;
    TYPE ,COMMA ;TYPE ','
    TYPE ,NONE ;TYPE 'NONE'
1$: MOV #TSTQUE+2,R1 ;R1 = ADDRESS OF FIRST ENTRY IN QUE
    MOV R1,TSTQUE ;INITIALIZE ENTRY POINTER
    MOV #1,R2 ;R2 = DEVICE POINTER
    CLR R3 ;R3 = DEVICE NUMBER
2$: BIT R2,R0 ;IS THIS DEVICE IN MAP ?
    BEQ 3$ ;NO
    TYPE ,COMMA ;TYPE ','
    MOV R3,(R1) ;YES - ENTER DEVICE NUMBER IN QUE
    MOV R3,-(SP) ;SAVE R3 FOR TYPEOUT
    TYPOS ;GO TYPE--OCTAL ASCII
    .BYTE 1 ;TYPE 1 DIGIT(S)
    .BYTE 0 ;SUPPRESS LEADING ZEROS
    MOVBL ATNTBL(R3),1(R1) ;ENTER ATTENTION BIT IN QUE
3$: ADD #2,R1 ;ADVANCE ENTRY POINTER
    ASL R2 ;ADVANCE DEVICE POINTER
    TSTB R2 ;DONE ALL DEVICES ?
    BEQ 4$ ;YES
    INC R3 ;ADVANCE DEVICE NUMBER
    BR 2$ ;ENTER NEXT DEVICE
4$: CLR (R1) ;TERMINATE TEST QUE
    TYPE ,$CRLF ;TYPE CR-LF

;SIZE FOR CLOCK
    JSR PC,$IZCLK ;SEE IF CLOCK PRESENT
    BR 6$ ;YES - CLOCK IS PRESENT
    TYPE ,65$ ;TYPE ASCII STRING
    BR 64$ ;GET OVER THE ASCII
    .:65$: .ASCIIZ <CRLF>/NO 'L' OR 'P' CLOCK/
64$: TST @#42 ;ANY MONITOR PRESENT ?
    BNE 5$ ;BR IF YES
    JMP START ;JUMP TO START
5$: JMP $GET42 ;RETURN CONTROL TO MONITOR
6$:

READY: NOP ;READY TO START TEST
    TSTB $DEVN ;ANY DRIVES IN MAP ?
    BNE 2$ ;BR IF YES
    TST @#42 ;ANY MONITOR PRESENT ?
    BNE 1$ ;BR IF YES
    JMP START ;JUMP TO START
1$: JMP $GET42 ;RETURN CONTROL TO MONITOR
2$: CLRB $TSTNM ;RESET TEST NUMBER
    CLR $TIMES ;INITIALIZE NUMBER OF ITERATIONS
    CLR CTLFG ;CLEAR CONTROL-C FLAG
    JSR PC,$TKINT ;INITIALIZE TTY
    MOV #PR5,-(SP) ;PUT NEW PS ON STACK
    MOV #64$,-(SP) ;PUT NEW PC ON STACK
    RTI ;POP NEW PC AND PS

```

```
007750
50 007750 117737 171512 001234 64$: MOVB @TSTQUE,$UNIT ;LOAD DRIVE NUMBER
51 007756 005037 001512 CLR MEDENB ;CLEAR MEDIA ENABLE
52
53 ;CLEAR MASSBUS CONTROLLER, SELECT DRIVE AND DETERMINE THE LAST TRACK
54 ;OF THE DIFFERENT DRIVE TYPES
55
56 007762 012737 002000 001334 MOV #TA4,LSTRK ;ASSUME LAST TRACK FOR RM02/3 = 4.
57 007770 013700 001276 MOV $BASE,R0 ;R0 = UNIBUS ADDRESS
58 007774 012760 000040 000010 MOV #CLR,RMCS2(R0) ;CLEAR MASSBUS
59 010002 117760 171460 000010 MOVB @TSTQUE,RMCS2(R0) ;SELECT DEVICE UNDER TEST
60 010010 016002 000026 MOV RMDT(R0),R2 ;GET RMDT AND
61 010014 042702 177770 BIC #177770,R2 ;SAVE DRIVE TYPE BITS
62 010020 022702 000007 CMP #7,R2 ;IS IT AN RM05 ?
63 010024 001003 BNE 3$ ;NO, MUST BE AN RM02 OR RM03
64 010026 012737 011000 001334 MOV #TA16.TA2,LSTRK ;YES--SET LAST TRACK - 18.
65
66 ;TYPE DRIVE NUMBER TO BE TESTED($UNIT)
67
68 010034 104401 001217 3$: TYPE , $CRLF ;CR-LF
69 010040 104401 071422 TYPE ,MSGDRV ;TYPE 'DRIVE'
70 010044 013746 001234 MOV $UNIT,-(SP) ;SAVE $UNIT FOR TYPEOUT
;TYPE DRIVE NUMBER
;GO TYPE--OCTAL ASCII
;TYPE 2 DIGIT(S)
;SUPPRESS LEADING ZEROS
;THESE TWO LOOPS ARE ADDED TO
;WAIT FOR TTY
010050 104403 TYPOS
010052 002 .BYTE 2
010053 000 .BYTE 0
71 010054 005004 CLR R4
72 010056 005304 DEC R4
73 010060 001376 BNE .-2
74 010062 005304 DEC R4
75 010064 001376 BNE .-2
```


1
2

*TEST 1 CONTROLLER ACCESS TEST

TST1:

010066	000004		SCOPE		:SCOPE CALL
010066	000240		NOP		:START OF TEST
010070	000240		MOV	#STACK,SP	:INITIALIZE STACK POINTER
010072	012706	001100	MOV	\$BASE,R0	:R0 = UNIBUS ADDRESS
010076	013700	001276	MOV	TSTQUE,R1	:(R1) = DEVICE BEING TESTED
010102	013701	001466	MOV	#1,\$TESTN	::SET TEST NUMBER IN APT MAIL BOX
010106	012737	000001 001226	MOV		
3					
4	010114	005001	CLR	R1	
5	010116	013746 0000C4	MOV	ERRVEC,-(SP)	::PUSH ERRVEC ON STACK
	010122	013746 000006	MOV	ERRVEC+2,-(SP)	::PUSH ERRVEC+2 ON STACK
6	010126	012737 010230 000004	MOV	#1\$,ERRVEC	
7	010134	012737 000300 000006	MOV	#PR6,ERRVEC+2	
8					
9	010142	110160 000001	MOVB	R1,RMCS1+1(R0)	:MOVE HI BYTE TO RMCS1
10	010146	010160 000002	MOV	R1,RMWC(R0)	:MOVE WORD COUNT REGISTER
11	010152	016002 000002	MOV	RMWC(R0),R2	
12	010156	010160 000004	MOV	R1,RMBA(R0)	:MOVE BUS ADDRESS REGISTER
13	010162	016002 000004	MOV	RMBA(R0),R2	
14	010166	016046 000010	MOV	RMCS2(R0),-(SP)	::PUSH RMCS2(R0) ON STACK
15	010172	010160 000010	MOV	R1,RMCS2(R0)	:MOVE CONTROL STATUS REGISTER
16	010176	016002 000010	MOV	RMCS2(R0),R2	
17	010202	012660 000010	MOV	(SP)+,RMCS2(R0)	::POP STACK INTO RMCS2(R0)
18	010206	010160 000022	MOV	R1,RMDB(R0)	:MOVE DATA BUFFER
19	010212	016002 000022	MOV	RMDB(R0),R2	
20	010216	012637 000006	MOV	(SP)+,ERRVEC+2	::POP STACK INTO ERRVEC+2
	010222	012637 000004	MOV	(SP)+,ERRVEC	::POP STACK INTO ERRVEC
21	010226	000417	BR	3\$	:NO BUS TIMEOUT OCCURRED
22					
23	010230	022626	1\$: CMP	(SP)+,(SP)+	:ADJUST STACK
24	010232	012637 000006	MOV	(SP)+,ERRVEC+2	::POP STACK INTO ERRVEC+2
	010236	012637 000004	MOV	(SP)+,ERRVEC	::POP STACK INTO ERRVEC
25	010242	104110	EMT	110	
26	010244	005737 000042	TST	@#42	:STAND ALONE MODE ?
27	010250	001002	BNE	2\$	:NO!!
28	010252	000137 005422	JMP	START1	:YES-GO GET \$BASE
29	010256	005037 001300	2\$: CLR	\$DEVM	:FUDGE NO DRIVES IN MAP
30	010262	000137 037044	JMP	\$EOP	:RETURN TO \$EOP
31	010266		3\$:		
32					
33					

*TEST 2 WRITE, READ ZEROS

TST2:

010266	000004		SCOPE		:SCOPE CALL
010266	000240		NOP		:START OF TEST
010270	000240		MOV	#STACK,SP	:INITIALIZE STACK POINTER
010272	012706	001100	MOV	\$BASE,R0	:R0 = UNIBUS ADDRESS
010276	013700	001276	MOV	TSTQUE,R1	:(R1) = DEVICE BEING TESTED
010302	013701	001466	MOV	#2,\$TESTN	::SET TEST NUMBER IN APT MAIL BOX
010306	012737	000002 001226	MOV		
34					
35					
36					

:LOOP #1 FORMAT,WRITE,READ

```

37
38 010314
39
40
41 010314 004737 037316
    010320 154130
;PREPARE DEVICE FOR TEST
    JSR PC,TSTPRP
    .WORD 154130
;PREPARE DEVICE FOR TEST
;TASK DESCRIPTOR AS FOLLOWS:
;SELECT DEVICE & VERIFY DEVICE AVAILABLE
;CLEAR CONTROLLER & SELECT DEVICE
;VERIFY CONTROLLER CLEAR OPERATION
;PACK ACKNOWLEDGE IF VOLUME NOT VALID
;VERIFY PACK ACKNOWLEDGE
;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
;VERIFY RECALIBRATION
;GO TO 2$ IF NO ERROR
;RETURN HERE IF ERROR
;ERROR # DEFINED BY TSTPRP SUBROUTINE
;GO TO 25$ IF ERROR

    010322 000404
    010324 000240
    010326 104000
    010330 000137 011274
    BR 2$
    NOP
    EMT
    JMP 25$

42
43
44 010334
45 010334 012737 000000 001446
46 010342 012737 000000 001420
47 010350 012737 106436 001416
48 010356 012737 177376 001414
49 010364 012737 010000 001444
50 010372 012737 000063 001412
    MOV #0,RMDCO
    MOV #0,RMDAO
    MOV #BUFONE,RMBAD
    MOV #-256,RMWCO
    MOV #FMT16,RMOFO
    MOV #WH!GO,RMCS10
;CYLINDER = 0
;TRACK - 0, SECTOR 0
;BUS ADDRESS
;2 + 256. WORDS (2'S COMP)
;16 BIT FORMAT
;WRITE HEADER AND DATA COMMAND

51
52
    010400 004737 040242
    010404 000405
    010406 104401 070426
    010412 104000
    010414 000137 011274
;VERIFY THAT SECTOR IS NOT BAD
    JSR PC,BADSCT
    BR 3$
    TYPE ,SCTMSG
    EMT
    JMP 25$
;CALL BAD SECTOR MODULE
;GO TO 3$ IF NO ERROR
;TYPE BAD SECTOR MESSAGE
;ERROR # DEFINED BY BADSCT SUBROUTINE
;GO TO 25$ IF ERROR

53 010420
54 010420 012737 072060 001174
55 010426 012737 000001 001176
56 010434 004737 042174
    MOV #ZEROS,$TMP0
    MOV #1,$TMP1
    JSR PC,GENBUF
;STARTING ADDRESS OF PATTERN
;RANGE OF PATTERN
;GO GENERATE BUFFER FOR FORMAT

57
58
;SETUP PARAMETERS AND EXECUTE COMMAND
    MOV #PUTINX,R2
    MOV #RMDC,(R2)+
    MOV #RMDA,(R2)+
    MOV #RMBAD,(R2)+
    MOV #RMWC,(R2)+
    MOV #RMOF,(R2)+
    MOV #RMCS1,(R2)+
    MOV #200,(R2)
;R2 BYTE ENTRY POSITION
;TERMINATE TABLE

59 010440 012702 001553
60 010444 112722 000034
61 010450 112722 000006
62 010454 112722 000004
63 010460 112722 000002
64 010464 112722 000032
65 010470 112722 000000
66 010474 112712 000200
67 010500
68 010500 004737 043330
    010504 000404
    010506 000240
    010510 104000
    010512 000137 011274
    JSR PC,PUT
    BR 5$
    NOP
    EMT
    JMP 25$
;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
;GO TO 5$ IF NO ERROR
;RETURN HERE IF ERROR
;ERROR # DEFINED BY PUT SUBROUTINE
;GO TO 25$ IF ERROR

69 010516
70
71
    010516 004737 042774
;SETUP GET INDEX TABLE TO READ ALL REGISTERS
    JSR PC,GETSTS
;GO TO GETSTS SUBROUTINE
  
```

```

72
73 ;WAIT FOR COMMAND TO COMPLETE
010522 004737 043672 JSR PC,TIMOUT ;GO TO TIMOUT SUBROUTINE
74
75 ;GO GET REGISTER STATUS
76 010526 004737 043060 JSR PC,GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
010532 000404 BR 6$ ;GO TO 6$ IF NO ERROR
010534 000240 NOP ;RETURN HERE IF ERROR
010536 104000 EMT ;ERROR # DEFINED BY GET SUBROUTINE
010540 000137 011274 JMP 25$ ;GO TO 25$ IF ERROR
77 010544 6$:
78
79 ;VERIFY RESULTS OF WRITE COMMAND
80 010544 004737 056572 JSR PC,DASTS ;GO VERIFY RESULTS OF DATA TRANSFER
010550 000405 BR 7$ ;GO TO 7$ IF NO ERROR
010552 000240 NOP ;RETURN HERE IF ERROR
010554 104000 EMT ;ERROR # DEFINED BY DASTS SUBROUTINE
010556 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
010560 000137 011274 JMP 25$ ;GO TO 25$ IF ERROR
81 010564 7$:
82
83 ;MOVE LOOP ADDRESSES TO NEXT OPERATION
84 010564 012737 010574 001124 MOV #8$, $LPERR
85 010572 000410 BR 9$ ;SKIP TO WRITE OPERATION
86
87 ;*****
88 ;LOOP #2 WRITE, READ
89
90 010574 8$:
91
92 ;PREPARE DEVICE FOR TEST
93 010574 004737 037316 JSR PC,TSTPRP ;PREPARE DEVICE FOR TEST
010600 154130 .WORD 154130 ;TASK DESCRIPTOR AS FOLLOWS:
;SELECT DEVICE & VERIFY DEVICE AVAILABLE
;CLEAR CONTROLLER & SELECT DEVICE
;VERIFY CONTROLLER CLEAR OPERATION
;PACK ACKNOWLEDGE IF VOLUME NOT VALID
;VERIFY PACK ACKNOWLEDGE
;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
;VERIFY RECALIBRATION
;GO TO 9$ IF NO ERROR
;RETURN HERE IF ERROR
;ERROR # DEFINED BY TSTPRP SUBROUTINE
;GO TO 25$ IF ERROR
010602 000404 BR 9$
010604 000240 NOP
010606 104000 EMT
010610 000137 011274 JMP 25$
94 010614 9$:
95
96 ;SETUP PARAMETERS AND EXECUTE COMMAND
97 010614 10$:
98 010614 012737 106442 001416 MOV #BUFONE+4, RMBAD ;MOVE MEMORY ADDRESS
99 010622 012737 177400 001414 MOV #-256, RMWCO ;CHANGE WORD COUNT
100 010630 012737 000061 001412 MOV #WD!GO, RMCS10 ;WRITE DATA COMMAND
101 010636 012702 001553 MOV #PUTINX, R2 ;LOAD PUT REGISTER INDEX TABLE
102 010642 112722 000006 MOV #RMDA, (R2)+
103 010646 112722 000034 MOV #RMDC, (R2)+
104 010652 112722 000032 MOV #RMOF, (R2)+
105 010656 112722 000004 MOV #RMSA, (R2)+
106 010662 112722 000002 MOV #RMWC, (R2)+

```

```
107 010666 112722 000000      MOVB    #RMCS1,(R2)+
108 010672 112722 000200      MOVB    #200,(R2)+          ;TERMINATE TABLE
109
110 010676 004737 043330      JSR      PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
    010702 000404              BR      11$          ;GO TO 11$ IF NO ERROR
    010704 000240              NOP              ;RETURN HERE IF ERROR
    010706 104000              EMT              ;ERROR # DEFINED BY PUT SUBROUTINE
    010710 000137 011274      JMP      25$          ;GO TO 25$ IF ERROR
111 010714
112
113      ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
    010714 004737 042774      JSR      PC,GETSTS      ;GO TO GETSTS SUBROUTINE
114
115      ;WAIT FOR COMMAND TO COMPLETE
    010720 004737 043672      JSR      PC,TIMOUT      ;GO TO TIMOUT SUBROUTINE
116
117      ;GO GET REGISTER STATUS
118 010724 004737 043060      JSR      PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
    010730 000404              BR      12$          ;GO TO 12$ IF NO ERROR
    010732 000240              NOP              ;RETURN HERE IF ERROR
    010734 104000              EMT              ;ERROR # DEFINED BY GET SUBROUTINE
    010736 000137 011274      JMP      25$          ;GO TO 25$ IF ERROR
119 010742
120
121      ;VERIFY RESULTS OF WRITE COMMAND
122 010742 004737 044056      JSR      PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
    010746 000405              BR      13$          ;GO TO 13$ IF NO ERROR
    010750 000240              NOP              ;RETURN HERE IF ERROR
    010752 104000              EMT              ;ERROR # DEFINED BY PRIERR SUBROUTINE
    010754 004736              JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
    010756 000137 011274      JMP      25$          ;GO TO 25$ IF ERROR
123 010762
124 010762 004737 056572      JSR      PC,DASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
    010766 000405              BR      14$          ;GO TO 14$ IF NO ERROR
    010770 000240              NOP              ;RETURN HERE IF ERROR
    010772 104000              EMT              ;ERROR # DEFINED BY DASTS SUBROUTINE
    010774 004736              JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
    010776 000137 011274      JMP      25$          ;GO TO 25$ IF ERROR
125 011002
126 011002 004737 044710      JSR      PC,SECERR      ;GO CHECK FOR SECONDARY ERRORS
    011006 000405              BR      15$          ;GO TO 15$ IF NO ERROR
    011010 000240              NOP              ;RETURN HERE IF ERROR
    011012 104000              EMT              ;ERROR # DEFINED BY SECERR SUBROUTINE
    011014 004736              JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
    011016 000137 011274      JMP      25$          ;GO TO 25$ IF ERROR
127 011022
128
129      ;CHANGE LOOP ADDRESSES
130 011022 012737 011032 001124  MOV      #16$,$LPERR
131 011030 000410              BR      17$          ;SKIP TO NEXT OPERATION
132
133      ;*****
134      ;LOOP #3          READ
135
136 011032
137
138      ;PREPARE DEVICE FOR TEST
```

```

139 011032 004737 037316      JSR    PC,TSTPRP      ;PREPARE DEVICE FOR TEST
      011036 154130              .WORD    154130      ;TASK DESCRIPTOR AS FOLLOWS:
                                          ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
                                          ;CLEAR CONTROLLER & SELECT DEVICE
                                          ;VERIFY CONTROLLER CLEAR OPERATION
                                          ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                          ;VERIFY PACK ACKNOWLEDGE
                                          ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
                                          ;VERIFY PECALIBRATION
      011040 000404              BR      17$          ;GO TO 17$ IF NO ERROR
      011042 000240              NOP                     ;RETURN HERE IF ERROR
      011044 104000              EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
      011046 000137 011274      JMP      25$          ;GO TO 25$ IF ERROR
140 011052                      17$:
141
142                      ;SETUP PARAMETERS AND EXECUTE COMMAND
143 011052                      18$:
144 011052 012737 107446 001416  MOV     #BUFTWO+4,RMBA0      ;CHANGE MEMORY ADDRESS
145 011060 012737 000071 001412  MOV     #RD!GO,RMCS10      ;READ DATA COMMAND
146 011066 012702 001553          MOV     #PUTINX,R2          ;LOAD PUT REGISTER INDEX TABLE
147 011072 112722 000006          MOVVB   #RMDA,(R2)+
148 011076 112722 000032          MOVVB   #RMOF,(R2)+
149 011102 112722 000034          MOVVB   #RMDC,(R2)+
150 011106 112722 000004          MOVVB   #RMBA,(R2)+
151 011112 112722 000002          MOVVB   #RMWC,(R2)+
152 011116 112722 000000          MOVVB   #RMCS1,(R2)+
153 011122 112712 000200          MOVVB   #200,(R2)
154
155 011126 004737 043330      JSR      PC,PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      011132 000404              BR      19$          ;GO TO 19$ IF NO ERROR
      011134 000240              NOP                     ;RETURN HERE IF ERROR
      011136 104000              EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
      011140 000137 011274      JMP      25$          ;GO TO 25$ IF ERROR
156 011144                      19$:
157
158                      ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
      011144 004737 042774      JSR      PC,GETSTS      ;GO TO GETSTS SUBROUTINE
159
160                      ;WAIT FOR COMMAND TO COMPLETE
      011150 004737 043672      JSR      PC,TIMOUT      ;GO TO TIMOUT SUBROUTINE
161
162                      ;GO GET REGISTER STATUS
      011154 004737 043060      JSR      PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      011160 000404              BR      20$          ;GO TO 20$ IF NO ERROR
      011162 000240              NOP                     ;RETURN HERE IF ERROR
      011164 104000              EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
      011166 000137 011274      JMP      25$          ;GO TO 25$ IF ERROR
164 011172                      20$:
165
166                      ;VERIFY RESULTS OF READ COMMAND
      011172 004737 044056      JSR      PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
      011176 000405              BR      21$          ;GO TO 21$ IF NO ERROR
      011200 000240              NOP                     ;RETURN HERE IF ERROR
      011202 104000              EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
      011204 004736              JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      011206 000137 011274      JMP      25$          ;GO TO 25$ IF ERROR
168 011212                      21$:

```

```

169 011212 004737 056572      JSR      PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
      011216 000405      BR      22$      ;GO TO 22$ IF NO ERROR
      011220 000240      NOP      ;RETURN HERE IF ERROR
      011222 104000      EMT      ;ERROR # DEFINED BY DTASTS SUBROUTINE
      011224 004736      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      011226 000137 011274      JMP      25$      ;GO TO 25$ IF ERROR
170 011232      22$:
171 011232 004737 044710      JSR      PC,SECERR      ;GO CHECK FOR SECONDARY ERRORS
      011236 000405      BR      23$      ;GO TO 23$ IF NO ERROR
      011240 000240      NOP      ;RETURN HERE IF ERROR
      011242 104000      EMT      ;ERROR # DEFINED BY SECERR SUBROUTINE
      011244 004736      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      011246 000137 011274      JMP      25$      ;GO TO 25$ IF ERROR
172 011252      23$:
173
174      ;GO VERIFY DATA
175 011252 004737 042432      JSR      PC,CMPBUF      ;GO COMPARE WRITE, READ DATA BUFFERS
      011256 106442      .WORD      BUFSIZE+4      ;STARTING ADDRESS OF WRITE BUFFER
      011260 107446      .WORD      BUFSIZE+4      ;STARTING ADDRESS OF READ BUFFER
      011262 000404      BR      24$      ;GO TO 24$ IF NO ERROR
      011264 000240      NOP      ;RETURN HERE IF ERROR
      011266 104000      EMT      ;ERROR # DEFINED BY CMPBUF SUBROUTINE
      011270 000137 011274      JMP      25$      ;GO TO 25$ IF ERROR
176 011274      24$:
177
178 011274      25$:
179
180      ;*****
      ;*TEST 3      WRITE, WRITE CHECK ZEROS
      ;*****
      TST3:
      011274      SCOPE      ;SCOPE CALL
      011276 000004      NOP      ;START OF TEST
      011276 000240      MOV      #STACK,SP      ;INITIALIZE STACK POINTER
      011300 012706 001100      MOV      $BASE,R0      ;R0 = UNIBUS ADDRESS
      011304 013700 001276      MOV      TSTQUE,R1      ;(R1) = DEVICE BEING TESTED
      011310 013701 001466      MOV      #3,$TESTN      ;SET TEST NUMBER IN APT MAIL BOX
      011314 012737 000003 001226
181
182      ;*****
183      ;LOOP #1      FORMAT,WRITE,WRITE CHECK DATA
184
185 011322      1$:
186
187      ;PREPARE DEVICE FOR TEST
188 011322 004737 037316      JSR      PC,TSTPRP      ;PREPARE DEVICE FOR TEST
      011326 154130      .WORD      154130      ;TASK DESCRIPTOR AS FOLLOWS:
      ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
      ;CLEAR CONTROLLER & SELECT DEVICE
      ;VERIFY CONTROLLER CLEAR OPERATION
      ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
      ;VERIFY PACK ACKNOWLEDGE
      ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
      ;VERIFY RECALIBRATION
      011330 000404      BR      2$      ;GO TO 2$ IF NO ERROR
      011332 000240      NOP      ;RETURN HERE IF ERROR
      011334 104000      EMT      ;ERROR # DEFINED BY TSTPRP SUBROUTINE
      011336 000137 012252      JMP      24$      ;GO TO 24$ IF ERROR

```



```

189
190 ;LOAD PARAMETERS AND GENERATE DATA BUFFER
191 011342 2$:
192 011342 012737 000000 001446 MOV #0,RMDCO ;CYLINDER = 0
193 011350 012737 000000 001420 MOV #0,RMDAO ;TRACK = 0, SECTOR = 0
194 011356 012737 106436 001416 MOV #BUFONE,RMBAO ;BUS ADDRESS
195 011364 012737 177376 001414 MOV #-258,RMWCO ;2 + 256. WORDS (2'S COMP)
196 011372 012737 010000 001444 MOV #FMT16,RMOFO ;16 BIT FORMAT
197 011400 012737 000063 001412 MOV #WH!GO,RMCS10 ;WRITE HEADER AND DATA COMMAND
198
199 ;VERIFY THAT SECTOR IS NOT BAD
011406 004737 040242 JSR PC,BADSCT ;CALL BAD SECTOR MODULE
011412 000405 BR 3$ ;GO TO 3$ IF NO ERROR
011414 104401 070426 TYPE ,SCTMSG ;TYPE BAD SECTOR MESSAGE
011420 104000 EMT ;ERROR # DEFINED BY BADSCT SUBROUTINE
011422 000137 012252 JMP 24$ ;GO TO 24$ IF ERROR
200 011426 3$:
201 011426 012737 072060 001174 MOV #ZEROS,$TMP0 ;STARTING ADDRESS OF PATTERN
202 011434 012737 000001 001176 MOV #1,$TMP1 ;RANGE OF PATTERN
203 011442 004737 042174 JSR PC,GENBUF ;GO GENERATE BUFFER FOR FORMAT
204
205 ;SETUP PARAMETERS AND EXECUTE COMMAND
206 011446 012702 001553 MOV #PUTINX,R2 ;R2 = BYTE ENTRY POSITION
207 011452 112722 000034 MOVB #RMDC,(R2)+
208 011456 112722 000006 MOVB #RMDA,(R2)+
209 011462 112722 000004 MOVB #RMBA,(R2)+
210 011466 112722 000002 MOVB #RMWC,(R2)+
211 011472 112722 000032 MOVB #RMOF,(R2)+
212 011476 112722 000000 MOVB #RMCS1,(R2)+
213 011502 112712 000200 MOVB #200,(R2) ;TERMINATE TABLE
214 011506
215 011506 004737 043330 4$: JSR PC,PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
011512 000404 BR 5$ ;GO TO 5$ IF NO ERROR
011514 000240 NOP ;RETURN HERE IF ERROR
011516 104000 EMT ;ERROR # DEFINED BY PUT SUBROUTINE
011520 000137 012252 JMP 24$ ;GO TO 24$ IF ERROR
216 011524
217
218 ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
011524 004737 042774 JSR PC,GETSTS ;GO TO GETSTS SUBROUTINE
219
220 ;WAIT FOR COMMAND TO COMPLETE
011530 004737 043672 JSR PC,TIMOUT ;GO TO TIMOUT SUBROUTINE
221
222 ;GO GET REGISTER STATUS
223 011534 004737 043060 JSR PC,GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
011540 000404 BR 6$ ;GO TO 6$ IF NO ERROR
011542 000240 NOP ;RETURN HERE IF ERROR
011544 104000 EMT ;ERROR # DEFINED BY GET SUBROUTINE
011546 000137 012252 JMP 24$ ;GO TO 24$ IF ERROR
224 011552
225
226 ;VERIFY RESULTS OF WRITE COMMAND
227 011552 004737 056572 JSR PC,DASTS ;GO VERIFY RESULTS OF DATA TRANSFER
011556 000405 BR 7$ ;GO TO 7$ IF NO ERROR
011560 000240 NOP ;RETURN HERE IF ERROR
011562 104000 EMT ;ERROR # DEFINED BY DASTS SUBROUTINE

```

```

011564 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
011566 000137 012252 JMP 24$ ;GO TO 24$ IF ERROR
228 011572 7$:
229
230 ;MOVE LOOP ADDRESSES TO NEXT OPERATION
231 011572 012737 011602 001124 MOV #8$,$LPERR
232 011600 000410 BR 9$ ;SKIP TO WRITE OPERATION
233
234 ;*****
235 ;LOOP #2 WRITE,WRITE CHECK DATA
236
237 011602 8$:
238
239 ;PREPARE DEVICE FOR TEST
240 011602 004737 037316 JSR PC,TSTPRP ;PREPARE DEVICE FOR TEST
011606 154130 .WORD 154130 ;TASK DESCRIPTOR AS FOLLOWS:
;SELECT DEVICE & VERIFY DEVICE AVAILABLE
;CLEAR CONTROLLER & SELECT DEVICE
;VERIFY CONTROLLER CLEAR OPERATION
;PACK ACKNOWLEDGE IF VOLUME NOT VALID
;VERIFY PACK ACKNOWLEDGE
;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
;VERIFY RECALIBRATION
;GO TO 9$ IF NO ERROR
;RETURN HERE IF ERROR
;ERROR # DEFINED BY TSTPRP SUBROUTINE
;GO TO 24$ IF ERROR

011610 000404 BR 9$
011612 000240 NOP
011614 104000 EMT
011616 000137 012252 JMP 24$
241 011622 9$:
242
243 ;SETUP PARAMETERS AND EXECUTE COMMAND
244 011622 10$:
245 011622 012737 106442 001416 MOV #BUFONE+4,RMBAD ;CHANGE MEMORY ADDRESS
246 011630 012737 177400 001414 MOV #-256.,RMWCO ;CHANGE WORD COUNT
247 011636 012737 000061 001412 MOV #WD!GO,RMCS10 ;WRITE DATA COMMAND
248 011644 012702 001553 MOV #PUTINX,R2 ;LOAD PUT REGISTER INDEX TABLE
249 011650 112722 000006 MOVB #RMDA,(R2)+
250 011654 112722 000034 MOVB #RMDC,(R2)+
251 011660 112722 000032 MOVB #RMOF,(R2)+
252 011664 112722 000004 MOVB #RMBB,(R2)+
253 011670 112722 000002 MOVB #RMWC,(R2)+
254 011674 112722 000000 MOVB #RMCS1,(R2)+
255 011700 112722 000200 MOVB #200,(R2)+ ;TERMINATE TABLE
256
257 011704 004737 043330 JSR PC,PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
011710 000404 BR 11$ ;GO TO 11$ IF NO ERROR
011712 000240 NOP ;RETURN HERE IF ERROR
011714 104000 EMT ;ERROR # DEFINED BY PUT SUBROUTINE
011716 000137 012252 JMP 24$ ;GO TO 24$ IF ERROR
258 011722 11$:
259
260 ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
011722 004737 042774 JSR PC,GETSTS ;GO TO GETSTS SUBROUTINE
261
262 ;WAIT FOR COMMAND TO COMPLETE
011726 004737 043672 JSR PC,TIMOUT ;GO TO TIMOUT SUBROUTINE
263
264 ;GO GET REGISTER STATUS

```



```

265 011732 004737 043060      JSR    PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      011736 000404          BR      12$      ;GO TO 12$ IF NO ERROR
      011740 000240          NOP          ;RETURN HERE IF ERROR
      011742 104000          EMT          ;ERROR # DEFINED BY GET SUBROUTINE
      011744 000137 012252      JMP      24$      ;GO TO 24$ IF ERROR
266 011750
267
268
269 011750 004737 044056      ;VERIFY RESULTS OF WRITE COMMAND
      011754 000405          JSR    PC,PRIERR  ;GO CHECK FOR PRIMARY FRROPS
      011756 000240          BR      13$      ;GO TO 13$ IF NO ERROR
      011760 104000          NOP          ;RETURN HERE IF ERROR
      011762 004736          EMT          ;ERROR # DEFINED BY PRIERR SUBROUTINE
      011764 000137 012252      JSR    PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
      011764 000137 012252      JMP      24$      ;GO TO 24$ IF ERROR
270 011770
271 011770 004737 0565/2      13$:      JSR    PC,DTASTS  ;GO VERIFY RESULTS OF DATA TRANSFER
      011774 000405          BR      14$      ;GO TO 14$ IF NO ERROR
      011776 000240          NOP          ;RETURN HERE IF ERROR
      012000 104000          EMT          ;ERROR # DEFINED BY DTASTS SUBROUTINE
      012002 004736          JSR    PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
      012004 000137 012252      JMP      24$      ;GO TO 24$ IF ERROR
272 012010
273 012010 004737 044710      14$:      JSR    PC,SECERR  ;GO CHECK FOR SECONDARY ERRORS
      012014 000405          BR      15$      ;GO TO 15$ IF NO ERROR
      012016 000240          NOP          ;RETURN HERE IF ERROR
      012020 104000          EMT          ;ERROR # DEFINED BY SECERR SUBROUTINE
      012022 004736          JSR    PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
      012024 000137 012252      JMP      24$      ;GO TO 24$ IF ERROR
274 012030
275
276
277 012030 012737 012040 001124 ;CHANGE LOOP ADDRESSES
      012036 000410          MOV      #16$,$LPERR
      012036 000410          BR      17$      ;SKIP TO NEXT OPERATION
278
279
280
281
282
283 012040
284
285
286 012040 004737 037316      ;PREPARE DEVICE FOR TEST
      012044 154130          JSR    PC,TSTPRP  ;TASK DESCRIPTOR AS FOLLOWS:
      012044 154130          .WORD 154130      ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
      012046 000404          ;CLEAR CONTROLLER & SELECT DEVICE
      012050 000240          ;VERIFY CONTROLLER CLEAR OPERATION
      012052 104000          ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
      012054 000137 012252      ;VERIFY PACK ACKNOWLEDGE
      012054 000137 012252      ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
      012054 000137 012252      ;VERIFY RECALIBRATION
      012054 000137 012252      BR      17$      ;GO TO 17$ IF NO ERROR
      012054 000137 012252      NOP          ;RETURN HERE IF ERROR
      012054 000137 012252      EMT          ;ERROR # DEFINED BY TSTPRP SUBROUTINE
      012054 000137 012252      JMP      24$      ;GO TO 24$ IF ERROR
287 012060
288
289
290 012060      17$:      ;SETUP PARAMETERS AND EXECUTE COMMAND
      012060
  
```

291	012060	012737	000051	001412	MOV	#WCD!GO,RMCS10	;WRITE CHECK DATA DATA COMMAND
292	012066	012702	001553		MOV	#PUTJNX,R2	;LOAD PUT REGISTER INDEX TABLE
293	012072	112722	000006		MOVB	#RMDA,(R2)+	
294	012076	112722	000032		MOVB	#RMOF,(R2)+	
295	012102	112722	000034		MOVB	#RMDC,(R2)+	
296	012106	112722	000004		MOVB	#RMEA,(R2)+	
297	012112	112722	000002		MOVB	#RMWC,(R2)+	
298	012116	112722	000000		MOVB	#RMCS1,(R2)+	
299	012122	112712	000200		MOVB	#200,(R2)	
300							
301	012126	004737	043330		JSR	PC,PUT	;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
	012132	000404			BR	19\$	;GO TO 19\$ IF NO ERROR
	012134	000240			NOP		;RETURN HERE IF ERROR
	012136	104000			EMT		;ERROR # DEFINED BY PUT SUBROUTINE
	012140	000137	012252		JMP	24\$	;GO TO 24\$ IF ERROR
302	012144			19\$:			
303							
304							
	012144	004737	042774		JSR	PC,GETSTS	;GO TO GETSTS SUBROUTINE
305							
306							
	012150	004737	043672		JSR	PC,TIMOUT	;GO TO TIMEOUT SUBROUTINE
307							
308							
	012154	004737	043060		JSR	PC,GET	;GO READ REGISTER(S) WITH GET SUBROUTINE
	012160	000404			BR	20\$	;GO TO 20\$ IF NO ERROR
	012162	000240			NOP		;RETURN HERE IF ERROR
	012164	104000			EMT		;ERROR # DEFINED BY GET SUBROUTINE
	012166	000137	012252		JMP	24\$	;GO TO 24\$ IF ERROR
310	012172			20\$:			
311							
312							
	012172	004737	044056		JSR	PC,PRIERR	;GO CHECK FOR PRIMARY ERRORS
	012176	000405			BR	21\$	;GO TO 21\$ IF NO ERROR
	012200	000240			NOP		;RETURN HERE IF ERROR
	012202	104000			EMT		;ERROR # DEFINED BY PRIERR SUBROUTINE
	012204	004736			JSR	PC,@(SP)+	;GO BACK FOR MORE ERROR CHECKS
	012206	000137	012252		JMP	24\$	;GO TO 24\$ IF ERROR
314	012212			21\$:			
	012212	004737	056572		JSR	PC,DTASTS	;GO VERIFY RESULTS OF DATA TRANSFER
	012216	000405			BR	22\$	;GO TO 22\$ IF NO ERROR
	012220	000240			NOP		;RETURN HERE IF ERROR
	012222	104000			EMT		;ERROR # DEFINED BY DTASTS SUBROUTINE
	012224	004736			JSR	PC,@(SP)+	;GO BACK FOR MORE ERROR CHECKS
	012226	000137	012252		JMP	24\$	;GO TO 24\$ IF ERROR
316	012232			22\$:			
	012232	004737	044710		JSR	PC,SECERR	;GO CHECK FOR SECONDARY ERRORS
	012236	000405			BR	23\$	;GO TO 23\$ IF NO ERROR
	012240	000240			NOP		;RETURN HERE IF ERROR
	012242	104000			EMT		;ERROR # DEFINED BY SECERP SUBROUTINE
	012244	004736			JSR	PC,@(SP)+	;GO BACK FOR MORE ERROR CHECKS
	012246	000137	012252		JMP	24\$	;GO TO 24\$ IF ERROR
318	012252			23\$:			
319							
320	012252			24\$:			
321							
322							

;*****

```

; *TEST 4 WRITE, WRITE CHECK ZEROS W/ WCE ERROR
;*****
ST4:
012252 000004 SCOPE ;SCOPE CALL
012252 000240 NOP ;START OF TEST
012254 000240 MOV #STACK,SP ;INITIALIZE STACK POINTER
012256 012706 001100 MOV $BASE,R0 ;R0 = UNIBUS ADDRESS
012262 013700 001276 MOV TSTQUE,R1 ;(R1) = DEVICE BEING TESTED
012266 013701 001466 MOV #4,$TESTN ;SET TEST NUMBER IN APT MAIL BOX
012272 012737 000004 001226

323
324
325
326
327 012300
328
329
330 012300 004737 037316 ;PREPARE DEVICE FOR TEST
012304 154130 JSR PC,TSTPRP ;PREPARE DEVICE FOR TEST
;TASK DESCRIPTOR AS FOLLOWS:
;SELECT DEVICE & VERIFY DEVICE AVAILABLE
;CLEAR CONTROLLER & SELECT DEVICE
;VERIFY CONTROLLER CLEAR OPERATION
;PACK ACKNOWLEDGE IF VOLUME NOT VALID
;VERIFY PACK ACKNOWLEDGE
;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
;VERIFY RECALIBRATION
;GO TO 2$ IF NO ERROR
;RETURN HERE IF ERROR
;ERROR # DEFINED BY TSTPRP SUBROUTINE
;GO TO 29$ IF ERROR

012306 000404 BR 2$
012310 000240 NOP
012312 104000 EMT
012314 000137 013446 JMP 29$

331
332 ;LOAD PARAMETERS AND GENERATE DATA BUFFER
333 2$:
334 012320 MOV #0,RMDCO ;CYLINDER = 0
335 012326 MOV #0,RMDAO ;TRACK = 0, SECTOR 0
336 012334 MOV #BUFONE,RMBAO ;BUS ADDRESS
337 012342 MOV #-258,RMWCO ;2 + 256 WORDS (2'S COMP)
338 012350 MOV #FMT16,RMOFO ;16 BIT FORMAT
339 012356 MOV #WH!GO,RMCS10 ;WRITE HEADER AND DATA COMMAND
340
341 ;VERIFY THAT SECTOR IS NOT BAD
012364 004737 040242 JSR PC,BADSCT ;CALL BAD SECTOR MODULE
012370 000405 BR 3$ ;GO TO 3$ IF NO ERROR
012372 104401 070426 TYPE ,SCTMSG ;TYPE BAD SECTOR MESSAGE
012376 104000 EMT ;ERROR # DEFINED BY BADSCT SUBROUTINE
012400 000137 013446 JMP 29$ ;GO TO 29$ IF ERROR

342 012404
343 012404 012737 072060 001174 3$: MOV #ZEROS,$TMP0 ;STARTING ADDRESS OF PATTERN
344 012412 012737 000001 001176 MOV #1,$TMP1 ;RANGE OF PATTERN
345 012420 004737 042174 JSR PC,GENBUF ;GO GENERATE BUFFER FOR FORMAT

346
347 012424 012702 001553 MOV #PUTINX,R2 ;R2 - BYTE ENTRY POSITION
348 012430 112722 000034 MOVB #RMDC,(R2)+
349 012434 112722 000006 MOVB #RMDA,(R2)+
350 012440 112722 000004 MOVB #RMBA,(R2)+
351 012444 112722 000002 MOVB #RMWC,(R2)+
352 012450 112722 000032 MOVB #RMOF,(R2)+
353 012454 112722 000000 MOVB #RMCS1,(R2)+

```

```

354 012460 112712 000200          MOVB    #200,(R2)          ;TERMINATE TABLE
355 012464          4$:          JSR      PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
356 012464 004737 043330          BR      5$          ;GO TO 5$ IF NO ERROR
          012470 000404          NOP          ;RETURN HERE IF ERROR
          012472 000240          EMT          ;ERROR # DEFINED BY PUT SUBROUTINE
          012474 104000          JMP      29$          ;GO TO 29$ IF ERROR
          012476 000137 013446
357 012502          5$:
358
359          ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
          012502 004737 042774          JSR      PC,GETSTS          ;GO TO GETSTS SUBROUTINE
360
361          ;WAIT FOR COMMAND TO COMPLETE
          012506 004737 043672          JSR      PC,TIMOUT          ;GO TO TIMEOUT SUBROUTINE
362
363          ;GO EGT REGISTER STATUS
          012512 004737 043060          JSR      PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
          012516 000404          BR      6$          ;GO TO 6$ IF NO ERROR
          012520 000240          NOP          ;RETURN HERE IF ERROR
          012522 104000          EMT          ;ERROR # DEFINED BY GET SUBROUTINE
          012524 000137 013446          JMP      29$          ;GO TO 29$ IF ERROR
365 012530          6$:
366
367          ;VERIFY RESULTS OF WRITE COMMAND
          012530 004737 056572          JSR      PC,DASTS          ;GO VERIFY RESULTS OF DATA TRANSFER
          012534 000405          BR      7$          ;GO TO 7$ IF NO ERROR
          012536 000240          NOP          ;RETURN HERE IF ERROR
          012540 104000          EMT          ;ERROR # DEFINED BY DASTS SUBROUTINE
          012542 004736          JSR      PC,@(SP)+          ;GO BACK FOR MORE ERROR CHECKS
          012544 000137 013446          JMP      29$          ;GO TO 29$ IF ERROR
369 012550          7$:
370
371          ;MOVE LOOP ADDRESSES TO NEXT OPERATION
          012550 012737 012560 001124          MOV      #8$,$LPERR
          012556 000410          BR      9$          ;SKIP TO WRITE OPERATION
374
375          ;*****
376          ;LOOP #2          WRITE,WRITE CHECK DATA
377
378 012560          8$:
379
380          ;PREPARE DEVICE FOR TEST
          012560 004737 037316          JSR      PC,TSTPRP          ;PREPARE DEVICE FOR TEST
          012564 154130          .WORD    154130          ;TASK DESCRIPTOR AS FOLLOWS:
          ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
          ;CLEAR CONTROLLER & SELECT DEVICE
          ;VERIFY CONTROLLER CLEAR OPERATION
          ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
          ;VERIFY PACK ACKNOWLEDGE
          ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
          ;VERIFY RECALIBRATION
          012566 000404          BR      9$          ;GO TO 9$ IF NO ERROR
          012570 000240          NOP          ;RETURN HERE IF ERROR
          012572 104000          EMT          ;ERROR # DEFINED BY TSTPRP SUBROUTINE
          012574 000137 013446          JMP      29$          ;GO TO 29$ IF ERROR
382 012600          9$:
383

```

```

384 ;SETUP PARAMETERS AND EXECUTE COMMAND
385 10$:
386 012600 MOV #256, RMWCO ;CHANGE WORD COUNT
387 012606 MOV #BUFONE+4, RMBAO ;CHANGE MEMORY ADDRESS
388 012614 MOV #WD!GO, RMCS10 ;WRITE DATA COMMAND
389 012622 MOV #PUTINX, R2 ;LOAD PUT REGISTER INDEX TABLE
390 012626 MOVB #RMDA, (R2)+
391 012632 MOVB #RMDC, (R2)+
392 012636 MOVB #RMOF, (R2)+
393 012642 MOVB #RMBB, (R2)+
394 012646 MOVB #RMWC, (R2)+
395 012652 MOVB #RMCS1, (R2)+
396 012656 MOVB #200, (R2)+ ;TERMINATE TABLE
397
398 012662 JSR PC, PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
    012666 BR 11$ ;GO TO 11$ IF NO ERROR
    012670 NOP ;RETURN HERE IF ERROR
    012672 EMT ;ERROR # DEFINED BY PUT SUBROUTINE
    012674 JMP 29$ ;GO TO 29$ IF ERROR
399 11$:
400 ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
401 JSR PC, GETSTS ;GO TO GETSTS SUBROUTINE
402
403 ;WAIT FOR COMMAND TO COMPLETE
    012704 JSR PC, TIMEOUT ;GO TO TIMEOUT SUBROUTINE
404
405 ;GO READ STATUS FOR WRITE COMMAND
406 JSR PC, GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
    012714 BR 12$ ;GO TO 12$ IF NO ERROR
    012716 NOP ;RETURN HERE IF ERROR
    012720 EMT ;ERROR # DEFINED BY GET SUBROUTINE
    012722 JMP 29$ ;GO TO 29$ IF ERROR
407 12$:
408
409 ;CHECK FOR ERRORS DURING WRITE OPERATION
410 JSR PC, PRIERR ;GO CHECK FOR PRIMARY ERRORS
    012732 BR 13$ ;GO TO 13$ IF NO ERROR
    012734 NOP ;RETURN HERE IF ERROR
    012736 EMT ;ERROR # DEFINED BY PRIERR SUBROUTINE
    012740 JSR PC, @ (SP)+ ;GO BACK FOR MORE ERROR CHECKS
    012742 JMP 29$ ;GO TO 29$ IF ERROR
411 13$:
412 JSR PC, DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
    012752 BR 14$ ;GO TO 14$ IF NO ERROR
    012754 NOP ;RETURN HERE IF ERROR
    012756 EMT ;ERROR # DEFINED BY DTASTS SUBROUTINE
    012760 JSR PC, @ (SP)+ ;GO BACK FOR MORE ERROR CHECKS
    012762 JMP 29$ ;GO TO 29$ IF ERROR
413 14$:
414 JSR PC, SECERR ;GO CHECK FOR SECONDARY ERRORS
    012772 BR 15$ ;GO TO 15$ IF NO ERROR
    012774 NOP ;RETURN HERE IF ERROR
    012776 EMT ;ERROR # DEFINED BY SECERR SUBROUTINE
    013000 JSR PC, @ (SP)+ ;GO BACK FOR MORE ERROR CHECKS
    013002 JMP 29$ ;GO TO 29$ IF ERROR
415 15$:
  
```

```

416
417
418 013006 012737 013014 001124 ;CHANGE LOOP ADDRESSES
419 MOV #16$, $LPERR
420
421 ;*****
422 ;LOOP #3 WRITE CHECK DATA
423 013014 16$:
424
425 013014 012703 000001 MOV #1, R3 ;R3+WCE BIT POSITION
426 013020 050337 107440 17$: BIS R3, BUFTWO-2 ;CHANGE LAST WORD OF BUFFER
427
428 ;PREPARE DEVICE FOR TEST
429 013024 004737 037316 JSR PC, TSTPRP ;PREPARE DEVICE FOR TEST
013030 154130 .WORD 154130 ;TASK DESCRIPTOR AS FOLLOWS:
;SELECT DEVICE & VERIFY DEVICE AVAILABLE
;CLEAR CONTROLLER & SELECT DEVICE
;VERIFY CONTROLLER CLEAR OPERATION
;PACK ACKNOWLEDGE IF VOLUME NOT VALID
;VERIFY PACK ACKNOWLEDGE
;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
;VERIFY RECALIBRATION
;GO TO 18$ IF NO ERROR
;RETURN HERE IF ERROR
;ERROR # DEFINED BY TSTPRP SUBROUTINE
;GO TO 29$ IF ERROR

013032 000404 BR 18$
013034 000240 NOP
013036 104000 EMT
013040 000137 013446 JMP 29$
430 013044 18$:
431
432 ;READ DATA FROM DEVICE
433 013044 19$:
434 013044 012737 000051 001412 MOV #WCD!GO, RMCS10 ;WRITE CHECK DATA DATA COMMAND
435 013052 012702 001553 MOV #PUTINX, R2 ;LOAD PUT REGISTER INDEX TABLE
436 013056 112722 000006 MOVB #RMDA, (R2)+
437 013062 112722 000032 MOVB #RMOF, (R2)+
438 013066 112722 000034 MOVB #RMDC, (R2)+
439 013072 112722 000004 MOVB #RMSA, (R2)+
440 013076 112722 000002 MOVB #RMWC, (R2)+
441 013102 112722 000000 MOVB #RMCS1, (R2)+
442 013106 112712 000200 MOVB #200, (R2)
443
444 013112 004737 043330 JSR PC, PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
013116 000404 BR 20$ ;GO TO 20$ IF NO ERROR
013120 000240 NOP ;RETURN HERE IF ERROR
013122 104000 EMT ;ERROR # DEFINED BY PUT SUBROUTINE
013124 000137 013446 JMP 29$ ;GO TO 29$ IF ERROR
445 013130 20$:
446
447 ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
013130 004737 042774 JSR PC, GETSTS ;GO TO GETSTS SUBROUTINE
448
449 ;WAIT FOR COMMAND TO COMPLETE
013134 004737 043672 JSR PC, TIMOUT ;GO TO TIMOUT SUBROUTINE
450
451 ;GO READ STATUS FOR WRITE CHECK DATA OPERATION
452 013140 004737 043060 JSR PC, GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
013144 000404 BR 21$ ;GO TO 21$ IF NO ERROR
013146 000240 NOP ;RETURN HERE IF ERROR

```



```

013150 104000 EMT ;ERROR # DEFINED BY GET SUBROUTINE
013152 000137 013446 JMP 29$ ;GO TO 29$ IF ERROR
453 013156 21$:
454
455 ;CHECK FOR ERRORS DURING WRITE CHECK DATA OPERATION
456 013156 004737 044056 JSR PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
013162 000405 BR 22$ ;GO TO 22$ IF NO ERROR
013164 000240 NOP ;RETURN HERE IF ERROR
013166 104000 EMT ;ERROR # DEFINED BY PRIERR SUBROUTINE
013170 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
013172 000137 013446 JMP 29$ ;GO TO 29$ IF ERROR
457 013176 22$:
458 013176 032737 040000 001346 BIT #WCE,RMCS2I ;WAS 'WCE' DETECTED??
459 013204 001030 BNE 24$ ;YES!!
460
461 013206 004737 056572 JSR PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
013212 000405 BR 23$ ;GO TO 23$ IF NO ERROR
013214 000240 NOP ;RETURN HERE IF ERROR
013216 104000 EMT ;ERROR # DEFINED BY DTASTS SUBROUTINE
013220 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
013222 000137 013446 JMP 29$ ;GO TO 29$ IF ERROR
462 013226 23$:
463 013226 013737 001346 001140 MOV RMCS2I,$GDDAT ;EXPECTED STATUS
464 013234 052737 040000 001140 BIS #WCE,$GDDAT
465 013242 013737 001346 001142 MOV RMCS2I,$BDDAT ;RECEIVED STATUS
466 013250 010337 001174 MOV R3,$TMP0 ;FAILING BIT POSITION
467 013254 012737 107440 001176 MOV #BUFTWO-2,$IMP1 ;FAILING ADDRESS
468 013262 104337 EMT 337
469 013264 000470 BR 29$
470 013266 24$:
471 013266 112737 000022 001524 MOVB #RMDB,GETINX ;SETUP GET INDEX TABLE
013274 112737 000200 001525 MOVB #200,GETINX+1 ;SETUP TERMINATOR BYTE
013302 004737 043060 JSR PC,GET ;GO READ RMDB VIA GET SUBROUTINE
013306 000404 BR 25$ ;GO TO 25$ IF NO ERROR
013310 000240 NOP ;RETURN HERE IF ERROR
013312 104000 EMT ;ERROR DEFINED BY GET SUBROUTINE
013314 000137 013446 JMP 29$ ;GO TO 29$ IF ERROR
472 013320 25$:
473 013320 013737 001360 001142 MOV RMDBI,$BDDAT ;RECEIVED DATA
474 013326 013737 107440 001140 MOV BUFTWO-2,$GDDAT ;EXPECTED DATA
475 013334 040337 001140 BIC R3,$GDDAT
476 013340 012737 107440 001134 MOV #BUFTWO-2,$GDADR ;EXPECTED ADDRESS
477 013346 013737 001342 001136 MOV RMBAI,$BDADR ;RECEIVED ADDRESS
478 013354 162737 0000C2 001136 SUB #2,$BDADR ;ADJUST BUS ADDRESS
479 013362 023737 001134 001136 CMP $GDADR,$BDADR ;ADDRESSES OK??
480 013370 001402 BEQ 26$ ;YES.!
481 013372 104340 EMT 340
482 013374 000424 BR 29$
483 013376 023737 001140 001142 26$: CMP $GDDAT,$BDDAT ;DATA OK??
484 013404 001402 BEQ 27$ ;YES.!
485 013406 104341 EMT 341
486 013410 000416 BR 29$
487 013412 27$:
488
489 013412 004737 044710 JSR PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
013416 000405 BR 28$ ;GO TO 28$ IF NO ERROR
013420 000240 NOP ;RETURN HERE IF ERROR

```

```

013422 104000      EMT
013424 004736      JSR      PC,@(SP)+
013426 000137 013446  JMP      29$      ;ERROR # DEFINED BY SECERR SUBROUTINE
                                ;GO BACK FOR MORE ERROR CHECKS
                                ;GO TO 29$ IF ERROR
490 013432      28$:      BIC      R3,BUFTWO-2      ;RESTORE DATA PATTERN
491 013432 040337 107440      ASL      R3      ;SHIFT TO NEXT BIT
492 013436 006303      BEQ      29$      ;EXIT IF DONE
493 013440 001402      JMP      17$      ;REPEAT TEST FOR NEXT DATA BIT
494 013442 000137 013020
495 013446
496
497
;*****
;*TEST 5      WRITE, READ ONES
;*****
TST5:
013446      SCOPE      ;SCOPE CALL
013446 000004      NOP      ;START OF TEST
013450 000240      MOV      #STACK,SP      ;INITIALIZE STACK POINTER
013452 012706 001100      MOV      $BASE,R0      ;R0 = UNIBUS ADDRESS
013456 013700 001276      MOV      TSTQUE,R1      ;(R1) = DEVICE BEING TESTED
013462 013701 001466      MOV      #5,$TESTN      ;SET TEST NUMBER IN APT MAIL BOX
013466 012737 000005 001226
498
499
500      ;*****
501      ;LOOP #1      FORMAT,WRITE,READ
502 013474      1$:
503
504      ;PREPARE THE DEVICE FOR TEST
505 013474 004737 037316      JSR      PC,TSTPRP      ;PREPARE DEVICE FOR TEST
013500 154130      .WORD      154130      ;TASK DESCRIPTOR AS FOLLOWS:
                                ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
                                ;CLEAR CONTROLLER & SELECT DEVICE
                                ;VERIFY CONTROLLER CLEAR OPERATION
                                ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                ;VERIFY PACK ACKNOWLEDGE
                                ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
                                ;VERIFY RECALIBRATION
                                ;GO TO 2$ IF NO ERROR
                                ;RETURN HERE IF ERROR
                                ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                ;GO TO 25$ IF ERROR
013502 000404      BR      2$
013504 000240      NOP
013506 104000      EMT
013510 000137 014454      JMP      25$
506
507      ;LOAD PARAMETERS AND GENERATE DATA BUFFER
508 013514      2$:
509 013514 012737 000000 001446      MOV      #0,RMDCO      ;CYLINDER = 0
510 013522 012737 000000 001420      MOV      #0,RMDAO      ;TRACK = 0, SECTOR 0
511 013530 012737 106436 001416      MOV      #BUFONE,RMBAO      ;BUS ADDRESS
512 013536 012737 177376 001414      MOV      #-258.,RMWCO      ;2 + 256. WORDS (2'S COMP)
513 013544 012737 010000 001444      MOV      #FMT16,RMOFO      ;16 BIT FORMAT
514 013552 012737 000063 001412      MOV      #WH!GO,RMCSTO      ;WRITE HEADER AND DATA COMMAND
515
516      ;VERIFY THAT SECTOR IS NOT BAD
013560 004737 040242      JSR      PC,BADSCT      ;CALL BAD SECTOR MODULE
013564 000405      BR      3$      ;GO TO 3$ IF NO ERROR
013566 104401 070426      TYPE      ,SCTMSG      ;TYPE BAD SECTOR MESSAGE
013572 104000      EMT
013574 000137 014454      JMP      25$      ;ERROR # DEFINED BY BADSCT SUBROUTINE
                                ;GO TO 25$ IF ERROR
517 013600      3$:

```



```

518 013600 012737 072016 001174      MOV    #ONES,$TMP0      ;STARTING ADDRESS OF PATTERN
519 013606 012737 000001 001176      MOV    #1,$TMP1        ;RANGE OF PATTERN
520 013614 004737 042174              JSR    PC,GENBUF        ;GO GENERATE BUFFER FOR FORMAT
521
522      ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATION
523 013620 012702 001553      MOV    #PUTINX,R2      ;R2 = BYTE ENTRY POSITION
524 013624 112722 000034      MOVB   #RMDX,(R2)+
525 013630 112722 000006      MOVB   #RMDA,(R2)+
526 013634 112722 000004      MOVB   #RMDA,(R2)+
527 013640 112722 000002      MOVB   #RMDC,(R2)+
528 013644 112722 000032      MOVB   #RMOF,(R2)+
529 013650 112722 000000      MOVB   #RMCST,(R2)+
530 013654 112712 000200      MOVB   #200,(R2)      ;TERMINATE TABLE
531 013660
532
533      ;FORMAT THE DRIVE
534 013660 004737 043330      JSR    PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      013664 000404      BR     5$          ;GO TO 5$ IF NO ERROR
      013666 000240      NOP          ;RETURN HERE IF ERROR
      013670 104000      EMT          ;ERROR # DEFINED BY PUT SUBROUTINE
      013672 000137 014454      JMP     25$         ;GO TO 25$ IF ERROR
535 013676
536
537      ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
      013676 004737 042774      JSR    PC,GETSTS      ;GO TO GETSTS SUBROUTINE
538
539      ;WAIT FOR COMMAND TO COMPLETE
      013702 004737 043672      JSR    PC,TIMOUT      ;GO TO TIMEOUT SUBROUTINE
540
541      ;GO READ STATUS FOR FORMAT OPERATION
542 013706 004737 043060      JSR    PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
      013712 000404      BR     6$          ;GO TO 6$ IF NO ERROR
      013714 000240      NOP          ;RETURN HERE IF ERROR
      013716 104000      EMT          ;ERROR # DEFINED BY GET SUBROUTINE
      013720 000137 014454      JMP     25$         ;GO TO 25$ IF ERROR
543 013724
544
545      ;VERIFY NO ERRORS DURING FORMAT
546 013724 004737 056572      JSR    PC,DASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
      013730 000405      BR     7$          ;GO TO 7$ IF NO ERROR
      013732 000240      NOP          ;RETURN HERE IF ERROR
      013734 104000      EMT          ;ERROR # DEFINED BY DASTS SUBROUTINE
      013736 004736      JSR    PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      013740 000137 014454      JMP     25$         ;GO TO 25$ IF ERROR
547 013744
548
549      ;MOVE LOOP ADDRESSES TO NEXT OPERATION
550 013744 012737 013754 001124      MOV    #8$,$LPERR
551 013752 000410      BR     9$          ;SKIP TO WRITE OPERATION
552
553      ;*****
554      ;LOOP #2      WRITE,READ
555
556 013754
557
558
559      ;PREPARE DEVICE FOR TEST

```

Line	Address	Offset	Label	Operation	PC	TSTPRP	Comments
560	013754	004737	037316	JSR	PC	TSTPRP	:PREPARE DEVICE FOR TEST
	013760	154130		.WORD	154130		:TASK DESCRIPTOR AS FOLLOWS:
							:SELECT DEVICE & VERIFY DEVICE AVAILABLE
							:CLEAR CONTROLLER & SELECT DEVICE
							:VERIFY CONTROLLER CLEAR OPERATION
							:PACK ACKNOWLEDGE IF VOLUME NOT VALID
							:VERIFY PACK ACKNOWLEDGE
							:RECALIBRATE IF 'SKI' OR 'PIP' IS SET
							:VERIFY RECALIBRATION
	013762	000404		BR	9\$		:GO TO 9\$ IF NO ERROR
	013764	000240		NOP			:RETURN HERE IF ERROR
	013766	104000		EMT			:ERROR # DEFINED BY TSTPRP SUBROUTINE
	013770	000137	014454	JMP	25\$		:GO TO 25\$ IF ERROR
561	013774			9\$:			
562							
563							
564	013774						:WRITE DATA TO THE DRIVE
565	013774	012737	106442	001416	MOV	#BUFONE+4,RMBAD	:CHANGE MEMORY ADDRESS
566	014002	012737	177400	001414	MOV	#-256.,RMWCO	:CHANGE WORD COUNT
567	014010	012737	000061	001412	MOV	#WD!GO,RMCS10	:WRITE DATA COMMAND
568	014016	012702	001553		MOV	#PUTINX,R2	:LOAD PUT REGISTER INDEX TABLE
569	014022	112722	000006		MOVB	#RMDA,(R2)+	
570	014026	112722	000034		MOVB	#RMDC,(R2)+	
571	014032	112722	000032		MOVB	#RMOF,(R2)+	
572	014036	112722	000004		MOVB	#RMDA,(R2)+	
573	014042	112722	000002		MOVB	#RMWC,(R2)+	
574	014046	112722	000000		MOVB	#RMCS1,(R2)+	
575	014052	112722	0002C0		MOVB	#200,(R2)+	:TERMINATE TABLE
576							
577	014056	004737	043330		JSR	PC,PUT	:GO WRITE REGISTER(S) WITH PUT SUBROUTINE
	014062	000404			BR	11\$	:GO TO 11\$ IF NO ERROR
	014064	000240			NOP		:RETURN HERE IF ERROR
	014066	104000			EMT		:ERROR # DEFINED BY PUT SUBROUTINE
	014070	000137	014454		JMP	25\$	:GO TO 25\$ IF ERROR
578	014074			11\$:			
579							
580							:SETUP GET INDEX TABLE TO READ ALL REGISTERS
	014074	004737	042774		JSR	PC,GETSTS	:GO TO GETSTS SUBROUTINE
581							
582							:WAIT FOR COMMAND TO COMPLETE
	014100	004737	043672		JSR	PC,TIMOUT	:GO TO TIMOUT SUBROUTINE
583							
584							:GO READ STATUS FOR WRITE COMMAND
585	014104	004737	043060		JSR	PC,GET	:GO READ REGISTER(S) WITH GET SUBROUTINE
	014110	000404			BR	12\$	:GO TO 12\$ IF NO ERROR
	014112	000240			NOP		:RETURN HERE IF ERROR
	014114	104000			EMT		:ERROR # DEFINED BY GET SUBROUTINE
	014116	000137	014454		JMP	25\$	:GO TO 25\$ IF ERROR
586	014122			12\$:			
587							
588							:CHECK FOR ERRORS DURING WRITE OPERATION
589	014122	004737	044056		JSR	PC,PRIERR	:GO CHECK FOR PRIMARY ERRORS
	014126	000405			BR	13\$	:GO TO 13\$ IF NO ERROR
	014130	000240			NOP		:RETURN HERE IF ERROR
	014132	104000			EMT		:ERROR # DEFINED BY PRIERR SUBROUTINE
	014134	004736			JSR	PC,@(SP)+	:GO BACK FOR MORE ERROR CHECKS
	014136	000137	014454		JMP	25\$	:GO TO 25\$ IF ERROR

```

590 014142      13$:
591 014142 004737 056572      JSR      PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
      014146 000405      BR      14$      ;GO TO 14$ IF NO ERROR
      014150 000240      NOP      ;RETURN HERE IF ERROR
      014152 104000      EMT      ;ERROR # DEFINED BY DTASTS SUBROUTINE
      014154 004736      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      014156 000137 014454      JMP      25$      ;GO TO 25$ IF ERROR

592 014162      14$:
593 014162 004737 044710      JSR      PC,SECERR      ;GO CHECK FOR SECONDARY ERRORS
      014166 000405      BR      15$      ;GO TO 15$ IF NO ERROR
      014170 000240      NOP      ;RETURN HERE IF ERROR
      014172 104000      EMT      ;ERROR # DEFINED BY SECERR SUBROUTINE
      014174 004736      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      014176 000137 014454      JMP      25$      ;GO TO 25$ IF ERROR

594 014202      15$:
595
596      ;CHANGE LOOP ADDRESSES
597 014202 012737 014212 001124      MOV      #16$,$LPERR
598 014210 000410      BR      17$      ;SKIP TO NEXT OPERATION
599
600      ;*****
601      ;LOOP #3      READ
602
603 014212      16$:
604
605      ;PREPARE DEVICE FOR TEST
606 014212 004737 037316      JSR      PC,TSTPRP      ;PREPARE DEVICE FOR TEST
      014216 154130      .WORD      154130      ;TASK DESCRIPTOR AS FOLLOWS:
      ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
      ;CLEAR CONTROLLER & SELECT DEVICE
      ;VERIFY CONTROLLER CLEAR OPERATION
      ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
      ;VERIFY PACK ACKNOWLEDGE
      ;RECALIBRATE IF 'SKI' OR 'PIP' IS SFT
      ;VERIFY RECALIBRATION
      ;GO TO 17$ IF NO ERROR
      ;RETURN HERE IF ERROR
      ;ERROR # DEFINED BY TSTPRP SUBROUTINE
      ;GO TO 25$ IF ERROR

      014220 000404      BR      17$
      014222 000240      NOP
      014224 104000      EMT
      014226 000137 014454      JMP      25$

607 014232      17$:
608
609      ;READ DATA FROM DEVICE
610 014232      18$:
611 014232 012737 107446 001416      MOV      #BUFTWO+4,RMBAD      ;CHANGE MEMORY ADDRESS
612 014240 012737 000071 001412      MOV      #RD!GO,RMCS10      ;READ DATA COMMAND
613 014246 012702 001553      MOV      #PUTINX,R2      ;LOAD PUT REGISTER INDEX TABLE
614 014252 112722 000006      MOV      #RMDA,(R2)+
615 014256 112722 000032      MOV      #RMOF,(R2)+
616 014262 112722 000034      MOV      #RMDC,(R2)+
617 014266 112722 000004      MOV      #RMBB,(R2)+
618 014272 112722 000002      MOV      #RMWC,(R2)+
619 014276 112722 000000      MOV      #RMCS1,(R2)+
620 014302 112712 000200      MOV      #200,(R2)
621
622 014306 004737 043330      JSR      PC,PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      014312 000404      BR      19$      ;GO TO 19$ IF NO ERROR
      014314 000240      NOP      ;RETURN HERE IF ERROR
  
```

```

    014316 104000
    014320 000137 014454
623 014324
624
625
    014324 004737 042774
626
627
    014330 004737 043672
628
629
630 014334 004737 043060
    014340 000404
    014342 000240
    014344 104000
    014346 000137 014454
631 014352
632
633
634 014352 004737 044056
    014356 000405
    014360 000240
    014362 104000
    014364 004736
    014366 000137 014454
635 014372
636 014372 004737 056572
    014376 000405
    014400 000240
    014402 104000
    014404 004736
    014406 000137 014454
637 014412
638 014412 004737 044710
    014416 000405
    014420 000240
    014422 104000
    014424 004736
    014426 000137 014454
639 014432
640 014432 004737 042432
    014436 106442
    014440 107446
    014442 000404
    014444 000240
    014446 104000
    014450 000137 014454
641 014454
642
643 014454
644
645

    014454
    014454 000004
    014456 000240

    EMT
    JMP 25$
19$:
    ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
    JSR PC,GETSTS ;GO TO GETSTS SUBROUTINE
    ;WAIT FOR COMMAND TO COMPLETE
    JSR PC,TIMOUT ;GO TO TIMEOUT SUBROUTINE
    ;GO READ STATUS FOR READ OPERATION
    JSR PC,GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
    BR 20$ ;GO TO 20$ IF NO ERROR
    NOP ;RETURN HERE IF ERROR
    EMT
    JMP 25$ ;ERROR # DEFINED BY GET SUBROUTINE
    ;GO TO 25$ IF ERROR
20$:
    ;CHECK FOR ERRORS DURING READ OPERATION
    JSR PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
    BR 21$ ;GO TO 21$ IF NO ERROR
    NOP ;RETURN HERE IF ERROR
    EMT
    JSR PC,@(SP)+ ;ERROR # DEFINED BY PRIERR SUBROUTINE
    JMP 25$ ;GO BACK FOR MORE ERROR CHECKS
    ;GO TO 25$ IF ERROR
21$:
    JSR PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
    BR 22$ ;GO TO 22$ IF NO ERROR
    NOP ;RETURN HERE IF ERROR
    EMT
    JSR PC,@(SP)+ ;ERROR # DEFINED BY DTASTS SUBROUTINE
    JMP 25$ ;GO BACK FOR MORE ERROR CHECKS
    ;GO TO 25$ IF ERROR
22$:
    JSR PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
    BR 23$ ;GO TO 23$ IF NO ERROR
    NOP ;RETURN HERE IF ERROR
    EMT
    JSR PC,@(SP)+ ;ERROR # DEFINED BY SECERR SUBROUTINE
    JMP 25$ ;GO BACK FOR MORE ERROR CHECKS
    ;GO TO 25$ IF ERROR
23$:
    JSR PC,CMPBUF ;GO COMPARE WRITE, READ DATA BUFFERS
    .WORD BUFOFF+4 ;STARTING ADDRESS OF WRITE BUFFER
    .WORD BUFTWO+4 ;STARTING ADDRESS OF READ BUFFER
    BR 24$ ;GO TO 24$ IF NO ERROR
    NOP ;RETURN HERE IF ERROR
    EMT
    JMP 25$ ;ERROR # DEFINED BY CMPBUF SUBROUTINE
    ;GO TO 25$ IF ERROR
24$:
25$:
    ;*****
    ;*TEST 6 WRITE, WRITE CHECK ONES
    ;*****
    TST6:
    SCOPE ;SCOPE CALL
    NOP ;START OF TEST
  
```

```
014460 012706 001100      MOV      #STACK,SP      ;INITIALIZE STACK POINTER
014464 013700 001276      MOV      $BASE,R0      ;R0 = UNIBUS ADDRESS
014470 013701 001466      MOV      TSTQUE,R1      ;(R1) = DEVICE BEING TESTED
014474 012737 000006 001226  MOV      #6,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

646
647      ;*****
648      ;LOOP #1      FORMAT,WRITE,WRITE CHECK DATA
649
650 014502      1$:
651
652      ;PREPARE THE DEVICE FOR TEST
653 014502 004737 037316      JSR      PC,TSTPRP      ;PREPARE DEVICE FOR TEST
014506 154130      .WORD      154130      ;TASK DESCRIPTOR AS FOLLOWS:
                                           ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
                                           ;CLEAR CONTROLLER & SELECT DEVICE
                                           ;VERIFY CONTROLLER CLEAR OPERATION
                                           ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                           ;VERIFY PACK ACKNOWLEDGE
                                           ;RECALIBRATE IF 'SKI' OR 'PIP' IS SFT
                                           ;VERIFY RECALIBRATION
                                           ;GO TO 2$ IF NO ERROR
                                           ;RETURN HERE IF ERROR
                                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                           ;GO TO 24$ IF ERROR

014510 000404      BR      2$
014512 000240      NOP
014514 104000      EMT
014516 000137 015432      JMP      24$

654
655      ;LOAD PARAMETERS AND GENERATE DATA BUFFER
656 014522      2$:
657 014522 012737 000000 001446      MOV      #0,RMDCO      ;CYLINDER = 0
658 014530 012737 000000 001420      MOV      #0,RMDAO      ;TRACK = 0, SECTOR = 0
659 014536 012737 106436 001416      MOV      #BUFONE,RMBAO      ;BUS ADDRESS
660 014544 012737 177376 001414      MOV      #-258.,RMWCO      ;2 + 256. WORDS (2'S COMP)
661 014552 012737 010000 001444      MOV      #FMT16,RMOFO      ;16 BIT FORMAT
662 014560 012737 000063 001412      MOV      #WH!GO,RMCS10      ;WRITE HEADER AND DATA COMMAND
663
664      ;VERIFY THAT SECTOR IS NOT BAD
014566 004737 040242      JSR      PC,BADSCT      ;CALL BAD SECTOR MODULE
014572 000405      BR      3$      ;GO TO 3$ IF NO ERROR
014574 104401 070426      TYPE      ,SCTMSG      ;TYPE BAD SECTOR MESSAGE
014600 104000      EMT      ;ERROR # DEFINED BY BADSCT SUBROUTINE
014602 000137 015432      JMP      24$      ;GO TO 24$ IF ERROR

665 014606      3$:
666 014606 012737 072016 001174      MOV      #ONES,$TMP0      ;STARTING ADDRESS OF PATTERN
667 014614 012737 000001 001176      MOV      #1,$TMP1      ;RANGE OF PATTERN
668 014622 004737 042174      JSR      PC,GENBUF      ;GO GENERATE BUFFER FOR FORMAT
669
670      ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATION
671 014626 012702 001553      MOV      #PUTINX,R2      ;R2 = BYTE ENTRY POSITION
672 014632 112722 000034      MOV      #RMDC,(R2)+
673 014636 112722 000006      MOV      #RMDA,(R2)+
674 014642 112722 000004      MOV      #RMBA,(R2)+
675 014646 112722 000002      MOV      #RMWC,(R2)+
676 014652 112722 000032      MOV      #RMOF,(R2)+
677 014656 112722 000000      MOV      #RMCS1,(R2)+
678 014662 112712 000200      MOV      #200,(R2)      ;TERMINATE TABLE
679 014666
680
681      4$:
      ;FORMAT THE DRIVE
```

```

682 014666 004737 043330      JSR      PC,PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      014672 000404      BR      5$      ;GO TO 5$ IF NO ERROR
      014674 000240      NOP      ;RETURN HERE IF ERROR
      014676 104000      EMT      ;ERROR # DEFINED BY PUT SUBROUTINE
      014700 000137 015432      JMP      24$      ;GO TO 24$ IF ERROR
683 014704      5$:
684
685      ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
      014704 004737 042774      JSR      PC,GETSTS      ;GO TO GETSTS SUBROUTINE
686
687      ;WAIT FOR COMMAND TO COMPLETE
      014710 004737 043672      JSR      PC,TIMOUT      ;GO TO TIMOUT SUBROUTINE
688
689      ;GO READ STATUS FOR FORMAT OPERATION
690 014714 004737 043060      JSR      PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      014720 000404      BR      6$      ;GO TO 6$ IF NO ERROR
      014722 000240      NOP      ;RETURN HERE IF ERROR
      014724 104000      EMT      ;ERROR # DEFINED BY GET SUBROUTINE
      014726 000137 015432      JMP      24$      ;GO TO 24$ IF ERROR
691 014732      6$:
692
693      ;VERIFY NO ERRORS DURING FORMAT
694 014732 004737 056572      JSR      PC,DFASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
      014736 000405      BR      7$      ;GO TO 7$ IF NO ERROR
      014740 000240      NOP      ;RETURN HERE IF ERROR
      014742 104000      EMT      ;ERROR # DEFINED BY DFASTS SUBROUTINE
      014744 004736      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      014746 000137 015432      JMP      24$      ;GO TO 24$ IF ERROR
695 014752      7$:
696
697      ;MOVE LOOP ADDRESSES TO NEXT OPERATION
698 014752 012737 014762 001124      MOV      #8$,SLPERR
699 014760 000410      BR      9$      ;SKIP TO WRITE OPERATION
700
701      ;*****
702      ;LOOP #2      WRITE,WRITE CHECK DATA
703
704 014762      8$:
705
706
707      ;PREPARE DEVICE FOR WRITE OPERATION
708 014762 004737 037316      JSR      PC,TSTPRP      ;PREPARE DEVICE FOR TEST
      014766 154130      .WORD      154130      ;TASK DESCRIPTOR AS FOLLOWS:
      ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
      ;CLEAR CONTROLLER & SELECT DEVICE
      ;VERIFY CONTROLLER CLEAR OPERATION
      ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
      ;VERIFY PACK ACKNOWLEDGE
      ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
      ;VERIFY RECALIBRATION
      014770 000404      BR      9$      ;GO TO 9$ IF NO ERROR
      014772 000240      NOP      ;RETURN HERE IF ERROR
      014774 104000      EMT      ;ERROR # DEFINED BY TSTPRP SUBROUTINE
      014776 000137 015432      JMP      24$      ;GO TO 24$ IF ERROR
709 015002      9$:
710
711      ;WRITE DATA TO THE DRIVE

```



```

712 015002      10$:
713 015002 012737 106442 001416  MOV    #BUFONE+4,RMBA0      ;CHANGE MEMORY ADDRESS
714 015010 012737 177400 001414  MOV    #-256.,RMWCO      ;CHANGE WORD COUNT
715 015016 012737 000061 001412  MOV    #WD!GO,RMCS10    ;WRITE DATA COMMAND
716 015024 012702 001553      MOV    #PUTINX,R2      ;LOAD PUT REGISTER INDEX TABLE
717 015030 112722 000006      MOVB   #RMDA,(R2)+
718 015034 112722 000034      MOVB   #RMDC,(R2)+
719 015040 112722 000032      MOVB   #RMOF,(R2)+
720 015044 112722 000004      MOVB   #RMDA,(R2)+
721 015050 112722 000002      MOVB   #RMWC,(R2)+
722 015054 112722 000000      MOVB   #RMCS1,(R2)+
723 015060 112722 000200      MOVB   #200,(R2)+      ;TERMINATE TABLE
724
725 015064 004737 043330      JSR     PC,PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      015070 000404      BR      11$      ;GO TO 11$ IF NO ERROR
      015072 000240      NOP      ;RETURN HERE IF ERROR
      015074 104000      EMT      ;ERROR # DEFINED BY PUT SUBROUTINE
      015076 000137 015432      JMP     24$      ;GO TO 24$ IF ERROR
726 015102
727
728 015102 004737 042774      ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
      JSR     PC,GETSTS      ;GO TO GETSTS SUBROUTINE
729
730 015106 004737 043672      ;WAIT FOR COMMAND TO COMPLETE
      JSR     PC,TIMOUT      ;GO TO TIMEOUT SUBROUTINE
731
732 015112 004737 043060      ;GO READ STATUS FOR WRITE COMMAND
733 015116 000404      JSR     PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      015120 000240      BR      12$      ;GO TO 12$ IF NO ERROR
      015122 104000      NOP      ;RETURN HERE IF ERROR
      015124 000137 015432      EMT      ;ERROR # DEFINED BY GET SUBROUTINE
      015130      JMP     24$      ;GO TO 24$ IF ERROR
734
735      12$:
736      ;CHECK FOR ERRORS DURING WRITE OPERATION
737 015130 004737 044056      JSR     PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
      015134 000405      BR      13$      ;GO TO 13$ IF NO ERROR
      015136 000240      NOP      ;RETURN HERE IF ERROR
      015140 104000      EMT      ;ERROR # DEFINED BY PRIERR SUBROUTINE
      015142 004736      JSR     PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      015144 000137 015432      JMP     24$      ;GO TO 24$ IF ERROR
738 015150
739 015150 004737 056572      13$:
      JSR     PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
      015154 000405      BR      14$      ;GO TO 14$ IF NO ERROR
      015156 000240      NOP      ;RETURN HERE IF ERROR
      015160 104000      EMT      ;ERROR # DEFINED BY DTASTS SUBROUTINE
      015162 004736      JSR     PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      015164 000137 015432      JMP     24$      ;GO TO 24$ IF ERROR
740 015170
741 015170 004737 044710      14$:
      JSR     PC,SECERR      ;GO CHECK FOR SECONDARY ERRORS
      015174 000405      BR      15$      ;GO TO 15$ IF NO ERROR
      015176 000240      NOP      ;RETURN HERE IF ERROR
      015200 104000      EMT      ;ERROR # DEFINED BY SECERR SUBROUTINE
      015202 004736      JSR     PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      015204 000137 015432      JMP     24$      ;GO TO 24$ IF ERROR
742 015210
743      15$:

```

```

744 ;CHANGE LOOP ADDRESSES
745 015210 012737 015220 001124 MOV #16$,SLPERR
746 015216 000410 BR 17$ ;SKIP TO NEXT OPERATION
747
748 ;*****
749 ;LOOP #3 WRITE CHECK DATA
750
751 015220 16$:
752
753 ;PREPARE DEVICE FOR WRITE CHECK DATA OPERATION
754 015220 004737 037316 JSR PC,TSTPRP ;PREPARE DEVICE FOR TEST
      015224 154130 .WORD 154130 ;TASK DESCRIPTOR AS FOLLOWS:
                                ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
                                ;CLEAR CONTROLLER & SELECT DEVICE
                                ;VERIFY CONTROLLER CLEAR OPERATION
                                ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                ;VERIFY PACK ACKNOWLEDGE
                                ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
                                ;VERIFY RECALIBRATION
                                ;GO TO 17$ IF NO ERROR
                                ;RETURN HERE IF ERROR
                                ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                ;GO TO 24$ IF ERROR
      015226 000404 BR 17$
      015230 000240 NOP
      015232 104000 EMT
      015234 000137 015432 JMP 24$
755 015240 17$:
756
757 ;WRITE CHECK DATA DATA FROM DEVICE
758 015240 18$:
759 015240 012737 000051 001412 MOV #WCD!GO,RMCS10 ;WRITE CHECK DATA DATA COMMAND
760 015246 012702 001553 MOV #PUTINX,R2 ;LOAD PUT REGISTER INDEX TABLE
761 015252 112722 000006 MOV #RMDC,(R2)+
762 015256 112722 000032 MOV #RMBA,(R2)+
763 015262 112722 000034 MOV #RMWC,(R2)+
764 015266 112722 000004 MOV #RMCS1,(R2)+
765 015272 112722 000002 MOV #200,(R2)
766 015276 112722 000000
767 015302 112712 000200
768
769 015306 004737 043330 JSR PC,PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      015312 000404 BR 19$ ;GO TO 19$ IF NO ERROR
      015314 000240 NOP ;RETURN HERE IF ERROR
      015316 104000 EMT ;ERROR # DEFINED BY PUT SUBROUTINE
      015320 000137 015432 JMP 24$ ;GO TO 24$ IF ERROR
770 015324 19$:
771
772 ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
      015324 004737 042774 JSR PC,GETSTS ;GO TO GETSTS SUBROUTINE
773
774 ;WAIT FOR COMMAND TO COMPLETE
      015330 004737 043672 JSR PC,TIMOUT ;GO TO TIMOUT SUBROUTINE
775
776 ;GO READ STATUS FOR WRITE CHECK DATA OPERATION
777 015334 004737 043060 JSR PC,GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
      015340 000404 BR 20$ ;GO TO 20$ IF NO ERROR
      015342 000240 NOP ;RETURN HERE IF ERROR
      015344 104000 EMT ;ERROR # DEFINED BY GET SUBROUTINE
      015346 000137 015432 JMP 24$ ;GO TO 24$ IF ERROR
778 015352 20$:

```



```

779
780 ;CHECK FOR ERRORS DURING WRITE CHECK DATA OPERATION
781 015352 004737 044056      JSR    PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
                                BR      21$          ;GO TO 21$ IF NO ERROR
                                NOP                    ;RETURN HERE IF ERROR
                                EMT                    ;ERROR # DEFINED BY PRIERR SUBROUTINE
                                JSR    PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
                                JMP     24$          ;GO TO 24$ IF ERROR
782 015372 004737 056572      21$: JSR    PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
                                BR      22$          ;GO TO 22$ IF NO ERROR
                                NOP                    ;RETURN HERE IF ERROR
                                EMT                    ;ERROR # DEFINED BY DTASTS SUBROUTINE
                                JSR    PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
                                JMP     24$          ;GO TO 24$ IF ERROR
783 015372 004737 056572
784 015412 004737 044710      22$: JSR    PC,SECERR      ;GO CHECK FOR SECONDARY ERRORS
                                BR      23$          ;GO TO 23$ IF NO ERROR
                                NOP                    ;RETURN HERE IF ERROR
                                EMT                    ;ERROR # DEFINED BY SECERR SUBROUTINE
                                JSR    PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
                                JMP     24$          ;GO TO 24$ IF ERROR
785 015412 004737 044710
786 015432 000137 015432      23$: JSR    PC,SECERR      ;GO CHECK FOR SECONDARY ERRORS
                                BR      23$          ;GO TO 23$ IF NO ERROR
                                NOP                    ;RETURN HERE IF ERROR
                                EMT                    ;ERROR # DEFINED BY SECERR SUBROUTINE
                                JSR    PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
                                JMP     24$          ;GO TO 24$ IF ERROR
787
788 015432 000137 015432      24$:
789
790 ;*****
; *TEST 7      WRITE, WRITE CHECK ONES W/ WCE ERROR
;*****
TST7:
015432 000004      SCOPE          ;SCOPE CALL
015433 000240      NOP            ;START OF TEST
015434 000240      MOV     #STACK,SP ;INITIALIZE STACK POINTER
015435 012706 001100  MOV     $BASE,R0 ;R0 = UNIBUS ADDRESS
015436 013700 001276  MOV     TSTQUE,R1 ; (R1) = DEVICE BEING TESTED
015437 013701 001466  MOV     #7,$TESTN ;SET TEST NUMBER IN APT MAIL BOX
015452 012737 000007 001226

791
792 ;*****
793 ;LOOP #1      FORMAT,WRITE,WRITE CHECK DATA
794
795 015460      1$:
796
797 ;PREPARE THE DEVICE FOR FORMAT OPERATION
798 015460 004737 037316      JSR    PC,TSTPRP      ;PREPARE DEVICE FOR TEST
                                .WORD  154130        ;TASK DESCRIPTOR AS FOLLOWS:
                                                        ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
                                                        ;CLEAR CONTROLLER & SELECT DEVICE
                                                        ;VERIFY CONTROLLER CLEAR OPERATION
                                                        ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                        ;VERIFY PACK ACKNOWLEDGE
                                                        ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
                                                        ;VERIFY RECALIBRATION
                                BR      2$            ;GO TO 2$ IF NO ERROR
                                NOP                    ;RETURN HERE IF ERROR
                                EMT                    ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                JMP     29$          ;GO TO 29$ IF ERROR
799 015500      2$:
015466 000404      BR      2$
015470 000240      NOP
015472 104000      EMT
015474 000137 016630      JMP     29$
015500

```

```

800
801 ;LOAD PARAMETERS AND GENERATE DATA BUFFER
802 015500 012737 000000 001446 MOV #0,RMDCO ;CYLINDER = 0
803 015506 012737 000000 001420 MOV #0,RMDAO ;TRACK = 0, SECTOR = 0
804 015514 012737 106436 001416 MOV #RUFONE,RMBAD ;BUS ADDRESS
805 015522 012737 177376 001414 MOV #-258.,RMWCO ;WORD COUNT = 1 SECTOR
806 015530 012737 010000 001444 MOV #FMT16,RMOFO ;16 BIT FORMAT
807 015536 012737 000063 001412 MOV #WH!GO,RMCS10 ;WRITE HEADER AND DATA COMMAND
808
809 ;VERIFY THAT SECTOR IS NOT BAD
      015544 004737 040242 JSR PC,BADSCT ;CALL BAD SECTOR MODULE
      015550 000405 BR 3$ ;GO TO 3$ IF NO ERROR
      015552 104401 070426 TYPE ,SCTMSG ;TYPE BAD SECTOR MESSAGE
      015556 104000 EMT ;ERROR # DEFINED BY BADSCT SUBROUTINE
      015560 000137 016630 JMP 29$ ;GO TO 29$ IF ERROR
810 3$:
811 015564 012737 072016 001174 MOV #ONES,$TMP0 ;STARTING ADDRESS OF PATTERN
812 015572 012737 000001 001176 MOV #1,$TMP1 ;RANGE OF PATTERN
813 015600 004737 042174 JSR PC,GENBUF ;GO GENERATE BUFFER FOR FORMAT
814
815 ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATION
816 015604 012702 001553 MOV #PUTINX,R2 ;R2 = BYTE ENTRY POSITION
817 015610 112722 000034 MOVB #RMDC,(R2)+
818 015614 112722 000006 MOVB #RMDA,(R2)+
819 015620 112722 000004 MOVB #RMBAD,(R2)+
820 015624 112722 000002 MOVB #RMWC,(R2)+
821 015630 112722 000032 MOVB #RMOF,(R2)+
822 015634 112722 000000 MCVB #RMCS1,(R2)+
823 015640 112712 000200 MOVB #200,(R2) ;TERMINATE TABLE
824
825 4$:
826 ;FORMAT THE DRIVE
827 015644 004737 043330 JSR PC,PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      015650 000404 BR 5$ ;GO TO 5$ IF NO ERROR
      015652 000240 NOP ;RETURN HERE IF ERROR
      015654 104000 EMT ;ERROR # DEFINED BY PUT SUBROUTINE
      015656 000137 016630 JMP 29$ ;GO TO 29$ IF ERROR
828 5$:
829
830 ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
      015662 004737 042774 JSR PC,GETSTS ;GO TO GETSTS SUBROUTINE
831
832 ;WAIT FOR COMMAND TO COMPLETE
      015666 004737 043672 JSR PC,TIMOUT ;GO TO TIMEOUT SUBROUTINE
833
834 ;GO READ STATUS FOR FORMAT OPERATION
835 015672 004737 043060 JSR PC,GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
      015676 000404 BR 6$ ;GO TO 6$ IF NO ERROR
      015700 000240 NOP ;RETURN HERE IF ERROR
      015702 104000 EMT ;ERROR # DEFINED BY GET SUBROUTINE
      015704 000137 016630 JMP 29$ ;GO TO 29$ IF ERROR
836 6$:
837
838 ;VERIFY NO ERRORS DURING FORMAT
839 015710 004737 056572 JSR PC,DASTS ;GO VERIFY RESULTS OF DATA TRANSFER
      015714 000405 BR 7$ ;GO TO 7$ IF NO ERROR
      015716 000240 NOP ;RETURN HERE IF ERROR

```

```

015720 104000      EMT
015722 004736      JSR      PC,@(SP)+      ;ERROR # DEFINED BY DTASTS SUBROUTINE
015724 000137 016630 JMP      29$      ;GO BACK FOR MORE ERROR CHECKS
840 015730      7$:      ;GO TO 29$ IF ERROR
841
842      ;MOVE LOOP ADDRESSES TO NEXT OPERATION
843 015730 012737 015740 001124 MOV      #8$,$LPERR
844 015736 000410      BR      9$      ;SKIP TO WRITE OPERATION
845
846      ;*****
847      ;LOOP #2      WRITE,WRITE CHECK DATA
848
849 015740      8$:
850
851      ;PREPARE DEVICE FOR WRITE OPERATION
852 015740 004737 037316 JSR      PC,TSTPRP      ;PREPARE DEVICE FOR TEST
015744 154130      .WORD 154130      ;TASK DESCRIPTOR AS FOLLOWS:
      ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
      ;CLEAR CONTROLLER & SELECT DEVICE
      ;VERIFY CONTROLLER CLEAR OPERATION
      ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
      ;VERIFY PACK ACKNOWLEDGE
      ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
      ;VERIFY RECALIBRATION
015746 000404      BR      9$      ;GO TO 9$ IF NO ERROR
015750 000240      NOP
015752 104000      EMT
015754 000137 016630 JMP      29$      ;RETURN HERE IF ERROR
853 015760      9$:      ;ERROR # DEFINED BY TSTPRP SUBROUTINE
854      ;GO TO 29$ IF ERROR
855
856      ;WRITE DATA TO THE DRIVE
857 015760 012737 177400 001414 10$: MOV      #-256.,RMWCO      ;CHANGE WORD COUNT
858 015766 012737 106442 001416      MOV      #BUFONE+4,RMBAD      ;CHANGE MEMORY ADDRESS
859 015774 012737 000061 001412      MOV      #WD!GO,RMCS10      ;WRITE DATA COMMAND
860 016002 012702 001553      MOV      #PUTINX,R2      ;LOAD PUT REGISTER INDEX TABLE
861 016006 112722 000006      MOV      #RMDA,(R2)+
862 016012 112722 000034      MOV      #RMDC,(R2)+
863 016016 112722 000032      MOV      #RMOF,(R2)+
864 016022 112722 000004      MOV      #RMEA,(R2)+
865 016026 112722 000002      MOV      #RMWC,(R2)+
866 016032 112722 000000      MOV      #RMCS1,(R2)+
867 016036 112722 000200      MOV      #200,(R2)+      ;TERMINATE TABLE
868
869 016042 004737 043330 JSR      PC,PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
016046 000404      BR      11$      ;GO TO 11$ IF NO ERROR
016050 000240      NOP      ;RETURN HERE IF ERROR
016052 104000      EMT
016054 000137 016630 JMP      29$      ;ERROR # DEFINED BY PUT SUBROUTINE
870 016060      11$:      ;GO TO 29$ IF ERROR
871
872      ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
016060 004737 042774 JSR      PC,GETSTS      ;GO TO GETSTS SUBROUTINE
873
874      ;WAIT FOR COMMAND TO COMPLETE
016064 004737 043672 JSR      PC,TIMOUT      ;GO TO TIMOUT SUBROUTINE
875

```

```

876 ;GO READ STATUS FOR WRITE COMMAND
877 016070 004737 043060 JSR PC,GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
      016074 000404 BR 12$ ;GO TO 12$ IF NO ERROR
      016076 000240 NOP ;RETURN HERE IF ERROR
      016100 104000 EMT ;ERROR # DEFINED BY GET SUBROUTINE
      016102 000137 016630 JMP 29$ ;GO TO 29$ IF ERROR
878 016106 12$:
879
880 ;CHECK FOR ERRORS DURING WRITE OPERATION
881 016106 004737 044056 JSR PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
      016112 000405 BR 13$ ;GO TO 13$ IF NO ERROR
      016114 000240 NOP ;RETURN HERE IF ERROR
      016116 104000 EMT ;ERROR # DEFINED BY PRIERR SUBROUTINE
      016120 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
      016122 000137 016630 JMP 29$ ;GO TO 29$ IF ERROR
882 016126 13$:
883 016126 004737 056572 JSR PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
      016132 000405 BR 14$ ;GO TO 14$ IF NO ERROR
      016134 000240 NOP ;RETURN HERE IF ERROR
      016136 104000 EMT ;ERROR # DEFINED BY DTASTS SUBROUTINE
      016140 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
      016142 000137 016630 JMP 29$ ;GO TO 29$ IF ERROR
884 016146 14$:
885 016146 004737 044710 JSR PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
      016152 000405 BR 15$ ;GO TO 15$ IF NO ERROR
      016154 000240 NOP ;RETURN HERE IF ERROR
      016156 104000 EMT ;ERROR # DEFINED BY SECERR SUBROUTINE
      016160 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
      016162 000137 016630 JMP 29$ ;GO TO 29$ IF ERROR
886 016166 15$:
887
888 ;CHANGE LOOP ADDRESSES
889 016166 012737 016174 001124 MOV #16$,$LPERR
890
891 ;*****
892 ;LOOP #3 WRITE CHECK DATA
893
894 016174 16$:
895
896 016174 012703 000001 MOV #1,R3 ;R3+WCE BIT POSITION
897 016200 040337 107440 BIC R3,BUFTWO-2 ;CHANGE LAST WORD OF BUFFER
898
899 ;PREPARE DEVICE FOR WRITE CHECK DATA OPERATION
900 016204 004737 037316 JSR PC,TSTPRP ;PREPARE DEVICE FOR TEST
      016210 154130 .WORD 154130 ;TASK DESCRIPTOR AS FOLLOWS:
      ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
      ;CLEAR CONTROLLER & SELECT DEVICE
      ;VERIFY CONTROLLER CLEAR OPERATION
      ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
      ;VERIFY PACK ACKNOWLEDGE
      ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
      ;VERIFY RECALIBRATION
      016212 000404 BR 18$ ;GO TO 18$ IF NO ERROR
      016214 000240 NOP ;RETURN HERE IF ERROR
      016216 104000 EMT ;ERROR # DEFINED BY TSTPRP SUBROUTINE
      016220 000137 016630 JMP 29$ ;GO TO 29$ IF ERROR
901 016224 18$:
  
```

```

902
903 ;READ DATA FROM DEVICE
904 016224 012737 000051 001412 19$: MOV #WCD!GO,RMCS10 ;WRITE CHECK DATA DATA COMMAND
905 016224 012702 001553 MOV #PUTINX,R2 ;LOAD PUT REGISTER INDEX TABLE
906 016232 112722 000006 MOV #RMDA,(R2)+
907 016236 112722 000032 MOV #RMOF,(R2)+
908 016242 112722 000032 MOV #RMDC,(R2)+
909 016246 112722 000034 MOV #RMB3,(R2)+
910 016252 112722 000004 MOV #RMWC,(R2)+
911 016256 112722 000002 MOV #RMCS1,(R2)+
912 016262 112722 000000 MOV #200,(R2)
913 016266 112712 000200
914
915 016272 004737 043330 JSR PC,PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
    016276 000404 BR 20$ ;GO TO 20$ IF NO ERROR
    016300 000240 NOP ;RETURN HERE IF ERROR
    016302 104000 EMT ;ERROR # DEFINED BY PUT SUBROUTINE
    016304 000137 016630 JMP 29$ ;GO TO 29$ IF ERROR
916 016310
917
918 016310 004737 042774 ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
    JSR PC,GETSTS ;GO TO GETSTS SUBROUTINE
919
920 016314 004737 043672 ;WAIT FOR COMMAND TO COMPLETE
    JSR PC,TIMOUT ;GO TO TIMEOUT SUBROUTINE
921
922 ;GO READ STATUS FOR WRITE CHECK DATA OPERATION
923 016320 004737 043060 JSR PC,GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
    016324 000404 BR 21$ ;GO TO 21$ IF NO ERROR
    016326 000240 NOP ;RETURN HERE IF ERROR
    016330 104000 EMT ;ERROR # DEFINED BY GET SUBROUTINE
    016332 000137 016630 JMP 29$ ;GO TO 29$ IF ERROR
924 016336
925
926 ;CHECK FOR ERRORS DURING WRITE CHECK DATA OPERATION
927 016336 004737 044056 JSR PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
    016342 000405 BR 22$ ;GO TO 22$ IF NO ERROR
    016344 000240 NOP ;RETURN HERE IF ERROR
    016346 104000 EMT ;ERROR # DEFINED BY PRIERR SUBROUTINE
    016350 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
    016352 000137 016630 JMP 29$ ;GO TO 29$ IF ERROR
928 016356
929 016356 032737 040000 001346 22$: BIT #WCE,RMCS2I ;WAS 'WCE' DETECTED??
    016364 001030 BNE 24$ ;YES!!
930 016366 004737 056572 JSR PC,DASTS ;GO VERIFY RESULTS OF DATA TRANSFER
    016372 000405 BR 23$ ;GO TO 23$ IF NO ERROR
    016374 000240 NOP ;RETURN HERE IF ERROR
    016376 104000 EMT ;ERROR # DEFINED BY DASTS SUBROUTINE
    016400 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
    016402 000137 016630 JMP 29$ ;GO TO 29$ IF ERROR
932 016406
933 016406 013737 001346 001140 23$: MOV RMCS2I,$GDDAT ;EXPECTED STATUS
934 016414 052737 040000 001140 BIS #WCE,$GDDAT
935 016422 013737 001346 001142 MOV RMCS2I,$BDDAT ;RECEIVED STATUS
936 016430 010337 001174 MOV R3,$TMP0 ;FAILING BIT POSITION
937 016434 012737 107440 001176 MOV #BUFTWO-2,$TMP1 ;FAILING ADDRESS
938 016442 104337 EMT 337
  
```

```

939 016444 000471 BR 29$
940
941 016446 24$:
942 016446 112737 000022 001524 MOV #RMDB,GETINX ;SETUP GET INDEX TABLE
    016454 112737 000200 001525 MOV #200,GETINX+1 ;SETUP TERMINATOR BYTE
    016462 012737 016630 001360 MOV #29$,RMDB ;SET RMDB INPUT BUFFER = 29$
    016470 004737 043060 JSR PC,GET ;GO READ RMDB VIA GET SUBROUTINE
    016474 000402 BR 25$ ;GO TO 25$ IF NO ERROR
    016476 000240 NOP ;RETURN HERE IF ERROR
    016500 104000 EMT ;ERROR DEFINED BY GET SUBROUTINE
943
944 016502 25$:
945 016502 013737 001360 001142 MOV RMDBI,$BDDAT ;RECEIVED DATA
946 016510 013737 107440 001140 MOV BUFTWO-2,$GDDAT ;EXPECTED DATA
947 016516 050337 001140 BIS R3,$GDDAT
948 016522 012737 107440 001134 MOV #BUFTWO-2,$GDADR ;EXPECTED ADDRESS
949 016530 013737 001342 001136 MOV RMBAI,$BDADR ;RECEIVED ADDRESS
950 016536 162737 000002 001136 SUB #2,$BDADR ;CORRECT MEMORY ADDRESS
951 016544 023737 001134 001136 CMP $GDADR,$BDADR ;ADDRESSES OK??
952 016552 001402 BEQ 26$ ;YES!!
953 016554 104340 EMT 340
954 016556 000424 BR 29$
955 016560 023737 001140 001142 26$: CMP $GDDAT,$BDDAT ;DATA OK??
956 016566 001402 BEQ 27$ ;YES!
957 016570 104341 EMT 341
958 016572 000416 BR 29$
959 016574 27$:
960
961 016574 004737 044710 JSR PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
    016600 000405 BR 28$ ;GO TO 28$ IF NO ERROR
    016602 000240 NOP ;RETURN HERE IF ERROR
    016604 104000 EMT ;ERROR # DEFINED BY SECERR SUBROUTINE
    016606 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
    016610 000137 016630 JMP 29$ ;GO TO 29$ IF ERROR
962 016614 28$:
963 016614 050337 107440 BIS R3,BUFTWO-2 ;RESTORE DATA PATTERN
964 016620 006303 ASL R3 ;SHIFT TO NEXT BIT
965 016622 001402 BEQ 29$ ;EXIT IF DONE
966 016624 000137 016200 JMP 17$ ;REPEAT TEST FOR NEXT DATA BIT
967 016630 29$:
968
969
;*****
;*TEST 10 WRITE, WRITE CHECK MULTIPLE SECTORS
;*****
TST10:
    016630 000004 SCOPE ;SCOPE CALL
    016632 000240 NOP ;START OF TEST
    016634 012706 001100 MOV #STACK,SP ;INITIALIZE STACK POINTER
    016640 013700 001276 MOV $BASE,R0 ;R0 = UNIBUS ADDRESS
    016644 013701 001466 MOV TSTQUE,R1 ;(R1) = DEVICE BEING TESTED
    016650 012737 000010 001226 MOV #10,$TESTN ;SET TEST NUMBER IN APT MAIL BOX
970
971
972
973
974 016656 1$:
975
;*****
;LOOP #1 FORMAT,WRITE,WRITE CHECK
;*****

```



```

976 ;PREPARE THE DEVICE FOR FORMAT OPERATION
977 016656 004737 037316 JSR PC,TSTPRP ;PREPARE DEVICE FOR TEST
    016662 154130 .WORD 154130 ;TASK DESCRIPTOR AS FOLLOWS:
                                ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
                                ;CLEAR CONTROLLER & SELECT DEVICE
                                ;VERIFY CONTROLLER CLEAR OPERATION
                                ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                ;VERIFY PACK ACKNOWLEDGE
                                ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
                                ;VERIFY RECALIBRATION
                                ;GO TO 2$ IF NO ERROR
                                ;RETURN HERE IF ERROR
                                ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                ;GO TO 24$ IF ERROR

    016664 000404 BR 2$
    016666 000240 NOP
    016670 104000 EMT
    016672 000137 017634 JMP 24$
978 016676 2$:
979
980 ;LOAD PARAMETERS AND GENERATE DATA BUFFER
981 016676 012737 000000 001446 MOV #0,RMDCO ;CYLINDER = 0
982 016704 012737 000000 001420 MOV #0,RMDAO ;TRACK = 0, SECTOR - 0
983 016712 012737 106436 001416 MOV #BUFONE,RMBAO ;BUS ADDRESS
984 016720 012737 176774 001414 MOV #-258.*2,RMWCO ;WORD COUNT FOR 2 SECTORS (2'S COMP)
985 016726 012737 010000 001444 MOV #FMT16,RMOFO ;16 BIT FORMAT
986 016734 012737 000063 001412 MOV #WH!GO,RMCSTO ;WRITE HEADER AND DATA COMMAND
987
988 ;VERIFY THAT SECTOR IS NOT BAD
    016742 004737 040242 JSR PC,BADSCT ;CALL BAD SECTOR MODULE
    016746 000405 BR 3$ ;GO TO 3$ IF NO ERROR
    016750 104401 070426 TYPE ,SCTMSG ;TYPE BAD SECTOR MESSAGE
    016754 104000 EMT ;ERROR # DEFINED BY BADSCT SUBROUTINE
    016756 000137 017634 JMP 24$ ;GO TO 24$ IF ERROR
989 016762 3$:
990 016762 012737 072060 001174 MOV #ZEROS,$TMP0 ;STARTING ADDRESS OF PATTERN
991 016770 012737 000001 001176 MOV #1,$TMP1 ;RANGE OF PATTERN
992 016776 004737 042174 JSR PC,GENBUF ;GO GENERATE BUFFER FOR FORMAT
993
994 ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATION
995 017002 012702 001553 MOV #PUTINX,R2 ;R2 = BYTE ENTRY POSITION
996 017006 112722 000034 MOVB #RMDC,(R2)+
997 017012 112722 000006 MOVB #RMDA,(R2)+
998 017016 112722 000004 MOVB #RMBA,(R2)+
999 017022 112722 000002 MOVB #RMWC,(R2)+
1000 017026 112722 000032 MOVB #RMOF,(R2)+
1001 017032 112722 000000 MOVB #RMCS1,(R2)+
1002 017036 112712 000200 MOVB #200(R2) ;TERMINATE TABLE
1003 017042 4$:
1004
1005 ;FORMAT THE DRIVE
1006 017042 004737 043330 JSR PC,PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
    017046 000404 BR 5$ ;GO TO 5$ IF NO ERROR
    017050 000240 NOP ;RETURN HERE IF ERROR
    017052 104000 EMT ;ERROR # DEFINED BY PUT SUBROUTINE
    017054 000137 017634 JMP 24$ ;GO TO 24$ IF ERROR
1007 017060 5$:
1008
1009 ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
    017060 004737 042774 JSR PC,GETSTS ;GO TO GETSTS SUBROUTINE
1010

```

```

1011      ;WAIT FOR COMMAND TO COMPLETE
1012      JSR      PC,TIMOUT      ;GO TO TIMOUT SUBROUTINE
1013      ;GO READ STATUS FOR FORMAT OPERATION
1014      JSR      PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
      BR      6$      ;GO TO 6$ IF NO ERROR
      NOP      ;RETURN HERE IF ERROR
      EMT      ;ERROR # DEFINED BY GET SUBROUTINE
      JMP      24$      ;GO TO 24$ IF ERROR
1015      6$:
1016      ;VERIFY NO ERRORS DURING FORMAT
1017      JSR      PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
      BR      7$      ;GO TO 7$ IF NO ERROR
      NOP      ;RETURN HERE IF ERROR
      EMT      ;ERROR # DEFINED BY DTASTS SUBROUTINE
      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      JMP      24$      ;GO TO 24$ IF ERROR
1018      7$:
1019      ;MOVE LOOP ADDRESSES TO NEXT OPERATION
1020      MOV      #8$,$LPERR
1021      ;REGENERATE DATA BUFFER
1022      MOV      #-256.*2,RMWCO ;CHANGE WORD COUNT
1023      MOV      #BUFONE,RMBAO ;CHANGE BUS ADDRESS
1024      MOV      #WD,RMCS10 ;WRITE DATA
1025      MOV      #ONES,$TMP0 ;STARTING ADDRESS OF PATTERN
1026      MOV      #1,$TMP1 ;RANGE OF PATTERN
1027      JSR      PC,GENBUF
1028      BR      9$      ;SKIP TO WRITE OPERATION
1029      ;*****
1030      ;LOOP #2      WRITE,WRITE CHECK
1031      8$:
1032      ;PREPARE DEVICE FOR WRITE OPERATION
1033      JSR      PC,TSTPRP ;PREPARE DEVICE FOR TEST
      .WORD      154130 ;TASK DESCRIPTOR AS FOLLOWS:
1034      ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
1035      ;CLEAR CONTROLLER & SELECT DEVICE
1036      ;VERIFY CONTROLLER CLEAR OPERATION
1037      ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
1038      ;VERIFY PACK ACKNOWLEDGE
1039      ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
1040      ;VERIFY RECALIBRATION
      BR      9$      ;GO TO 9$ IF NO ERROR
      NOP      ;RETURN HERE IF ERROR
      EMT      ;ERROR # DEFINED BY TSTPRP SUBROUTINE
      JMP      24$      ;GO TO 24$ IF ERROR
1041      9$:
1042      ;WRITE DATA TO THE DRIVE
1043      10$:
1044      MOV      #WD!GO,RMCS10 ;WRITE DATA COMMAND
1045
      017064 004737 043672
      017070 004737 043060
      017074 000404
      017076 000240
      017100 104000
      017102 000137 017634
      017106
      017106 004737 056572
      017112 000405
      017114 000240
      017116 104000
      017120 004736
      017122 000137 017634
      017126
      017126 012737 017200 001124
      017134 012737 177000 001414
      017142 012737 106436 001416
      017150 012737 000060 001412
      017156 012737 072016 001174
      017164 012737 000001 001176
      017172 004737 042174
      017176 000410
      017200
      017200 004737 037316
      017204 154130
      017206 000404
      017210 000240
      017212 104000
      017214 000137 017634
      017220
      017220 012737 000061 001412
  
```


1046	017226	012702	001553	MOV	#PUTINX,R2	;LOAD PUT REGISTER INDEX TABLE
1047	017232	112722	000006	MOVB	#RMDA,(R2)+	
1048	017236	112722	000034	MOVB	#RMDC,(R2)+	
1049	017242	112722	000032	MOVB	#RMOF,(R2)+	
1050	017246	112722	000004	MOVB	#RMBA,(R2)+	
1051	017252	112722	000002	MOVB	#RMWC,(R2)+	
1052	017256	112722	000000	MOVB	#RMCS1,(R2)+	
1053	017262	112722	000200	MOVB	#200,(R2)+	;TERMINATE TABLE
1054						
1055	017266	004737	043330	JSR	PC,PUT	;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
	017272	000404		BR	11\$	;GO TO 11\$ IF NO ERROR
	017274	000240		NOP		;RETURN HERE IF ERROR
	017276	104000		EMT		;ERROR # DEFINED BY PUT SUBROUTINE
	017300	000137	017634	JMP	24\$	;GO TO 24\$ IF ERROR
1056	017304					
1057						
1058						
	017304	004737	042774	JSR	PC,GETSTS	;GO TO GETSTS SUBROUTINE
1059						
1060						
	017310	004737	043672	JSR	PC,TIMOUT	;GO TO TIMOUT SUBROUTINE
1061						
1062						
1063	017314	004737	043060	JSR	PC,GET	;GO READ REGISTER(S) WITH GET SUBROUTINE
	017320	000404		BR	12\$	;GO TO 12\$ IF NO ERROR
	017322	000240		NOP		;RETURN HERE IF ERROR
	017324	104000		EMT		;ERROR # DEFINED BY GET SUBROUTINE
	017326	000137	017634	JMP	24\$	;GO TO 24\$ IF ERROR
1064	017332					
1065						
1066						
1067	017332	004737	044056	JSR	PC,PRIERR	;GO CHECK FOR PRIMARY ERRORS
	017336	000405		BR	13\$	;GO TO 13\$ IF NO ERROR
	017340	000240		NOP		;RETURN HERE IF ERROR
	017342	104000		FMT		;ERROR # DEFINED BY PRIERR SUBROUTINE
	017344	004736		JSR	PC,@(SP)+	;GO BACK FOR MORE ERROR CHECKS
	017346	000137	017634	JMP	24\$	;GO TO 24\$ IF ERROR
1068	017352					
1069	017352	004737	056572	JSR	PC,DTASTS	;GO VERIFY RESULTS OF DATA TRANSFER
	017356	000405		BR	14\$	;GO TO 14\$ IF NO ERROR
	017360	000240		NOP		;RETURN HERE IF ERROR
	017362	104000		EMT		;ERROR # DEFINED BY DTASTS SUBROUTINE
	017364	004736		JSR	PC,@(SP)+	;GO BACK FOR MORE ERROR CHECKS
	017366	000137	017634	JMP	24\$	;GO TO 24\$ IF ERROR
1070	017372					
1071	017372	004737	044710	JSR	PC,SECERR	;GO CHECK FOR SECONDARY ERRORS
	017376	000405		BR	15\$	;GO TO 15\$ IF NO ERROR
	017400	000240		NOP		;RETURN HERE IF ERROR
	017402	104000		EMT		;ERROR # DEFINED BY SECERR SUBROUTINE
	017404	004736		JSR	PC,@(SP)+	;GO BACK FOR MORE ERROR CHECKS
	017406	000137	017634	JMP	24\$	;GO TO 24\$ IF ERROR
1072	017412					
1073						
1074						
1075	017412	012737	017422 001124	MOV	#16\$,\$LPERR	
1076	017420	000410		BR	17\$	;SKIP TO NEXT OPERATION
1077						

```

1078 ;*****
1079 ;LOOP #3      WRITE CHECK
1080
1081 017422      16$:
1082
1083 ;PREPARE DEVICE FOR READ OPERATION
1084 017422 004737 037316      JSR PC,TSTPRP      ;PREPARE DEVICE FOR TEST
      017426 154130      .WORD 154130      ;TASK DESCRIPTOR AS FOLLOWS:
      ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
      ;CLEAR CONTROLLER & SELECT DEVICE
      ;VERIFY CONTROLLER CLEAR OPERATION
      ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
      ;VERIFY PACK ACKNOWLEDGE
      ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
      ;VERIFY RECALIBRATION
      ;GO TO 17$ IF NO ERROR
      ;RETURN HERE IF ERROR
      ;ERROR # DEFINED BY TSTPRP SUBROUTINE
      ;GO TO 24$ IF ERROR
      017430 000404      BR      17$
      017432 000240      NOP
      017434 104000      EMT
      017436 000137 017634      JMP      24$
1085 017442      17$:
1086
1087 ;READ DATA FROM DEVICE
1088 017442      18$:
1089 017442 012737 000051 001412      MOV      #WCD!GO,RMCS10      ;READ DATA COMMAND
1090 017450 012702 001553      MOV      #PUTINX,R2      ;LOAD PUT REGISTER INDEX TABLE
1091 017454 112722 000006      MOVB     #RMDA,(R2)+
1092 017460 112722 000032      MOVB     #RMOF,(R2)+
1093 017464 112722 000034      MOVB     #RMDC,(R2)+
1094 017470 112722 000004      MOVB     #RMB A,(R2)+
1095 017474 112722 000002      MOVB     #RMWC,(R2)+
1096 017500 112722 000000      MOVB     #RMCS1,(R2)+
1097 017504 112712 000200      MOVB     #200,(R2)
1098
1099 017510 004737 043330      JSR      PC,PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      017514 000404      BR      19$      ;GO TO 19$ IF NO ERROR
      017516 000240      NOP      ;RETURN HERE IF ERROR
      017520 104000      EMT      ;ERROR # DEFINED BY PUT SUBROUTINE
      017522 000137 017634      JMP      24$      ;GO TO 24$ IF ERROR
1100 017526      19$:
1101
1102 ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
      017526 004737 042774      JSR      PC,GETSTS      ;GO TO GETSTS SUBROUTINE
1103
1104 ;WAIT FOR COMMAND TO COMPLETE
      017532 004737 043672      JSR      PC,TIMOUT      ;GO TO TIMOUT SUBROUTINE
1105
1106 ;GO READ STATUS FOR READ OPERATION
1107 017536 004737 043060      JSR      PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      017542 000404      BR      20$      ;GO TO 20$ IF NO ERROR
      017544 000240      NOP      ;RETURN HERE IF ERROR
      017546 104000      EMT      ;ERROR # DEFINED BY GET SUBROUTINE
      017550 000137 017634      JMP      24$      ;GO TO 24$ IF ERROR
1108 017554      20$:
1109
1110 ;CHECK FOR ERRORS DURING READ OPERATION
1111 017554 004737 044056      JSR      PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
      017560 000405      BR      21$      ;GO TO 21$ IF NO ERROR

```

```

017562 000240      NOP      ;RETURN HERE IF ERROR
017564 104000      EMT      ;ERROR # DEFINED BY TSTPRP SUBROUTINE
017566 004736      JSR      ;GO BACK FOR MORE ERROR CHECKS
017570 000137 017634 JMP      24$ ;GO TO 24$ IF ERROR

1112 017574      JSR      ;GO VERIFY RESULTS OF DATA TRANSFER
1113 017574 004737 056572 BR      22$ ;GO TO 22$ IF NO ERROR
017600 000405      NOP      ;RETURN HERE IF ERROR
017602 000240      EMT      ;ERROR # DEFINED BY DTASTS SUBROUTINE
017604 104000      JSR      ;GO BACK FOR MORE ERROR CHECKS
017606 004736      JMP      24$ ;GO TO 24$ IF ERROR
1114 017610 000137 017634
1115 017614 004737 044710      JSR      ;GO CHECK FOR SECONDARY ERRORS
017620 000405      BR      23$ ;GO TO 23$ IF NO ERROR
017622 000240      NOP      ;RETURN HERE IF ERROR
017624 104000      EMT      ;ERROR # DEFINED BY SECERR SUBROUTINE
017626 004736      JSR      ;GO BACK FOR MORE ERROR CHECKS
017630 000137 017634 JMP      24$ ;GO TO 24$ IF ERROR

1116 017634      JSR      ;GO CHECK FOR SECONDARY ERRORS
1117      BR      23$ ;GO TO 23$ IF NO ERROR
1118 017634      NOP      ;RETURN HERE IF ERROR
1119      EMT      ;ERROR # DEFINED BY SECERR SUBROUTINE
1120      JSR      ;GO BACK FOR MORE ERROR CHECKS
1120      JMP      24$ ;GO TO 24$ IF ERROR

;*****
;*TEST 11      WRITE, READ W/ IMPLIED SEEK
;*****
TST11:
017634      SCOPE      ;SCOPE CALL
017634 000004      NOP      ;START OF TEST
017636 000240      MOV      #STACK,SP ;INITIALIZE STACK POINTER
017640 012706 001100 MOV      $BASE,R0 ;R0 = UNIBUS ADDRESS
017644 013700 001276 MOV      TSTQUE,R1 ;(R1) = DEVICE BEING TESTED
017650 013701 001466 MOV      #11,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
017654 012737 000011 001226

1121
1122
1123 ;*****
1124 ;LOOP #1      FORMAT,WRITE,READ
1125 017662      1$:
1126
1127 ;PREPARE THE DEVICE FOR FORMAT OPERATION
1128 017662 004737 037316 JSR      PC,TSTPRP ;PREPARE DEVICE FOR TEST
017666 154130      .WORD 154130 ;TASK DESCRIPTOR AS FOLLOWS:
;SELECT DEVICE & VERIFY DEVICE AVAILABLE
;CLEAR CONTROLLER & SELECT DEVICE
;VERIFY CONTROLLER CLEAR OPERATION
;PACK ACKNOWLEDGE IF VOLUME NOT VALID
;VERIFY PACK ACKNOWLEDGE
;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
;VERIFY RECALIBRATION
017670 000404      BR      2$ ;GO TO 2$ IF NO ERROR
017672 000240      NOP      ;RETURN HERE IF ERROR
017674 104000      EMT      ;ERROR # DEFINED BY TSTPRP SUBROUTINE
017676 000137 021054 JMP      31$ ;GO TO 31$ IF ERROR

1129
1130 ;LOAD PARAMETERS AND GENERATE DATA BUFFER
1131 017702      2$:
1132 017702 012737 000000 001446 MOV      #0,RMDCO ;CYLINDER = 0
1133 017710 012737 000000 001420 MOV      #0,RMDAO ;TRACK = 0, SECTOR = 0

```

```

1134 017716 013737 001446 021056      MOV      RMDCO,32$      ;SAVE DESIRED CYLINDER
1135 017724 012737 106436 001416      MOV      #BUFOE,RMBAO    ;BUS ADDRESS
1136 017732 012737 177376 001414      MOV      #-256,RMWCO    ;2 + 256. WORDS (2'S COMP)
1137 017740 012737 010000 001444      MOV      #FMT16,RMOFU   ;16 BIT FORMAT
1138 017746 012737 000063 001412      MOV      #WH!GO,RMCS10  ;WRITE HEADER AND DATA COMMAND
1139
1140      ;VERIFY THAT SECTOR IS NOT BAD
      JSR      PC,BADSCT    ;CALL BAD SECTOR MODULE
      BR       3$          ;GO TO 3$ IF NO ERROR
      TYPE     ,SLTMSG      ;TYPE BAD SECTOR MESSAGE
      EMT      ;ERROR # DEFINED BY BADSCT SUBROUTINE
      JMP      31$         ;GO TO 31$ IF ERROR
1141 017774      3$:      MOV      #ONES,$TMP0      ;STARTING ADDRESS OF PATTERN
1142 017774 012737 072016 001174      MOV      #1,$TMP1      ;RANGE OF PATTERN
1143 020002 012737 000001 001176      JSR      PC,GENBUF      ;GO GENERATE BUFFER FOR FORMAT
1144 020010 004737 042174
1145
1146      ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATION
      MOV      #PUTINX,R2    ;R2 = BYTE ENTRY POSITION
1147 020014 012702 001553      MOV      #RMDC,(R2)+
1148 020020 112722 000034      MOV      #RMDA,(R2)+
1149 020024 112722 000006      MOV      #RMDA,(R2)+
1150 020030 112722 000004      MOV      #RMDA,(R2)+
1151 020034 112722 000002      MOV      #RMDA,(R2)+
1152 020040 112722 000032      MOV      #RMDA,(R2)+
1153 020044 112722 000000      MOV      #RMDA,(R2)+
1154 020050 112712 000200      MOV      #200,(R2)    ;TERMINATE TABLE
1155 020054
1156
1157      ;FORMAT THE DRIVE
      JSR      PC,PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
1158 020054 004737 043330      BR       5$          ;GO TO 5$ IF NO ERROR
      NOP      ;RETURN HERE IF ERROR
      EMT      ;ERROR # DEFINED BY PUT SUBROUTINE
      JMP      31$         ;GO TO 31$ IF ERROR
1159 020072      5$:
1160
1161      ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
      JSR      PC,GETSTS    ;GO TO GETSTS SUBROUTINE
1162 020072 004737 042774
1163
1164      ;WAIT FOR COMMAND TO COMPLETE
      JSR      PC,TIMOUT    ;GO TO TIMEOUT SUBROUTINE
1165 020076 004737 043672
1166
1167      ;GO READ STATUS FOR FORMAT OPERATION
      JSR      PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
1168 020102 004737 043060      BR       6$          ;GO TO 6$ IF NO ERROR
      NOP      ;RETURN HERE IF ERROR
      EMT      ;ERROR # DEFINED BY GET SUBROUTINE
      JMP      31$         ;GO TO 31$ IF ERROR
1169 020110 000404
1170 020112 104000
1171 020114 000137 021054      6$:
1172
1173      ;VERIFY NO ERRORS DURING FORMAT
      JSR      PC,DASTS     ;GO VERIFY RESULTS OF DATA TRANSFER
1174 020120 004737 056572      BR       7$          ;GO TO 7$ IF NO ERROR
      NOP      ;RETURN HERE IF ERROR
      EMT      ;ERROR # DEFINED BY DASTS SUBROUTINE
      JSR      PC,@(SP)+    ;GO BACK FOR MORE ERROR CHECKS
      JMP      31$         ;GO TO 31$ IF ERROR
1175 020124 000405
1176 020126 000240
1177 020130 104000
1178 020132 004736
1179 020134 000137 021054

```

```

1171 020140      7$:
1172
1173      ;MOVE LOOP ADDRESSES TO NEXT OPERATION
1174 020140 012737 020150 001124      MOV      #8$,SLPERR
1175 020146 000410      BR      9$      ;SKIP TO WRITE OPERATION
1176
1177      ;*****
1178      ;LOOP #2      WRITE,READ
1179
1180 020150      8$:
1181
1182      ;PREPARE DEVICE FOR WRITE OPERATION
1183 020150 004737 037316      JSR      PC,TSTPRP      ;PREPARE DEVICE FOR TEST
      020154 154130      .WORD      154130      ;TASK DESCRIPTOR AS FOLLOWS:
      ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
      ;CLEAR CONTROLLER & SELECT DEVICE
      ;VERIFY CONTROLLER CLEAR OPERATION
      ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
      ;VERIFY PACK ACKNOWLEDGE
      ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
      ;VERIFY RECALIBRATION
      ;GO TO 9$ IF NO ERROR
      ;RETURN HERE IF ERROR
      ;ERROR # DEFINED BY TSTPRP SUBROUTINE
      ;GO TO 31$ IF ERROR
      BR      9$
      NOP
      EMT
      JMP      31$
1184 020170      9$:
1185 020170 012737 001466 001446      MOV      #822.,RMDCO      ;SEEK TO LAST CYLINDER
1186 020176 012737 000005 001412      MOV      #SEEK!GO,RMCS10      ;WRITE SEEK COMMAND
1187
1188 020204 004737 043330      JSR      PC,PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      020210 000404      BR      10$      ;GO TO 10$ IF NO ERROR
      020212 000240      NOP      ;RETURN HERE IF ERROR
      020214 104000      EMT      ;ERROR # DEFINED BY PUT SUBROUTINE
      020216 000137 021054      JMP      31$      ;GO TO 31$ IF ERROR
1189 020222
1190
1191      ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
      JSR      PC,GETSTS      ;GO TO GETSTS SUBROUTINE
1192 020222 004737 042774
1193      ;WAIT FOR COMMAND TO COMPLETE
      JSR      PC,TIMOUT      ;GO TO TIMOUT SUBROUTINE
1194 020226 004737 043672
1195
1196      ;GO GET REGISTER STATUS
      JSR      PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      020236 000404      BR      11$      ;GO TO 11$ IF NO ERROR
      020240 000240      NOP      ;RETURN HERE IF ERROR
      020242 104000      EMT      ;ERROR # DEFINED BY GET SUBROUTINE
      020244 000137 021054      JMP      31$      ;GO TO 31$ IF ERROR
1197 020250      11$:
1198
1199      ;VERIFY RESULTS OF SEEK COMMAND
      JSR      PC,SEKSTS      ;GO VERIFY RESULTS OF SEEK OPERATION
      020254 000405      BR      12$      ;GO TO 12$ IF NO ERROR
      020256 000240      NOP      ;RETURN HERE IF ERROR
      020260 104000      EMT      ;ERROR # DEFINED BY SEKSTS SUBROUTINE
      020262 004736      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      020264 000137 021054      JMP      31$      ;GO TO 31$ IF ERROR
  
```

```

1201 020270      12$:
1202
1203      ;SETUP PARAMETERS AND EXECUTE WRITE DATA COMMAND
1204 020270      13$:
1205 020270 013737 021056 001446      MOV      32$,RMDCO      ;RESTORE CYLINDER
1206 020276 012737 177400 001414      MOV      #-256.,RMWCO      ;CHANGE WORD COUNT
1207 020304 012737 106442 001416      MOV      #BUFONE+4,RMBAO      ;CHANGE BUS ADDRESS
1208 020312 012737 000061 001412      MOV      #WD!GO,RMCS10      ;WRITE DATA COMMAND
1209 020320 012702 001553      MOV      #PUTINX,R2      ;LOAD PUT REGISTER INDEX TABLE
1210 020324 112722 000006      MOV      #RMDA,(R2)+
1211 020330 112722 000034      MOV      #RMDC,(R2)+
1212 020334 112722 000032      MOV      #RMOF,(R2)+
1213 020340 112722 000004      MOV      #RMBA,(R2)+
1214 020344 112722 000002      MOV      #RMWC,(R2)+
1215 020350 112722 000000      MOV      #RMCS1,(R2)+
1216 020354 112722 000200      MOV      #200,(R2)+      ;TERMINATE TABLE
1217
1218 020360 004737 043330      JSR      PC,PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      020364 000404      BR      14$      ;GO TO 14$ IF NO ERROR
      020366 000240      NOP      ;RETURN HERE IF ERROR
      020370 104000      EMT      ;ERROR # DEFINED BY PUT SUBROUTINE
      020372 000137 021054      JMP      31$      ;GO TO 31$ IF ERROR
1219 020376
1220
1221      ;WAIT FOR COMMAND TO COMPLETE
      020376 004737 043672      JSR      PC,TIMOUT      ;GO TO TIMEOUT SUBROUTINE
1222
1223      ;GO READ STATUS FOR WRITE COMMAND
1224 020402 004737 043060      JSR      PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      020406 000404      BR      15$      ;GO TO 15$ IF NO ERROR
      020410 000240      NOP      ;RETURN HERE IF ERROR
      020412 104000      EMT      ;ERROR # DEFINED BY GET SUBROUTINE
      020414 000137 021054      JMP      31$      ;GO TO 31$ IF ERROR
1225 020420
1226
1227      ;CHECK FOR ERRORS DURING WRITE OPERATION
1228 020420 004737 044056      JSR      PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
      020424 000405      BR      16$      ;GO TO 16$ IF NO ERROR
      020426 000240      NOP      ;RETURN HERE IF ERROR
      020430 104000      EMT      ;ERROR # DEFINED BY PRIERR SUBROUTINE
      020432 004736      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      020434 000137 021054      JMP      31$      ;GO TO 31$ IF ERROR
1229 020440
1230 020440 004737 056572      JSR      PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
      020444 000405      BR      17$      ;GO TO 17$ IF NO ERROR
      020446 000240      NOP      ;RETURN HERE IF ERROR
      020450 104000      EMT      ;ERROR # DEFINED BY DTASTS SUBROUTINE
      020452 004736      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      020454 000137 021054      JMP      31$      ;GO TO 31$ IF ERROR
1231 020460
1232 020460 004737 044710      JSR      PC,SECERR      ;GO CHECK FOR SECONDARY ERRORS
      020464 000405      BR      18$      ;GO TO 18$ IF NO ERROR
      020466 000240      NOP      ;RETURN HERE IF ERROR
      020470 104000      EMT      ;ERROR # DEFINED BY SECERR SUBROUTINE
      020472 004736      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      020474 000137 021054      JMP      31$      ;GO TO 31$ IF ERROR
1233 020500      18$:
  
```



```

1234
1235
1236 020500 012737 020510 001124 ;CHANGE LOOP ADDRESSES
1237 020506 000410 MOV #19$,SLPERR
1238 ;SKIP TO NEXT OPERATION
1239
1240 ;*****
1241 ;LOOP #3 READ
1242 020510 19$:
1243
1244 ;PREPARE DEVICE FOR READ OPERATION
1245 020510 004737 037316 JSR PC,TSTPRP ;PREPARE DEVICE FOR TEST
020514 154130 .WORD 154130 ;TASK DESCRIPTOR AS FOLLOWS:
;SELECT DEVICE & VERIFY DEVICE AVAILABLE
;CLEAR CONTROLLER & SELECT DEVICE
;VERIFY CONTROLLER CLEAR OPERATION
;PACK ACKNOWLEDGE IF VOLUME NOT VALID
;VERIFY PACK ACKNOWLEDGE
;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
;VERIFY RECALIBRATION
;GO TO 20$ IF NO ERROR
;RETURN HERE IF ERROR
;ERROR # DEFINED BY TSTPRP SUBROUTINE
;GO TO 31$ IF ERROR

020516 000404 BR 20$
020520 000240 NOP
020522 104000 EMT
020524 000137 021054 JMP 31$
1246 020530 20$:
1247 020530 012737 001466 001446 MOV #822.,RMDCO ;SEEK TO LAST CYLINDER
1248 020536 012737 000005 001412 MOV #SEEK!GO,RMCS10 ;WRITE SEEK COMMAND
1249
1250 020544 004737 043330 JSR PC,PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
020550 000404 BR 21$ ;GO TO 21$ IF NO ERROR
020552 000240 NOP ;RETURN HERE IF ERROR
020554 104000 EMT ;ERROR # DEFINED BY PUT SUBROUTINE
020556 000137 021054 JMP 31$ ;GO TO 31$ IF ERROR
1251 020562 21$:
1252
1253 ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
020562 004737 042774 JSR PC,GETSTS ;GO TO GETSTS SUBROUTINE
1254
1255 ;WAIT FOR COMMAND TO COMPLETE
020566 004737 043672 JSR PC,TIMOUT ;GO TO TIMEOUT SUBROUTINE
1256
1257 ;GO GET REGISTER STATUS
1258 020572 004737 043060 JSR PC,GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
020576 000404 BR 22$ ;GO TO 22$ IF NO ERROR
020600 000240 NOP ;RETURN HERE IF ERROR
020602 104000 EMT ;ERROR # DEFINED BY GET SUBROUTINE
020604 000137 021054 JMP 31$ ;GO TO 31$ IF ERROR
1259 020610 22$:
1260
1261 ;VERIFY RESULTS OF SEEK COMMAND
1262 020610 004737 051174 JSR PC,SEKSTS ;GC VERIFY RESULTS OF SEEK OPERATION
020614 000405 BR 23$ ;GO TO 23$ IF NO ERROR
020616 000240 NOP ;RETURN HERE IF ERROR
020620 104000 EMT ;ERROR # DEFINED BY SEKSTS SUBROUTINE
020622 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
020624 000137 021054 JMP 31$ ;GO TO 31$ IF ERROR
1263 020630 23$:

```

```

1264
1265      ;SETUP PARAMETERS AND EXECUTE READ DATA COMMAND
1266 020630 24$: MOV 32$,RMDCO ;RESTORE CYLINDER
1267 020630 013737 021056 001446 MOV #RD!GO,RMCS10 ;READ DATA COMMAND
1268 020636 012737 000071 001412 MOV #BUFTWO+4,RMBAD ;LOAD STARTING BUFFER ADDRESS
1269 020644 012737 107446 001416 MOV #PUTINX,R2 ;LOAD PUT REGISTER INDEX TABLE
1270 020652 012702 001553 MOVB #RMDA,(R2)+
1271 020656 112722 000006 MOVB #RMOF,(R2)+
1272 020662 112722 000032 MOVB #RMDC,(R2)+
1273 020666 112722 000034 MOVB #RMBB,(R2)+
1274 020672 112722 000004 MOVB #RMWC,(R2)+
1275 020676 112722 000002 MOVB #RMCS1,(R2)+
1276 020702 112722 000000 MOVB #200,(R2)
1277 020706 112712 000200
1278
1279 020712 004737 043330 JSR PC,PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      020716 000404 BR 25$ ;GO TO 25$ IF NO ERROR
      020720 000240 NOP ;RETURN HERE IF ERROR
      020722 104000 EMT ;ERROR # DEFINED BY PUT SUBROUTINE
      020724 000137 021054 JMP 31$ ;GO TO 31$ IF ERROR
1280 020730
1281
1282      ;WAIT FOR COMMAND TO COMPLETE
      020730 004737 043672 JSR PC,TIMOUT ;GO TO TIMEOUT SUBROUTINE
1283
1284      ;GO READ STATUS FOR READ OPERATION
1285 020734 004737 043060 JSR PC,GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
      020740 000404 BR 26$ ;GO TO 26$ IF NO ERROR
      020742 000240 NOP ;RETURN HERE IF ERROR
      020744 104000 EMT ;ERROR # DEFINED BY GET SUBROUTINE
      020746 000137 021054 JMP 31$ ;GO TO 31$ IF ERROR
1286 020752
1287
1288      ;CHECK FOR ERRORS DURING READ OPERATION
1289 020752 004737 044056 JSR PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
      020756 000405 BR 27$ ;GO TO 27$ IF NO ERROR
      020760 000240 NOP ;RETURN HERE IF ERROR
      020762 104000 EMT ;ERROR # DEFINED BY PRIERR SUBROUTINE
      020764 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
      020766 000137 021054 JMP 31$ ;GO TO 31$ IF ERROR
1290 020772
1291 020772 004737 056572 JSR PC,DASTS ;GO VERIFY RESULTS OF DATA TRANSFER
      020776 000405 BR 28$ ;GO TO 28$ IF NO ERROR
      021000 000240 NOP ;RETURN HERE IF ERROR
      021002 104000 EMT ;ERROR # DEFINED BY DASTS SUBROUTINE
      021004 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
      021006 000137 021054 JMP 31$ ;GO TO 31$ IF ERROR
1292 021012
1293 021012 004737 044710 JSR PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
      021016 000405 BR 29$ ;GO TO 29$ IF NO ERROR
      021020 000240 NOP ;RETURN HERE IF ERROR
      021022 104000 EMT ;ERROR # DEFINED BY SECERR SUBROUTINE
      021024 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
      021026 000137 021054 JMP 31$ ;GO TO 31$ IF ERROR
1294 021032
1295 021032 004737 042432 JSR PC,CMPBUF ;GO COMPARE WRITE, READ DATA BUFFERS
      021036 106442 .WORD BUFONE+4 ;STARTING ADDRESS OF WRITE BUFFER
  
```



```

021040 107446      .WORD  BUFTWO+4      ;STARTING ADDRESS OF READ BUFFER
021042 000404      BR      30$          ;GO TO 30$ IF NO ERROR
021044 000240      NOP                    ;RETURN HERE IF ERROR
021046 104000      EMT                    ;ERROR # DEFINED BY CMPBUF SUBROUTINE
021050 000137 021054 JMP      31$          ;GO TO 31$ IF ERROR
1296 021054      30$:
1297
1298 021054      31$:
1299 021054 000401      BR      33$
1300
1301 021056 000000      32$:      .WORD  0
1302
1303 021060      33$:
1304
1305      ;*****
      ;*TEST 12      WRITE, WRITE CHECK W/ HEAD SWITCHING
      ;*****
      TST12:
021060      SCOPE                    ;SCOPE CALL
021062 000004      NOP                    ;START OF TEST
021064 012706 001100  MOV      #STACK,SP  ;INITIALIZE STACK POINTER
021070 013700 001276  MOV      $BASE,R0   ;R0 = UNIBUS ADDRESS
021074 013701 001466  MOV      TSTQUE,R1  ;(R1) = DEVICE BEING TESTED
021100 012737 000012 001226 MOV      #12,$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
1306
1307      ;*****
1308      ;LOOP #1      FORMAT,WRITE,WRITE CHECK
1309
1310      ;LOAD PARAMETERS AND GENERATE DATA BUFFER
1311 021105      1$:
1312 021106 012737 000000 001446  MOV      #0,RMDCO      ;CYLINDER = 0
1313 021114 112737 000000 001421  MOV      #0,RMDAO+1    ;TRACK 0
1314 021122 112737 000037 001420 2$:  MOV      #31,RMDAO    ;SECTOR = 31.
1315 021130 012737 010000 001444  MOV      #FMT16,RMOFO    ;16 BIT FORMAT
1316 021136 012737 106436 001416  MOV      #BUFONE,RMBAO    ;BUS ADDRESS
1317 021144 012737 176774 001414  MOV      #-258.*2,RMWCO    ;WORD COUNT FOR 2 SECTORS (2'S COMP)
1318 021152 012737 000063 001412  MOV      #WH!GO,RMCS10    ;WRITE HEADER AND DATA COMMAND
1319
1320      ;VERIFY THAT SECTOR IS NOT BAD
021160 004737 040242  JSR      PC,BADSCT  ;CALL BAD SECTOR MODULE
021164 000405      BR      3$            ;GO TO 3$ IF NO ERROR
021166 104401 070426  TYPE      ,SCTMSG    ;TYPE BAD SECTOR MESSAGE
021172 104000      EMT                    ;ERROR # DEFINED BY BADSCT SUBROUTINE
021174 000137 022266  JMP      30$        ;GO TO 30$ IF ERROR
1321 021200      3$:
1322 021200 123727 001420 000037  CMPB      RMDAO,#31.    ;IS LAST TRACK ASSIGNED ?
1323 021206 001345      BNE      2$        ;BR IF NO
1324 021210 012737 072016 001174  MOV      #ONES,$TMP0    ;STARTING ADDRESS OF PATTERN
1325 021216 012737 000001 001176  MOV      #1,$TMP1     ;RANGE OF PATTERN
1326 021224 004737 042174      JSR      PC,GENBUF    ;GO GENERATE BUFFER FOR FORMAT
1327
1328      ;PREPARE THE DEVICE FOR FORMAT OPERATION
1329 021230 004737 037316  JSR      PC,TSTPRP  ;PREPARE DEVICE FOR TEST
021234 154130      .WORD  154130    ;TASK DESCRIPTOR AS FOLLOWS:
      ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
      ;CLEAR CONTROLLER & SELECT DEVICE
      ;VERIFY CONTROLLER CLEAR OPERATION

```

```

                                ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                ;VERIFY PACK ACKNOWLEDGE
                                ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
                                ;VERIFY RECALIBRATION
                                ;GO TO 4$ IF NO ERROR
                                ;RETURN HERE IF ERROR
                                ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                ;GO TO 30$ IF ERROR

021236 000404 BR 4$
021240 000240 NOP
021242 104000 EMT
021244 000137 022266 JMP 30$
1330 021250 4$:
1331
1332 ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATION
1333 021250 012702 001553 MOV #PUTINX,R2 ;R2 - BYTE ENTRY POSITION
1334 021254 112722 000034 MOVB #RMDC,(R2)+
1335 021260 112722 000006 MOVB #RMDA,(R2)+
1336 021264 112722 000004 MOVB #RMB A,(R2)+
1337 021270 112722 000002 MOVB #RMWC,(R2)+
1338 021274 112722 000032 MOVB #RMOF,(R2)+
1339 021300 112722 000000 MOVB #RMCS1,(R2)+
1340 021304 112712 000200 MOVB #200,(R2) ;TERMINATE TABLE
1341
1342 ;FORMAT THE DRIVE
1343 021310 004737 043330 JSR PC,PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
021314 000404 BR 5$ ;GO TO 5$ IF NO ERROR
021316 000240 NOP ;RETURN HERE IF ERROR
021320 104000 EMT ;ERROR # DEFINED BY PUT SUBROUTINE
021322 000137 022266 JMP 30$ ;GO TO 30$ IF ERROR
1344 021326 5$:
1345
1346 ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
021326 004737 042774 JSR PC,GETSTS ;GO TO GETSTS SUBROUTINE
1347
1348 ;WAIT FOR COMMAND TO COMPLETE
021332 004737 043672 JSR PC,TIMOUT ;GO TO TIMOUT SUBROUTINE
1349
1350 ;GO READ STATUS FOR FORMAT OPERATION
1351 021336 004737 043060 JSR PC,GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
021342 000404 BR 6$ ;GO TO 6$ IF NO ERROR
021344 000240 NOP ;RETURN HERE IF ERROR
021346 104000 EMT ;ERROR # DEFINED BY GET SUBROUTINE
021350 000137 022266 JMP 30$ ;GO TO 30$ IF ERROR
1352 021354 6$:
1353
1354 ;VERIFY NO ERRORS DURING FORMAT
1355 021354 004737 056572 JSR PC,DASTS ;GO VERIFY RESULTS OF DATA TRANSFER
021360 000405 BR 7$ ;GO TO 7$ IF NO ERROR
021362 000240 NOP ;RETURN HERE IF ERROR
021364 104000 EMT ;ERROR # DEFINED BY DASTS SUBROUTINE
021366 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
021370 000137 022266 JMP 30$ ;GO TO 30$ IF ERROR
1356 021374 7$:
1357
1358 ;MOVE LOOP ADDRESSES TO NEXT OPERATION
1359 021374 012737 021446 001124 MOV #8$,$LPERR
1360
1361 ;REGENERATE BUFFER
1362 021402 012737 177000 001414 MOV #-256,*2,RMWC ;CHANGE WORD COUNT
1363 021410 012737 106436 001416 MOV #BUFONE,RMBAO ;CHANGE BUS ADDRESS

```

```

1364 021416 012737 072060 001174      MOV    #ZEROS,$TMP0      ;STARTING ADDRESS
1365 021424 012737 000001 001176      MOV    #1,$TMP1        ;RANGE
1366 021432 012737 000060 001412      MOV    #WD,RMCS10      ;WRITE DATA
1367 021440 004737 042174              JSR    PC,GENBUF        ;GENERATE BUFFER
1368 021444 000410              BR     9$          ;SKIP TO WRITE OPERATION
1369
1370      ;*****
1371      ;LOOP #2      WRITE,WRITE CHECK
1372
1373 021446      8$:
1374
1375      ;PREPARE DEVICE FOR WRITE OPERATION
1376 021446 004737 037316      JSR    PC,TSTPRP      ;PREPARE DEVICE FOR TEST
      021452 154130      .WORD    154130      ;TASK DESCRIPTOR AS FOLLOWS:
      ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
      ;CLEAR CONTROLLER & SELECT DEVICE
      ;VERIFY CONTROLLER CLEAR OPERATION
      ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
      ;VERIFY PACK ACKNOWLEDGE
      ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
      ;VERIFY RECALIBRATION
      ;GO TO 9$ IF NO ERROR
      ;RETURN HERE IF ERROR
      ;ERROR # DEFINED BY TSTPRP SUBROUTINE
      ;GO TO 30$ IF ERROR
      021454 000404      BR     9$
      021456 000240      NOP
      021460 104000      EMT
      021462 000137 022266      JMP    30$
1377 021466      9$:
1378 021466 012737 000005 001412      MOV    #SEEK!GO,RMCS10      ;LOAD SEEK COMMAND
1379
1380 021474 004737 043330      JSR    PC,PUT        ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      021500 000404      BR     10$      ;GO TO 10$ IF NO ERROR
      021502 000240      NOP          ;RETURN HERE IF ERROR
      021504 104000      EMT          ;ERROR # DEFINED BY PUT SUBROUTINE
      021506 000137 022266      JMP    30$      ;GO TO 30$ IF ERROR
1381 021512      10$:
1382
1383      ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
      021512 004737 042774      JSR    PC,GETSTS      ;GO TO GETSTS SUBROUTINE
1384
1385      ;WAIT FOR COMMAND TO COMPLETE
      021516 004737 043672      JSR    PC,TIMOUT      ;GO TO TIMEOUT SUBROUTINE
1386
1387      ;GO GET REGISTER STATUS
1388 021522 004737 043060      JSR    PC,GET        ;GO READ REGISTER(S) WITH GET SUBROUTINE
      021526 000404      BR     11$      ;GO TO 11$ IF NO ERROR
      021530 000240      NOP          ;RETURN HERE IF ERROR
      021532 104000      EMT          ;ERROR # DEFINED BY GET SUBROUTINE
      021534 000137 022266      JMP    30$      ;GO TO 30$ IF ERROR
1389 021540      11$:
1390
1391      ;VERIFY RESULTS OF SEEK COMMAND
1392 021540 004737 051174      JSR    PC,SEKSTS      ;GO VERIFY RESULTS OF SEEK OPERATION
      021544 000405      BR     12$      ;GO TO 12$ IF NO ERROR
      021546 000240      NOP          ;RETURN HERE IF ERROR
      021550 104000      EMT          ;ERROR # DEFINED BY SEKSTS SUBROUTINE
      021552 004736      JSR    PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      021554 000137 022266      JMP    30$      ;GO TO 30$ IF ERROR
1393 021560      12$:

```

```

1394
1395
1396 021560
1397 021560 012737 000061 001412
1398 021566 012702 001553
1399 021572 112722 000006
1400 021576 112722 000034
1401 021602 112722 000032
1402 021606 112722 000004
1403 021612 112722 000002
1404 021616 112722 000000
1405 021622 112722 000200
1406
1407 021626 004737 043330
    021632 000404
    021634 000240
    021636 104000
    021640 000137 022266
1408 021644
1409
1410
    021644 004737 042774
1411
1412
    021650 004737 043672
1413
1414
1415 021654 004737 043060
    021660 000404
    021662 000240
    021664 104000
    021666 000137 022266
1416 021672
1417
1418
1419 021672 004737 044056
    021676 000405
    021700 000240
    021702 104000
    021704 004736
    021706 000137 022266
1420 021712
1421 021712 004737 056572
    021716 000405
    021720 000240
    021722 104000
    021724 004736
    021726 000137 022266
1422 021732
1423 021732 004737 044710
    021736 000405
    021740 000240
    021742 104000
    021744 004736
    021746 000137 022266
1424 021752
1425

;WRITE DATA TO THE DRIVE
13$:
    MOV    #WD!GO,RMC$10
    MOV    #PUTINX,R2
    MOV    #RMDA,(R2)+
    MOV    #RMDC,(R2)+
    MOV    #RMOF,(R2)+
    MOV    #RMB$A,(R2)+
    MOV    #RMWC,(R2)+
    MOV    #RMC$1,(R2)+
    MOV    #200,(R2)+

;WRITE DATA COMMAND
;LOAD PUT REGISTER INDEX TABLE

;TERMINATE TABLE

    JSR    PC,PUT
    BR     14$
    NOP
    EMT
    JMP    30$

;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
;GO TO 14$ IF NO ERROR
;RETURN HERE IF ERROR
;ERROR # DEFINED BY PUT SUBROUTINE
;GO TO 30$ IF ERROR

14$:
;SETUP GET INDEX TABLE TO READ ALL REGISTERS
    JSR    PC,GETSTS
;GO TO GETSTS SUBROUTINE

;WAIT FOR COMMAND TO COMPLETE
    JSR    PC,TIMOUT
;GO TO TIMOUT SUBROUTINE

;GO READ STATUS FOR WRITE COMMAND
    JSR    PC,GET
;GO READ REGISTER(S) WITH GET SUBROUTINE
    BR     15$
;GO TO 15$ IF NO ERROR
    NOP
;RETURN HERE IF ERROR
    EMT
;ERROR # DEFINED BY GET SUBROUTINE
    JMP    30$
;GO TO 30$ IF ERROR

15$:
;CHECK FOR ERRORS DURING WRITE OPERATION
    JSR    PC,PRIERR
;GO CHECK FOR PRIMARY ERRORS
    BR     16$
;GO TO 16$ IF NO ERROR
    NOP
;RETURN HERE IF ERROR
    EMT
;ERROR # DEFINED BY PRIERR SUBROUTINE
    JSR    PC,@(SP)+
;GO BACK FOR MORE ERROR CHECKS
    JMP    30$
;GO TO 30$ IF ERROR

16$:
    JSR    PC,DTASTS
;GO VERIFY RESULTS OF DATA TRANSFER
    BR     17$
;GO TO 17$ IF NO ERROR
    NOP
;RETURN HERE IF ERROR
    EMT
;ERROR # DEFINED BY DTASTS SUBROUTINE
    JSR    PC,@(SP)+
;GO BACK FOR MORE ERROR CHECKS
    JMP    30$
;GO TO 30$ IF ERROR

17$:
    JSR    PC,SECERR
;GO CHECK FOR SECONDARY ERRORS
    BR     18$
;GO TO 18$ IF NO ERROR
    NOP
;RETURN HERE IF ERROR
    EMT
;ERROR # DEFINED BY SECERR SUBROUTINE
    JSR    PC,@(SP)+
;GO BACK FOR MORE ERROR CHECKS
    JMP    30$
;GO TO 30$ IF ERROR

18$:

```

```

1426 ;CHANGE LOOP ADDRESSES
1427 021752 012737 021762 001124 MOV #19$, $LPERR
1428 021760 000410 BR 20$ ;SKIP TO NEXT OPERATION
1429
1430 ;*****
1431 ;LOOP #3 WRITE CHECK
1432
1433 021762 19$:
1434
1435 ;PREPARE DEVICE FOR READ OPERATION
1436 021762 004737 037316 JSR PC, TSTPRP ;PREPARE DEVICE FOR TEST
      021766 154130 .WORD 154130 ;TASK DESCRIPTOR AS FOLLOWS:
      ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
      ;CLEAR CONTROLLER & SELECT DEVICE
      ;VERIFY CONTROLLER CLEAR OPERATION
      ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
      ;VERIFY PACK ACKNOWLEDGE
      ;RECALIBRATE IF 'SKI' OR 'PIP' IS SFT
      ;VERIFY RECALIBRATION
      ;GO TO 20$ IF NO ERROR
      ;RETURN HERE IF ERROR
      ;ERROR # DEFINED BY TSTPRP SUBROUTINE
      ;GO TO 30$ IF ERROR
      021770 000404 BR 20$
      021772 000240 NOP
      021774 104000 EMT
      021776 000137 022266 JMP 30$
1437 022002 20$:
1438 022002 012737 000005 001412 MOV #SEEK!GO, RMCS10 ;LOAD SEEK COMMAND
1439
1440 022010 004737 043330 JSR PC, PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      022014 000404 BR 21$ ;GO TO 21$ IF NO ERROR
      022016 000240 NOP ;RETURN HERE IF ERROR
      022020 104000 EMT ;ERROR # DEFINED BY PUT SUBROUTINE
      022022 000137 022266 JMP 30$ ;GO TO 30$ IF ERROR
1441 022026 21$:
1442
1443 ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
      022026 004737 042774 JSR PC, GETSTS ;GO TO GETSTS SUBROUTINE
1444
1445 ;WAIT FOR COMMAND TO COMPLETE
      022032 004737 043672 JSR PC, TIMEOUT ;GO TO TIMEOUT SUBROUTINE
1446
1447 ;GO READ REGISTER STATUS
      022036 004737 043060 JSR PC, GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
      022042 000404 BR 22$ ;GO TO 22$ IF NO ERROR
      022044 000240 NOP ;RETURN HERE IF ERROR
      022046 104000 EMT ;ERROR # DEFINED BY GET SUBROUTINE
      022050 000137 022266 JMP 30$ ;GO TO 30$ IF ERROR
1449 022054 22$:
1450
1451 ;GO VERIFY RESULTS OF SEEK
      022054 004737 051174 JSR PC, SEKSTS ;GO VERIFY RESULTS OF SEEK OPERATION
      022060 000405 BR 23$ ;GO TO 23$ IF NO ERROR
      022062 000240 NOP ;RETURN HERE IF ERROR
      022064 104000 EMT ;ERROR # DEFINED BY SEKSTS SUBROUTINE
      022066 004736 JSR PC, @ (SP)+ ;GO BACK FOR MORE ERROR CHECKS
      022070 000137 022266 JMP 30$ ;GO TO 30$ IF ERROR
1453 022074 23$:
1454
1455 ;READ DATA FROM DEVICE

```

```

1456 022074      24$:
1457 022074 012737 000051 001412  MOV    #WCD.GO, RMCS10      ;READ DATA COMMAND
1458 022102 012702 001553      MOV    #PUTINX, R2      ;LOAD PUT REGISTER INDEX TABLE
1459 022106 112722 000006      MOVB   #RMDA, (R2)+
1460 022112 112722 000032      MOVB   #RMOF, (R2)+
1461 022116 112722 000034      MOVB   #RMDC, (R2)+
1462 022122 112722 000004      MOVB   #RMB3A, (R2)+
1463 022126 112722 000002      MOVB   #RMWC, (R2)+
1464 022132 112722 000000      MOVB   #RMCS1, (R2)+
1465 022136 112712 000200      MOVB   #200, (R2)
1466
1467 022142 004737 043330      JSR     PC, PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      022146 000404      BR      25$      ;GO TO 25$ IF NO ERROR
      022150 000240      NOP      ;RETURN HERE IF ERROR
      022152 104000      EMT      ;ERROR # DEFINED BY PUT SUBROUTINE
      022154 000137 022266      JMP     30$      ;GO TO 30$ IF ERROR
1468 022160      25$:
1469
1470      ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
      JSR     PC, GETSTS      ;GO TO GETSTS SUBROUTINE
1471
1472      ;WAIT FOR COMMAND TO COMPLETE
      JSR     PC, TIMEOUT      ;GO TO TIMEOUT SUBROUTINE
1473
1474      ;GO READ STATUS FOR READ OPERATION
1475 022170 004737 043060      JSR     PC, GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      022174 000404      BR      26$      ;GO TO 26$ IF NO ERROR
      022176 000240      NOP      ;RETURN HERE IF ERROR
      022200 104000      EMT      ;ERROR # DEFINED BY GET SUBROUTINE
      022202 000137 022266      JMP     30$      ;GO TO 30$ IF ERROR
1476 022206      26$:
1477
1478      ;CHECK FOR ERRORS DURING READ OPERATION
1479 022206 004737 044056      JSR     PC, PRIERR      ;GO CHECK FOR PRIMARY ERRORS
      022212 000405      BR      27$      ;GO TO 27$ IF NO ERROR
      022214 000240      NOP      ;RETURN HERE IF ERROR
      022216 104000      EMT      ;ERROR # DEFINED BY PRIERR SUBROUTINE
      022220 004736      JSR     PC, @ (SP)+      ;GO BACK FOR MORE ERROR CHECKS
      022222 000137 022266      JMP     30$      ;GO TO 30$ IF ERROR
1480 022226      27$:
1481 022226 004737 056572      JSR     PC, DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
      022232 000405      BR      28$      ;GO TO 28$ IF NO ERROR
      022234 000240      NOP      ;RETURN HERE IF ERROR
      022236 104000      EMT      ;ERROR # DEFINED BY DTASTS SUBROUTINE
      022240 004736      JSR     PC, @ (SP)+      ;GO BACK FOR MORE ERROR CHECKS
      022242 000137 022266      JMP     30$      ;GO TO 30$ IF ERROR
1482 022246      28$:
1483 022246 004737 044710      JSR     PC, SECERR      ;GO CHECK FOR SECONDARY ERRORS
      022252 000405      BR      29$      ;GO TO 29$ IF NO ERROR
      022254 000240      NOP      ;RETURN HERE IF ERROR
      022256 104000      EMT      ;ERROR # DEFINED BY SECERR SUBROUTINE
      022260 004736      JSR     PC, @ (SP)+      ;GO BACK FOR MORE ERROR CHECKS
      022262 000137 022266      JMP     30$      ;GO TO 30$ IF ERROR
1484 022266      29$:
1485
1486      30$:
1487

```



```

1488      022266      000004      001100      001226
1489      022266      000240
1490      022270      012706      001276
1491      022272      013700      001466
1492      022276      013701      000013
1493      022302      012737
1494      022314
1495      022314      012737      000000      001446
1496      022322      013737      001334      001420
1497      022330      112737      000037      001420
1498      022336      012737      010000      001444
1499      022344      012737      106436      001416
1500      022352      012737      176774      001414
1501      022360      012737      000063      001412
1502
1503      022366      004737      040242
1504      022372      000405
1505      022374      104401      070426
1506      022400      104000
1507      022402      000137      023310
1508      022406      123737      001421      001335
1509      022414      001342
1510      022416      012737      072060      001174
1511      022424      012737      000001      001176
1512      022432      004737      042174
1513      022436      004737      037316
1514      022442      154130
1515
1516      022444      000404
1517      022446      000240
1518      022450      104000
1519      022452      000137      023310
1520      022456
1521
1522      022456      012702      001553
1523      022462      112722      000034
1524      022466      112722      000006
  
```

```

*****
*TEST 13      WRITE, WRITE CHECK W/ MID-TRANSFER SEEK
*****
TST13:
      SCOPE      :SCOPE CALL
      NOP      :START OF TEST
      MOV      #STACK,SP      :INITIALIZE STACK POINTER
      MOV      $BASE,R0      :R0 = UNIBUS ADDRESS
      MOV      TSTQUE,R1      : (R1) = DEVICE BEING TESTED
      MOV      #13,$TESTN      :SET TEST NUMBER IN APT MAIL BOX

*****
;LOOP #1      FORMAT,WRITE,WRITE CHECK

;LOAD PARAMETERS AND GENERATE DATA BUFFER
1$:      MOV      #0,RMDCO      :CYLINDER = 0
2$:      MOV      LSTRK,RMDAO      :SET LAST TRACK AND
      MOV      #31,RMDAO      :LAST SECTOR
      MOV      #FMT16,RMOFO      :16 BIT FORMAT
      MOV      #BUFONE,RMBAO      :BUS ADDRESS
      MOV      #-258,*2,RMWCO      :WORD COUNT FOR 2 SECTORS (2* COMP)
      MOV      #WH!GO,RMCS10      :WRITE HEADER AND DATA COMMAND

;VERIFY THAT SECTOR IS NOT BAD
      JSR      PC,BADSCT      :CALL BAD SECTOR MODULE
      BR      3$      :GO TO 3$ IF NO ERROR
      TYPE      ,SCTMSG      :TYPE BAD SECTOR MESSAGE
      EMT      :ERROR # DEFINED BY BADSCT SUBROUTINE
      JMP      25$      :GO TO 25$ IF ERROR

3$:      CMPB      RMDAO+1,LSTRK+1      :IS LAST TRACK ASSIGNED ?
      BNE      2$      :BR IF NO
      MOV      #ZEROS,$TMP0      :STARTING ADDRESS OF PATTERN
      MOV      #1,$TMP1      :RANGE OF PATTERN
      JSR      PC,GENBUF      :GO GENERATE BUFFER FOR FORMAT

;PREPARE THE DEVICE FOR FORMAT OPERATION
      JSR      PC,TSTPRP      :PREPARE DEVICE FOR TEST
      .WORD      154130      :TASK DESCRIPTOR AS FOLLOWS:
      :SELECT DEVICE & VERIFY DEVICE AVAILABLE
      :CLEAR CONTROLLER & SELECT DEVICE
      :VERIFY CONTROLLER CLEAR OPERATION
      :PACK ACKNOWLEDGE IF VOLUME NOT VALID
      :VERIFY PACK ACKNOWLEDGE
      :RECALIBRATE IF 'SKI' OR 'PIP' IS SET
      :VERIFY RECALIBRATION
      BR      4$      :GO TO 4$ IF NO ERROR
      NOP      :RETURN HERE IF ERROR
      EMT      :ERROR # DEFINED BY TSTPRP SUBROUTINE
      JMP      25$      :GO TO 25$ IF ERROR

4$:

;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATION
      MOV      #PUTINX,R2      :R2 - BYTE ENTRY POSITION
      MOV      #RMDC,(R2)+
      MOV      #RMDA,(R2)+
  
```

```

1519 022472 112722 000004      MOV8  #RMA,(R2)+
1520 022476 112722 000002      MOV8  #RMC,(R2)+
1521 022502 112722 000032      MOV8  #RMOF,(R2)+
1522 022506 112722 000000      MOV8  #RMC51,(R2)+
1523 022512 112712 000200      MOV8  #200,(R2)          ;TERMINATE TABLE
1524 022516
1525 022516 004737 043330      5$: JSR    PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      022522 000404          BR     6$          ;GO TO 6$ IF NO ERROR
      022524 000240          NOP          ;RETURN HERE IF ERROR
      022526 104000          EMT          ;ERROR # DEFINED BY PUT SUBROUTINE
      022530 000137 023310      JMP     25$         ;GO TO 25$ IF ERROR
1526 022534
1527
1528      022534 004737 042774      6$: ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
      JSR    PC,GETSTS        ;GO TO GETSTS SUBROUTINE
1529
1530      022540 004737 043672      ;WAIT FOR COMMAND TO COMPLETE
      JSR    PC,TIMOUT        ;GO TO TIMEOUT SUBROUTINE
1531
1532      ;GO READ STATUS FOR FORMAT OPERATION
1533 022544 004737 043060      JSR    PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
      022550 000404          BR     7$          ;GO TO 7$ IF NO ERROR
      022552 000240          NOP          ;RETURN HERE IF ERROR
      022554 104000          EMT          ;ERROR # DEFINED BY GET SUBROUTINE
      022556 00013 023310      JMP     25$         ;GO TO 25$ IF ERROR
1534 022562
1535
1536      ;VERIFY NO ERRORS DURING FORMAT
1537 022562 004737 056572      JSR    PC,DASTS        ;GO VERIFY RESULTS OF DATA TRANSFER
      022566 000405          BR     8$          ;GO TO 8$ IF NO ERROR
      022570 000240          NOP          ;RETURN HERE IF ERROR
      022572 104000          EMT          ;ERROR # DEFINED BY DASTS SUBROUTINE
      022574 004736          JSR    PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      022576 000137 023310      JMP     25$         ;GO TO 25$ IF ERROR
1538 022602
1539
1540      ;MOVE LOOP ADDRESSES TO NEXT OPERATION
1541 022602 012737 022654 001124      MOV     #9$,$LPERR
1542
1543      ;REGENERATE BUFFER
1544 022610 012737 000060 001412      MOV     #WD,RMC510          ;WRITE DATA
1545 022616 012737 177000 001414      MOV     #-256,*2,RMWC0      ;CHANGE WORD COUNT
1546 022624 012737 106436 001416      MOV     #BUFONE,RMBAD      ;CHANGE BUS ADDRESS
1547 022632 012737 072016 001174      MOV     #ONES,$TMP0      ;PATTERN ADDRESS
1548 022640 012737 000001 001176      MOV     #1,$TMP1          ;PATTERN RANGE
1549 022646 004737 042174          JSR    PC,GENBUF
1550 022652 000410          BR     10$          ;SKIP TO WRITE OPERATION
1551
1552      ;*****
1553      ;LOOP #2      WRITE,WRITE CHECK
1554
1555 022654      9$:
1556
1557      ;PREPARE DEVICE FOR WRITE OPERATION
1558 022654 004737 037316      JSR    PC,ISTPRP      ;PREPARE DEVICE FOR TEST
      022660 154130      .WOR    154130      ;TASK DESCRIPTOR AS FOLLOWS:
      ;SELECT DEVICE & VERIFY DEVICE AVAILABLE

```


;CLEAR CONTROLLER & SELECT DEVICE
 ;VERIFY CONTROLLER CLEAR OPERATION
 ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
 ;VERIFY PACK ACKNOWLEDGE
 ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
 ;VERIFY RECALIBRATION
 ;GO TO 10\$ IF NO ERROR
 ;RETURN HERE IF ERROR
 ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 ;GO TO 25\$ IF ERROR

1559	022667	000404		BR	10\$	
1560	022664	000240		NOP		
1561	022666	104000		EMT		
1562	022670	000137	023310	JMP	25\$	
1563	022674					10\$:
1564	022674	012737	000061			;WRITE DATA TO THE DRIVE
1565	022702	012702	001553	1\$:		
1566	022706	112722	000006	MOV	#WD!GO,RMCS10	;WRITE DATA COMMAND
1567	022712	112722	000034	MOV	#PUTINX,R2	;LOAD PUT REGISTER INDEX TABLE
1568	022716	112722	000032	MOVB	#RMDA,(R2)+	
1569	022722	112722	000004	MOVB	#RMDC,(R2)+	
1570	022726	112722	000002	MOVB	#RMOF,(R2)+	
1571	022732	112722	000000	MOVB	#RMDA,(R2)+	
1572	022736	112722	000200	MOVB	#RMCJ,(R2)+	
1573	022742	004737	043330	MOVB	#200,(R2)+	;TERMINATE TABLE
1574	022746	000404		JSR	PC,PUT	;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
1575	022750	000240		BR	12\$	;GO TO 12\$ IF NO ERROR
1576	022752	104000		NOP		;RETURN HERE IF ERROR
1577	022754	000137	023310	EMT		;ERROR # DEFINED BY PUT SUBROUTINE
1578	022760			JMP	25\$	;GO TO 25\$ IF ERROR
1579	022760	004737	042774			12\$:
1580	022764	004737	043672			;SETUP GET INDEX TABLE TO READ ALL REGISTERS
1581	022770	004737	043060	JSR	PC,GETSTS	;GO TO GETSTS SUBROUTINE
1582	022774	000404				
1583	022776	000240				
1584	023000	104000				
1585	023002	000137	023310			
1586	023006					
1587	023012	000405	044056			
1588	023014	000240				
1589	023016	104000				
1590	023020	004736				
1591	023022	000137	023310			
1592	023026					
1593	023032	000405				
1594	023034	000240				
1595	023036	104000				
1596						
1597						
1598						
1599						
1600						
1601						
1602						
1603						
1604						
1605						
1606						
1607						
1608						
1609						
1610						
1611						
1612						
1613						
1614						
1615						
1616						
1617						
1618						
1619						
1620						
1621						
1622						
1623						
1624						
1625						
1626						
1627						
1628						
1629						
1630						
1631						
1632						
1633						
1634						
1635						
1636						
1637						
1638						
1639						
1640						
1641						
1642						
1643						
1644						
1645						
1646						
1647						
1648						
1649						
1650						
1651						
1652						
1653						
1654						
1655						
1656						
1657						
1658						
1659						
1660						
1661						
1662						
1663						
1664						
1665						
1666						
1667						
1668						
1669						
1670						
1671						
1672						
1673						
1674						
1675						
1676						
1677						
1678						
1679						
1680						
1681						
1682						
1683						
1684						
1685						
1686						
1687						
1688						
1689						
1690						
1691						
1692						
1693						
1694						
1695						
1696						
1697						
1698						
1699						
1700						
1701						
1702						
1703						
1704						
1705						
1706						
1707						
1708						
1709						
1710						
1711						
1712						
1713						
1714						
1715						
1716						
1717						
1718						
1719						
1720						
1721						
1722						
1723						
1724						
1725						
1726						
1727						
1728						
1729						
1730						
1731						
1732						
1733						
1734						
1735						
1736						
1737						
1738						
1739						
1740						
1741						
1742						
1743						
1744						
1745						
1746						
1747						
1748						
1749						
1750						
1751						
1752						
1753						
1754						
1755						
1756						
1757						
1758						
1759						
1760						
1761						
1762						
1763						
1764						
1765						
1766						
1767						
1768						
1769						
1770						
1771						
1772						
1773						
1774						
1775						
1776						
1777						
1778						
1779						
1780						
1781						
1782						
1783						
1784						
1785						
1786						
1787						
1788						
1789						
1790						
1791						
1792						
1793						
1794						
1795						
1796						
17						

```

023040 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
023042 000137 023310 JMP 25$ ;GO TO 25$ IF ERROR
1588 023046 15$: JSR PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
1589 023046 004737 044710 BR 16$ ;GO TO 16$ IF NO ERROR
023052 000405 NOP ;RETURN HERE IF ERROR
023054 000240 EMT ;ERROR # DEFINED BY SECERR SUBROUTINE
023056 104000 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
023060 004736 JMP 25$ ;GO TO 25$ IF ERROR
023062 000137 023310
1590 023066
1591
1592 ;CHANGE LOOP ADDRESSES
1593 023066 012737 023076 001124 MOV #17$,SLPERR
1594 023074 000410 BR 18$ ;SKIP TO NEXT OPERATION
1595
1596 ;*****
1597 ;LOOP #3 WRITE CHECK
1598
1599 023076 17$:
1600
1601 ;PREPARE DEVICE FOR READ OPERATION
1602 023076 004737 037316 JSR PC,TSTPRP ;PREPARE DEVICE FOR TEST
023102 154130 .WORD 154130 ;TASK DESCRIPTOR AS FOLLOWS:
;SELECT DEVICE & VERIFY DEVICE AVAILABLE
;CLEAR CONTROLLER & SELECT DEVICE
;VERIFY CONTROLLER-CLEAR OPERATION
;PACK ACKNOWLEDGE IF VOLUME NOT VALID
;VERIFY PACK ACKNOWLEDGE
;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
;VERIFY RECALIBRATION
;GO TO 18$ IF NO ERROR
;RETURN HERE IF ERROR
;ERROR # DEFINED BY TSTPRP SUBROUTINE
;GO TO 25$ IF ERROR
023104 000404 BR 18$
023106 000240 NOP
023110 104000 EMT
023112 000137 023310 JMP 25$
1603 023116 18$:
1604
1605 ;READ DATA FROM DEVICE
1606 023116 19$:
1607 023116 012737 000051 001412 MOV #WCD!GO,RMCS10 ;READ DATA COMMAND
1608 023124 012702 001553 MOV #PUTINX,R2 ;LOAD PUT REGISTER INDEX TABLE
1609 023130 112722 000006 MOVB #RMDA,(R2)+
1610 023134 112722 000032 MOVB #RMOF,(R2)+
1611 023140 112722 000034 MOVB #RMDC,(R2)+
1612 023144 112722 000004 MOVB #RMBB,(R2)+
1613 023150 112722 000002 MOVB #RMWC,(R2)+
1614 023154 112722 000000 MOVB #RMCS1,(R2)+
1615 023160 112712 000200 MOVB #200,(R2)
1616
1617 023164 004737 043330 JSR PC,PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
023170 000404 BR 20$ ;GO TO 20$ IF NO ERROR
023172 000240 NOP ;RETURN HERE IF ERROR
023174 104000 EMT ;ERROR # DEFINED BY PUT SUBROUTINE
023176 000137 023310 JMP 25$ ;GO TO 25$ IF ERROR
1618 023202 20$:
1619
1620 ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
023202 004737 042774 JSR PC,GETSTS ;GO TO GETSTS SUBROUTINE

```

```

1621
1622      023206 004737 043672      ;WAIT FOR COMMAND TO COMPLETE
                                   JSR      PC,TIMOUT      ;GO TO TIMOUT SUBROUTINE
1623
1624      023212 004737 043060      ;GO READ STATUS FOR READ OPERATION
1625      023216 000404              JSR      PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
                                   BR       21$           ;GO TO 21$ IF NO ERROR
                                   NOP      ;RETURN HERE IF ERROR
                                   EMT      ;ERROR # DEFINED BY GET SUBROUTINE
                                   JMP      25$           ;GO TO 25$ IF ERROR
                                   21$:
1626      023224 000137 023310
1627
1628      023230 004737 044056      ;CHECK FOR ERRORS DURING READ OPERATION
1629      023234 000405              JSR      PC,PRIERR    ;GO CHECK FOR PRIMARY ERRORS
                                   BR       22$           ;GO TO 22$ IF NO ERROR
                                   NOP      ;RETURN HERE IF ERROR
                                   EMT      ;ERROR # DEFINED BY PRIERR SUBROUTINE
                                   JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
                                   JMP      25$           ;GO TO 25$ IF ERROR
                                   22$:
1630      023240 004736 023310
1631      023242 000137 023310
1632      023250 004737 056572      JSR      PC,DTASTS    ;GO VERIFY RESULTS OF DATA TRANSFER
1633      023254 000405              BR       23$           ;GO TO 23$ IF NO ERROR
                                   NOP      ;RETURN HERE IF ERROR
                                   EMT      ;ERROR # DEFINED BY DTASTS SUBROUTINE
                                   JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
                                   JMP      25$           ;GO TO 25$ IF ERROR
                                   23$:
1634      023270 004737 044710      JSR      PC,SECERR    ;GO CHECK FOR SECONDARY ERRORS
1635      023274 000405              BR       24$           ;GO TO 24$ IF NO ERROR
                                   NOP      ;RETURN HERE IF ERROR
                                   EMT      ;ERROR # DEFINED BY SECERR SUBROUTINE
                                   JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
                                   JMP      25$           ;GO TO 25$ IF ERROR
                                   24$:
1636      023276 000240
1637      023300 104000
1638      023302 004736 023310
1639      023304 000137 023310
1640
1641      023310
1642
1643      023310 000004
1644      023312 000240
1645      023314 012706 001100
1646      023320 013700 001276
1647      023324 013701 001466
1648      023330 012737 000014 001226
                                   TST14:
                                   SCOPE      ;SCOPE CALL
                                   NOP      ;START OF TEST
                                   MOV      #STACK,SP      ;INITIALIZE STACK POINTER
                                   MOV      $BASE,R0      ;R0 = UNIBUS ADDRESS
                                   MOV      TSTQUE,R1      ;(R1) = DEVICE BEING TESTED
                                   MOV      #14,$TESTN     ;SET TEST NUMBER IN APT MAIL BOX
                                   ;*****
                                   ;*TEST 14      WRITE, READ W/ HCE ERROR
                                   ;*****
                                   ;LOOP #1      FORMAT,WRITE,READ
                                   1$:      MOV      #0,R4      ;R4 = 1ST HEADER WORD
                                   2$:      MOV      #1,R3      ;R3 = HCE BIT
                                   ;PREPARE THE DEVICE FOR FORMAT OPERATION
                                   JSR      PC,ISTPRP      ;PREPARE DEVICE FOR TEST
  
```

	023352	154130		.WORD	154130	:TASK DESCRIPTOR AS FOLLOWS:
						:SELECT DEVICE & VERIFY DEVICE AVAILABLE
						:CLEAR CONTROLLER & SELECT DEVICE
						:VERIFY CONTROLLER CLEAR OPERATION
						:PACK ACKNOWLEDGE IF VOLUME NOT VALID
						:VERIFY PACK ACKNOWLEDGE
						:RECALIBRATE IF 'SKI' OR 'PIP' IS SET
						:VERIFY RECALIBRATION
	023354	000404		BR	3\$	:GO TO 3\$ IF NO ERROR
	023356	000240		NOP		:RETURN HERE IF ERROR
	023360	104000		EMT		:ERROR # DEFINED BY 'STPRP SUBROUTINE
	023362	000137	025156	JMP	46\$	:GO TO 46\$ IF ERROR
1649						
1650						
1651	023366					:LOAD PARAMETERS AND GENERATE DATA BUFFER
1652	023366	012737	000000	001446	3\$: MOV #0,RMDCO	:CYLINDER = 0
1653	023374	012737	000000	001420	MOV #0,RMDAO	:TRACK = 0, SECTOR = 0
1654	023402	012737	106436	001416	MOV #BUFONE,RMBAO	:BUS ADDRESS
1655	023410	012737	177376	001414	MOV #-258.,RMWCO	:2 + 256. WORDS (2'S COMP)
1656	023416	012737	010000	001444	MOV #FMT16,RMOFO	:16 BIT FORMAT
1657	023424	012737	000063	001412	MOV #WH!GO,RMC\$10	:WRITE HEADER AND DATA COMMAND
1658						
1659						
	023432	004737	040242		:VERIFY THAT SECTOR IS NOT BAD	
	023436	000405			JSR PC,BADSCT	:CALL BAD SECTOR MODULE
	023440	104401	070426		BR 4\$	:GO TO 4\$ IF NO ERROR
	023444	104000			TYPE ,SCTMSG	:TYPE BAD SECTOR MESSAGE
	023446	000137	025004		EMT	:ERROR # DEFINED BY BADCST SUBROUTINE
1660	023452				JMP 42\$	:GO TO 42\$ IF ERROR
1661	023452	012737	072060	001174	4\$: MOV #ZEROS,\$TMP0	:STARTING ADDRESS OF PATTERN
1662	023460	012737	000001	001176	MOV #1,\$TMP1	:RANGE OF PATTERN
1663	023466	004737	042174		JSR PC,GENBUF	:GO GENERATE BUFFER FOR FORMAT
1664						
1665						
1666	023472	030364	106436		:CHANGE HEADER WORD TO FORCE HCE DURING WRITE & READ	
1667	023476	001403			BIT R3,BUFONE(R4)	:SET OR RESET FOR HCE??
1668	023500	040364	106436		BEQ 5\$	
1669	023504	000402			BIC R3,BUFONE(R4)	:RESET BIT FOR HCE
1670	023506	050364	106436		BR 6\$	
1671					5\$: BIS R3,BUFONE(R4)	:SET FOR HCE
1672						
1673	023512				:LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION	
1674	023512	012702	001553		6\$: MOV #PUTINX,R2	:R2 - BYTE ENTRY POSITION
1675	023516	112722	000034		MOVB #RMDC,(R2)+	
1676	023522	112722	000006		MOVB #RMDA,(R2)+	
1677	023526	112722	000004		MOVB #RMBA,(R2)+	
1678	023532	112722	000002		MOVB #RMWC,(R2)+	
1679	023536	112722	000032		MOVB #RMOF,(R2)+	
1680	023542	112722	000000		MOVB #RMCS1,(R2)+	
1681	023546	112712	000200		MOVB #200,(R2)	:TERMINATE TABLE
1682	023552				7\$: JSR PC,PUT	:GO WRITE REGISTER(S) WITH PUT SUBROUTINE
1683	023552	004737	043330		BR 8\$	:GO TO 8\$ IF NO ERROR
	023556	000404			NOP	:RETURN HERE IF ERROR
	023560	000240			EMT	:ERROR # DEFINED BY PUT SUBROUTINE
	023562	104000			JMP 42\$	:GO TO 42\$ IF ERROR
	023564	000137	025004			
1684	023570				8\$: JSR PC,PUT	:GO WRITE REGISTER(S) WITH PUT SUBROUTINE
					BR 8\$	:GO TO 8\$ IF NO ERROR
					NOP	:RETURN HERE IF ERROR
					EMT	:ERROR # DEFINED BY PUT SUBROUTINE
					JMP 42\$	:GO TO 42\$ IF ERROR

```

1685
1686      023570  004737  042774      ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
      JSR      PC,GETSTS      ;GO TO GETSTS SUBROUTINE
1687
1688      023574  004737  043672      ;WAIT FOR COMMAND TO COMPLETE
      JSR      PC,TIMOUT      ;GO TO TIMOUT SUBROUTINE
1689
1690      023600  004737  043060      ;GO READ STATUS FOR FORMAT OPERATION
1691      023604  000404      JSR      PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      023606  000240      BR      9$      ;GO TO 9$ IF NO ERROR
      023610  104000      NOP      ;RETURN HERE IF ERROR
      023612  000137  025004      EMT      ;ERROR # DEFINED BY GET SUBROUTINE
      023616      JMP      42$      ;GO TO 42$ IF ERROR
1692      9$:
1693
1694      023616  004737  056572      ;VERIFY NO ERRORS DURING FORMAT
1695      023622  000405      JSR      PC,DASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
      023624  000240      BR      10$      ;GO TO 10$ IF NO ERROR
      023626  104000      NOP      ;RETURN HERE IF ERROR
      023630  004736      EMT      ;ERROR # DEFINED BY DASTS SUBROUTINE
      023632  000137  025004      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      023636      JMP      42$      ;GO TO 42$ IF ERROR
1696      10$:
1697
1698      023636  012737  023646  00124  ;MOVE LOOP ADDRESSES TO NEXT OPERATION
1699      023644  000410      MOV      #11$,$LPERR
1700      BR      12$      ;SKIP TO WRITE OPERATION
1701
1702      ;*****
1703      ;LOOP #2      WRITE,READ
1704
1705      023646      11$:
1706
1707      023646  004737  037316      ;PREPARE DEVICE FOR WRITE OPERATION
1708      023652  154130      JSR      PC,TSTPRP      ;PREPARE DEVICE FOR TEST
      ;TASK DESCRIPTOR AS FOLLOWS:
      ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
      ;CLEAR CONTROLLER & SELECT DEVICE
      ;VERIFY CONTROLLER CLEAR OPERATION
      ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
      ;VERIFY PACK ACKNOWLEDGE
      ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
      ;VERIFY RECALIBRATION
      023654  000404      BR      12$      ;GO TO 12$ IF NO ERROR
      023656  000240      NOP      ;RETURN HERE IF ERROR
      023660  104000      EMT      ;ERROR # DEFINED BY TSTPRP SUBROUTINE
      023662  000137  025004      JMP      42$      ;GO TO 42$ IF ERROR
1709      12$:
1710
1711      023666      ;WRITE DATA TO THE DRIVE
1712      13$:
1713      023666  012737  106442  001416  MOV      #BUFONE+4,RMBAD ;CHANGE BUS ADDRESS
1714      023674  012737  177400  001414  MOV      #-256.,RMWCO ;CHANGE WORD COUNT
1715      023702  012737  000061  001412  MOV      #WD.GO,RMCS10 ;WRITE DATA COMMAND
1716      023710  012702  001553      MOV      #PUTINX,R2 ;LOAD PUT REGISTER INDEX TABLE
1717      023714  112722  000006      MOVB     #RMDA,(R2)+
1718      023720  112722  000034      MOVB     #RMDC,(R2)+
  
```

```

1719 023724 112722 000032      MOVB  #RMOF,(R2)+
1720 023730 112722 000004      MOVB  #RMSA,(R2)+
1721 023734 112722 000002      MOVB  #RMWC,(R2)+
1722 023740 112722 000000      MOVB  #RMCS1,(R2)+
1723 023744 112722 000200      MOVB  #200,(R2)+      ;TERMINATE TABLE
1724
1725 023750 004737 043330      JSR    PC,PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      023754 000404      BR     14$      ;GO TO 14$ IF NO ERROR
      023756 000240      NOP      ;RETURN HERE IF ERROR
      023760 104000      EMT      ;ERROR # DEFINED BY PUT SUBROUTINE
      023762 000137 025004      JMP     42$      ;GO TO 42$ IF ERROR
1726 023766
1727
1728      ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
      023766 004737 042774      JSR    PC,GETSTS      ;GO TO GETSTS SUBROUTINE
1729
1730      ;WAIT FOR COMMAND TO COMPLETE
      023772 004737 043672      JSR    PC,TIMOUT      ;GO TO TIMEOUT SUBROUTINE
1731
1732      ;GO READ STATUS FOR WRITE COMMAND
1733 023776 004737 043060      JSR    PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      024002 000404      BR     15$      ;GO TO 15$ IF NO ERROR
      024004 000240      NOP      ;RETURN HERE IF ERROR
      024006 104000      EMT      ;ERROR # DEFINED BY GET SUBROUTINE
      024010 000137 025004      JMP     42$      ;GO TO 42$ IF ERROR
1734 024014
1735
1736      ;CHECK FOR ERRORS DURING WRITE OPERATION
1737 024014 004737 044056      JSR    PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
      024020 000405      BR     16$      ;GO TO 16$ IF NO ERROR
      024022 000240      NOP      ;RETURN HERE IF ERROR
      024024 104000      EMT      ;ERROR # DEFINED BY PRIERR SUBROUTINE
      024026 004736      JSR    PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      024030 000137 025004      JMP     42$      ;GO TO 42$ IF ERROR
1738
1739      ;DECODE THE TYPE OF ERROR PRODUCED BY THE BIT POSITION OF R3
1740 024034
1741 024034 005704      TST     R4      ;IS THIS THE FIRST HEADER WORD ?
1742 024036 001006      BNE     17$      ;NO !!
1743
1744 024040 032703 010000      BIT     #FMT16,R3      ;SHOULD FER BE SET ?
1745 024044 001037      BNE     20$      ;YES !!
1746 024046 032703 140000      BIT     #MSE.USE,R3      ;SHOULD BSE BE SET ?
1747 024052 001061      BNE     22$      ;YES !!
1748
1749      ;VERIFY THAT A HEADER COMPARE ERROR WAS DETECTED
1750 024054
1751 024054 032737 000200 001352      BIT     #HCE,RMER11      ;WAS HCE DETECTED ??
1752 024062 001102      BNE     24$      ;YES !!
1753
1754 024064 004737 056572      JSR    PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
      024070 000405      BR     18$      ;GO TO 18$ IF NO ERROR
      024072 000240      NOP      ;RETURN HERE IF ERROR
      024074 104000      EMT      ;ERROR # DEFINED BY DTASTS SUBROUTINE
      024076 004736      JSR    PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      024100 000137 025004      JMP     42$      ;GO TO 42$ IF ERROR
1755 024104
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900

```



```

1756 024104 013737 001352 001142      MOV      RMER11,$BDDAT      ;RECEIVED STATUS
1757 024112 012737 000200 001140      MOV      #HCE,$GDDAT      ;EXPECTED STATUS
1758 024120 010437 001174              MOV      R4,$TMP0          ;R4 = HEADER WORD NUMBER AND
1759 024124 001002              BNE      19$              ;ADJUST NUMBER.
1760 024126 005237 001174              INC      $TMP0
1761 024132 010337 001176      19$:    MOV      R3,$TMP1          ;R3 = BIT POSITION
1762 024136 104344              EMT      344
1763 024140 000137 025004              JMP      42$
1764
1765      ;VERIFY THAT A FORMAT ERROR WAS DETECTED
1766 024144      20$:    BIT      #FER,RMER11      ;WAS FER DETECTED ??
1767 024144 032737 000020 001352      BNE      24$              ;YES !!
1768 024152 001046
1769
1770 024154 004737 056572              JSR      PC,DTASTS          ;GO VERIFY RESULTS OF DATA TRANSFER
                                BR      21$          ;GO TO 21$ IF NO ERROR
                                NOP                     ;RETURN HERE IF ERROR
                                EMT      104000         ;ERROR # DEFINED BY DTASTS SUBROUTINE
                                JSR      PC,@(SP)+        ;GO BACK FOR MORE ERROR CHECKS
                                JMP      42$          ;GO TO 42$ IF ERROR
1771 024174      21$:    MOV      #FER,$GDDAT      ;EXPECTED STATUS
1772 024174 012737 000020 001140      MOV      RMER11,$BDDAT      ;RECEIVED STATUS
1773 024202 013737 001352 001142      EMT      343
1774 024210 104343              JMP      42$
1775 024212 000137 025004
1776
1777      ;VERIFY THAT A BAD SECTOR ERROR WAS DETECTED
1778 024216      22$:    BIT      #BSE,RMER21      ;WAS BSE DETECTED ??
1779 024216 032737 100000 001400      BNE      24$              ;YES !!
1780 024224 001021
1781
1782 024226 004737 056572              JSR      PC,DTASTS          ;GO VERIFY RESULTS OF DATA TRANSFER
                                BR      23$          ;GO TO 23$ IF NO ERROR
                                NOP                     ;RETURN HERE IF ERROR
                                EMT      104000         ;ERROR # DEFINED BY DTASTS SUBROUTINE
                                JSR      PC,@(SP)+        ;GO BACK FOR MORE ERROR CHECKS
                                JMP      42$          ;GO TO 42$ IF ERROR
1783 024246      23$:    MOV      RMER21,$BDDAT      ;RECEIVED STATUS
1784 024246 013737 001400 001142      MOV      #BSE,$GDDAT      ;EXPECTED STATUS
1785 024254 012737 100000 001140      EMT      345
1786 024262 104345              JMP      42$
1787 024264 000137 025004
1788
1789      ;CHECK FOR OTHER ERRORS
1790 024270      24$:    JSR      PC,SECERR          ;GO CHECK FOR SECONDARY ERRORS
1791 024270 004737 044710      BR      25$          ;GO TO 25$ IF NO ERROR
                                NOP                     ;RETURN HERE IF ERROR
                                EMT      104000         ;ERROR # DEFINED BY SECERR SUBROUTINE
                                JSR      PC,@(SP)+        ;GO BACK FOR MORE ERROR CHECKS
                                JMP      42$          ;GO TO 42$ IF ERROR
1792 024310      25$:
1793
1794      ;CHANGE LOOP ADDRESSES
1795 024310 012737 024316 001124      MOV      #26$,$LPERR
1796
1797      ;*****

```

```

1798          ;LOOP #3          READ
1799
1800 024316    26$:
1801
1802          ;PREPARE DEVICE FOR READ OPERATION
1803 024316    004737    037316    JSR    PC,TSTPRP    ;PREPARE DEVICE FOR TEST
      024322    154130    .WORD    154130    ;TASK DESCRIPTOR AS FOLLOWS:
                                          ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
                                          ;CLEAR CONTROLLER & SELECT DEVICE
                                          ;VERIFY CONTROLLER CLEAR OPERATION
                                          ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                          ;VERIFY PACK ACKNOWLEDGE
                                          ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
                                          ;VERIFY RECALIBRATION
                                          ;GO TO 27$ IF NO ERROR
                                          ;RETURN HERE IF ERROR
                                          ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                          ;GO TO 42$ IF ERROR
      024324    000404    BR        27$
      024326    000240    NOP
      024330    104000    EMT
      024332    000137    025004    JMP     42$
1804 024336    27$:
1805
1806          ;READ DATA FROM DEVICE
1807 024336    28$:
1808 024336    012737    107446    001416    MOV     #BUFTWO+4,RMBA0    ;CHANGE BUS ADDRESS
1809 024344    012737    177400    001414    MOV     #-256.,RMWCO    ;CHANGE WORD COUNT
1810 024352    012737    000071    001412    MOV     #RD!GO,RMCST0    ;READ DATA COMMAND
1811 024360    012702    001553    MOV     #PUTINX,R2    ;LOAD PUT REGISTER INDEX TABLE
1812 024364    112722    000006    MOV     #RMDA,(R2)+
1813 024370    112722    000032    MOV     #RMOF,(R2)+
1814 024374    112722    000034    MOV     #RMDC,(R2)+
1815 024400    112722    000004    MOV     #RMBB,(R2)+
1816 024404    112722    000002    MOV     #RMWC,(R2)+
1817 024410    112722    000000    MOV     #RMCS1,(R2)+
1818 024414    112712    000200    MOV     #200,(R2)
1819
1820 024420    004737    043330    JSR     PC,PUT    ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      024424    000404    BR        29$    ;GO TO 29$ IF NO ERROR
      024426    000240    NOP    ;RETURN HERE IF ERROR
      024430    104000    EMT    ;ERROR # DEFINED BY PUT SUBROUTINE
      024432    000137    025004    JMP     42$    ;GO TO 42$ IF ERROR
1821 024436    29$:
1822
1823          ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
      024436    004737    042774    JSR     PC,GETSTS    ;GO TO GETSTS SUBROUTINE
1824
1825          ;WAIT FOR COMMAND TO COMPLETE
      024442    004737    043672    JSR     PC,TIMOUT    ;GO TO TIMEOUT SUBROUTINE
1826
1827          ;GO READ STATUS FOR READ OPERATION
1828 024446    004737    043060    JSR     PC,GET    ;GO READ REGISTER(S) WITH GET SUBROUTINE
      024452    000404    BR        30$    ;GO TO 30$ IF NO ERROR
      024454    000240    NOP    ;RETURN HERE IF ERROR
      024456    104000    EMT    ;ERROR # DEFINED BY GET SUBROUTINE
      024460    000137    025004    JMP     42$    ;GO TO 42$ IF ERROR
1829 024464    30$:
1830
1831          ;CHECK FOR ERRORS DURING READ OPERATION
1832 024464    004737    044056    JSR     PC,PRIERR    ;GO CHECK FOR PRIMARY ERRORS
  
```


	024470	000405			BR	31\$	:GO TO 31\$ IF NO ERROR
	024472	000240			NOP		:RETURN HERE IF ERROR
	024474	104000			EMT		:ERROR # DEFINED BY PRIERR SUBROUTINE
	024476	004736			JSR	PC,@(SP)+	:GO BACK FOR MORE ERROR CHECKS
	024500	000137	025004		JMP	42\$	:GO TO 42\$ IF ERROR
1833	024504			31\$:			
1834	024504	005704			TST	R4	:IS THIS THE FIRST HEADER WORD ?
1835	024506	001006			BNE	32\$	:NO !!
1836							
1837	024510	032703	010000		BIT	#FMT16,R3	:SHOULD FER BE SET ?
1838	024514	001037			BNE	35\$	:YES !!
1839	024516	032703	140000		BIT	#MSE.USE,R3	:SHOULD BSE BE SET ?
1840	024522	001061			BNE	37\$	:YES !!
1841							
1842							:VERIFY THAT A HEADER COMPARE ERROR WAS DETECTED
1843	024524			32\$:			
1844	024524	032737	000200	001352	BIT	#HCE,RMER11	:WAS HCE DETECTED ??
1845	024532	001102			BNE	39\$	:YES !!
1846							
1847	024534	004737	056572		JSR	PC,DTASTS	:GO VERIFY RESULTS OF DATA TRANSFER
	024540	000405			BR	33\$	:GO TO 33\$ IF NO ERROR
	024542	000240			NOP		:RETURN HERE IF ERROR
	024544	104000			EMT		:ERROR # DEFINED BY DTASTS SUBROUTINE
	024546	004736			JSR	PC,@(SP)+	:GO BACK FOR MORE ERROR CHECKS
	024550	000137	025004		JMP	42\$	:GO TO 42\$ IF ERROR
1848	024554			33\$:			
1849	024554	013737	001352	001142	MOV	RMER11,\$BDDAT	:RECEIVED STATUS
1850	024562	012737	000200	001140	MOV	#HCE,\$GDDAT	:EXPECTED STATUS
1851	024570	010437	001174		MOV	R4,\$TMP0	:R4 = HEADER WORD NUMBER AND
1852	024574	001002			BNE	34\$	:ADJUST NUMBER.
1853	024576	005237	001174		INC	\$TMP0	
1854	024602	010337	001176		34\$:	MOV	R3,\$TMP1
1855	024606	104344			EMT	344	:R3 = BIT POSITION
1856	024610	000137	025004		JMP	42\$	
1857							
1858							:VERIFY THAT A FORMAT ERROR WAS DETECTED
1859	024614			35\$:			
1860	024614	032737	000020	001352	BIT	#FER,RMER11	:WAS FER DETECTED ??
1861	024622	001046			BNE	39\$	:YES !!
1862							
1863	024624	004737	056572		JSR	PC,DTASTS	:GO VERIFY RESULTS OF DATA TRANSFER
	024630	000405			BR	36\$	:GO TO 36\$ IF NO ERROR
	024632	000240			NOP		:RETURN HERE IF ERROR
	024634	104000			EMT		:ERROR # DEFINED BY DTASTS SUBROUTINE
	024636	004736			JSR	PC,@(SP)+	:GO BACK FOR MORE ERROR CHECKS
	024640	000137	025004		JMP	42\$	:GO TO 42\$ IF ERROR
1864	024644			36\$:			
1865	024644	012737	000020	001142	MOV	#FER,\$GDDAT	:EXPECTED STATUS
1866	024652	013737	001352		MOV	RMER11,\$BDDAT	:RECEIVED STATUS
1867	024660	104343			EMT	343	
1868	024662	000137	025004		JMP	42\$	
1869							
1870							:VERIFY THAT A BAD SECTOR ERROR WAS DETECTED
1871	024666			37\$:			
1872	024666	032737	100000	001400	BIT	#BSE,RMER21	:WAS BSE DETECTED ??
1873	024674	001021			BNE	39\$	:YES !!
1874							

```

1875 024676 004737 056572      JSR    PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
      024702 000405              BR      38$              ;GO TO 38$ IF NO ERROR
      024704 000240              NOP                      ;RETURN HERE IF ERROR
      024706 104000              EMT                      ;ERROR # DEFINED BY DTASTS SUBROUTINE
      024710 004736              JSR    PC,@(SP)+          ;GO BACK FOR MORE ERROR CHECKS
      024712 000137 025004      JMP     42$              ;GO TO 42$ IF ERROR

1876 024716                      38$:
1877 024716 013737 001400 001142  MOV     RMER2I,$BDDAT    ;RECEIVED STATUS
1878 024724 012737 100000 001140  MOV     #BSE,$GDDAT    ;EXPECTED STAIUS
1879 024732 104345              EMT      345
1880 024734 000137 000350      JMP     350
1881
1882                      ;CHECK FOR OTHER ERRORS
1883 024740                      39$:
1884 024740 004737 044710      JSR    PC,SECERR          ;GO CHECK FOR SECONDARY ERRORS
      024744 000405              BR      40$              ;GO TO 40$ IF NO ERROR
      024746 000240              NOP                      ;RETURN HERE IF ERROR
      024750 104000              EMT                      ;ERROR # DEFINED BY SECERR SUBROUTINE
      024752 004736              JSR    PC,@(SP)+          ;GO BACK FOR MORE ERROR CHECKS
      024754 000137 025004      JMP     42$              ;GO TO 42$ IF ERROR

1885 024760                      40$:
1886 024760 006303              ASL      R3                ;SHIFT TO NEXT BIT
1887 024762 001006              BNE     41$              ;REPEAT IF NOT DONE
1888
1889 024764 005704              TST      R4                ;SECOND HEADER WORD DONE??
1890 024766 001006              BNE     42$              ;YES!!
1891 024770 012704 000002      MOV     #2,R4              ;SETUP FOR 2ND HEADER WORD
1892 024774 012703 000001      MOV     #1,R3              ;SETUP FOR HCE
1893 025000                      41$:
1894 025000 000137 023346      JMP     2$
1895 025004                      42$:
1896 025004 000464              BR      46$              ;EXIT IF ERROR
1897                      ;*****
1898                      ;*REFORMAT SECTOR THAT WAS WRITTEN WITH BAD HEADER
1899                      ;*****
1900 025006                      43$:
1901 025006 012737 177776 001414  MOV     #-2,RMWCO          ;ONLY TWO HEAD WORDS
1902 025014 012737 010000 001444  MOV     #FMT16,RMOFO        ;ALWAYS IN 16 BITS MODE
1903 025022 012737 106436 001416  MOV     #BUFONE,RMBAO        ;BUFFER ADDRESS,REFORMAT THE SECTOR
1904 025030 012737 000062 001412  MOV     #WH,RMCS10        ;WRITE HEAD AND DATA COMMAND
1905 025036 004737 042174      JSR     PC,GENBUF          ;SET UP THE BUFFER
1906
1907 025042 004737 037316      JSR     PC,TSTPRP          ;PREPARE DEVICE FOR TEST
      025046 054130      .WORD    054130          ;TASK DESCRIPTOR AS FOLLOWS:
      ;CLEAR CONTROLLER & SELECT DEVICE
      ;VERIFY CONTROLLER CLEAR OPERATION
      ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
      ;VERIFY PACK ACKNOWLEDGE
      ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
      ;VERIFY RECALIBRATION
      025050 000404              BR      44$              ;GO TO 44$ IF NO ERROR
      025052 000240              NOP                      ;RETURN HERE IF ERROR
      025054 104000              EMT                      ;ERROR # DEFINED BY TSTPRP SUBROUTINE
      025056 000137 025062      JMP     44$              ;GO TO 44$ IF ERROR

1908 025062                      44$:
1909 025062 012737 000063 001412  MOV     #WH!GO,RMCS10        ;FORMAT THE SECTOR
1910 025070 012702 001553      MOV     #PUTINX,R2          ;SET UP THE REGS
  
```

1911	025074	112722	000006		MOVB	#RMDA,(R2)+	
1912	025100	112722	000032		MOVB	#RMOF,(R2)+	
1913	025104	112722	000034		MOVB	#RMDC,(R2)+	
1914	025110	112722	000004		MOVB	#RMB A,(R2)+	
1915	025114	112722	000002		MOVB	#RMWC,(R2)+	
1916	025120	112722	000000		MOVB	#RMCS1,(R2)+	
1917	025124	112722	000200		MOVB	#200,(R2)+	
1918	025130	004737	043330		JSR	PC,PUT	;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
	025134	000404			BR	45\$	;GO TO 45\$ IF NO ERROR
	025136	000240			NOP		;RETURN HERE IF ERROR
	025140	104000			EMI		;ERROR # DEFINED BY PUT SUBROUTINE
	025142	000137	025156		JMP	46\$	;GO TO 46\$ IF ERROR
1919	025146	000240		45\$:	NOP		
1920	025150	004737	043672		JSR	PC,TIMOUT	;WAIT FOR FINISH
1921	025154	000240			NOP		
1922	025156			46\$:			
1923							
1924							
;*****							
;*TEST 15 WRITE, READ W/ HCI							
;*****							
TST15:							
	025156				SCOPE		;SCOPE CALL
	025156	000004			NOP		;START OF TEST
	025160	000240			MOV	#STACK,SP	;INITIALIZE STACK POINTER
	025162	012706	001100		MOV	\$BASE,R0	;R0 = UNIBUS ADDRESS
	025166	013700	001276		MOV	TSTQUE,R1	;(R1) = DEVICE BEING TESTED
	025172	013701	001466		MOV	#15,\$TESTN	;SET TEST NUMBER IN APT MAIL BOX
	025176	012737	000015	001226			
1925							
1926							
1927							
1928							
1929	025204	012704	000000				
1930	025210	012703	000001				
1931	025214						
1932							
1933							
1934	025214	004737	037316				
	025220	154130			JSR	PC,TSTPRP	;PREPARE DEVICE FOR TEST
					.WORD	154130	;TASK DESCRIPTOR AS FOLLOWS:
							;SELECT DEVICE & VERIFY DEVICE AVAILABLE
							;CLEAR CONTROLLER & SELECT DEVICE
							;VERIFY CONTROLLER CLEAR OPERATION
							;PACK ACKNOWLEDGE IF VOLUME NOT VALID
							;VERIFY PACK ACKNOWLEDGE
							;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
							;VERIFY RECALIBRATION
	025222	000404			BR	3\$	;GO TO 3\$ IF NO ERROR
	025224	000240			NOP		;RETURN HERE IF ERROR
	025226	104000			EMI		;ERROR # DEFINED BY TSTPRP SUBROUTINE
	025230	000137	026562		JMP	41\$	;GO TO 41\$ IF ERROR
1935							
1936							
1937	025234						
1938	025234	012737	000000	001446	MOV	#0,RMDCO	;CYLINDER = 0
1939	025242	012737	000000	001420	MOV	#0,RMDAO	;TRACK = 0, SECTOR = 0
1940	025250	012737	106436	001416	MOV	#BUFONE,RMBAO	;BUS ADDRESS
1941	025256	012737	177376	001414	MOV	#-258.,RMWCO	;2 + 256. WORDS (2' COMP)
1942	025264	012737	010000	001444	MOV	#FMT16,RMOFO	;16 BIT FORMAT

```

1943 025272 012737 000063 001412      MOV      #WH!GO,RMCS10      ;WRITE HEADER AND DATA COMMAND
1944
1945      ;VERIFY THAT SECTOR IS NOT BAD
      JSR      PC,BADSCT      ;CALL BAD SECTOR MODULE
      BR      4$      ;GO TO 4$ IF NO ERROR
      TYPE     ,SCTMSG      ;TYPE BAD SECTOR MESSAGE
      EMT      ;ERROR # DEFINED BY BADSCT SUBROUTINE
      JMP      37$      ;GO TO 37$ IF ERROR
1946 025320      4$:
1947 025320 012737 072016 001174      MOV      #ONES,$TMP0      ;STARTING ADDRESS OF PATTERN
1948 025326 012737 000001 001176      MOV      #1,$TMP1      ;RANGE OF PATTERN
1949 025334 004737 042174      JSR      PC,GENBUF      ;GO GENERATE BUFFER FOR FORMAT
1950
1951      ;CHANGE HEADER WORD TO FORCE ERROR DURING WRITE
1952 025340 030364 106436      BIT      R3,BUFONE(R4)      ;SET OR RESET BIT??
1953 025344 001403      BEQ      5$
1954 025346 040364 106436      BIC      R3,BUFONE(R4)      ;RESET BIT
1955 025352 000402      BR      6$
1956 025354 050364 106436      5$:      BIS      R3,BUFONE(R4)      ;SET BIT
1957
1958      ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATION
1959 025360      6$:
1960 025360 012702 001553      MOV      #PUTINX,R2      ;R2 = BYTE ENTRY POSITION
1961 025364 112722 000034      MOVB     #RMDC,(R2)+
1962 025370 112722 000006      MOVB     #RMDA,(R2)+
1963 025374 112722 000004      MOVB     #RMBB,(R2)+
1964 025400 112722 000002      MOVB     #RMWC,(R2)+
1965 025404 112722 000032      MOVB     #RMOF,(R2)+
1966 025410 112722 000000      MOVB     #RMCS1,(R2)+
1967 025414 112712 000200      MOVB     #200,(R2)      ;TERMINATE TABLE
1968 025420      7$:
1969
1970      ;FORMAT THE DRIVE
1971 025420 004737 043330      JSR      PC,PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      025424 000404      BR      8$      ;GO TO 8$ IF NO ERROR
      025426 000240      NOP      ;RETURN HERE IF ERROR
      025430 104000      EMT      ;ERROR # DEFINED BY PUT SUBROUTINE
      025432 000137 026410      JMP      37$      ;GO TO 37$ IF ERROR
1972 025436      8$:
1973
1974      ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
      JSR      PC,GETSTS      ;GO TO GETSTS SUBROUTINE
1975
1976      ;WAIT FOR COMMAND TO COMPLETE
      JSR      PC,TIMOUT      ;GO TO TIMEOUT SUBROUTINE
1977
1978      ;GO READ STATUS FOR FORMAT OPERATION
1979 025446 004737 043060      JSR      PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      025452 000404      BR      9$      ;GO TO 9$ IF NO ERROR
      025454 000240      NOP      ;RETURN HERE IF ERROR
      025456 104000      EMT      ;ERROR # DEFINED BY GET SUBROUTINE
      025460 000137 026410      JMP      37$      ;GO TO 37$ IF ERROR
1980 025464      9$:
1981
1982      ;VERIFY NO ERRORS DURING FORMAT
1983 025464 004737 056572      JSR      PC,DASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
      025470 000405      BR      10$      ;GO TO 10$ IF NO ERROR

```

025472	000240			NOP		:RETURN HERE IF ERROR
025474	104000			EMT		:ERROR # DEFINED BY DTASTS SUBROUTINE
025476	004736			JSR	PC,@(SP)+	:GO BACK FOR MORE ERROR CHECKS
025500	000137	026410		JMP	37\$	:GO TO 37\$ IF ERROR
1984	025504			10\$:		
1985						
1986				:MOVE LOOP ADDRESSES TO NEXT OPERATION		
1987	025504	012737	025514 001124	MOV	#11\$,\$LPERR	
1988	025512	000410		BR	12\$	:SKIP TO WRITE OPERATION
1989						
1990				:*****		
1991				:LOOP #2	WRITE,READ	
1992						
1993	025514			11\$:		
1994						
1995				:PREPARE DEVICE FOR WRITE OPERATION		
1996	025514	004737	037316	JSR	PC,TSTPRP	:PREPARE DEVICE FOR TEST
	025520	154130		.WORD	154130	:TASK DESCRIPTOR AS FOLLOWS:
						:SELECT DEVICE & VERIFY DEVICE AVAILABLE
						:CLEAR CONTROLLER & SELECT DEVICE
						:VERIFY CONTROLLER CLEAR OPERATION
						:PACK ACKNOWLEDGE IF VOLUME NOT VALID
						:VERIFY PACK ACKNOWLEDGE
						:RECALIBRATE IF 'SKI' OR 'PIP' IS SET
						:VERIFY RECALIBRATION
						:GO TO 12\$ IF NO ERROR
						:RETURN HERE IF ERROR
						:ERROR # DEFINED BY TSTPRP SUBROUTINE
						:GO TO 37\$ IF ERROR
	025522	000404		BR	12\$	
	025524	000240		NOP		
	025526	104000		EMT		
	025530	000137	026410	JMP	37\$	
1997	025534			12\$:		
1998						
1999				:WRITE DATA TO THE DRIVE		
2000	025534			13\$:		
2001	025534	012737	012000 001444	MOV	#FMT16!HCI,RMOFO	:INHIBIT HEADER COMPARE
2002	025542	012737	000061 001412	MOV	#WD!GO,RMCS10	:WRITE DATA COMMAND
2003	025550	012737	106442 001416	MOV	#BUFONE+4,RMBA0	:LOAD STARTING BUFFER ADDRESS
2004	025556	012702	001553	MOV	#PUTINX,R2	:LOAD PUT REGISTER INDEX TABLE
2005	025562	112722	000006	MOVB	#RMDA,(R2)+	
2006	025566	112722	000034	MOVB	#RMDC,(R2)+	
2007	025572	112722	000032	MOVB	#RMOF,(R2)+	
2008	025576	112722	000004	MOVB	#RMBA,(R2)+	
2009	025602	112722	000002	MOVB	#RMWC,(R2)+	
2010	025606	112722	000000	MOVB	#RMCS1,(R2)+	
2011	025612	112722	000200	MOVB	#200,(R2)+	:TERMINATE TABLE
2012						
2013	025616	004737	043330	JSR	PC,PUT	:GO WRITE REGISTER(S) WITH PUT SUBROUTINE
	025622	000404		BR	14\$	:GO TO 14\$ IF NO ERROR
	025624	000240		NOP		:RETURN HERE IF ERROR
	025626	104000		EMT		:ERROR # DEFINED BY PUT SUBROUTINE
	025630	000137	026410	JMP	37\$	:GO TO 37\$ IF ERROR
2014	025634			14\$:		
2015						
2016				:SETUP GET INDEX TABLE TO READ ALL REGISTERS		
	025634	004737	042774	JSR	PC,GETSTS	:GO TO GETSTS SUBROUTINE
2017						
2018				:WAIT FOR COMMAND TO COMPLETE		
	025640	004737	043672	JSR	PC,TIMOUT	:GO TO TIMOUT SUBROUTINE

```

20*9
2020 ;GO READ STATUS FOR WRITE COMMAND
2021 025644 004737 043060 JSR PC,GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
    025650 000404 BR 15$ ;GO TO 15$ IF NO ERROR
    025652 000240 NOP ;RETURN HERE IF ERROR
    025654 104000 EMT ;ERROR # DEFINED BY GET SUBROUTINE
    025656 000137 026410 JMP 37$ ;GO TO 37$ IF ERROR
2022 025662 15$:
2023
2024 ;CHECK FOR ERRORS DURING WRITE OPERATION
2025 025662 004737 044056 JSR PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
    025666 000405 BR 16$ ;GO TO 16$ IF NO ERROR
    025670 000240 NOP ;RETURN HERE IF ERROR
    025672 104000 EMT ;ERROR # DEFINED BY PRIERR SUBROUTINE
    025674 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
    025676 000137 026410 JMP 37$ ;GO TO 37$ IF ERROR
2026 025702 005037 001140 16$: CLR $GDDAT ;EXPECTED STATUS
2027
2028 025706 032737 000220 001352 BIT #HCE!FER,RMER1I ;ANY ERROR?
2029 025714 001407 BEQ 17$
2030 025716 013737 001352 001142 MOV RMER1I,$BDDAT ;RECEIVED STATUS
2031 025724 042737 177557 001142 BIC #^C<HCE!FER>,$BDDAT ;CLEAR DONT CARES
2032 025732 000412 BR 18$
2033 025734 17$:
2034 025734 013737 001400 001142 MOV RMER2I,$BDDAT ;RECEIVED STATUS
2035 025742 032737 100000 001400 BIT #BSE,RMER2I ;ANY BAD SECTOR ERROR ?
2036 025750 001406 BEQ 19$ ;NO !!
2037 025752 042737 077777 001142 BIC #^CBSE,$BDDAT ;SAVE BSE FOR ERROR
2038 025760 18$:
2039 025760 104346 EMT 346
2040 025762 000137 026410 JMP 37$
2041 025766 19$:
2042
2043 025766 20$:
2044 025766 004737 056572 JSR PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
    025772 000405 BR 21$ ;GO TO 21$ IF NO ERROR
    025774 000240 NOP ;RETURN HERE IF ERROR
    025776 104000 EMT ;ERROR # DEFINED BY DTASTS SUBROUTINE
    026000 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
    026002 000137 026410 JMP 37$ ;GO TO 37$ IF ERROR
2045 026006 21$:
2046 026006 004737 044710 JSR PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
    026012 000405 BR 22$ ;GO TO 22$ IF NO ERROR
    026014 000240 NOP ;RETURN HERE IF ERROR
    026016 104000 EMT ;ERROR # DEFINED BY SECERR SUBROUTINE
    026020 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
    026022 000137 026410 JMP 37$ ;GO TO 37$ IF ERROR
2047 026026 22$:
2048
2049 ;CHANGE LOOP ADDRESSES
2050 026026 012737 026036 001124 MOV #23$,$LPERR
2051 026034 000410 BR 24$
2052
2053 ;*****
2054 ;LOOP #3 READ
2055
2056 026036 23$:
  
```



```

2057
2058
2059 026036 004737 037316      ;PREPARE DEVICE FOR READ OPERATION
      026042 154130      JSR PC,TSTPRP      ;PREPARE DEVICE FOR TEST
      .WORD 154130      .WORD 154130      ;TASK DESCRIPTOR AS FOLLOWS:
                                          ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
                                          ;CLEAR CONTROLLER & SELECT DEVICE
                                          ;VERIFY CONTROLLER CLEAR OPERATION
                                          ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                          ;VERIFY PACK ACKNOWLEDGE
                                          ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
                                          ;VERIFY RECALIBRATION
                                          ;GO TO 24$ IF NO ERROR
                                          ;RETURN HERE IF ERROR
                                          ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                          ;GO TO 37$ IF ERROR

      026044 000404      BR 24$
      026046 000240      NOP
      026050 104000      EMT
      026052 000137 026410    JMP 37$
2060 026056      24$:
2061
2062      ;READ DATA FROM DEVICE
2063 026056      25$:
2064 026056 012737 000071 001412    MOV #RD!GO,RMCS10      ;READ DATA COMMAND
2065 026064 012737 107446 001416    MOV #BUFTWO+4,RMBAO      ;LOAD STARTING BUFFER ADDRESS
2066 026072 012702 001553      MOV #PUTINX,R2      ;LOAD PUT REGISTER INDEX TABLE
2067 026076 112722 000006    MOVB #RMDA,(R2)+
2068 026102 112722 000032    MOVB #RMOF,(R2)+
2069 026106 112722 000034    MOVB #RMDC,(R2)+
2070 026112 112722 000004    MOVB #RMB A,(R2)+
2071 026116 112722 000002    MOVB #RMWC,(R2)+
2072 026122 112722 000000    MOVB #RMCS1,(R2)+
2073 026126 112712 000200    MOVB #200,(R2)
2074
2075 026132 004737 043330      JSR PC,PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      026136 000404      BR 26$      ;GO TO 26$ IF NO ERROR
      026140 000240      NOP      ;RETURN HERE IF ERROR
      026142 104000      EMT      ;ERROR # DEFINED BY PUT SUBROUTINE
      026144 000137 026410    JMP 37$      ;GO TO 37$ IF ERROR
2076 026150      26$:
2077
2078      ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
      026150 004737 042774      JSR PC,GETSTS      ;GO TO GETSTS SUBROUTINE
2079
2080      ;WAIT FOR COMMAND TO COMPLETE
      026154 004737 043672      JSR PC,TIMOUT      ;GO TO TIMEOUT SUBROUTINE
2081
2082      ;GO READ STATUS FOR READ OPERATION
2083 026160 004737 043060      JSR PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      026164 000404      BR 27$      ;GO TO 27$ IF NO ERROR
      026166 000240      NOP      ;RETURN HERE IF ERROR
      026170 104000      EMT      ;ERROR # DEFINED BY GET SUBROUTINE
      026172 000137 026410    JMP 37$      ;GO TO 37$ IF ERROR
2084 026176      27$:
2085
2086      ;CHECK FOR ERRORS DURING READ OPERATION
2087 026176 004737 044056      JSR PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
      026202 000405      BR 28$      ;GO TO 28$ IF NO ERROR
      026204 000240      NOP      ;RETURN HERE IF ERROR
      026206 104000      EMT      ;ERROR # DEFINED BY PRIERR SUBROUTINE
      026210 004736      JSR PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
  
```

```

2088 026212 000137 026410      JMP      37$      ;GO TO 37$ IF ERROR
2089 026216 005037 001140      CLR      $GDDAT      ;EXPECTED STATUS
2090 026222 032737 000220 001352      BIT      #HCE!FER,RMER1I      ;ANY ERROR ?
2091 026230 001407              BEQ      29$      ;NO!!
2092 026232 013737 001352 001142      MOV      RMER1I,$BDDAT      ;RECEIVED STATUS
2093 026240 042737 177557 001142      BIC      #^C<HCE.FER>,$BDDAT      ;CLEAR DONT CARES
2094 026246 000412              BR      30$
2095 026250              29$:      MOV      RMER2I,$BDDAT      ;RECEIVED STATUS
2096 026250 013737 001400 001142      BIT      #BSE,RMER2I      ;ANY BAD SECTOR ERROR ?
2097 026256 032737 100000 001400      BEQ      31$      ;NO !
2098 026264 001406              BIC      #^CBSE,$BDDAT      ;CLEAR DONT CARES
2099 026266 042737 077777 001142      30$:      EMT      346
2100 026274              JMP      37$
2101 026274 104346              31$:
2102 026276 000137 026410              32$:
2103 026302              JSR      PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
2104              BR      33$      ;GO TO 33$ IF NO ERROR
2105 026302 004737 056572      NOP      ;RETURN HERE IF ERROR
2106 026306 000405      EMT      ;ERROR # DEFINED BY DTASTS SUBROUTINE
2107 026310 000240      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
2108 026312 104000      JMP      37$      ;GO TO 37$ IF ERROR
2109 026314 004736      33$:      JSR      PC,SECERR      ;GO CHECK FOR SECONDARY ERRORS
2110 026316 000137 026410      BR      34$      ;GO TO 34$ IF NO ERROR
2111 026322              NOP      ;RETURN HERE IF ERROR
2112 026322 004737 044710      EMT      ;ERROR # DEFINED BY SECERR SUBROUTINE
2113 026326 000405      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
2114 026330 000240      JMP      37$      ;GO TO 37$ IF ERROR
2115 026332 104000      34$:      JSR      PC,CMPBUF      ;GO COMPARE WRITE, READ DATA BUFFERS
2116 026334 004736      .WORD      BUFOFF+4      ;STARTING ADDRESS OF WRITE BUFFER
2117 026336 000137 026410      .WORD      BUFTWO+4      ;STARTING ADDRESS OF READ BUFFER
2118 026342              BR      35$      ;GO TO 35$ IF NO ERROR
2119 026342 004737 042432      NOP      ;RETURN HERE IF ERROR
2120 026346 106442      EMT      ;ERROR # DEFINED BY CMPBUF SUBROUTINE
2121 026350 107446      JMP      37$      ;GO TO 37$ IF ERROR
2122 026352 000404      35$:      ASL      R3      ;SHIFT HCE BIT
2123 026354 000240      BNE      36$      ;CONTINUE IF NOT DONE
2124 026356 104000      TST      R4      ;SECOND HEADER DONE??
2125 026360 000137 026410      BNE      37$      ;YES.!
2126 026364 006303      MOV      #1,R3      ;START WITH BIT 0
2127 026366 001006      MOV      #2,R4      ;DO SECOND HEADER WORD
2128 026370 005704      36$:      JMP      2$
2129 026372 001006      37$:      BR      41$      ;EXIT IF ERROR
2130 026374 012703 000001      *****
2131 026400 012704 000002      ;*REFORMAT SECTOR THAT WAS WRITTEN WITH BAD HEADER
2132 026404              ;*****
2133 026410 000464      38$:      MOV      #-2,RMWCO      ;WORD COUNT
2134 026412 012737 177776 001414      MOV      #FMT16,RMOFO      ;IN 16 BIT MODE
2135 026420 012737 010000 001444

```


2128	026426	012737	106436	001416		MOV	#BUFONE,RMBAO	:BUFFER ADDRESS
2129	026434	012737	000062	001412		MOV	#WH,RMCSTO	:FORMAT COMMAND
2130	026442	004737	042174			JSR	PC,GENBUF	:SET UP BUFFER
2131								
2132	026446	004737	037316			JSR	PC,TSTPRP	:PREPARE DEVICE FOR TEST
	026452	054130				.WORD	054130	:TASK DESCRIPTOR AS FOLLOWS:
								:CLEAR CONTROLLER & SELECT DEVICE
								:VERIFY CONTROLLER CLEAR OPERATION
								:PACK ACKNOWLEDGE IF VOLUME NOT VALID
								:VERIFY PACK ACKNOWLEDGE
								:RECALIBRATE IF 'SKI' OR 'PIP' IS SET
								:VERIFY RECALIBRATION
	026454	000404				BR	39\$	:GO TO 39\$ IF NO ERROR
	026456	000240				NOP		:RETURN HERE IF ERROR
	026460	104000				EMT		:ERROR # DEFINED BY TSTPRP SUBROUTINE
	026462	000137	026466			JMP	39\$	:GO TO 39\$ IF ERROR
2133	026466				39\$:			
2134	026466	012737	000063	001412		MOV	#WH!GO,RMCSTO	
2135	026474	012702	001553			MOV	#PUTINX,R2	:TABLE ADDRESS
2136	026500	112722	000006			MOVB	#RMDA,(R2)+	
2137	026504	112722	000032			MOVB	#RMDF,(R2)+	
2138	026510	112722	000034			MOVB	#RMDC,(R2)+	
2139	026514	112722	000004			MOVB	#RMDA,(R2)+	
2140	026520	112722	000002			MOVB	#RMWC,(R2)+	
2141	026524	112722	000000			MOVB	#RMCST,(R2)+	
2142	026530	112722	000200			MOVB	#200,(R2)+	
2143	026534	004737	043330			JSR	PC,PUT	:GO WRITE REGISTER(S) WITH PUT SUBROUTINE
	026540	000404				BR	40\$	:GO TO 40\$ IF NO ERROR
	026542	000240				NOP		:RETURN HERE IF ERROR
	026544	104000				EMT		:ERROR # DEFINED BY PUT SUBROUTINE
	026546	000137	026562			JMP	41\$	:GO TO 41\$ IF ERROR
2144	026552	000240			40\$:	NOP		
2145	026554	004737	043672			JSR	PC,TIMOUT	
2146	026560	000240				NOP		
2147	026562				41\$:			
2148								
2149								
								::*****
								::*TEST 16 WRITE, READ W/ IVC ERROR
								::*****
	026562				TST16:			
	026562	000004				SCOPE		:SCOPE CALL
	026564	000240				NOP		:START OF TEST
	026566	012706	001100			MOV	#STACK,SP	:INITIALIZE STACK POINTER
	026572	013700	001276			MOV	\$BASE,R0	:R0 = UNIBUS ADDRESS
	026576	013701	001466			MOV	TSTQUE,R1	:(R1) = DEVICE BEING TESTED
	026602	012737	000016	001226		MOV	#16,\$TESTN	::SET TEST NUMBER IN APT MAIL BOX
2150								
2151								::*****
2152								::LOOP #1 WRITE,READ
2153								
2154								
2155	026610							:LOAD PARAMETERS AND GENERATE DATA BUFFER
					1\$:			
2156	026610	012737	000000	001446		MOV	#0,RMDCO	:CYLINDER = 0
2157	026616	012737	000000	001420		MOV	#0,RMDAO	:TRACK = 0, SECTOR = 0
2158	026624	012737	106436	001416		MOV	#BUFONE,RMBAO	:BUS ADDRESS
2159	026632	012737	177400	001414		MOV	#-256.,RMWCO	:256. WORDS (2'S COMP)
2160	026640	012737	010000	001444		MOV	#FMT16,RMOFO	:16 BIT FORMAT

```

2161 026646 012737 000060 001412      MOV      #WD,RMCS10      ;WRITE HEADER AND DATA COMMAND
2162
2163      ;VERIFY THAT SECTOR IS NOT BAD
      026654 004737 040242      JSR      PC,BADSCT      ;CALL BAD SECTOR MODULE
      026660 000405      BR      2$      ;GO TO 2$ IF NO ERROR
      026662 104401 070426      TYPE      ,SCTMSG      ;TYPE BAD SECTOR MESSAGE
      026666 104000      EMT      ;ERROR # DEFINED BY BADSCT SUBROUTINE
      026670 000137 027602      JMP      22$      ;GO TO 22$ IF ERROR
2164 026674
2165 026674 012737 001420 001174      MOV      #RMDAO,$TMP0      ;USE SECTOR FOR DATA PATTERN
2166 026702 012737 000001 001176      MOV      #1,$TMP1
2167 026710 004737 042174      JSR      PC,GENBUF      ;GO GENERATE DATA BUFFER
2168
2169      ;PREPARE DEVICE FOR WRITE OPERATION
2170 026714 004737 037316      JSR      PC,TSTPRP      ;PREPARE DEVICE FOR TEST
      026720 154130      .WORD      154130      ;TASK DESCRIPTOR AS FOLLOWS:
      ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
      ;CLEAR CONTROLLER & SELECT DEVICE
      ;VERIFY CONTROLLER CLEAR OPERATION
      ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
      ;VERIFY PACK ACKNOWLEDGE
      ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
      ;VERIFY RECALIBRATION
      026722 000404      BR      3$      ;GO TO 3$ IF NO ERROR
      026724 000240      NOP      ;RETURN HERE IF ERROR
      026726 104000      EMT      ;ERROR # DEFINED BY TSTPRP SUBROUTINE
      026730 000137 027602      JMP      22$      ;GO TO 22$ IF ERROR
2171 026734
2172
2173      ;RESET VOLUME VALID
2174 026734 112737 000024 001553      MOV      #RMMR1,PUTINX      ;SETUP PUT INDEX TABLE
      026742 112737 000200 001554      MOV      #200,PUTINX+1      ;SET TERMINATOR BYTE
      026750 012737 000001 001436      MOV      #DMD,RMMR10      ;SET RMMR1 OUTPUT BUFFER - DMD
      026756 004737 043330      JSR      PC,PUT      ;GO WRITE RMMR1 VIA PUT SUBROUTINE
      026762 000404      BR      4$      ;GO TO 4$ IF NO ERROR
      026764 000240      NOP      ;RETURN HERE IF ERROR
      026766 104000      EMT      ;ERROR DEFINED BY PUT SUBROUTINE
      026770 000137 027602      JMP      22$      ;GO TO 22$ IF ERROR
2175 026774
2176
2177      ;SETUP PARAMETERS AND EXECUTE WRITE DATA COMMAND
2178 026774
2179 026774 012737 000061 001412      MOV      #WD!GO,RMCS10      ;WRITE DATA COMMAND
2180 027002 012737 000000 001436      MOV      #0,RMMR10      ;RESET DIAGNOSTIC MODE
2181 027010 012702 001553      MOV      #PUTINX,R2      ;LOAD PUT REGISTER INDEX TABLE
2182 027014 112722 000024      MOV      #RMMR1,(R2)+
2183 027020 112722 000006      MOV      #RMDA,(R2)+
2184 027024 112722 000034      MOV      #RMDC,(R2)+
2185 027030 112722 000032      MOV      #RMOF,(R2)+
2186 027034 112722 000002      MOV      #RMWC,(R2)+
2187 027040 112722 000004      MOV      #RMB A,(R2)+
2188 027044 112722 000000      MOV      #RMCS1,(R2)+
2189 027050 112722 000200      MOV      #200,(R2)+      ;TERMINATE TABLE
2190
2191 027054 004737 043330      JSR      PC,PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      027060 000404      BR      6$      ;GO TO 6$ IF NO ERROR
      027062 000240      NOP      ;RETURN HERE IF ERROR

```

```

027064 104000 EMT ;ERROR # DEFINED BY PUT SUBROUTINE
027066 000137 027602 JMP 22$ ;GO TO 22$ IF ERROR
2192 027072 6$:
2193
2194 ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
027072 004737 042774 JSR PC,GETSTS ;GO TO GETSTS SUBROUTINE
2195
2196 ;WAIT FOR COMMAND TO COMPLETE
027076 004737 043672 JSR PC,TIMOUT ;GO TO TIMOUT SUBROUTINE
2197
2198 ;GO READ STATUS FOR WRITE COMMAND
2199 027102 004737 043060 JSR PC,GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
027106 000404 BR 7$ ;GO TO 7$ IF NO ERROR
027110 000240 NOP ;RETURN HERE IF ERROR
027112 104000 EMT ;ERROR # DEFINED BY GET SUBROUTINE
027114 000137 027602 JMP 22$ ;GO TO 22$ IF ERROR
2200 027120 7$:
2201
2202 ;CHECK FOR ERRORS DURING WRITE OPERATION
2203 027120 004737 044056 JSR PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
027124 000405 BR 8$ ;GO TO 8$ IF NO ERROR
027126 000240 NOP ;RETURN HERE IF ERROR
027130 104000 EMT ;ERROR # DEFINED BY PRIERR SUBROUTINE
027132 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
027134 000137 027602 JMP 22$ ;GO TO 22$ IF ERROR
2204 027140 8$:
2205 027140 032737 010000 001400 BIT #IVC,RMER2I ;WAS "IVC" DETECTED??
2206 027146 001024 BNE 10$ ;YES!
2207
2208 027150 004737 056572 JSR PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
027154 000405 BR 9$ ;GO TO 9$ IF NO ERROR
027156 000240 NOP ;RETURN HERE IF ERROR
027160 104000 EMT ;ERROR # DEFINED BY DTASTS SUBROUTINE
027162 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
027164 000137 027602 JMP 22$ ;GO TO 22$ IF ERROR
2209 027170 9$:
2210 027170 013737 001400 001142 MOV RMER2I,$BDDAT ;RECEIVED STATUS
2211 027176 013737 001400 001140 MOV RMER2I,$GDDAT ;EXPECTED STATUS
2212 027204 052737 010000 001140 BIS #IVC,$GDDAT
2213 027212 104342 EMT 342
2214 027214 000137 027602 JMP 22$
2215 027220 10$:
2216 027220 004737 044710 JSR PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
027224 000405 BR 11$ ;GO TO 11$ IF NO ERROR
027226 000240 NOP ;RETURN HERE IF ERROR
027230 104000 EMT ;ERROR # DEFINED BY SECERR SUBROUTINE
027232 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
027234 000137 027602 JMP 22$ ;GO TO 22$ IF ERROR
2217 027240 11$:
2218
2219 ;CHANGE LOOP ADDRESSES
2220 027240 012737 027250 001124 MOV #12$,$LPERR
2221 027246 000410 BR 13$
2222
2223 ;*****
2224 ;LOOP #2 READ
2225

```

```

2226 027250      12$:
2227
2228
2229 027250 004737 037316      ;PREPARE DEVICE FOR READ OPERATION
      027254 154130      JSR PC,TSTPRP      ;PREPARE DEVICE FOR TEST
      .WORD 154130      .WORD 154130      ;TASK DESCRIPTOR AS FOLLOWS:
      ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
      ;CLEAR CONTROLLER & SELECT DEVICE
      ;VERIFY CONTROLLER CLEAR OPERATION
      ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
      ;VERIFY PACK ACKNOWLEDGE
      ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
      ;VERIFY RECALIBRATION
      ;GO TO 13$ IF NO ERROR
      ;RETURN HERE IF ERROR
      ;ERROR # DEFINED BY TSTPRP SUBROUTINE
      ;GO TO 22$ IF ERROR

      027256 000404      BR 13$
      027260 000240      NOP
      027262 104000      EMT
      027264 000137 027602  JMP 22$

2230 027270      13$:
2231
2232      ;RESET VOLUME VALID
2233 027270 112737 000024 001553  MOVB #RMMR1,PUTINX      ;SETUP PUT INDEX TABLE
      027276 112737 000200 001554  MOVB #200,PUTINX+1      ;SET TERMINATOR BYTE
      027304 012737 000001 001436  MOV #DMD,RMMR10      ;SET RMMR1 OUTPUT BUFFER - DMD
      027312 004737 043330      JSR PC,PUT      ;GO WRITE RMMR1 VIA PUT SUBROUTINE
      027316 000404      BR 14$      ;GO TO 14$ IF NO ERROR
      027320 000240      NOP      ;RETURN HERE IF ERROR
      027322 104000      EMT      ;ERROR DEFINED BY PUT SUBROUTINE
      027324 000137 027602  JMP 22$      ;GO TO 22$ IF ERROR
2234 027330
2235
2236      ;READ DATA FROM DEVICE
2237 027330      15$:
2238 027330 012737 000071 001412  MOV #RD!GO,RMCS10      ;READ DATA COMMAND
2239 027336 012737 107442 001416  MOV #BUFTWO,RMBA0      ;LOAD STARTING BUFFER ADDRESS
2240 027344 012737 000000 001436  MOV #0,RMMR10      ;RESET DIANOSTIC MODE
2241 027352 012702 001553      MOV #PUTINX,R2      ;LOAD PUT REGISTER INDEX TABLE
2242 027356 112722 000024      MOVB #RMMR1,(R2)+
2243 027362 112722 000006      MOVB #RMDA,(R2)+
2244 027366 112722 000032      MOVB #RMOF,(R2)+
2245 027372 112722 000034      MOVB #RMDC,(R2)+
2246 027376 112722 000004      MOVB #RMBA,(R2)+
2247 027402 112722 000002      MOVB #RMWC,(R2)+
2248 027406 112722 000000      MOVB #RMCS1,(R2)+
2249 027412 112712 000200      MOVB #200,(R2)
2250
2251 027416 004737 043330      JSR PC,PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      027422 000404      BR 16$      ;GO TO 16$ IF NO ERROR
      027424 000240      NOP      ;RETURN HERE IF ERROR
      027426 104000      EMT      ;ERROR # DEFINED BY PUT SUBROUTINE
      027430 000137 027602  JMP 22$      ;GO TO 22$ IF ERROR

2252 027434      16$:
2253
2254      ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
      027434 004737 042774  JSR PC,GETSTS      ;GO TO GETSTS SUBROUTINE

2255
2256      ;WAIT FOR COMMAND TO COMPLETE
      027440 004737 043672  JSR PC,TIMOUT      ;GO TO TIMOUT SUBROUTINE
2257

```

```

2258      ;GO READ STATUS FOR READ OPERATION
2259 027444 004737 043060      JSR      PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      027450 000404      BR      17$      ;GO TO 17$ IF NO ERROR
      027452 000240      NOP      ;RETURN HERE IF ERROR
      027454 104000      EMT      ;ERROR # DEFINED BY GET SUBROUTINE
      027456 000137 027602      JMP      22$      ;GO TO 22$ IF ERROR
2260 027462      17$:
2261
2262      ;CHECK FOR ERRORS DURING READ OPERATION
2263 027462 004737 044056      JSR      PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
      027466 000405      BR      18$      ;GO TO 18$ IF NO ERROR
      027470 000240      NOP      ;RETURN HERE IF ERROR
      027472 104000      EMT      ;ERROR # DEFINED BY PRIERR SUBROUTINE
      027474 004736      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      027476 000137 027602      JMP      22$      ;GO TO 22$ IF ERROR
2264 027502      18$:
2265 027502 032737 010000 001400      BIT      #IVC,RMER2I      ;WAS 'IVC' DETECTED??
2266 027510 001024      BNE      20$      ;YES!!
2267
2268 027512 004737 056572      JSR      PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
      027516 000405      BR      19$      ;GO TO 19$ IF NO ERROR
      027520 000240      NOP      ;RETURN HERE IF ERROR
      027522 104000      EMT      ;ERROR # DEFINED BY DTASTS SUBROUTINE
      027524 004736      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      027526 000137 027602      JMP      22$      ;GO TO 22$ IF ERROR
2269 027532      19$:
2270 027532 013737 001400 001140      MOV      RMER2I,$GDDAT      ;EXPECTED STATUS
2271 027540 052737 010000 001140      BIS      #IVC,$GDDAT
2272 027546 013737 001400 001142      MOV      RMER2I,$BDDAT      ;RECEIVED STATUS
2273 027554 104342      EMT      342
2274 027556 000137 027602      JMP      22$
2275 027562      20$:
2276 027562 004737 044710      JSR      PC,SECERR      ;GO CHECK FOR SECONDARY ERRORS
      027566 000405      BR      21$      ;GO TO 21$ IF NO ERROR
      027570 000240      NOP      ;RETURN HERE IF ERROR
      027572 104000      EMT      ;ERROR # DEFINED BY SECERR SUBROUTINE
      027574 004736      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      027576 000137 027602      JMP      22$      ;GO TO 22$ IF ERROR
2277 027602      21$:
2278
2279 027602      22$:
2280
2281      ;*****
      ;*TEST 17      WRITE, READ W/ ABORT
      ;*****
      TST17:
      027602 000004      SCOPE      ;SCOPE CALL
      027604 000240      NOP      ;START OF TEST
      027606 012706 001100      MOV      #STACK,SP      ;INITIALIZE STACK POINTER
      027612 013700 001276      MOV      $BASE,R0      ;R0 = UNIBUS ADDRESS
      027616 013701 001466      MOV      TSTQUE,R1      ;(R1) = DEVICE BEING TESTED
      027622 012737 000017 001226      MOV      #17,$TESTN      ;SET TEST NUMBER IN APT MAIL BOX
2282
2283      ;*****
2284      ;LOOP #1      WRITE,READ
2285
2286      ;LOAD PARAMETERS AND GENERATE DATA BUFFER

```

2287	027630	012737	000000	001446	MOV	#0,RMDCO	;CYLINDER = 0
2288	027636	012737	000000	001420	MOV	#0,RMDAO	;TRACK = 0, SECTOR = 0
2289	027644	012737	106436	001416	MOV	#BUFONE,RMBAO	;BUS ADDRESS
2290	027652	012737	177400	001414	MOV	#-256,RMWCO	;256. WORDS (2'S COMP)
2291	027660	012737	010000	001444	MOV	#FMT16,RMOFO	;16 BIT FORMAT
2292	027666	012737	000061	001412	MOV	#WD!GO,RMCS10	;WRITE DATA COMMAND
2293	027674				1\$:		
2294							
2295							
2296	027674	004737	037316				
	027700	154130					
					JSR	PC,TSTPRP	;PREPARE DEVICE FOR TEST
					.WORD	154130	;TASK DESCRIPTOR AS FOLLOWS:
							;SELECT DEVICE & VERIFY DEVICE AVAILABLE
							;CLEAR CONTROLLER & SELECT DEVICE
							;VERIFY CONTROLLER CLEAR OPERATION
							;PACK ACKNOWLEDGE IF VOLUME NOT VALID
							;VERIFY PACK ACKNOWLEDGE
							;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
							;VERIFY RECALIBRATION
							;GO TO 2\$ IF NO ERROR
							;RETURN HERE IF ERROR
							;ERROR # DEFINED BY TSTPRP SUBROUTINE
							;GO TO 21\$ IF ERROR
	027702	000404			BR	2\$	
	027704	000240			NOP		
	027706	104000			EMT		
	027710	000137	030520		JMP	21\$	
2297	027714				2\$:		
2298							
2299							
2300	027714	112737	000014	001553	MOV	#RMER1,PUTINX	;SETUP PUT INDEX TABLE
	027722	112737	000200	001554	MOV	#200,PUTINX+1	;SET TERMINATOR BYTE
	027730	012737	040000	001426	MOV	#UNS,RMER10	;SET RMER1 OUTPUT BUFFER = UNS
	027736	004737	043330		JSR	PC,PUT	;GO WRITE RMER1 VIA PUT SUBROUTINE
	027742	000404			BR	3\$	;GO TO 3\$ IF NO ERROR
	027744	000240			NOP		;RETURN HERE IF ERROR
	027746	104000			EMT		;ERROR DEFINED BY PUT SUBROUTINE
	027750	000137	030520		JMP	21\$	;GO TO 21\$ IF ERROR
2301	027754				3\$:		
2302							
2303							
2304	027754						
2305	027754	012737	000061	001412	MOV	#WD!GO,RMCS10	;WRITE DATA COMMAND
2306	027762	012702	001553		MOV	#PUTINX,R2	;LOAD PUT REGISTER INDEX TABLE
2307	027766	112722	000006		MOVB	#RMDA,(R2)+	
2308	027772	112722	000034		MOVB	#RMDC,(R2)+	
2309	027776	112722	000032		MOVB	#RMOF,(R2)+	
2310	030002	112722	000004		MOVB	#RMBA,(R2)+	
2311	030006	112722	000002		MOVB	#RMWC,(R2)+	
2312	030012	112722	000000		MOVB	#RMCS1,(R2)+	
2313	030016	112722	000200		MOVB	#200,(R2)+	;TERMINATE TABLE
2314							
2315	030022	004737	043330		JSR	PC,PUT	;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
	030026	000404			BR	5\$	;GO TO 5\$ IF NO ERROR
	030030	000240			NOP		;RETURN HERE IF ERROR
	030032	104000			EMT		;ERROR # DEFINED BY PUT SUBROUTINE
	030034	000137	030520		JMP	21\$	;GO TO 21\$ IF ERROR
2316	030040				5\$:		
2317							
2318							
	030040	004737	042774				
2319							
					JSR	PC,GETSTS	;GO TO GETSTS SUBROUTINE


```

2320 ;GO GET REGISTER STATUS
2321 030044 004737 043060 JSR PC,GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
030050 000404 BR 6$ ;GO TO 6$ IF NO ERROR
030052 000240 NOP ;RETURN HERE IF ERROR
030054 104000 EMT ;ERROR # DEFINED BY GET SUBROUTINE
030056 000137 030520 JMP 21$ ;GO TO 21$ IF ERROR

2322 030062 6$:
2323 030062 032737 020000 001350 BIT #PIP,RMDSI ;WAS COMMAND EXECUTED?
2324 030070 001412 BEQ 7$ ;NO, GO WAIT
2325 030072 013737 001350 001140 MOV RMDSI,$GDDAT ;EXPECTED STATUS
2326 030100 042737 020000 001140 BIC #PIP,$GDDAT
2327 030106 013737 001350 001142 MOV RMDSI,$BDDAT ;RECEIVED STATUS
2328 030114 104347 EMT 347
2329 030116 7$:
2330
2331 ;WAIT FOR COMMAND TO COMPLETE
JSR PC,TIMOUT ;GO TO TIMEOUT SUBROUTINE

2332
2333 ;GO GET REGISTER STATUS
2334 030122 004737 043060 JSR PC,GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
030126 000404 BR 8$ ;GO TO 8$ IF NO ERROR
030130 000240 NOP ;RETURN HERE IF ERROR
030132 104000 EMT ;ERROR # DEFINED BY GET SUBROUTINE
030134 000137 030520 JMP 21$ ;GO TO 21$ IF ERROR

2335 030140 8$:
2336 030140 004737 044056 JSR PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
030144 000405 BR 9$ ;GO TO 9$ IF NO ERROR
030146 000240 NOP ;RETURN HERE IF ERROR
030150 104000 EMT ;ERROR # DEFINED BY PRIERR SUBROUTINE
030152 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
030154 000137 030520 JMP 21$ ;GO TO 21$ IF ERROR

2337 030160 9$:
2338 030160 004737 044710 JSR PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
030164 000405 BR 10$ ;GO TO 10$ IF NO ERROR
030166 000240 NOP ;RETURN HERE IF ERROR
030170 104000 EMT ;ERROR # DEFINED BY SECERR SUBROUTINE
030172 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
030174 000137 030520 JMP 21$ ;GO TO 21$ IF ERROR

2339 030200 10$:
2340
2341 ;CHANGE LOOP ADDRESSES
2342 030200 012737 030210 001124 MOV #11$,SLPERR
2343 030206 000410 BR 12$
2344
2345 ;*****
2346 ;LOOP #2 READ
2347
2348 030210 11$:
2349
2350 ;PREPARE DEVICE FOR READ OPERATION
2351 030210 004737 037316 JSR PC,TSTPRP ;PREPARE DEVICE FOR TEST
030214 154130 .WORD 154130 ;TASK DESCRIPTOR AS FOLLOWS:
;SELECT DEVICE & VERIFY DEVICE AVAILABLE
;CLEAR CONTROLLER & SELECT DEVICE
;VERIFY CONTROLLER CLEAR OPERATION
;PACK ACKNOWLEDGE IF VOLUME NOT VALID
;VERIFY PACK ACKNOWLEDGE

```

```

030216 000404 BR 12$
030220 000240 NOP
030222 104000 EMT
030224 000137 030520 JMP 21$
2352 030230 12$:
2353
2354 ;SET UNSAFE ERROR
2355 030230 112737 000014 001553 MOV #RMER1,PUTINX ;SETUP PUT INDEX TABLE
030236 112737 000200 001554 MOV #200,PUTINX+1 ;SET TERMINATOR BYTE
030244 012737 040000 001426 MOV #UNS,RMER10 ;SET RMER1 OUTPUT BUFFER = UNS
030252 004737 043330 JSR PC,PUT ;GO WRITE RMER1 VIA PUT SUBROUTINE
030256 000404 BR 13$ ;GO TO 13$ IF NO ERROR
030260 000240 NOP ;RETURN HERE IF ERROR
030262 104000 EMT ;ERROR DEFINED BY PUT SUBROUTINE
030264 000137 030520 JMP 21$ ;GO TO 21$ IF ERROR
2356 030270 13$:
2357
2358 ;READ DATA FROM DEVICE
2359 030270 14$:
2360 030270 012737 000071 001412 MOV #RD!GO,RMCS10 ;READ DATA COMMAND
2361 030276 012702 001553 MOV #PUTINX,R2 ;LOAD PUT REGISTER INDEX TABLE
2362 030302 112722 000006 MOV #RMDA,(R2)+
2363 030306 112722 000032 MOV #RMOF,(R2)+
2364 030312 112722 000034 MOV #RMDC,(R2)+
2365 030316 112722 000004 MOV #RMB A,(R2)+
2366 030322 112722 000002 MOV #RMWC,(R2)+
2367 030326 112722 000000 MOV #RMCS1,(R2)+
2368 030332 112712 000200 MOV #200,(R2)
2369
2370 030336 004737 043330 JSR PC,PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
030342 000404 BR 15$ ;GO TO 15$ IF NO ERROR
030344 000240 NOP ;RETURN HERE IF ERROR
030346 104000 EMT ;ERROR # DEFINED BY PUT SUBROUTINE
030350 000137 030520 JMP 21$ ;GO TO 21$ IF ERROR
2371 030354 15$:
2372
2373 ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
030354 004737 042774 JSR PC,GETSTS ;GO TO GETSTS SUBROUTINE
2374
2375 ;SEE IF DEVICE STARTED COMMAND
2376 030360 004737 043060 JSR PC,GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
030364 000404 BR 16$ ;GO TO 16$ IF NO ERROR
030366 000240 NOP ;RETURN HERE IF ERROR
030370 104000 EMT ;ERROR # DEFINED BY GET SUBROUTINE
030372 000137 030520 JMP 21$ ;GO TO 21$ IF ERROR
2377 030376 16$:
2378 030376 032737 020000 001350 BIT #PIP,RMDSI ;WAS COMMAND EXECUTED??
2379 030404 001414 BEQ 17$ ;NO!!
2380 030406 013737 001350 001140 MOV RMDSI,$GDDAT ;EXPECTED STATUS
2381 030414 042737 020000 001140 BIC #PIP,$GDDAT
2382 030422 013737 001350 001142 MOV RMDSI,$BDDAT ;RECEIVED STATUS
2383 030430 104347 EMT 347
2384 030432 000137 030520 JMP 21$
2385 030436 17$:
2386

```


Address	Op Code	Op 1	Op 2	Op 3	Op 4	Op 5	Op 6	Op 7	Op 8	Op 9	Op 10	Op 11	Op 12	Op 13	Op 14	Op 15	Op 16	Op 17	Op 18	Op 19	Op 20	Op 21	Op 22	Op 23	Op 24	Op 25	Op 26	Op 27	Op 28	Op 29	Op 30	Op 31	Op 32	Op 33	Op 34	Op 35	Op 36	Op 37	Op 38	Op 39	Op 40	Op 41	Op 42	Op 43	Op 44	Op 45	Op 46	Op 47	Op 48	Op 49	Op 50	Op 51	Op 52	Op 53	Op 54	Op 55	Op 56	Op 57	Op 58	Op 59	Op 60	Op 61	Op 62	Op 63	Op 64	Op 65	Op 66	Op 67	Op 68	Op 69	Op 70	Op 71	Op 72	Op 73	Op 74	Op 75	Op 76	Op 77	Op 78	Op 79	Op 80	Op 81	Op 82	Op 83	Op 84	Op 85	Op 86	Op 87	Op 88	Op 89	Op 90	Op 91	Op 92	Op 93	Op 94	Op 95	Op 96	Op 97	Op 98	Op 99	Op 100	Op 101	Op 102	Op 103	Op 104	Op 105	Op 106	Op 107	Op 108	Op 109	Op 110	Op 111	Op 112	Op 113	Op 114	Op 115	Op 116	Op 117	Op 118	Op 119	Op 120	Op 121	Op 122	Op 123	Op 124	Op 125	Op 126	Op 127	Op 128	Op 129	Op 130	Op 131	Op 132	Op 133	Op 134	Op 135	Op 136	Op 137	Op 138	Op 139	Op 140	Op 141	Op 142	Op 143	Op 144	Op 145	Op 146	Op 147	Op 148	Op 149	Op 150	Op 151	Op 152	Op 153	Op 154	Op 155	Op 156	Op 157	Op 158	Op 159	Op 160	Op 161	Op 162	Op 163	Op 164	Op 165	Op 166	Op 167	Op 168	Op 169	Op 170	Op 171	Op 172	Op 173	Op 174	Op 175	Op 176	Op 177	Op 178	Op 179	Op 180	Op 181	Op 182	Op 183	Op 184	Op 185	Op 186	Op 187	Op 188	Op 189	Op 190	Op 191	Op 192	Op 193	Op 194	Op 195	Op 196	Op 197	Op 198	Op 199	Op 200	Op 201	Op 202	Op 203	Op 204	Op 205	Op 206	Op 207	Op 208	Op 209	Op 210	Op 211	Op 212	Op 213	Op 214	Op 215	Op 216	Op 217	Op 218	Op 219	Op 220	Op 221	Op 222	Op 223	Op 224	Op 225	Op 226	Op 227	Op 228	Op 229	Op 230	Op 231	Op 232	Op 233	Op 234	Op 235	Op 236	Op 237	Op 238	Op 239	Op 240	Op 241	Op 242	Op 243	Op 244	Op 245	Op 246	Op 247	Op 248	Op 249	Op 250	Op 251	Op 252	Op 253	Op 254	Op 255	Op 256	Op 257	Op 258	Op 259	Op 260	Op 261	Op 262	Op 263	Op 264	Op 265	Op 266	Op 267	Op 268	Op 269	Op 270	Op 271	Op 272	Op 273	Op 274	Op 275	Op 276	Op 277	Op 278	Op 279	Op 280	Op 281	Op 282	Op 283	Op 284	Op 285	Op 286	Op 287	Op 288	Op 289	Op 290	Op 291	Op 292	Op 293	Op 294	Op 295	Op 296	Op 297	Op 298	Op 299	Op 300	Op 301	Op 302	Op 303	Op 304	Op 305	Op 306	Op 307	Op 308	Op 309	Op 310	Op 311	Op 312	Op 313	Op 314	Op 315	Op 316	Op 317	Op 318	Op 319	Op 320	Op 321	Op 322	Op 323	Op 324	Op 325	Op 326	Op 327	Op 328	Op 329	Op 330	Op 331	Op 332	Op 333	Op 334	Op 335	Op 336	Op 337	Op 338	Op 339	Op 340	Op 341	Op 342	Op 343	Op 344	Op 345	Op 346	Op 347	Op 348	Op 349	Op 350	Op 351	Op 352	Op 353	Op 354	Op 355	Op 356	Op 357	Op 358	Op 359	Op 360	Op 361	Op 362	Op 363	Op 364	Op 365	Op 366	Op 367	Op 368	Op 369	Op 370	Op 371	Op 372	Op 373	Op 374	Op 375	Op 376	Op 377	Op 378	Op 379	Op 380	Op 381	Op 382	Op 383	Op 384	Op 385	Op 386	Op 387	Op 388	Op 389	Op 390	Op 391	Op 392	Op 393	Op 394	Op 395	Op 396	Op 397	Op 398	Op 399	Op 400	Op 401	Op 402	Op 403	Op 404	Op 405	Op 406	Op 407	Op 408	Op 409	Op 410	Op 411	Op 412	Op 413	Op 414	Op 415	Op 416	Op 417	Op 418

```

2409 ;LOAD PARAMETERS AND GENERATE DATA BUFFER
2410 030566 012737 000000 001446 MOV #0,RMDCO ;CYLINDER 0
2411 030574 012737 000000 001420 3$: MOV #0,RMDAO ;TRACK = 0, SECTOR = 0
2412 030602 012737 106436 001416 MOV #BUFONE,RMBAD ;BUS ADDRESS
2413 030610 012737 177376 001414 MOV #-258,RMWCO ;2 + 256. WORDS (2'S COMP)
2414 030616 012737 010000 001444 MOV #FMT16,RMOFO ;16 BIT FORMAT
2415 030624 012737 000063 001412 MOV #WH!GO,RMCS10 ;WRITE HEADER AND DATA COMMAND
2416
2417 ;VERIFY THAT SECTOR IS NOT BAD
      030632 004737 040242 JSR PC,BADSCT ;CALL BAD SECTOR MODULE
      030636 000405 BR 4$ ;GO TO 4$ IF NO ERROR
      030640 104401 070426 TYPE ,SCTMSG ;TYPE BAD SECTOR MESSAGE
      030644 104000 EMT ;ERROR # DEFINED BY BADSCT SUBROUTINE
      030646 000137 031556 JMP 28$ ;GO TO 28$ IF ERROR
2418 030652
2419 030652 012737 072060 001174 4$: MOV #ZEROS,$TMP0 ;STARTING ADDRESS OF PATTERN
2420 030660 012737 000001 001176 MOV #1,$TMP1 ;RANGE OF PATTERN
2421 030666 004737 042174 JSR PC,GENBUF ;GO GENERATE BUFFER FOR FORMAT
2422
2423 ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATION
2424 030672 012702 001553 MOV #PUTINX,R2 ;R2 = BYTE ENTRY POSITION
2425 030676 112722 000034 MOVB #RMDC,(R2)+
2426 030702 112722 000006 MOVB #RMDA,(R2)+
2427 030706 112722 000004 MOVB #RMBAD,(R2)+
2428 030712 112722 000002 MOVB #RMWC,(R2)+
2429 030716 112722 000032 MOVB #RMOF,(R2)+
2430 030722 112722 000000 MOVB #RMCS1,(R2)+
2431 030726 112712 000200 MOVB #200,(R2) ;TERMINATE TABLE
2432 030732
2433 030732 004737 043330 5$: JSR PC,PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      030736 000404 BR 6$ ;GO TO 6$ IF NO ERROR
      030740 000240 NOP ;RETURN HERE IF ERROR
      030742 104000 EMT ;ERROR # DEFINED BY PUT SUBROUTINE
      030744 000137 031556 JMP 28$ ;GO TO 28$ IF ERROR
2434 030750
2435
2436 ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
      030750 004737 042774 JSR PC,GETSTS ;GO TO GETSTS SUBROUTINE
2437
2438 ;WAIT FOR COMMAND TO COMPLETE
      030754 004737 043672 JSR PC,TIMOUT ;GO TO TIMEOUT SUBROUTINE
2439
2440 ;GO READ STATUS FOR FORMAT OPERATION
2441 030760 004737 043060 JSR PC,GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
      030764 000404 BR 7$ ;GO TO 7$ IF NO ERROR
      030766 000240 NOP ;RETURN HERE IF ERROR
      030770 104000 EMT ;ERROR # DEFINED BY GET SUBROUTINE
      030772 000137 031556 JMP 28$ ;GO TO 28$ IF ERROR
2442 030776
2443
2444 ;VERIFY NO ERRORS DURING FORMAT
2445 030776 004737 056572 JSR PC,DASTS ;GO VERIFY RESULTS OF DATA TRANSFER
      031002 000405 BR 8$ ;GO TO 8$ IF NO ERROR
      031004 000240 NOP ;RETURN HERE IF ERROR
      031006 104000 EMT ;ERROR # DEFINED BY DASTS SUBROUTINE
      031010 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
      031012 000137 031556 JMP 28$ ;GO TO 28$ IF ERROR

```

```

2446 031016      8$:
2447
2448 ;MOVE LOOP ADDRESSES TO NEXT OPERATION
2449 031016 012737 031026 001124      MOV      #9$,SLPERR
2450 031024 000410      BR          10$      ;SKIP TO WRITE OPERATION
2451
2452 ;*****
2453 ;LOOP #2      WRITE,READ
2454
2455 031026      9$:
2456
2457 ;PREPARE DEVICE FOR WRITE OPERATION
2458
2459 031026 004737 037316      JSR      PC,TSTPRP      ;PREPARE DEVICE FOR TEST
      031032 054130      .WORD    054130      ;TASK DESCRIPTOR AS FOLLOWS:
                                           ;CLEAR CONTROLLER & SELECT DEVICE
                                           ;VERIFY CONTROLLER CLEAR OPERATION
                                           ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                           ;VERIFY PACK ACKNOWLEDGE
                                           ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
                                           ;VERIFY RECALIBRATION
                                           ;GO TO 10$ IF NO ERROR
                                           ;RETURN HERE IF ERROR
                                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                           ;GO TO 28$ IF ERROR
      031034 000404      BR          10$
      031036 000240      NOP
      031040 104000      EMT
      031042 000137 031556      JMP      28$
2460 031046      10$:
2461
2462 031046      11$:
2463
2464 ;WRITE DATA TO THE DRIVE
2465 031046 012737 177400 001414      MOV      #-256.,RMWCO      ;256. WORDS (2'S COMP)
2466 031054 012737 106442 001416      MOV      #BUFONE+4,RMBAD      ;CHANGE ADDRESS
2467 031062 012737 000061 001412      MOV      #WD!GO,RMCS10      ;WRITE DATA COMMAND
2468 031070 012702 001553      MOV      #PUTINX,R2      ;LOAD PUT REGISTER INDEX TABLE
2469 031074 112722 000006      MOVB      #RMDA,(R2)+
2470 031100 112722 000034      MOVB      #RMDC,(R2)+
2471 031104 112722 000032      MOVB      #RMOF,(R2)+
2472 031110 112722 000004      MOVB      #RMBB,(R2)+
2473 031114 112722 000002      MOVB      #RMWC,(R2)+
2474 031120 112722 000000      MOVB      #RMCS1,(R2)+
2475 031124 112722 000200      MOVB      #200,(R2)+      ;TERMINATE TABLE
2476
2477 031130 004737 043330      JSR      PC,PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      031134 000404      BR          12$      ;GO TO 12$ IF NO ERROR
      031136 000240      NOP      ;RETURN HERE IF ERROR
      031140 104000      EMT      ;ERROR # DEFINED BY PUT SUBROUTINE
      031142 000137 031556      JMP      28$      ;GO TO 28$ IF ERROR
2478 031146      12$:
2479
2480 ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
      031146 004737 042774      JSR      PC,GETSTS      ;GO TO GETSTS SUBROUTINE
2481
2482 ;WAIT FOR COMMAND TO COMPLETE
      031152 004737 043672      JSR      PC,TIMOUT      ;GO TO TIMOUT SUBROUTINE
2483
2484 ;GO READ STATUS FOR WRITE COMMAND
2485 031156 004737 043060      JSR      PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
  
```

```

031162 000404 BR 13$ ;GO TO 13$ IF NO ERROR
031164 000240 NOP ;RETURN HERE IF ERROR
031166 104000 EMT ;ERROR # DEFINED BY GET SUBROUTINE
031170 000137 031556 JMP 28$ ;GO TO 28$ IF ERROR
2486 031174 13$:
2487
2488 ;CHECK FOR ERRORS DURING WRITE OPERATION
2489 031174 004737 044056 JSR PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
031200 000405 BR 14$ ;GO TO 14$ IF NO ERROR
031202 000240 NOP ;RETURN HERE IF ERROR
031204 104000 EMT ;ERROR # DEFINED BY PRIERR SUBROUTINE
031206 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
031210 000137 031556 JMP 28$ ;GO TO 28$ IF ERROR
2490 031214 14$:
2491 031214 004737 056572 JSR PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
031220 000405 BR 15$ ;GO TO 15$ IF NO ERROR
031222 000240 NOP ;RETURN HERE IF ERROR
031224 104000 EMT ;ERROR # DEFINED BY DTASTS SUBROUTINE
031226 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
031230 000137 031556 JMP 28$ ;GO TO 28$ IF ERROR
2492 031234 15$:
2493
2494 031234 16$:
2495 031234 004737 044710 JSR PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
031240 000405 BR 17$ ;GO TO 17$ IF NO ERROR
031242 000240 NOP ;RETURN HERE IF ERROR
031244 104000 EMT ;ERROR # DEFINED BY SECERR SUBROUTINE
031246 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
031250 000137 031556 JMP 28$ ;GO TO 28$ IF ERROR
2496 031254 17$:
2497
2498 ;CHANGE LOOP ADDRESSES
2499 031254 012737 031264 001124 MOV #18$,$LPERR
2500 031262 000410 BR 19$ ;SKIP TO NEXT OPERATION
2501
2502 ;*****
2503 ;LOOP #3 READ
2504
2505 031264 18$:
2506
2507 ;PREPARE DEVICE FOR READ OPERATION
2508 031264 004737 037316 JSR PC,TSTPRP ;PREPARE DEVICE FOR TEST
031270 054130 .WORD 054130 ;TASK DESCRIPTOR AS FOLLOWS:
;CLEAR CONTROLLER & SELECT DEVICE
;VERIFY CONTROLLER CLEAR OPERATION
;PACK ACKNOWLEDGE IF VOLUME NOT VALID
;VERIFY PACK ACKNOWLEDGE
;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
;VERIFY RECALIBRATION
031272 000404 BR 19$ ;GO TO 19$ IF NO ERROR
031274 000240 NOP ;RETURN HERE IF ERROR
031276 104000 EMT ;ERROR # DEFINED BY TSTPRP SUBROUTINE
031300 000137 031556 JMP 28$ ;GO TO 28$ IF ERROR
2509 031304 19$:
2510
2511 031304 20$:
2512

```

```

2513      ;READ DATA FROM DEVICE
2514 031304 012737 107446 001416      MOV      #BUFTWO+4,RMBAD      ;CHANGE ADDRESS
2515 031312 012737 177400 001414      MOV      #-256.,RMWCO      ;256. WORDS (2'S COMP)
2516 031320 012737 000071 001412      MOV      #RD!GO,RMCS10      ;READ DATA COMMAND
2517 031326 012702 001553              MOV      #PUTINX,R2      ;LOAD PUT REGISTER INDEX TABLE
2518 031332 112722 000006              MOV      #RMDA,(R2)+
2519 031336 112722 000032              MOV      #RMOF,(R2)+
2520 031342 112722 000034              MOV      #RMDC,(R2)+
2521 031346 112722 000004              MOV      #RMDA,(R2)+
2522 031352 112722 000002              MOV      #RMWC,(R2)+
2523 031356 112722 000000              MOV      #RMCS1,(R2)+
2524 031362 112712 000200              MOV      #200,(R2)
2525
2526 031366 004737 043330              JSR      PC,PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      031372 000404              BR      21$      ;GO TO 21$ IF NO ERROR
      031374 000240              NOP      ;RETURN HERE IF ERROR
      031376 104000              EMT      ;ERROR # DEFINED BY PUT SUBROUTINE
      031400 000137 031556              JMP      28$      ;GO TO 28$ IF ERROR
2527 031404
2528
2529      ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
      031404 004737 042774              JSR      PC,GETSTS      ;GO TO GETSTS SUBROUTINE
2530
2531      ;WAIT FOR COMMAND TO COMPLETE
      031410 004737 043672              JSR      PC,TIMOUT      ;GO TO TIMOUT SUBROUTINE
2532
2533      ;GO READ STATUS FOR READ OPERATION
2534 031414 004737 043060              JSR      PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      031420 000404              BR      22$      ;GO TO 22$ IF NO ERROR
      031422 000240              NOP      ;RETURN HERE IF ERROR
      031424 104000              EMT      ;ERROR # DEFINED BY GET SUBROUTINE
      031426 000137 031556              JMP      28$      ;GO TO 28$ IF ERROR
2535 031432
2536
2537      ;CHECK FOR ERRORS DURING READ OPERATION
2538 031432 004737 044056              JSR      PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
      031436 000405              BR      23$      ;GO TO 23$ IF NO ERROR
      031440 000240              NOP      ;RETURN HERE IF ERROR
      031442 104000              EMT      ;ERROR # DEFINED BY PRIERR SUBROUTINE
      031444 004736              JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      031446 000137 031556              JMP      28$      ;GO TO 28$ IF ERROR
2539 031452
2540 031452 004737 056572              JSR      PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
      031456 000405              BR      24$      ;GO TO 24$ IF NO ERROR
      031460 000240              NOP      ;RETURN HERE IF ERROR
      031462 104000              EMT      ;ERROR # DEFINED BY DTASTS SUBROUTINE
      031464 004736              JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      031466 000137 031556              JMP      28$      ;GO TO 28$ IF ERROR
2541 031472
2542
2543      ;GO CHECK FOR SECONDARY ERRORS
2544 031472 004737 044710              JSR      PC,SECERR      ;GO TO 26$ IF NO ERROR
      031476 000405              BR      26$      ;RETURN HERE IF ERROR
      031500 000240              NOP      ;ERROR # DEFINED BY SECERR SUBROUTINE
      031502 104000              EMT      ;GO BACK FOR MORE ERROR CHECKS
      031504 004736              JSR      PC,@(SP)+      ;GO TO 28$ IF ERROR
      031506 000137 031556              JMP      28$
  
```

```

2545 031512
2546 031512 004737 042432
      031516 106442
      031520 107446
      031522 000404
      031524 000240
      031526 104000
      031530 000137 031556
2547 031534
2548 031534 062737 000200 001446
2549 031542 023727 001446 001400
2550 031550 101002
2551 031552 000137 030574
2552 031556
2553
2554

      031556
      031556 000004
      031560 000240
      031562 012706 001100
      031566 013700 001276
      031572 013701 001466
      031576 012737 000021 001226
2555 031604
2556
2557
2558
2559
2560 031604 004737 037316
      031610 054130

      031612 000404
      031614 000240
      031616 104000
      031620 000137 032562
2561 031624
2562
2563
2564 031624 012737 000577 001446
2565 031632 013737 001334 001420
2566 031640 112737 000037 001420
2567 031646 012737 106436 001416
2568 031654 012737 176774 001414
2569 031662 012737 010000 001444
2570 031670 012737 000063 001412
2571
2572

      031676 004737 040242
      031702 000405
      031704 104401 070426

26$:  JSR      PC,CMPBUF      ;GO COMPARE WRITE, READ DATA BUFFERS
      .WORD    BUFOne+4      ;STARTING ADDRESS OF WRITE BUFFER
      .WORD    BUFTwo+4      ;STARTING ADDRESS OF READ BUFFER
      BR       27$           ;GO TO 27$ IF NO ERROR
      NOP      ;RETURN HERE IF ERROR
      EMT      ;ERROR # DEFINED BY CMPBUF SUBROUTINE
      JMP      28$           ;GO TO 28$ IF ERROR

27$:  ADD      #128.,RMDCO     ;ADVANCE TO NEXT THRESHOLD
      CMP      RMDCO,#768.    ;DONE??
      BHI      28$           ;YES!!
      JMP      3$            ;DO NEXT CURRENT LEVEL

28$:

;*****
;*TEST 21      WRITE, WRITE CHECK W/ CURRENT LEVEL SWITCHING
;*****
TST21:
      SCOPE      ;SCOPE CALL
      NOP      ;START OF TEST
      MOV      #STACK,SP     ;INITIALIZE STACK POINTER
      MOV      $BASE,R0      ;R0 = UNIBUS ADDRESS
      MOV      TSTQUE,R1     ;(R1) = DEVICE BEING TESTED
      MOV      #21,$TESTN    ;SET TEST NUMBER IN APT MAIL BOX

1$:

;*****
;LOOP #1      FORMAT,WRITE,READ
;*****

      JSR      PC,TSTPRP     ;PREPARE DEVICE FOR TEST
      .WORD    054130        ;TASK DESCRIPTOR AS FOLLOWS:
                              ;CLEAR CONTROLLER & SELECT DEVICE
                              ;VERIFY CONTROLLER CLEAR OPERATION
                              ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                              ;VERIFY PACK ACKNOWLEDGE
                              ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
                              ;VERIFY RECALIBRATION
                              ;GO TO 2$ IF NO ERROR
                              ;RETURN HERE IF ERROR
                              ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                              ;GO TO 26$ IF ERROR

      BR       2$

2$:

;LOAD PARAMETERS AND GENERATE DATA BUFFER
      MOV      #383.,RMDCO   ;CYLINDER - 383.
      MOV      LSTRK,RMDAO   ;SET LAST TRACK AND
      MOV      #31.,RMDAO   ;LAST SECTOR
      MOVB     #BUFOne,RMBAO ;BUS ADDRESS
      MOV      #-258.*2,RMWCO ;2 SECTORS (2'S COMP)
      MOV      #FMT16,RMOFC  ;16 BIT FORMAT
      MOV      #WH!GO,RMC$10 ;WRITE HEADER AND DATA COMMAND

;VERIFY THAT SECTOR IS NOT BAD
      JSR      PC,BAD$CT     ;CALL BAD SECTOR MODULE
      BR       3$           ;GO TO 3$ IF NO ERROR
      TYPE     ,SCTMSG       ;TYPE BAD SECTOR MESSAGE
  
```



```

031710 104000 EMT ;ERROR # DEFINED BY BADSC1 SUBROUTINE
031712 000137 032562 JMP 26$ ;GO TO 26$ IF ERROR
2573 031716 3$: MOV #ONES,$TMP0 ;STARTING ADDRESS OF PATTERN
2574 031716 012737 072016 001174 MOV #1,$TMP1 ;RANGE OF PATTERN
2575 031724 012737 000001 001176 JSR PC,GENBUF ;GO GENERATE BUFFER FOR FORMAT
2576 031732 004737 042174
2577
2578 ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATION
2579 031736 012702 001553 MOV #PUTINX,R2 ;R2 = BYTE ENTRY POSITION
2580 031742 112722 000034 MOVB #RMDC,(R2)+
2581 031746 112722 000006 MOVB #RMDA,(R2)+
2582 031752 112722 000004 MOVB #RMBB,(R2)+
2583 031756 112722 000002 MOVB #RMWC,(R2)+
2584 031762 112722 000032 MOVB #RMOF,(R2)+
2585 031766 112722 000000 MOVB #RMCS1,(R2)+
2586 031772 112712 000200 MOVB #200,(R2) ;TERMINATE TABLE
2587 031776
2588 031776 004737 043330 4$: JSR PC,PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
032002 000404 BR 5$ ;GO TO 5$ IF NO ERROR
032004 000240 NOP ;RETURN HERE IF ERROR
032006 104000 EMT ;ERROR # DEFINED BY PUT SUBROUTINE
032010 000137 032562 JMP 26$ ;GO TO 26$ IF ERROR
2589 032014
2590
2591 ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
032014 004737 042774 JSR PC,GETSTS ;GO TO GETSTS SUBROUTINE
2592
2593 ;WAIT FOR COMMAND TO COMPLETE
032020 004737 043672 JSR PC,TIMOUT ;GO TO TIMOUT SUBROUTINE
2594
2595 ;GO READ STATUS FOR FORMAT OPERATION
2596 032024 004737 043060 JSR PC,GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
032030 000404 BR 6$ ;GO TO 6$ IF NO ERROR
032032 000240 NOP ;RETURN HERE IF ERROR
032034 104000 EMT ;ERROR # DEFINED BY GET SUBROUTINE
032036 000137 032562 JMP 26$ ;GO TO 26$ IF ERROR
2597 032042
2598
2599 ;VERIFY NO ERRORS DURING FORMAT
2600 032042 004737 056572 JSR PC,DASTS ;GO VERIFY RESULTS OF DATA TRANSFER
032046 000405 BR 7$ ;GO TO 7$ IF NO ERROR
032050 000240 NOP ;RETURN HERE IF ERROR
032052 104000 EMT ;ERROR # DEFINED BY DASTS SUBROUTINE
032054 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
032056 000137 032562 JMP 26$ ;GO TO 26$ IF ERROR
2601 032062
2602
2603 ;MOVE LOOP ADDRESSES TO NEXT OPERATION
2604 032062 012737 032072 001124 MOV #8$,$LPERR
2605 032070 000410 BR 9$ ;SKIP TO WRITE OPERATION
2606
2607 ;*****
2608 ;LOOP #2 WRITE,READ
2609
2610 032072
2611
2612
  
```

```

2613                                     ;PREPARE DEVICE FOR WRITE OPERATION
2614 032072 004737 037316                JSR    PC,TSTPRP    ;PREPARE DEVICE FOR TEST
                                .WORD    054130              ;TASK DESCRIPTOR AS FOLLOWS:
                                                ;CLEAR CONTROLLER & SELECT DEVICE
                                                ;VERIFY CONTROLLER CLEAR OPERATION
                                                ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                ;VERIFY PACK ACKNOWLEDGE
                                                ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
                                                ;VERIFY RECALIBRATION
                                                ;GO TO 9$ IF NO ERROR
                                                ;RETURN HERE IF ERROR
                                                ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                                ;GO TO 26$ IF ERROR

                                032100 000404                BR      9$
                                032102 000240                NOP
                                032104 104000                EMT
                                032106 000137 032562         JMP      26$

2615 032112                                     9$:
2616
2617 032112                                     10$:
2618
2619                                     ;WRITE DATA TO THE DRIVE
2620 032112 012737 177000 001414          MOV     #-256.*2,RMWCO    ;2 SECTORS (2'S COMP)
2621 032120 012737 106436 001416          MOV     #BUFONE,RMBA0    ;CHANGE ADDRESS
2622 032126 012737 000061 001412          MOV     #WD!GO,RMCS10    ;WRITE DATA COMMAND
2623 032134 012737 072016 001174          MOV     #ONES,$TMP0    ;STARTING ADDRESS OF PATTERN
2624 032142 012737 000001 001176          MOV     #1,$TMP1      ;RANGE OF PATTERN
2625 032150 004737 042174                  JSR     PC,GENBUF      ;GO GENERATE BUFFER FOR FORMAT
2626 032154 012702 001553                  MOV     #PUTINX,R2      ;LOAD PUT REGISTER INDEX TABLE
2627 032160 112722 000006                  MOVB    #RMDA,(R2)+
2628 032164 112722 000034                  MOVB    #RMDC,(R2)+
2629 032170 112722 000032                  MOVB    #RMOF,(R2)+
2630 032174 112722 000004                  MOVB    #RMBB,(R2)+
2631 032200 112722 000002                  MOVB    #RMWC,(R2)+
2632 032204 112722 000000                  MOVB    #RMCS1,(R2)+
2633 032210 112722 000200                  MOVB    #200,(R2)+    ;TERMINATE TABLE
2634
2635 032214 004737 043330                  JSR     PC,PUT        ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
                                032220 000404                BR      11$    ;GO TO 11$ IF NO ERROR
                                032222 000240                NOP        ;RETURN HERE IF ERROR
                                032224 104000                EMT        ;ERROR # DEFINED BY PUT SUBROUTINE
                                032226 000137 032562         JMP      26$    ;GO TO 26$ IF ERROR
2636 032232                                     11$:
2637
2638                                     ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
                                032232 004737 042774          JSR     PC,GETSTS    ;GO TO GETSTS SUBROUTINE
2639
2640                                     ;WAIT FOR COMMAND TO COMPLETE
                                032236 004737 043672          JSR     PC,TIMOUT    ;GO TO TIMOUT SUBROUTINE
2641
2642                                     ;GO READ STATUS FOR WRITE COMMAND
2643 032242 004737 043060                  JSR     PC,GET        ;GO READ REGISTER(S) WITH GET SUBROUTINE
                                032246 000404                BR      12$    ;GO TO 12$ IF NO ERROR
                                032250 000240                NOP        ;RETURN HERE IF ERROR
                                032252 104000                EMT        ;ERROR # DEFINED BY GET SUBROUTINE
                                032254 000137 032562         JMP      26$    ;GO TO 26$ IF ERROR
2644 032260                                     12$:
2645
2646                                     ;CHECK FOR ERRORS DURING WRITE OPERATION
2647 032260 004737 044056                  JSR     PC,PRIERR    ;GO CHECK FOR PRIMARY ERRORS
                                032264 000405                BR      13$    ;GO TO 13$ IF NO ERROR

```



```

032266 000240      NOP      ;RETURN HERE IF ERROR
032270 104000      EMT      ;ERROR # DEFINED BY PRIERR SUBROUTINE
032272 004736      JSR      PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
032274 000137 032562 JMP      26$      ;GO TO 26$ IF ERROR
2648 032300      13$:      JSR      PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
2649 032300 004737 056572 BR      14$      ;GO TO 14$ IF NO ERROR
032304 000405      NOP      ;RETURN HERE IF ERROR
032306 000240      EMT      ;ERROR # DEFINED BY DTASTS SUBROUTINE
032310 104000      JSR      PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
032312 004736      JMP      26$      ;GO TO 26$ IF ERROR
032314 000137 032562
2650 032320      14$:
2651
2652 032320      15$:      JSR      PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
2653 032320 004737 044710 BR      16$      ;GO TO 16$ IF NO ERROR
032324 000405      NOP      ;RETURN HERE IF ERROR
032326 000240      EMT      ;ERROR # DEFINED BY SECERR SUBROUTINE
032330 104000      JSR      PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
032332 004736      JMP      26$      ;GO TO 26$ IF ERROR
032334 000137 032562
2654 032340      16$:
2655
2656      ;CHANGE LOOP ADDRESSES
2657 032340 012737 032350 001124 MOV      #17$,$LPERR
2658 032346 000410      BR      18$      ;SKIP TO NEXT OPERATION
2659
2660      ;*****
2661      ;LOOP #3      READ
2662
2663 032350      17$:
2664
2665      ;PREPARE DEVICE FOR READ OPERATION
2666 032350 004737 037316 JSR      PC,TSTPRP ;PREPARE DEVICE FOR TEST
032354 054130      .WORD 054130 ;TASK DESCRIPTOR AS FOLLOWS:
                                ;CLEAR CONTROLLER & SELECT DEVICE
                                ;VERIFY CONTROLLER CLEAR OPERATION
                                ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                ;VERIFY PACK ACKNOWLEDGE
                                ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
                                ;VERIFY RECALIBRATION
                                ;GO TO 18$ IF NO ERROR
                                ;RETURN HERE IF ERROR
                                ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                ;GO TO 26$ IF ERROR
032356 000404      BR      18$
032360 000240      NOP
032362 104000      EMT
032364 000137 032562 JMP      26$
2667 032370      18$:
2668
2669 032370      19$:
2670
2671      ;READ DATA FROM DEVICE
2672 032370 012737 000051 001412 MOV      #WCD!GO,RMCS10 ;READ DATA COMMAND
2673 032376 012702 001553 MOV      #PUTINX,R2 ;LOAD PUT REGISTER INDEX TABLE
2674 032402 112722 000006 MOVB     #RMDA,(R2)+
2675 032406 112722 000032 MOVB     #RMOF,(R2)+
2676 032412 112722 000034 MOVB     #RMDC,(R2)+
2677 032416 112722 000004 MOVB     #RMB A,(R2)+
2678 032422 112722 000002 MOVB     #RMWC,(R2)+
2679 032426 112722 000000 MOVB     #RMCS1,(R2)+

```

```

2680 032432 112712 000200      MOVB    #200,(R2)
2681
2682 032436 004737 043330      JSR     PC,PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      032442 000404      BR      20$      ;GO TO 20$ IF NO ERROR
      032444 000240      NOP      ;RETURN HERE IF ERROR
      032446 104000      EMT      ;ERROR # DEFINED BY PUT SUBROUTINE
      032450 000137 032562      JMP     26$      ;GO TO 26$ IF ERROR
2683 032454
2684
2685      ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
      032454 004737 042774      JSR     PC,GETSTS    ;GO TO GETSTS SUBROUTINE
2686
2687      ;WAIT FOR COMMAND TO COMPLETE
      032460 004737 043672      JSR     PC,TIMOUT    ;GO TO TIMEOUT SUBROUTINE
2688
2689      ;GO READ STATUS FOR READ OPERATION
2690 032464 004737 043060      JSR     PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      032470 000404      BR      21$      ;GO TO 21$ IF NO ERROR
      032472 000240      NOP      ;RETURN HERE IF ERROR
      032474 104000      EMT      ;ERROR # DEFINED BY GET SUBROUTINE
      032476 000137 032562      JMP     26$      ;GO TO 26$ IF ERROR
2691 032502
2692
2693      ;CHECK FOR ERRORS DURING READ OPERATION
2694 032502 004737 044056      JSR     PC,PRIERR    ;GO CHECK FOR PRIMARY ERRORS
      032506 000405      BR      22$      ;GO TO 22$ IF NO ERROR
      032510 000240      NOP      ;RETURN HERE IF ERROR
      032512 104000      EMT      ;ERROR # DEFINED BY PRIERR SUBROUTINE
      032514 004736      JSR     PC,@(SP)+    ;GO BACK FOR MORE ERROR CHECKS
      032516 000137 032562      JMP     26$      ;GO TO 26$ IF ERROR
2695 032522
2696 032522 004737 056572      JSR     PC,DTASTS    ;GO VERIFY RESULTS OF DATA TRANSFER
      032526 000405      BR      23$      ;GO TO 23$ IF NO ERROR
      032530 000240      NOP      ;RETURN HERE IF ERROR
      032532 104000      EMT      ;ERROR # DEFINED BY DTASTS SUBROUTINE
      032534 004736      JSR     PC,@(SP)+    ;GO BACK FOR MORE ERROR CHECKS
      032536 000137 032562      JMP     26$      ;GO TO 26$ IF ERROR
2697 032542
2698
2699      ;
2700 032542 004737 044710      JSR     PC,SECERR    ;GO CHECK FOR SECONDARY ERRORS
      032546 000405      BR      25$      ;GO TO 25$ IF NO ERROR
      032550 000240      NOP      ;RETURN HERE IF ERROR
      032552 104000      EMT      ;ERROR # DEFINED BY SECERR SUBROUTINE
      032554 004736      JSR     PC,@(SP)+    ;GO BACK FOR MORE ERROR CHECKS
      032556 000137 032562      JMP     26$      ;GO TO 26$ IF ERROR
2701 032562
2702
2703 032562
2704
2705
      ;*****
      ;*TEST 22      WRITE, READ EARLY PEAK SHIFT
      ;*****
TST22:
      032562 000004      SCOPE      ;SCOPE CALL
      032564 000240      NOP      ;START OF TEST
      032566 012706 001100      MOV     #STACK,SP    ;INITIALIZE STACK POINTER
  
```

```

032572 013700 001276      MOV    $BASE,R0      ;R0 = UNIBUS ADDRESS
032574 013701 001466      MOV    TSTQUE,R1     ;(R1) = DEVICE BEING TESTED
032602 012737 000022 001226  MOV    #22,$TESTN   ;:SET TEST NUMBER IN APT MAIL BOX

2706
2707
2708
2709
2710 032610
2711
2712
2713 032610 004737 037316
    032614 154130

;*****
;LOOP #1      FORMAT,WRITE,READ

1$:
;PREPARE THE DEVICE FOR FORMAT OPERATION
    JSR    PC,TSTPRP      ;PREPARE DEVICE FOR TEST
    .WORD  154130         ;TASK DESCRIPTOR AS FOLLOWS:
                        ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
                        ;CLEAR CONTROLLER & SELECT DEVICE
                        ;VERIFY CONTROLLER CLEAR OPERATION
                        ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                        ;VERIFY PACK ACKNOWLEDGE
                        ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
                        ;VERIFY RECALIBRATION
                        ;GO TO 2$ IF NO ERROR
                        ;RETURN HERE IF ERROR
                        ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                        ;GO TO 25$ IF ERROR

    BR     2$
    NOP
    EMT
    JMP    25$

2714 032630
2715
2716
2717 032630 012737 000000 001446
2718 032636 012737 000000 001420
2719 032644 012737 106436 001416
2720 032652 012737 177376 001414
2721 032660 012737 010000 001444
2722 032666 012737 000063 001412
2723
2724
    BR     2$
    NOP
    EMT
    JMP    25$

;LOAD PARAMETERS AND GENERATE DATA BUFFER
    MOV    #0,RMDCO      ;CYLINDER = 0
    MOV    #0,RMDAO      ;TRACK = 0, SECTOR = 0
    MOV    #BUFONE,RMBAO  ;BUS ADDRESS
    MOV    #-258,RMWCO    ;2 + 256. WORDS (2'S COMP)
    MOV    #FMT16,RMOFO   ;16 BIT FORMAT
    MOV    #WH!GO,RMCS10  ;WRITE HEADER AND DATA COMMAND

;VERIFY THAT SECTOR IS NOT BAD
    JSR    PC,BADSCT      ;CALL BAD SECTOR MODULE
    BR     3$             ;GO TO 3$ IF NO ERROR
    TYPE    ,SCTMSG       ;TYPE BAD SECTOR MESSAGE
    EMT
    JMP    25$            ;ERROR # DEFINED BY BADSCT SUBROUTINE
                        ;GO TO 25$ IF ERROR

3$:
    MOV    #EARLY,$TMP0   ;STARTING ADDRESS OF PATTERN
    MOV    #1,$TMP1       ;RANGE OF PATTERN
    JSR    PC,GENBUF      ;GO GENERATE BUFFER FOR FORMAT

;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATION
    MOV    #PUTINX,R2     ;R2 = BYTE ENTRY POSITION
    MOVB   #RMDC,(R2)+
    MOVB   #RMDA,(R2)+
    MOVB   #RMBA,(R2)+
    MOVB   #RMWC,(R2)+
    MOVB   #RMOF,(R2)+
    MOVB   #RMCS1,(R2)+
    MOVB   #200,(R2)

;TERMINATE TABLE

4$:
    JSR    PC,PUT         ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
    BR     5$             ;GO TO 5$ IF NO ERROR
    NOP                  ;RETURN HERE IF ERROR
  
```

```

2741 033004 104000          EMT          ;ERROR # DEFINED BY PUT SUBROUTINE
2742 033006 000137 033570   JMP          ;GO TO 25$ IF ERROR
2743 033012          5$:
                ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
                JSR      PC,GETSTS      ;GO TO GETSTS SUBROUTINE
2744 033012 004737 042774
2745          ;WAIT FOR COMMAND TO COMPLETE
                JSR      PC,TIMOUT      ;GO TO TIMEOUT SUBROUTINE
2746 033016 004737 043672
2747          ;GO READ STATUS FOR FORMAT OPERATION
2748 033022 004737 043060   JSR      PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
                BR        6$           ;GO TO 6$ IF NO ERROR
                NOP        ;RETURN HERE IF ERROR
                EMT        ;ERROR # DEFINED BY GET SUBROUTINE
                JMP        25$         ;GO TO 25$ IF ERROR
2749 033026 000404
2750 033030 000240
2751 033032 104000
2752 033034 000137 033570   6$:
                ;VERIFY NO ERRORS DURING FORMAT
                JSR      PC,DASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
                BR        7$           ;GO TO 7$ IF NO ERROR
                NOP        ;RETURN HERE IF ERROR
                EMT        ;ERROR # DEFINED BY DASTS SUBROUTINE
                JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
                JMP        25$         ;GO TO 25$ IF ERROR
2753 033040 004737 056572   7$:
2754 033044 000405
2755 033046 000240
2756 033050 104000
2757 033052 004736
2758 033054 000137 033570   ;MOVE LOOP ADDRESSES TO NEXT OPERATION
                MOV      #8$,$LPERR
                BR        9$
2759 033060
2760          ;*****
2761          ;LOOP #2      WRITE,READ
2762 033070          8$:
2763          ;PREPARE DEVICE FOR WRITE OPERATION
2764 033070 004737 037316   JSR      PC,TSTPRP      ;PREPARE DEVICE FOR TEST
2765 033074 154130          .WORD      154130      ;TASK DESCRIPTOR AS FOLLOWS:
                ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
                ;CLEAR CONTROLLER & SELECT DEVICE
                ;VERIFY CONTROLLER CLEAR OPERATION
                ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                ;VERIFY PACK ACKNOWLEDGE
                ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
                ;VERIFY RECALIBRATION
                BR        9$           ;GO TO 9$ IF NO ERROR
                NOP        ;RETURN HERE IF ERROR
                EMT        ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                JMP        25$         ;GO TO 25$ IF ERROR
2766 033076 000404
2767 033100 000240
2768 033102 104000
2769 033104 000137 033570   9$:
                BR        9$
                NOP
                EMT
                JMP        25$
2770 033110          10$:
                ;WRITE DATA TO THE DRIVE
2771 033110 012737 177400 001414   MOV      #-256.,RMWCO      ;CHANGE WORD COUNT
2772 033116 012737 106442 001416   MOV      #BUFONE+4,RMBAO    ;CHANGE ADDRESS
2773 033124 012737 000061 001412   MOV      #WD.GO,RMC$10    ;WRITE DATA COMMAND
  
```

2773	033132	012702	001553	MOV	#PUTINX,R2	;LOAD PUT REGISTER INDEX TABLE
2774	033136	112722	000006	MOVB	#RMDA,(R2)+	
2775	033142	112722	000034	MOVB	#RMDC,(R2)+	
2776	033146	112722	000032	MOVB	#RMOF,(R2)+	
2777	033152	112722	000004	MOVB	#RMBA,(R2)+	
2778	033156	112722	000002	MOVB	#RMWC,(R2)+	
2779	033162	112722	000000	MOVB	#RMCS1,(R2)+	
2780	033166	112722	000200	MOVB	#200,(R2)+	;TERMINATE TABLE
2781						
2782	033172	004737	043330	JSR	PC,PUT	;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
	033176	000404		BR	11\$	;GO TO 11\$ IF NO ERROR
	033200	000240		NOP		;RETURN HERE IF ERROR
	033202	104000		EMT		;ERROR # DEFINED BY PUT SUBROUTINE
	033204	000137	033570	JMP	25\$	;GO TO 25\$ IF ERROR
2783	033210					
2784						
2785						
	033210	004737	042774	JSR	PC,GETSTS	;GO TO GETSTS SUBROUTINE
2786						
2787						
	033214	004737	043672	JSR	PC,TIMOUT	;GO TO TIMEOUT SUBROUTINE
2788						
2789						
	033220	004737	043060	JSR	PC,GET	;GO READ REGISTER(S) WITH GET SUBROUTINE
2790	033224	000404		BR	12\$	;GO TO 12\$ IF NO ERROR
	033226	000240		NOP		;RETURN HERE IF ERROR
	033230	104000		EMT		;ERROR # DEFINED BY GET SUBROUTINE
	033232	000137	033570	JMP	25\$	;GO TO 25\$ IF ERROR
2791	033236					
2792						
2793						
	033236	004737	044056	JSR	PC,PRIERR	;GO CHECK FOR PRIMARY ERRORS
2794	033242	000405		BR	13\$	;GO TO 13\$ IF NO ERROR
	033244	000240		NOP		;RETURN HERE IF ERROR
	033246	104000		EMT		;ERROR # DEFINED BY PRIERR SUBROUTINE
	033250	004736		JSR	PC,@(SP)+	;GO BACK FOR MORE ERROR CHECKS
	033252	000137	033570	JMP	25\$	;GO TO 25\$ IF ERROR
2795	033256					
2796	033256	004737	056572	JSR	PC,DTASTS	;GO VERIFY RESULTS OF DATA TRANSFER
	033262	000405		BR	14\$	;GO TO 14\$ IF NO ERROR
	033264	000240		NOP		;RETURN HERE IF ERROR
	033266	104000		EMT		;ERROR # DEFINED BY DTASTS SUBROUTINE
	033270	004736		JSR	PC,@(SP)+	;GO BACK FOR MORE ERROR CHECKS
	033272	000137	033570	JMP	25\$	;GO TO 25\$ IF ERROR
2797	033276					
2798	033276	004737	044710	JSR	PC,SECERR	;GO CHECK FOR SECONDARY ERRORS
	033302	000405		BR	15\$	;GO TO 15\$ IF NO ERROR
	033304	000240		NOP		;RETURN HERE IF ERROR
	033306	104000		EMT		;ERROR # DEFINED BY SECERR SUBROUTINE
	033310	004736		JSR	PC,@(SP)+	;GO BACK FOR MORE ERROR CHECKS
	033312	000137	033570	JMP	25\$	;GO TO 25\$ IF ERROR
2799	033316					
2800						
2801						
2802	033316	012737	033326 001124	MOV	#16\$,\$LPERR	
2803	033324	000410		BR	17\$	
2804						

```

2805
2806
2807
2808 033326
2809
2810
2811 033326 004737 037316
      033332 154130
      JSR PC,TSTPRP      ;PREPARE DEVICE FOR TEST
      .WORD 154130      ;TASK DESCRIPTOR AS FOLLOWS:
                          ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
                          ;CLEAR CONTROLLER & SELECT DEVICE
                          ;VERIFY CONTROLLER CLEAR OPERATION
                          ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                          ;VERIFY PACK ACKNOWLEDGE
                          ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
                          ;VERIFY RECALIBRATION
                          ;GO TO 17$ IF NO ERROR
                          ;RETURN HERE IF ERROR
                          ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                          ;GO TO 25$ IF ERROR

      BR 17$
      NOP
      EMT
      JMP 25$

2812 033334 000404
2813 033336 000240
2814 033340 104000
2815 033342 000137 033570
2816 033346
2817 033346 012737 000071 001412
2818 033354 012737 107446 001416
2819 033362 012702 001553
2820 033366 112722 000006
2821 033372 112722 000032
2822 033376 112722 000034
2823 033402 112722 000004
2824 033406 112722 000002
2825 033412 112722 000000
2826 033416 112712 000200
2827 033422 004737 043330
      JSR PC,PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      BR 19$          ;GO TO 19$ IF NO ERROR
      NOP             ;RETURN HERE IF ERROR
      EMT             ;ERROR # DEFINED BY PUT SUBROUTINE
      JMP 25$         ;GO TO 25$ IF ERROR

2828 033426 000404
2829 033430 000240
2830 033432 104000
2831 033434 000137 033570
2832 033440
2833 033440 004737 042774
2834 033444 004737 043672
2835 033450 004737 043060
      JSR PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      BR 20$          ;GO TO 20$ IF NO ERROR
      NOP             ;RETURN HERE IF ERROR
      EMT             ;ERROR # DEFINED BY GET SUBROUTINE
      JMP 25$         ;GO TO 25$ IF ERROR

2836 033454 000404
2837 033456 000240
2838 033460 104000
2839 033462 000137 033570
2840 033466
2841 033466 004737 044056
      JSR PC,PRIERR    ;GO CHECK FOR PRIMARY ERRORS
  
```



```

033472 000405 BR 21$ :GO TO 21$ IF NO ERROR
033474 000240 NOP :RETURN HERE IF ERROR
033476 104000 EMT :ERROR # DEFINED BY PRIERP SUBROUTINE
033500 004736 JSR PC,@(SP)+ :GO BACK FOR MORE ERROR CHECKS
033502 000137 033570 JMP 25$ :GO TO 25$ IF ERROR
2840 033506 21$: JSR PC,DTASTS :GO VERIFY RESULTS OF DATA TRANSFER
2841 033506 004737 056572 BR 22$ :GO TO 22$ IF NO ERROR
033512 000405 NOP :RETURN HERE IF ERROR
033514 000240 EMT :ERROR # DEFINED BY DTASTS SUBROUTINE
033516 104000 JSR PC,@(SP)+ :GO BACK FOR MORE ERROR CHECKS
033520 004736 JMP 25$ :GO TO 25$ IF ERROR
2842 033526 22$: JSR PC,SECERR :GO CHECK FOR SECONDARY ERROR
2843 033526 004737 044710 BR 23$ :GO TO 23$ IF NO ERROR
033532 000405 NOP :RETURN HERE IF ERROR
033534 000240 EMT :ERROR # DEFINED BY SECERR SUBROUTINE
033540 004736 JSR PC,@(SP)+ :GO BACK FOR MORE ERROR CHECKS
033542 000137 033570 JMP 25$ :GO TO 25$ IF ERROR
2844 033546 23$: JSR PC,CMPBUF :GO COMPARE WRITE, READ DATA BUFFERS
2845 033546 004737 042432 .WORD BUFOFF+4 :STARTING ADDRESS OF WRITE BUFFER
033552 106442 .WORD BUFTWO+4 :STARTING ADDRESS OF READ BUFFER
033554 107446 BR 24$ :GO TO 24$ IF NO ERROR
033556 000404 NOP :RETURN HERE IF ERROR
033560 000240 EMT :ERROR # DEFINED BY CMPBUF SUBROUTINE
033562 104000 JMP 25$ :GO TO 25$ IF ERROR
2846 033570 24$:
2847 25$:
2848 033570
2849
2850
;*****
;*TEST 23 WRITE, READ MIXED PEAK SHIFT
;*****
TST23:
033570 SCOPE :SCOPE CALL
033570 000004 NOP :START OF TEST
033572 000240 MOV #STACK,SP :INITIALIZE STACK POINTER
033574 012706 001100 MOV $BASE,R0 :R0 - UNIBUS ADDRESS
033600 013700 001276 MOV TSTQUE,R1 : (R1) - DEVICE BEING TESTED
033604 013701 001466 MOV #23,$TESTN :SET TEST NUMBER IN APT MAIL BOX
033610 012737 000023 001226
2851
2852
2853 ;*****
2854 ;LOOP #1 FORMAT,WRITE,READ
2855 033616 1$:
2856
2857 ;PREPARE THE DEVICE FOR FORMAT OPERATION
2858 033616 004737 037316 JSR PC,TSTPRP :PREPARE DEVICE FOR TEST
033622 154130 .WORD 154130 :TASK DESCRIPTOR AS FOLLOWS:
:SELECT DEVICE & VERIFY DEVICE AVAILABLE
:CLEAR CONTROLLER & SELECT DEVICE
:VERIFY CONTROLLER CLEAR OPERATION
:PACK ACKNOWLEDGE IF VOLUME NOT VALID
:VERIFY PACK ACKNOWLEDGE
:RECALIBRATE IF 'SKI' OR 'PIP' IS SET
:VERIFY RECALIBRATION

```

```

033624 000404 BR 2$ ;GO TO 2$ IF NO ERROR
033626 000240 NOP ;RETURN HERE IF ERROR
033630 104000 EMT ;ERROR # DEFINED BY TSTPRP SUBROUTINE
033632 000137 034576 JMP 25$ ;GO TO 25$ IF ERROR
2859 033636 2$:
2860
2861 ;LOAD PARAMETERS AND GENERATE DATA BUFFER
2862 033636 012737 000000 001446 MOV #0,RMDCO ;CYLINDER = 0
2863 033644 012737 000000 001420 MOV #0,RMDAO ;TRACK = 0, SECTOR = 0
2864 033652 012737 106436 001416 MOV #BJFONE,RMBAO ;BUS ADDRESS
2865 033660 012737 177376 001414 MOV #-256,RMWCO ;2 + 256. WORDS (2'S COMP)
2866 033666 012737 010000 001444 MOV #FMT16,RMOFO ;16 BIT FORMAT
2867 033674 012737 000063 001412 MOV #WH!GO,RMCS10 ;WRITE HEADER AND DATA COMMAND
2868
2869 ;VERIFY THAT SECTOR IS NOT BAD
033702 004737 040242 JSR PC,BADSCT ;CALL BAD SECTOR MODULE
033706 000405 BR 3$ ;GO TO 3$ IF NO ERROR
033710 104401 070426 TYPE ,SCTMSG ;TYPE BAD SECTOR MESSAGE
033714 104000 EMT ;ERROR # DEFINED BY BADSCT SUBROUTINE
033716 000137 034576 JMP 25$ ;GO TO 25$ IF ERROR
2870 033722 3$:
2871 033722 012737 071756 001174 MOV #MIXED,$TMP0 ;STARTING ADDRESS OF PATTERN
2872 033730 012737 000200 001176 MOV #128,$TMP1 ;RANGE OF PATTERN
2873 033736 004737 042174 JSR PC,GENBUF ;GO GENERATE BUFFER FOR FORMAT
2874
2875 ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATION
2876 033742 012702 001553 MOV #PUTINX,R2 ;R2 = BYTE ENTRY POSITION
2877 033746 112722 000034 MOVB #RMDC,(R2)+
2878 033752 112722 000006 MOVB #RMDA,(R2)+
2879 033756 112722 000004 MOVB #RMBA,(R2)+
2880 033762 112722 000002 MOVB #RMWC,(R2)+
2881 033766 112722 000032 MOVB #RMOF,(R2)+
2882 033772 112722 000000 MOVB #RMCS1,(R2)+
2883 033776 112712 000200 MOVB #200,(R2) ;TERMINATE TABLE
2884 034002 4$:
2885 034002 004737 043330 JSR PC,PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
034006 000404 BR 5$ ;GO TO 5$ IF NO ERROR
034010 000240 NOP ;RETURN HERE IF ERROR
034012 104000 EMT ;ERROR # DEFINED BY PUT SUBROUTINE
034014 000137 034576 JMP 25$ ;GO TO 25$ IF ERROR
2886 034020 5$:
2887
2888 ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
034020 004737 042774 JSR PC,GETSTS ;GO TO GETSTS SUBROUTINE
2889
2890 ;WAIT FOR COMMAND TO COMPLETE
034024 004737 043672 JSR PC,TIMOUT ;GO TO TIMEOUT SUBROUTINE
2891
2892 ;GO READ STATUS FOR FORMAT OPERATION
2893 034030 004737 043060 JSR PC,GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
034034 000404 BR 6$ ;GO TO 6$ IF NO ERROR
034036 000240 NOP ;RETURN HERE IF ERROR
034040 104000 EMT ;ERROR # DEFINED BY GET SUBROUTINE
034042 000137 034576 JMP 25$ ;GO TO 25$ IF ERROR
2894 034046 6$:
2895
2896 ;VERIFY NO ERRORS DURING FORMAT

```



```

2897 034046 004737 056572      JSR    PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
      034052 000405      BR      7$      ;GO TO 7$ IF NO ERROR
      034054 000240      NOP      ;RETURN HERE IF ERROR
      034056 104000      EMT      ;ERROR # DEFINED BY DTASTS SUBROUTINE
      034060 004736      JSR    PC,a(SP)+ ;GO BACK FOR MORE ERROR CHECKS
      034062 000137 034576      JMP    25$      ;GO TO 25$ IF ERROR

2898 034066      7$:
2899
2900      ;MOVE LOOP ADDRESSES TO NEXT OPERATION
2901 034066 012737 034076 001124      MOV    #8$,SLPERR
2902 034074 000410      BR      9$      ;SKIP TO WRITE OPERATION
2903
2904      ;*****
2905      ;LOOP #2      WRITE,READ
2906
2907 034076      8$:
2908
2909      ;PREPARE DEVICE FOR WRITE OPERATION
2910 034076 004737 037316      JSR    PC,TSTPRP      ;PREPARE DEVICE FOR TEST
      034102 154130      .WORD    154130      ;TASK DESCRIPTOR AS FOLLOWS:
      ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
      ;CLEAR CONTROLLER & SELECT DEVICE
      ;VERIFY CONTROLLER CLEAR OPERATION
      ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
      ;VERIFY PACK ACKNOWLEDGE
      ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
      ;VERIFY RECALIBRATION
      034104 000404      BR      9$      ;GO TO 9$ IF NO ERROR
      034106 000240      NOP      ;RETURN HERE IF ERROR
      034110 104000      EMT      ;ERROR # DEFINED BY TSTPRP SUBROUTINE
      034112 000137 034576      JMP    25$      ;GO TO 25$ IF ERROR

2911 034116      9$:
2912
2913      ;WRITE DATA TO THE DRIVE
2914 034116      10$:
2915 034116 012737 177400 001414      MOV    #-256,,RMWCO      ;CHANGE WORD COUNT
2916 034124 012737 106442 001416      MOV    #BUFONE+4,RMBAO      ;CHANGE ADDRESS
2917 034132 012737 000061 001412      MOV    #WD!GO,RMCS10      ;WRITE DATA COMMAND
2918 034140 012702 001553      MOV    #PUTINX,R2      ;LOAD PUT REGISTER INDEX TABLE
2919 034144 112722 000006      MOV    #RMDA,(R2)+
2920 034150 112722 000034      MOV    #RMDC,(R2)+
2921 034154 112722 000032      MOV    #RMOF,(R2)+
2922 0 160 112722 000004      MOV    #RMBA,(R2)+
2923 034164 112722 000002      MOV    #RMWC,(R2)+
2924 034170 112722 000000      MOV    #RMCS1,(R2)+
2925 034174 112722 000200      MOV    #200,(R2)+      ;TERMINATE TABLE
2926
2927 034200 004737 043330      JSR    PC,PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      034204 000404      BR      11$      ;GO TO 11$ IF NO ERROR
      034206 000240      NOP      ;RETURN HERE IF ERROR
      034210 104000      EMT      ;ERROR # DEFINED BY PUT SUBROUTINE
      034212 000137 034576      JMP    25$      ;GO TO 25$ IF ERROR

2928 034216      11$:
2929
2930      ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
2931 034216 004737 042774      JSR    PC,GETSTS      ;GO TO GETSTS SUBROUTINE

```

```

2932      034222 004737 043672      ;WAIT FOR COMMAND TO COMPLETE
      JSR      PC,TIMOUT      ;GO TO TIMEOUT SUBROUTINE
2933
2934      ;GO READ STATUS FOR WRITE COMMAND
2935      034226 004737 043060      JSR      PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      034232 000404      BR      12$      ;GO TO 12$ IF NO ERROR
      034234 000240      NOP      ;RETURN HERE IF ERROR
      034236 104000      EMT      ;ERROR # DEFINED BY GET SUBROUTINE
      034240 000137 034576      JMP      25$      ;GO TO 25$ IF ERROR
2936      034244
2937
2938      ;CHECK FOR ERRORS DURING WRITE OPERATION
2939      034244 004737 044056      JSR      PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
      034250 000405      BR      13$      ;GO TO 13$ IF NO ERROR
      034252 000240      NOP      ;RETURN HERE IF ERROR
      034254 104000      EMT      ;ERROR # DEFINED BY PRIERR SUBROUTINE
      034256 004736      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      034260 000137 034576      JMP      25$      ;GO TO 25$ IF ERROR
2940      034264
2941      034264 004737 056572      JSR      PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
      034270 000405      BR      14$      ;GO TO 14$ IF NO ERROR
      034272 000240      NOP      ;RETURN HERE IF ERROR
      034274 104000      EMT      ;ERROR # DEFINED BY DTASTS SUBROUTINE
      034276 004736      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      034300 000137 034576      JMP      25$      ;GO TO 25$ IF ERROR
2942      034304
2943      034304 004737 044710      JSR      PC,SECERR      ;GO CHECK FOR SECONDARY ERRORS
      034310 000405      BR      15$      ;GO TO 15$ IF NO ERROR
      034312 000240      NOP      ;RETURN HERE IF ERROR
      034314 104000      EMT      ;ERROR # DEFINED BY SECERR SUBROUTINE
      034316 004736      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      034320 000137 034576      JMP      25$      ;GO TO 25$ IF ERROR
2944      034324
2945
2946      ;CHANGE LOOP ADDRESSES
2947      034324 012737 034334 001124      MOV      #16$,$LPERR
2948      034332 000410      BR      17$      ;SKIP TO NEXT OPERATION
2949
2950      ;*****
2951      ;LOOP #3      READ
2952
2953      034334
2954
2955      ;PREPARE DEVICE FOR READ OPERATION
2956      034334 004737 037316      JSR      PC,TSTPRP      ;PREPARE DEVICE FOR TEST
      034340 154130      .WORD      154130      ;TASK DESCRIPTOR AS FOLLOWS:
      ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
      ;CLEAR CONTROLLER & SELECT DEVICE
      ;VERIFY CONTROLLER CLEAR OPERATION
      ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
      ;VERIFY PACK ACKNOWLEDGE
      ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
      ;VERIFY RECALIBRATION
      034342 000404      BR      17$      ;GO TO 17$ IF NO ERROR
      034344 000240      NOP      ;RETURN HERE IF ERROR
      034346 104000      EMT      ;ERROR # DEFINED BY TSTPRP SUBROUTINE
      034350 000137 034576      JMP      25$      ;GO TO 25$ IF ERROR
  
```

```

2957 034354      17$:
2958
2959      ;READ DATA FROM DEVICE
2960 034354      18$:
2961 034354 012737 107446 001416      MOV      #BUFTWO+4,RMBAD      ;CHANGE BUFFER
2962 034362 012737 000071 001412      MOV      #RD!GO,RMCS10      ;READ DATA COMMAND
2963 034370 012702 001553              MOV      #PUTINX,R2      ;LOAD PUT REGISTER INDEX TABLE
2964 034374 112722 000006              MOV      #RMDA,(R2)+
2965 034400 112722 000032              MOV      #RMOF,(R2)+
2966 034404 112722 000034              MOV      #RMDC,(R2)+
2967 034410 112722 000004              MOV      #RMDA,(R2)+
2968 034414 112722 000002              MOV      #RMDC,(R2)+
2969 034420 112722 000000              MOV      #RMDC,(R2)+
2970 034424 112712 000200              MOV      #200,(R2)
2971
2972 034430 004737 043330              JSR      PC,PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
2973 034434 000404              BR      19$      ;GO TO 19$ IF NO ERROR
2974 034436 000240              NOP      ;RETURN HERE IF ERROR
2975 034440 104000              EMT      ;ERROR # DEFINED BY PUT SUBROUTINE
2976 034442 000137 034576              JMP      25$      ;GO TO 25$ IF ERROR
2977
2978      19$:
2979      ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
2980 034446 004737 042774              JSR      PC,GETSTS      ;GO TO GETSTS SUBROUTINE
2981
2982      ;WAIT FOR COMMAND TO COMPLETE
2983 034452 004737 043672              JSR      PC,TIMOUT      ;GO TO TIMEOUT SUBROUTINE
2984
2985      ;GO READ STATUS FOR READ OPERATION
2986 034456 004737 043060              JSR      PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
2987 034462 000404              BR      20$      ;GO TO 20$ IF NO ERROR
2988 034464 000240              NOP      ;RETURN HERE IF ERROR
2989 034466 104000              EMT      ;ERROR # DEFINED BY GET SUBROUTINE
2990 034470 000137 034576              JMP      25$      ;GO TO 25$ IF ERROR
2991
2992      20$:
2993      ;CHECK FOR ERRORS DURING READ OPERATION
2994 034474 004737 044056              JSR      PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
2995 034500 000405              BR      21$      ;GO TO 21$ IF NO ERROR
2996 034502 000240              NOP      ;RETURN HERE IF ERROR
2997 034504 104000              EMT      ;ERROR # DEFINED BY PRIERR SUBROUTINE
2998 034506 004736              JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
2999 034510 000137 034576              JMP      25$      ;GO TO 25$ IF ERROR
3000
3001      21$:
3002 034514 004737 056572              JSR      PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
3003 034520 000405              BR      22$      ;GO TO 22$ IF NO ERROR
3004 034522 000240              NOP      ;RETURN HERE IF ERROR
3005 034524 104000              EMT      ;ERROR # DEFINED BY DTASTS SUBROUTINE
3006 034526 004736              JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
3007 034530 000137 034576              JMP      25$      ;GO TO 25$ IF ERROR
3008
3009      22$:
3010 034534 004737 044710              JSR      PC,SECERR      ;GO CHECK FOR SECONDARY ERRORS
3011 034540 000405              BR      23$      ;GO TO 23$ IF NO ERROR
3012 034542 000240              NOP      ;RETURN HERE IF ERROR
3013 034544 104000              EMT      ;ERROR # DEFINED BY SECERR SUBROUTINE
3014 034546 004736              JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
3015 034550 000137 034576              JMP      25$      ;GO TO 25$ IF ERROR

```

2989 034554
 2990 034554 004737 C42432
 034560 106442
 034562 107446
 034564 000404
 034566 000240
 034570 104000
 034572 000137 034576
 2991 034576
 2992
 2993 034576
 2994
 2995

23\$: JSR PC,CMPBUF ;GO COMPARE WRITE, READ DATA BUFFERS
 .WORD BUFONE+4 ;STARTING ADDRESS OF WRITE BUFFER
 .WORD BUFTWO+4 ;STARTING ADDRESS OF READ BUFFER
 BR 24\$;GO TO 24\$ IF NO ERROR
 NOP ;RETURN HERE IF ERROR
 EMT ;ERROR # DEFINED BY CMPBUF SUBROUTINE
 JMP 25\$;GO TO 25\$ IF ERROR

24\$:

25\$:

 ;*TEST 24 WRITE, READ EACH TRACK

 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

 ;LOOP #1 FORMAT,WRITE,READ

1\$:

;PREPARE THE DEVICE FOR FORMAT OPERATION

JSR PC,TSTPRP ;PREPARE DEVICE FOR TEST
 .WORD 154130 ;TASK DESCRIPTOR AS FOLLOWS:
 ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
 ;CLEAR CONTROLLER & SELECT DEVICE
 ;VERIFY CONTROLLER CLEAR OPERATION
 ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
 ;VERIFY PACK ACKNOWLEDGE
 ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
 ;VERIFY RECALIBRATION
 BR 2\$;GO TO 2\$ IF NO ERROR
 NOP ;RETURN HERE IF ERROR
 EMT ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 JMP 26\$;GO TO 26\$ IF ERROR

2\$:

;LOAD PARAMETERS AND GENERATE DATA BUFFER

MOV #0,RMDCO ;CYLINDER = 0
 MOV #0,RMDAO ;TRACK = 0, SECTOR 0

3\$:

MOV #BUFONE,RMBAO ;BUS ADDRESS
 MOV #-258,RMWCO ;2 + 256 WORDS (2'S COMP)
 MOV #FMT16,RMOFO ;16 BIT FORMAT
 MOV #WH!GO,RMCS10 ;WRITE HEADER AND DATA COMMAND

;VERIFY THAT SECTOR IS NOT BAD

JSR PC,BADSET ;CALL BAD SECTOR MODULE
 BR 4\$;GO TO 4\$ IF NO ERROR
 TYPE ,SCTMSG ;TYPE BAD SECTOR MESSAGE

034576 000004
 034600 000240
 034602 012706 001100
 034606 013700 001276
 034612 013701 001466
 034616 012737 000024 001226
 2996
 2997
 2998
 2999
 3000 034624
 3001
 3002
 3003 034624 004737 037316
 034630 154130
 034632 000404
 034634 000240
 034636 104000
 034640 000137 035624
 3004 034644
 3005
 3006
 3007 034644 012737 000000 001446
 3008 034652 012737 000000 001420
 3009 034660
 3010 034660 012737 106436 001416
 3011 034666 012737 177376 001414
 3012 034674 012737 010000 001444
 3013 034702 012737 000063 001412
 3014
 3015
 034710 004737 040242
 034714 000405
 034716 104401 070426

```

034722 104000 EMT ;ERROR # DEFINED BY BADSCI SUBROUTINE
034724 000137 035624 JMP 26$ ;GO TO 26$ IF ERROR
3016 034730 4$: MOV #MIXED,$TMP0 ;STARTING ADDRESS OF PATTERN
3017 034730 012737 071756 001174 MOV #128,$TMP1 ;RANGE OF PATTERN
3018 034736 012737 000200 001176 JSR PC,GENBUF ;GO GENERATE BUFFER FOR FORMAT
3019 034744 004737 042774
3020
3021 ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATION
3022 034750 012702 001553 MOV #PUTINX,R2 ;R2 = BYTE ENTRY POSITION
3023 034754 112722 000034 MOVB #RMDX,(R2)+
3024 034760 112722 000006 MOVB #RMDA,(R2)+
3025 034764 112722 000004 MOVB #RMDA,(R2)+
3026 034770 112722 000002 MOVB #RMDC,(R2)+
3027 034774 112722 000032 MOVB #RMDA,(R2)+
3028 035000 112722 000000 MOVB #RMC1,(R2)+
3029 035004 112712 000200 MOVB #200,(R2) ;TERMINATE TABLE
3030 035010
3031
3032 ;FORMAT THE DRIVE
3033 035010 004737 043330 JSR PC,PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
035014 000404 BR 6$ ;GO TO 6$ IF NO ERROR
035016 000240 NOP ;RETURN HERE IF ERROR
035020 104000 EMT ;ERROR # DEFINED BY PUT SUBROUTINE
035022 000137 035624 JMP 26$ ;GO TO 26$ IF ERROR
3034 035026
3035
3036 ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
035026 004737 042774 JSR PC,GETSTS ;GO TO GETSTS SUBROUTINE
3037
3038 ;WAIT FOR COMMAND TO COMPLETE
035032 004737 043672 JSR PC,TIMOUT ;GO TO TIMEOUT SUBROUTINE
3039
3040 ;GO READ STATUS FOR FORMAT OPERATION
3041 035036 004737 043060 JSR PC,GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
035042 000404 BR 7$ ;GO TO 7$ IF NO ERROR
035044 000240 NOP ;RETURN HERE IF ERROR
035046 104000 EMT ;ERROR # DEFINED BY GET SUBROUTINE
035050 000137 035624 JMP 26$ ;GO TO 26$ IF ERROR
3042 035054
3043
3044 ;VERIFY NO ERRORS DURING FORMAT
3045 035054 004737 056572 JSR PC,DASTS ;GO VERIFY RESULTS OF DATA TRANSFER
035060 000405 BR 8$ ;GO TO 8$ IF NO ERROR
035062 000240 NOP ;RETURN HERE IF ERROR
035064 104000 EMT ;ERROR # DEFINED BY DASTS SUBROUTINE
035066 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
035070 000137 035624 JMP 26$ ;GO TO 26$ IF ERROR
3046 035074
3047
3048 ;MOVE LOOP ADDRESSES TO NEXT OPERATION
3049 035074 012737 035104 001124 MOV #9$,$LPERR
3050 035102 000410 BR 10$ ;SKIP TO WRITE OPERATION
3051
3052 ;*****
3053 ;LOOP #2 WRITE,READ
3054
3055 035104 9$:

```

Address	Op Code	Op Name	Op Data	Op Comment
3056				
3057				
3058	035104	004737	037316	
	035110	154130		
		JSR	PC,TSTPRP	;PREPARE DEVICE FOR WRITE OPERATION
		.WORD	154130	
				;PREPARE DEVICE FOR TEST
				;TASK DESCRIPTOR AS FOLLOWS:
				;SELECT DEVICE & VERIFY DEVICE AVAILABLE
				;CLEAR CONTROLLER & SELECT DEVICE
				;VERIFY CONTROLLER CLEAR OPERATION
				;PACK ACKNOWLEDGE IF VOLUME NOT VALID
				;VERIFY PACK ACKNOWLEDGE
				;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
				;VERIFY RECALIBRATION
	035112	000404		
	035114	000240		
	035116	104000		
	035120	000137	035624	
	035124			
3059				
3060				
3061				
3062	035124			
3063	035124	012737	106442	001416
3064	035132	012737	177400	001414
3065	035140	012737	000061	001412
3066	035146	012702	001553	
3067	035152	112722	000006	
3068	035156	112722	000034	
3069	035162	112722	000032	
3070	035166	112722	000004	
3071	035172	112722	000002	
3072	035176	112722	000000	
3073	035202	112722	000200	
3074				
3075	035206	004737	043330	
	035212	000404		
	035214	000240		
	035216	104000		
	035220	000137	035624	
3076	035224			
3077				
3078				
3079	035224	004737	042774	
3080				
3081	035230	004737	043672	
3082				
3083	035234	004737	043060	
	035240	000404		
	035242	000240		
	035244	104000		
	035246	000137	035624	
3084	035252			
3085				
3086				
3087	035252	004737	044056	
	035256	000405		
	035260	000240		
	035262	104000		
		JSR	PC,PRIERR	;CHECK FOR ERRORS DURING WRITE OPERATION
		BR	14\$	
		NOP		
		EMT		
				;GO CHECK FOR PRIMARY ERRORS
				;GO TO 14\$ IF NO ERROR
				;RETURN HERE IF ERROR
				;ERROR # DEFINED BY PRIERR SUBROUTINE


```

    035264 004736      JSR    PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
    035266 000137 035624  JMP    26$      ;GO TO 26$ IF ERROR
3088 035272
3089 035272 004737 056572 14$:      JSR    PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
    035276 000405      BR    15$      ;GO TO 15$ IF NO ERROR
    035300 000240      NOP                ;RETURN HERE IF ERROR
    035302 104000      EMT                ;ERROR # DEFINED BY DTASTS SUBROUTINE
    035304 004736      JSR    PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
    035306 000137 035624  JMP    26$      ;GO TO 26$ IF ERROR
3090 035312
3091 035312 004737 044710 15$:      JSR    PC,SECERR      ;GO CHECK FOR SECONDARY ERRORS
    035316 000405      BR    16$      ;GO TO 16$ IF NO ERROR
    035320 000240      NOP                ;RETURN HERE IF ERROR
    035322 104000      EMT                ;ERROR # DEFINED BY SECERR SUBROUTINE
    035324 004736      JSR    PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
    035326 000137 035624  JMP    26$      ;GO TO 26$ IF ERROR
3092 035332
3093
3094
3095 035332 012737 035342 001124 ;CHANGE LOOP ADDRESSES
3096 035340 000410      MOV    #17$,$LPERR
3097
3098
3099
3100
3101 035342
3102
3103
3104 035342 004737 037316 ;PREPARE DEVICE FOR READ OPERATION
    035346 154130      JSR    PC,TSTPRP      ;PREPARE DEVICE FOR TEST
    .WORD    154130      .WORD    154130      ;TASK DESCRIPTOR AS FOLLOWS:
    ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
    ;CLEAR CONTROLLER & SELECT DEVICE
    ;VERIFY CONTROLLER CLEAR OPERATION
    ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
    ;VERIFY PACK ACKNOWLEDGE
    ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
    ;VERIFY RECALIBRATION
    035350 000404      BR    18$      ;GO TO 18$ IF NO ERROR
    035352 000240      NOP                ;RETURN HERE IF ERROR
    035354 104000      EMT                ;ERROR # DEFINED BY TSTPRP SUBROUTINE
    035356 000137 035624  JMP    26$      ;GO TO 26$ IF ERROR
3105 035362
3106
3107
3108 035362
3109 035362 012737 107446 001416 ;READ DATA FROM DEVICE
3110 035370 012737 000071 001412 19$:      MOV    #BUFTWO+4,RMBA0      ;CHANGE BUS ADDRESS
3111 035376 012702 001553      MOV    #RD.GO,RMC$10      ;READ DATA COMMAND
3112 035402 112722 000006      MOV    #PUTINX,R2      ;LOAD PUT REGISTER INDEX TABLE
3113 035406 112722 000032      MOV    #RMDA,(R2)+
3114 035412 112722 000034      MOV    #RMOF,(R2)+
3115 035416 112722 000004      MOV    #RMDC,(R2)+
3116 035422 112722 000002      MOV    #RMBA,(R2)+
3117 035426 112722 000000      MOV    #RMWC,(R2)+
3118 035432 112712 000200      MOV    #RMC$1,(R2)+
3119
3120 035436 004737 043330      MOV    #200,(R2)
    JSR    PC,PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
  
```

```

035442 000404 BR 20$ ;GO TO 20$ IF NO ERROR
035444 000240 NOP ;RETURN HERE IF ERROR
035446 104000 EMT ;ERROR # DEFINED BY PUT SUBROUTINE
035450 000137 035624 JMP 26$ ;GO TO 26$ IF ERROR
3121 035454 20$:
3122
3123 ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
035454 004737 042774 JSR PC,GETSTS ;GO TO GETSTS SUBROUTINE
3124
3125 ;WAIT FOR COMMAND TO COMPLETE
035460 004737 043672 JSR PC,TIMOUT ;GO TO TIMEOUT SUBROUTINE
3126
3127 ;GO READ STATUS FOR READ OPERATION
3128 035464 004737 043060 JSR PC,GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
035470 000404 BR 21$ ;GO TO 21$ IF NO ERROR
035472 000240 NOP ;RETURN HERE IF ERROR
035474 104000 EMT ;ERROR # DEFINED BY GET SUBROUTINE
035476 000137 035624 JMP 26$ ;GO TO 26$ IF ERROR
3129 035502 21$:
3130
3131 ;CHECK FOR ERRORS DURING READ OPERATION
3132 035502 004737 044056 JSR PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
035506 000405 BR 22$ ;GO TO 22$ IF NO ERROR
035510 000240 NOP ;RETURN HERE IF ERROR
035512 104000 EMT ;ERROR # DEFINED BY PRIERR SUBROUTINE
035514 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
035516 000137 035624 JMP 26$ ;GO TO 26$ IF ERROR
3133 035522 22$:
3134 035522 004737 056572 JSR PC,DASTS ;GO VERIFY RESULTS OF DATA TRANSFER
035526 000405 BR 23$ ;GO TO 23$ IF NO ERROR
035530 000240 NOP ;RETURN HERE IF ERROR
035532 104000 EMT ;ERROR # DEFINED BY DASTS SUBROUTINE
035534 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
035536 000137 035624 JMP 26$ ;GO TO 26$ IF ERROR
3135 035542 23$:
3136 035542 004737 044710 JSR PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
035546 000405 BR 24$ ;GO TO 24$ IF NO ERROR
035550 000240 NOP ;RETURN HERE IF ERROR
035552 104000 EMT ;ERROR # DEFINED BY SECERR SUBROUTINE
035554 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
035556 000137 035624 JMP 26$ ;GO TO 26$ IF ERROR
3137 035562 24$:
3138 035562 004737 042432 JSR PC,CMPBUF ;GO COMPARE WRITE, READ DATA BUFFERS
035566 106442 .WORD BUFOne+4 ;STARTING ADDRESS OF WRITE BUFFER
035570 107446 .WORD BUFTwo+4 ;STARTING ADDRESS OF READ BUFFER
035572 000404 BR 25$ ;GO TO 25$ IF NO ERROR
035574 000240 NOP ;RETURN HERE IF ERROR
035576 104000 EMT ;ERROR # DEFINED BY CMPBUF SUBROUTINE
035600 000137 035624 JMP 26$ ;GO TO 26$ IF ERROR
3139 035604 25$:
3140 035604 105237 001421 INCB RMDAO+1 ;ADVANCE TO NEXT TRACK
3141 035610 123737 001421 001335 CMPB RMDAO+1,LSTRK+1 ;DONE ?
3142 035616 101002 BHI 26$ ;YES!!
3143 035620 000137 034660 JMP 3$ ;TEST NEXT TRACK
3144 035624 26$:
3145
3146 ;*****

```



```

; *TEST 25      READ, WRITE CHECK MULTIPLE SECTORS IN OFFSET MODE
; *****
TST25:
    035624      000004      SCOPE      ;SCOPE CALL
    035626      000240      NOP        ;START OF TEST
    035630      012706      001100      MOV     #STACK,SP      ;INITIALIZE STACK POINTER
    035634      013700      001276      MOV     $BASE,R0       ;R0 = UNIBUS ADDRESS
    035640      013701      001466      MOV     TSTQUE,R1      ;(R1) = DEVICE BEING TESTED
    035644      012737      000025      001226      MOV     #25,$TESTN ;:SET TEST NUMBER IN APT MAIL BOX

3147
3148
3149
3150 035652
3151
3152
3153 035652      004737      037316      JSR     PC,TSTPRP      ;PREPARE DEVICE FOR TEST
    035656      154130      .WORD      154130      ;PREPARE DEVICE FOR TEST
                                                ;TASK DESCRIPTOR AS FOLLOWS:
                                                ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
                                                ;CLEAR CONTROLLER & SELECT DEVICE
                                                ;VERIFY CONTROLLER CLEAR OPERATION
                                                ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                ;VERIFY PACK ACKNOWLEDGE
                                                ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
                                                ;VERIFY RECALIBRATION
    035660      000404      BR         2$      ;GO TO 2$ IF NO ERROR
    035662      000240      NOP        ;RETURN HERE IF ERROR
    035664      104000      EMT        ;ERROR # DEFINED BY TSTPRP SUBROUTINE
    035666      000137      037006      JMP     28$      ;GO TO 28$ IF ERROR

3154
3155
3156 035672
3157 035672      012737      000000      001446      MOV     #0,RMDCO      ;CYLINDER = 0
3158 035700      012737      000000      001420      MOV     #0,RMDAO      ;TRACK = 0, SECTOR 0
3159 035706      012737      106436      001416      MOV     #BUFONE,RMBAO      ;BUFFER ADDRESS
3160 035714      012737      176774      001414      MOV     #-258.*2,RMWCO      ;WORD COUNT FOR 2 SECTORS (2'S COMP)
3161 035722      012737      010000      001444      MOV     #FMT16,RMOFO      ;16 BIT MODE
3162 035730      012737      000063      001412      MOV     #WH'GO,RMCS10      ;WRITE HEADER
3163
3164
    035736      004737      040242      JSR     PC,BADSCT      ;CALL BAD SECTOR MODULE
    035742      000405      BR         3$      ;GO TO 3$ IF NO ERROR
    035744      104401      070426      TYPE     ,SCTMSG      ;TYPE BAD SECTOR MESSAGE
    035750      104000      EMT        ;ERROR # DEFINED BY BADSCT SUBROUTINE
    035752      000137      037006      JMP     28$      ;GO TO 28$ IF ERROR

3165 035756
3166 035756      012737      072060      001174      MOV     #ZEROS,$TMP0      ;DATA PATTERN
3167 035764      012737      000001      001176      MOV     #1,$TMP1
3168 035772      004737      042174      JSR     PC,GENBUF      ;TO GENERATE BUFFER FOR FORMAT
3169
3170
;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATION
    035776      012702      001553      MOV     #PUTINX,R2      ;BYTE ENTRY TABLE
    036002      112722      000034      MOVB    #RMDC,(R2)+
    036006      112722      000006      MOVB    #RMDA,(R2)+
    036012      112722      000004      MOVB    #RMBA,(R2)+
    036016      112722      000002      MOVB    #RMWC,(R2)+
    036022      112722      000032      MOVB    #RMOF,(R2)+
    036026      112722      000000      MOVB    #RMCS1,(R2)+
  
```

```

3178 036032 112712 000200      MOVB    #200,(R2)          ;TABLE TERMINATOR
3179 036036      4$:          JSR      PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
3180 036036 004737 043330      BR      5$          ;GO TO 5$ IF NO ERROR
      036042 000404          NOP          ;RETURN HERE IF ERROR
      036044 000240          EMT          ;ERROR # DEFINED BY PUT SUBROUTINE
      036046 104000          JMP      28$         ;GO TO 28$ IF ERROR
      036050 000137 037006      5$:
3181 036054      ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
3182      JSR      PC,GETSTS        ;GO TO GETSTS SUBROUTINE
3183 036054 004737 042774
3184      ;WAIT FOR COMMAND TO COMPLETE
3185      JSR      PC,TIMOUT        ;GO TO TIMOUT SUBROUTINE
3186 036060 004737 043672
3187      ;GO READ STATUS FOR FORMAT OPERATION
3188 036064 004737 043060      JSR      PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
      036070 000404      BR      6$          ;GO TO 6$ IF NO ERROR
      036072 000240      NOP          ;RETURN HERE IF ERROR
      036074 104000      EMT          ;ERROR # DEFINED BY GET SUBROUTINE
      036076 000137 037006      JMP      28$         ;GO TO 28$ IF ERROR
3189 036102      6$:
3190      ;VERIFY NO ERROR DURING FORMAT
3191      JSR      PC,DTASTS        ;GO VERIFY RESULTS OF DATA TRANSFER
3192 036102 004737 056572      BR      7$          ;GO TO 7$ IF NO ERROR
      036106 000405      NOP          ;RETURN HERE IF ERROR
      036110 000240      EMT          ;ERROR # DEFINED BY DTASTS SUBROUTINE
      036112 104000      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      036114 004736      JMP      28$         ;GO TO 28$ IF ERROR
      036116 000137 037006
3193      ;MOV LOOP ADDRESSES TO NEXT OPERATION
3194      7$:
3195 036122      MOV      #8$,$LPERR      ;ERROR LOOP ADDRESS
3196 036122 012737 036144 001124
3197      ;LOCATE BUFFER ADDRESS AND WORD COUNT
3198      MOV      #-256.*2,RMWCO      ;TWO SECTORS
3199 036130 012737 177000 001414      MOV      #BUFONE,RMBAO      ;BUFFER ADDRESS
3200 036136 012737 106436 001416
3201      ;*****
3202      ;LOOP 2 READ AND WRITE CHECK IN OFFSET MODE
3203 036144      8$:
3204      ;PREPARE DEVICE FOR READ OFFSET OPEATION
3205 036144 004737 037316      JSR      PC,TSTPRP      ;PREPARE DEVICE FOR TEST
      036150 154130      .WORD      154130      ;TASK DESCRIPTOR AS FOLLOWS:
      ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
      ;CLEAR CONTROLLER & SELECT DEVICE
      ;VERIFY CONTROLLER CLEAR OPERATION
      ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
      ;VERIFY PACK ACKNOWLEDGE
      ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
      ;VERIFY RECALIBRATION
      036152 000404      BR      9$          ;GO TO 9$ IF NO ERROR
      036154 000240      NOP          ;RETURN HERE IF ERROR
      036156 104000      EMT          ;ERROR # DEFINED BY TSTPRP SUBROUTINE
      036160 000137 037006      JMP      28$         ;GO TO 28$ IF ERROR
3206 036164      9$:
3207 036164      10$:

```

```

3208 ;READ OFFSET
3209 036164 012702 001553 MOV #PUTINX,R2 ;LOAD THE TABLE
3210 036170 112722 000006 MOVB #RMDA,(R2)+
3211 036174 112722 000034 MOVB #RMDC,(R2)+
3212 036200 112722 000032 MOVB #RMOF,(R2)+
3213 036204 112722 000004 MOVB #RMBA,(R2)+
3214 036210 112722 000002 MOVB #RMWC,(R2)+
3215 036214 112722 000200 MOVB #200,(R2)+ ;TABLE TERMINATOR
3216
3217 036220 004737 043330 JSR PC,PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
    036224 000404 BR 11$ ;GO TO 11$ IF NO ERROR
    036226 000240 NOP ;RETURN HERE IF ERROR
    036230 104000 EMT ;ERROR # DEFINED BY PUT SUBROUTINE
    036232 000137 037006 JMP 28$ ;GO TO 28$ IF ERROR
3218 036236 012737 000015 001412 11$: MOV #OFFSET.GO,RMCS10 ;OFFSET COMMAND
3219 036244 012702 001553 MOV #PUTINX,R2
3220 036250 112722 000000 MOVB #RMCS1,(R2)+
3221 036254 112722 000200 MOVB #200,(R2)+
3222
3223 036260 004737 043330 JSR PC,PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
    036264 000404 BR 12$ ;GO TO 12$ IF NO ERROR
    036266 000240 NOP ;RETURN HERE IF ERROR
    036270 104000 EMT ;ERROR # DEFINED BY PUT SUBROUTINE
    036272 000137 037006 JMP 28$ ;GO TO 28$ IF ERROR
3224 036276 12$:
3225
3226 ;WAIT FOR COMMAND TO COMPLETE
    JSR PC,TIMOUT ;GO TO TIMEOUT SUBROUTINE
3227
3228 036302 012737 000071 001412 MOV #RD!GO,RMCS10 ;READ DATA COMMAND
3229 036310 012702 001553 MOV #PUTINX,R2 ;TABLE INDEX
3230 036314 112722 000000 MOVB #RMCS1,(R2)+
3231 036320 112712 000200 MOVB #200,(R2) ;TABLE TERMINATOR
3232
3233 036324 004737 043330 JSR PC,PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
    036330 000404 BR 13$ ;GO TO 13$ IF NO ERROR
    036332 000240 NOP ;RETURN HERE IF ERROR
    036334 104000 EMT ;ERROR # DEFINED BY PUT SUBROUTINE
    036336 000137 037006 JMP 28$ ;GO TO 28$ IF ERROR
3234 036342 13$:
3235
3236 ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
    JSR PC,GETSTS ;GO TO GETSTS SUBROUTINE
3237
3238 ;WAIT FOR COMMAND TO COMPLETE
    JSR PC,TIMOUT ;GO TO TIMEOUT SUBROUTINE
3239
3240 ;READ STATUS FOR READ OFFSET OPERATION
3241 036352 004737 043060 JSR PC,GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
    036356 000404 BR 14$ ;GO TO 14$ IF NO ERROR
    036360 000240 NOP ;RETURN HERE IF ERROR
    036362 104000 EMT ;ERROR # DEFINED BY GET SUBROUTINE
    036364 000137 037006 JMP 28$ ;GO TO 28$ IF ERROR
3242
3243 ;CHECK ERROR DURING READ OFFSET
3244 036370 14$:
3245 036370 004737 044056 JSR PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
  
```

```

036374 000405 BR 15$ ;GO TO 15$ IF NO ERROR
036376 000240 NOP ;RETURN HERE IF ERROR
036400 104000 EMT ;ERROR # DEFINED BY PRIERR SUBROUTINE
036402 004736 JSR PC,a(SP)+ ;GO BACK FOR MORE ERROR CHECKS
036404 000137 037006 JMP 28$ ;GO TO 28$ IF ERROR
3246 036410 15$: JSR PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
3247 036410 004737 056572 BR 16$ ;GO TO 16$ IF NO ERROR
036414 000405 NOP ;RETURN HERE IF ERROR
036416 000240 EMT ;ERROR # DEFINED BY DTASTS SUBROUTINE
036420 104000 JSR PC,a(SP)+ ;GO BACK FOR MORE ERROR CHECKS
036422 004736 JMP 28$ ;GO TO 28$ IF ERROR
036424 000137 037006
3248 036430 16$: JSR PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
3249 036430 004737 044710 BR 17$ ;GO TO 17$ IF NO ERROR
036434 000405 NOP ;RETURN HERE IF ERROR
036436 000240 EMT ;ERROR # DEFINED BY SECERR SUBROUTINE
036440 104000 JSR PC,a(SP)+ ;GO BACK FOR MORE ERROR CHECKS
036442 004736 JMP 28$ ;GO TO 28$ IF ERROR
036444 000137 037006
3250 036450 17$:
3251
3252 ;CHANGE LOOP ADDRESS
3253 036450 012737 036460 001124 MOV #18$,SLPERR
3254 036456 000410 BR 19$ ;TO NEXT OPERATION
3255
3256 ;*****
3257 ;LOOP 3 WRITE CHECK OFFSET
3258 036460 18$:
3259
3260 ;PREPARE DEVICE FOR READ OPERATION
3261 036460 004737 037316 JSR PC,TSTPRP ;PREPARE DEVICE FOR TEST
036464 154130 .WORD 154130 ;TASK DESCRIPTOR AS FOLLOWS:
;SELECT DEVICE & VERIFY DEVICE AVAILABLE
;CLEAR CONTROLLER & SELECT DEVICE
;VERIFY CONTROLLER CLEAR OPERATION
;PACK ACKNOWLEDGE IF VOLUME NOT VALID
;VERIFY PACK ACKNOWLEDGE
;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
;VERIFY RECALIBRATION
036466 000404 BR 19$ ;GO TO 19$ IF NO ERROR
036470 000240 NOP ;RETURN HERE IF ERROR
036472 104000 EMT ;ERROR # DEFINED BY TSTPRP SUBROUTINE
036474 000137 037006 JMP 28$ ;GO TO 28$ IF ERROR
3262 036500 19$:
3263
3264 ;WRITE CHECK DATA IN OFFSET MODE
3265 036500 20$:
3266 036500 012702 001553 MOV #PUTINX,R2
3267 036504 112722 000006 MOVB #RMDA,(R2)+
3268 036510 112722 000032 MOVB #RMOF,(R2)+
3269 036514 112722 000034 MOVB #RMDC,(R2)+
3270 036520 112722 000004 MOVB #RIBA,(R2)+
3271 036524 112722 000002 MOVB #RMWC,(R2)+
3272 036530 112712 000200 MOVB #200,(R2) ;TABLE TERMINATOR
3273
3274 036534 004737 043330 JSR PC,PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
036540 000404 BR 21$ ;GO TO 21$ IF NO ERROR

```

```

036542 000240      NOP      ;RETURN HERE IF ERROR
036544 104000      EMT      ;ERROR # DEFINED BY PUT SUBROUTINE
036546 000137 037006 JMP      28$      ;GO TO 28$ IF ERROR

3275
3276      ;SETUP OFFSET MODE FIRST
3277 036552      21$:      MOV      #OFFSET!GO, RMCS10      ;OUTPUT BUFFER
3278 036552 012737 000015 001412 MOV      #PUTINX, R2
3279 036560 012702 001553      MOV      #RMCS1, (R2)+
3280 036564 112722 000000      MOV      #200, (R2)
3281 036570 112712 000200
3282
3283 036574 004737 043330      JSR      PC, PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
036600 000404      BR      22$      ;GO TO 22$ IF NO ERROR
036602 000240      NOP      ;RETURN HERE IF ERROR
036604 104000      EMT      ;ERROR # DEFINED BY PUT SUBROUTINE
036606 000137 037006 JMP      28$      ;GO TO 28$ IF ERROR

3284 036612      22$:
3285
3286      ;WAIT FOR COMMAND TO COMPLETE
036612 004737 043672      JSR      PC, TIMOUT      ;GO TO TIMOUT SUBROUTINE

3287
3288      ;EXECUTE WRITE CHECK DATA COMMAND
3289 036616 012737 000051 001412 MOV      #WCD!GO, RMCS10
3290 036624 012702 001553      MOV      #PUTINX, R2
3291 036630 112722 000000      MOV      #RMCS1, (R2)+
3292 036634 112712 000200      MOV      #200, (R2)
3293
3294 036640 004737 043330      JSR      PC, PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
036644 000404      BR      23$      ;GO TO 23$ IF NO ERROR
036646 000240      NOP      ;RETURN HERE IF ERROR
036650 104000      EMT      ;ERROR # DEFINED BY PUT SUBROUTINE
036652 000137 037006 JMP      28$      ;GO TO 28$ IF ERROR

3295 036656      23$:
3296
3297      ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
036656 004737 042774      JSR      PC, GETSTS      ;GO TO GETSTS SUBROUTINE

3298
3299      ;WAIT FOR COMMAND TO COMPLETE
036662 004737 043672      JSR      PC, TIMOUT      ;GO TO TIMOUT SUBROUTINE

3300
3301      ;READ STATUS FOR WRITE CHECK OFFSET OPERATION
3302 036666 004737 043060      JSR      PC, GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
036672 000404      BR      24$      ;GO TO 24$ IF NO ERROR
036674 000240      NOP      ;RETURN HERE IF ERROR
036676 104000      EMT      ;ERROR # DEFINED BY GET SUBROUTINE
036700 000137 037006 JMP      28$      ;GO TO 28$ IF ERROR

3303 036704      24$:
3304
3305      ;CHECK ERROR DURING WRITE CHECK OFFSET OPERATION
3306 036704 004737 044056      JSR      PC, PRIERR      ;GO CHECK FOR PRIMARY ERRORS
036710 000405      BR      25$      ;GO TO 25$ IF NO ERROR
036712 000240      NOP      ;RETURN HERE IF ERROR
036714 104000      EMT      ;ERROR # DEFINED BY PRIERR SUBROUTINE
036716 004736      JSR      PC, @ (SP)+      ;GO BACK FOR MORE ERROR CHECKS
036720 000137 037006 JMP      28$      ;GO TO 28$ IF ERROR

3307 036724      25$:
3308 036724 004737 056572      JSR      PC, DTAITS      ;GO VERIFY RESULTS OF DATA TRANSFER

```

	036730	000405		BR	26\$	:GO TO 26\$ IF NO ERROR
	036732	000240		NOP		:RETURN HERE IF ERROR
	036734	104000		EMT		:ERROR # DEFINED BY DTASTS SUBROUTINE
	036736	004736		JSR	PC,@(SP)+	:GO BACK FOR MORE ERROR CHECKS
	036740	000137	037006	JMP	28\$	:GO TO 28\$ IF ERROR
3309	036744					
3310	036744	004737	044710	JSR	PC,SECERR	:GO CHECK FOR SECONDARY ERRORS
	036750	000405		BR	27\$	:GO TO 27\$ IF NO ERROR
	036752	000240		NOP		:RETURN HERE IF ERROR
	036754	104000		EMT		:ERROR # DEFINED BY SECERR SUBROUTINE
	036756	004736		JSR	PC,@(SP)+	:GO BACK FOR MORE ERROR CHECKS
	036760	000137	037006	JMP	28\$	:GO TO 28\$ IF ERROR
3311	036764					
3312	036764	032737	000200 001444	BIT	#OFD,RMOFO	:IN FORWARD DIRECTION OF OFFSET ?
3313	036772	001005		BNE	28\$	:BRANCH IF SO
3314	036774	052737	000200 001444	BIS	#OFD,RMOFO	:SET TO FORWARD DIRECTION
3315	037002	000137	036122	JMP	7\$	:CHANGE OFFSET DIRECTION TEST AGAIN
3316	037006					


```

1      .SBTTL  END OF SUB-PASS ROUTINE
2
3      ;THIS IS THE END OF SUB-PASS ROUTINE. THIS ROUTINE IS USED TO
4      ;TERMINATE THE OPERATION OF THE CURRENT DEVICE UNDER TEST AND
5      ;SELECT THE NEXT DEVICE FOR TEST. IF THERE ARE NO MORE DEVICES
6      ;TO TEST, EXIT IS MADE TO '$EOP' ROUTINE. OTHERWISE, RETURN
7      ;IS MADE TO 'SHUT' ROUTINE.
8
9      037006 000004      $EOSP:  SCOPE
10     037010 000240      NOP
11     037012 013700 001466      MOV     TSTQUE,R0      ;GET POINTER TO TSTQUE
12     037016 062700 000002      ADD     #2,R0      ;ADJUST PCINTER TO NEXT DEVICE
13     037022 010037 001466      MOV     R0,TSTQUE      ;SAVE POINTER TO TSTQUE
14     037026 005710      TST     (R0)      ;ANY MORE DEVICES FOR TEST ?
15     037030 001402      BEQ     1$      ;BR IF NO
16     037032 000137 037262      JMP     SHUT      ;JUMP TO 'SHUT' ROUTINE
17     037036 012737 001470 001466 1$:  MOV     #TSTQUE+2,TSTQUE      ;INITIALIZE POINTER TO FIRST DEVICE IN
18                                     ;TEST QUE TABLE
19
20     .SBTTL  END OF PASS ROUTINE
  
```

```

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

037044      $EOP:
037044      NOP
037046 000240      CLR     $TSTNM      ;;ZERO THE TEST NUMBER
037052 005037 001116      CLR     $TIMES      ;;ZERO THE NUMBER OF ITERATIONS
037056 005237 001206      INC     $PASS      ;;INCREMENT THE PASS NUMBER
037062 042737 001230 001230      BIC     #100000,$PASS      ;;DON'T ALLOW A NEG. NUMBER
037070 005327      DEC     (PC)+      ;;LOOP?
037072 000001      $EOPCT: .WORD 1
037074 003066      BGT     $DOAGN      ;;YES
037076 012737      MOV     (PC)+,a(PC)+      ;;RESTORE COUNTER
037100 000001      $ENDCT: .WORD 1
037102 037072      $EOPCT
037104 104401 037112      TYPE     ,65$      ;;TYPE ASCII STRING
037110 000407      BR      64$      ;;GET OVER THE ASCII
;;65$: .ASCIIZ <12><15>/END PASS #/
64$:
037130      MOV     $PASS,-(SP)      ;;SAVE $PASS FOR TYPEOUT
037130 013746 001230      ;;TYPE PASS NUMBER
037134 104405      TYPDS      ;;GO TYPE--DECIMAL ASCII WITH SIGN
037136 005737 001126      TST     $ERTTL      ;;SEE IF ANY ERRORS THIS PASS
037142 001431      BEQ     $GT42P      ;;BR IF NO ERRORS TO REPORT
037144 104401 037152      TYPE     ,67$      ;;TYPE ASCII STRING
037150 000421      BR      66$      ;;GET OVER THE ASCII
;;67$: .ASCIIZ / TOTAL ERRORS SINCE LAST REPORT /
66$:
037214      MOV     $ERTTL,-(SP)      ;;SAVE $ERTTL FOR TYPEOUT
037214 013746 001126      ;;TOTAL NUMBER OF ERRORS
037220 104405      TYPDS      ;;GO TYPE--DECIMAL ASCII WITH SIGN
037222 005037 001126      CLR     $ERTTL      ;;CLEAR ERROR TOTAL
  
```

CZRM080 RM05/3/2 FCTNL TST 3
END OF PASS ROUTINE

MACRO V04.00 4-APR-81 01:15:59 PAGE 14-1

SEQ 0174

037226	104401	001217		\$GT42P: TYPE	,\$CRLF	::TYPE CARRIAGE RETURN, LINE FEED
037232	013700	000042		\$GET42: MOV	@#42,R0	::GET MONITOR ADDRESS
037236	001405			BEQ	\$DOAGN	::BRANCH IF NO MONITOR
037240	000005			RESET		::CLEAR THE WORLD
037242	004710			\$ENDAD: JSR	PC,(R0)	::GO TO MONITOR
037244	000240			NOP		::SAVE ROOM
037246	000240			NOP		::FOR
037250	000240			NOP		::ACT11
037252				\$DOAGN:		
037252	000137			JMP	@(PC)+	::RETURN
037254	037262			\$RTNAD: .WORD	SHUT	
037256	377	377	000	\$ENULL: .BYTE	-1,-1,0	::NULL CHARACTER STRING
				.EVEN		
21						
22	037262	005737	001326	SHUT: TST	CTLFG	:WAS CONTROL C FLAGGED ?
23	037266	001002		BNE	1\$	:BR IF YES
24	037270	000137	007670	JMP	READY	:CONTINUE
25	037274	005737	000042	1\$: TST	@#42	:ANY MONITOR PRESENT ?
26	037300	001002		BNE	2\$	:BR IF YES
27	037302	000137	005432	JMP	START	:GO TO START
28	037306	005037	001300	2\$: CLR	\$DEVN	:FUDGE NO DRIVES IN MAP
29	037312	000137	037044	JMP	\$EOP	:RETURN TO \$EOP

.SBTTL TEST PREPARATION MODULE

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

:CALL:

JSR	PC,TSTPRP	
.WORD	NN'NNN	TASK/VERIFY DESCRIPTOR
BR	??	RETURN HERE IF NO ERROR
NOP		RETURN HERE IF ERROR
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	@(SP),500\$	:STORE DESCRIPTOR
ADD	#6,(SP)	:MOVE SP TO USERS ERROR CALL
CLRB	@(SP)	:CLEAR ERROR CALL
SUB	#4,(SP)	:MOVE SP TO NO ERROR RETURN

JSR	PC,GETSTS	:SETUP TO READ ALL REGISTERS
JSR	PC,GET	:GET RMER2
BR	15\$	:BR IF NO ERROR DETECTED
BR	10\$	:GET OVER ERROR NUMBER
.WORD	0	:ERROR DEFINED BY GET SUBROUTINE
ADD	#4,(SP)	:XFER ERROR TO USER AND
MOVB	10\$-2,@(SP)	:GET ERROR NUMBER.
JMP	400\$	

10\$:	ADD	#4,(SP)	
15\$:	MOV	RMER2I,505\$	:GET RMER2 AND SAVE FOR LATER

;;*****

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

037316

037316	017637	000000	040236
037324	062716	000006	
037330	105076	000000	
037334	162716	000004	
037340	004737	042774	
037344	004737	043060	
037350	000411		
037352	000401		
037354	000000		
037356	062716	000004	
037362	113776	037354	000000
037370	000137	040226	
037374	013737	001400	040240

```

58                                     ;SELECT DEVICE AND VERIFY DEVICE AVAILABLE IF BIT 15 SET IN TASK
59 037402 005737 040236               TST    500$      ;SELECT DEVICE??
60 037406 100014                       BPL    30$      ;NO!!
61
62 037410 004737 050762               JSR    PC,DEVSEL  ;GO SELECT DEVICE
63 037414 000411                       BR     30$      ;NO ERROR - CONTINUE
64 037416 000401                       BR     20$
65 037420 000000                       .WORD  0         ;ERROR NUMBER FROM DEVSEL
66 037422 062716 000004 000000 20$:   ADD    #4,(SP)   ;TRANSFER ERROR TO USER
67 037426 113776 037420 000000       MOVW   20$,2,a(SP)
68 037434 000137 040226               JMP     400$
69
70                                     ;*****
71                                     ;CLEAR CONTROLLER IF BIT 14 IS SET IN TASK
72 037440 032737 040000 040236 30$:   BIT     #BIT14,500$  ;CLEAR CONTROLLER??
73 037440 001451                       BEQ    120$      ;NO!!
74
75                                     ;*****
76 037450 004737 052434               JSR    PC,CNTCLR  ;GO CLEAR CONTROLLER
77 037454 000411                       BR     60$      ;CONTINUE - NO ERROR
78 037456 000401                       BR     50$
79 037460 000000 000000 40$:   .WORD  0         ;ERROR NUMBER FROM CNTCLR
80 037462 062716 000004 000000 50$:   ADD    #4,(SP)   ;TRANSFER ERROR TO USER
81 037466 113776 037460 000000       MOVW   40$,a(SP)
82 037474 000137 040226               JMP     400$
83
84                                     ;*****
85                                     ;VERIFY CONTROLLER CLEAR IF BIT6 SET IN TASK
86 037500 032737 000100 040236 60$:   BIT     #BIT6,500$  ;VERIFY??
87 037500 001431                       BEQ    120$      ;NO!!
88
89                                     ;*****
90 037510 004737 043060               JSR    PC,GET     ;GO GET STATUS
91 037514 000411                       BR     90$      ;NO ERROR GETTING STATUS
92 037516 000401                       BR     80$
93 037520 000000 000000 70$:   .WORD  0         ;ERROR FROM GETTING STATUS
94 037522 062716 000004 000000 80$:   ADD    #4,(SP)   ;TRANSFER ERROR TO USER
95 037526 113776 037520 000000       MOVW   70$,a(SP)
96 037534 000137 040226               JMP     400$
97
98 037540 004737 052552 90$:   JSR    PC,CLRSTS  ;GO VERIFY STATUS CLEAR
99 037544 000412                       BR     120$     ;NO ERROR IN CLEAR
100 037546 000401                       BR     110$
101 037550 000000 000000 100$:   .WORD  0         ;ERROR IN STATUS CLEAR
102 037552 005726 000004 000000 110$:   TST     (SP)+    ;STRIP RETURN ADDRESS TO
103 037554 062716 000004 000000       ADD    #4,(SP)   ;SUBROUTINE AND TRANSFER
104 037560 113776 037550 000000       MOVW   100$,a(SP) ;ERROR TO USER
105 037566 000137 040226               JMP     400$
106
107                                     ;*****
108                                     ;EXECUTE PACK ACKNOWLEDGE IF BIT12 SET IN TASK AND VOLUME IS
109                                     ;NOT VALID
110 037572 032737 010000 040236 120$:   BIT     #BIT12,500$  ;PACK ACKNOWLEDGE??
111 037572 001503                       BEQ    240$      ;NO!!
112
113                                     ;*****
114 037602 004737 043060               JSR    PC,GET

```

```

115 037606 000411 BR 150$ ;NO ERROR GETTING RMDS
116 037610 000401 BR 140$
117 037612 000000 130$: .WORD 0
118 037614 062716 000004 140$: ADD #4,(SP) ;TRANSFER ERROR TO USER
119 037620 113776 037612 000000 MOVB 130$,a(SP)
120 037626 000137 040226 JMP 400$
121
122 037632 032737 000100 001350 150$: BIT #VV,RMDSI ;IS VOLUME VALID??
123 037640 001063 BNE 240$ ;YES!
124
125 037642 005037 001512 CLR MEDENB ;CLEAR MEDIA ENABLE
126 037646 012737 000023 001412 MOV #PAKACK!GO,RMCS10 ;LOAD PAK ACK COMMAND
127 037654 112737 000000 001553 MOVB #RMCS1,PUTINX ;SETUP REGISTER INDEX TABLE
128 037662 112737 000200 001554 MOVB #200,PUTINX+1
129 037670 004737 043330 JSR PC,PUT ;GO WRITE COMMAND
130 037674 000410 BR 180$ ;NO ERROR LOADING REGISTER
131 037676 000401 BR 170$
132 037700 000000 160$: .WORD 0 ;ERROR FROM PUT SUB
133 037702 062716 000004 170$: ADD #4,(SP) ;TRANSFER ERROR TO USER
134 037706 113776 037700 000000 MOVB 160$,a(SP)
135 037714 000544 BR 400$
136
137 037716 004737 043672 180$: JSR PC,TIMOUT ;WAIT FOR COMMAND TO COMPLETE
138
139 ;*****
140 ;VERIFY PACK ACKNOWLEDGE IF #BIT4 SET IN TASK
141 037722 032737 000020 040236 BIT #BIT4,500$ ;VERIFY PACK ACKNOWLEDGE??
142 037730 001427 BEQ 240$ ;NO!
143
144 037732 004737 043060 JSR PC,GET ;GO GET STATUS
145 037736 000410 BR 210$ ;NO ERROR GETTING STATUS
146 037740 000401 BR 200$
147 037742 000000 190$: .WORD 0 ;ERROR FROM GET SUB
148 037744 062716 000004 200$: ADD #4,(SP) ;TRANSFER ERROR TO USER
149 037750 113776 037742 000000 MOVB 190$,a(SP)
150 037756 000523 BR 400$
151
152 037760 004737 053432 210$: JSR PC,ACKSTS ;GO CHECK ACKNOWLEDGE
153 037764 000411 BR 240$ ;NO ERROR
154 037766 000401 BR 230$
155 037770 000000 220$: .WORD 0 ;PACK ACKNOWLEDGE ERROR
156 037772 005726 230$: TST (SP)+ ;STRIP RETURN TO SUB AND
157 037774 062716 000004 ADD #4,(SP) ;TRANSFER ERROR TO USER
158 040000 113776 037770 000000 MOVB 220$,a(SP)
159 040006 000507 BR 400$
160
161 ;*****
162 ;RECALIBRATE DRIVE IF BIT 11 IS SET IN TASK AND 'SKI' IS SET
163 ;OR 'PIP' IS ACTIVE.
164 040010 240$: BIT #BIT11,500$ ;RECALIBRATE??
165 040010 032737 004000 040236 BEQ 410$ ;NO!
166 040016 001505
167
168 040020 004737 043060 JSR PC,GET ;GO GET RMDS
169 040024 000410 BR 270$ ;NO ERROR GETTING RMDS
170 040026 000401 BR 260$
171 040030 000000 250$: .WORD 0 ;ERROR FROM GET SUB

```

```

172 040032 062716 000004 260$: ADD #4,(SP) ;TRANSFER ERROR TO USER
173 040036 113776 040030 000000 MOVB 250$,a(SP)
174 040044 000470 BR 400$
175
176 040046 032737 040000 040240 270$: BIT #SKI,505$ ;WAS SKI SET ?
177 040054 001004 BNE 280$ ;YES, GO RECALIBRATE
178 040056 032737 020000 001350 BIT #PIP,RMDSI ;IS PIP ACTIVE??
179 040064 001462 BEQ 410$ ;NO!!
180
181 040066 012737 000007 001412 280$: MOV #RLCAL!GO,RMCS10 ;LOAD RECALIBRATE COMMAND
182 040074 112737 000000 001553 MOVB #RMCS1,PUTINX ;AND REGISTER INDEX
183 040102 112737 000200 001554 MOVB #200,PUTINX+1
184 040110 004737 043330 JSR PC,PUT ;GO ISSUE RECALIBRATE
185 040114 000410 BR 300$ ;NO ERROR
186 040116 000401 BR 290$
187 040120 000000 .WORD 0 ;ERROR IN REGISTER TRANSFER
188 040122 062716 000004 290$: ADD #4,(SP) ;TRANSFER ERROR TO USER
189 040126 113776 040120 000000 MOVB 290$-2,a(SP)
190 040134 000434 BR 400$
191
192 040136 004737 043672 300$: JSR PC,TIMOUT ;WAIT FOR COMPLETION
193
194 ;*****
195 ;VERIFY RECALIBRATE IF BIT 3 SET IN TASK
196 040142 032737 000010 040236 BIT #BIT3,500$ ;VERIFY RECALIBRATE??
197 040150 001430 BEQ 410$ ;NO!!
198
199 040152 004737 043060 JSR PC,GET ;GO GET STATUS
200 040156 000410 BR 330$ ;NO ERROR GETTING STATUS
201 040160 000401 BR 320$
202 040162 000000 310$: .WORD 0 ;ERROR FROM GET
203 040164 062716 000004 320$: ADD #4,(SP) ;TRANSFER ERROR TO USER
204 040170 113776 040162 000000 MOVB 310$,a(SP)
205 040176 000413 BR 400$
206
207 040200 004737 054226 330$: JSR PC,RLSTS ;GO CHECK RECALIBRATE
208 040204 000412 BR 410$ ;NO ERROR DURING RECALIBRATE
209 040206 000401 BR 350$
210 040210 000000 340$: .WORD 0 ;ERROR DURING RECALIBRATE
211 040212 005726 350$: TST (SP)+ ;STRIP RETURN TO SUB AND
212 040214 062716 000004 ADD #4,(SP) ;TRANSFER ERROR TO USER
213 040220 113776 040210 000000 MOVB 340$,a(SP)
214 040226 162716 000002 400$: SUB #2,(SP) ;MOVE SP BACK BEFORE ERROR
215 040232 000240 410$: NOP
216 040234 000207 RTS PC ;RETURN TO USER
217
218 040236 000000 500$: .WORD 0 ;TASK/VERIFY DESCRIPTOR
219 040240 000000 505$: .WORD 0 ;CONTAINS RMER2

```

```
1 .SBTTL BAD SECTOR MODULE
2
3 ;THE MODULE IS INTENDED TO BE CALLED PRIOR TO CALLING THE BUFFER
4 ;GENERATOR SUBROUTINE, AND PRESERVES THE 'PUT BUFFER' SO THAT THE
5 ;BUFFER NEED ONLY BE FILLED ONCE FOR THE EXECUTION OF A FORMAT
6 ;OPERATION.
7
8 ;THE MODULE RETURNS TO THE CALLING TEST WITH THE APPROVED OR ASSIGNED
9 ;SECTOR IN THE PUT BUFFER AND ALSO IN LOCATIONS 'ASNDA' AND 'ASNDC'
10 ;SO THAT A REFERENCE IS AVAILABLE TO THE TEST OUTSIDE OF THE PUT BUFFER.
11
12 ;THE BAD SECTOR MODULE PERFORMS TWO MAJOR FUNCTIONS:
13 ; (1) RECOVER THE BAD SECTOR FILES AND
14 ; (2) APPROVE THE USAGE OF A SECTOR BASED ON INFORMATION IN
15 ; THE BAD SECTOR FILES OR ASSIGN A NEW SECTOR IF THE ONE
16 ; ELECTED IS NOT AVAILABLE FOR USE.
17
18 ;INFORMATION REQUIRED BY THE MODULE INCLUDES:
19 ; (1) .RMDCO - THE DESIRED CYLINDER,
20 ; (2) .RMDAO - THE TRACK AND SECTOR ADDRESS,
21 ; (3) .RMWCO - THE WORD COUNT,
22 ; (4) .RMCS10 - THE COMMAND,
23 ; (5) .RMOFO - THE FORMAT MODE
24
25 ;CALL:
26 JSR PC,BADSCT ;CALL SUBROUTINE
27 BR ??? ;RETURN HERE IF NO ERROR
28 TYPE ,MESSAGE ;RETURN HERE IF THE BAD SECTOR FILE
29 ;CANNOT BE RECOVERED.
30 ERROR N ;THE EMT OFFSET NUMBER 'N' IS DEFINED
31 ;BY BAD SECTOR MODULE.
32
33 BADSCT:
34 ADD #6,(SP) ;CLEAR ERROR NUMBER IN USER'S
35 CLRB @ (SP) ;ERROR CALL.
36 SUB #6,(SP)
37
38 ;TEST 'MEDIA ENABLE' TO DETERMINE WHETHER OR NOT THE BAD SECTOR FILES
39 ;HAVE BEEN RECOVERED.
40 TST MEDENB ;HAS BAD SECTOR FILES BEEN RECOVERED ?
41 BEQ 1$ ;BR IF NO
42 JMP 54$ ;YES, BAD SECTOR FILE IS AVAILABLE
43
44 ;RECOVER THE MANUFACTURES / USERS BAD SECTOR FILE FROM CYLINDER = 822.,
45 ;TRACK LAST TRACK (RM02/3 = 4 AND RM05 = 18.). ALSO, SAVE THE USER'S
46 ;PUT BUFFER
47 1$:
48 MOV R0,-(SP) ;PUSH R0 ON STACK
49 CLR R0 ;START WITH RMCS1
50 MOV PUTBUF(R0),BUFFER(R0)
51 ADD #2,R0 ;ADVANCE TO NEXT BUFFER POSITION
52 CMP #46,R0 ;END OF BUFFER
53 BHS 2$ ;NO !!
54
55 ;SET RETRY COUNT AND LOAD PUT BUFFER AND REGISTER INDEX TABLE
56 ;SETUP PARAMETERS TO READ SKIP SECTOR FILE FIRST (3 SECTORS)
57 MOV #3,68$ ;RETRY COUNT
```

040242	062716	000006	
040246	105076	000000	
040252	162716	000006	
040256	005737	001512	
040262	001402		
040264	000137	041542	
040270	010046		
040272	005000		
040274	016060	001412	106436
040302	062700	000002	
040306	022700	000046	
040312	103370		
040314	012737	000003	042160

```

57 040322 012737 001466 001446      MOV      #822.,RMDCO      ;DESIRED CYLINDER = 822.
58 040330 013737 001334 001420      MOV      LSTRK,RMDAO    ;STARTING LAST TRACK, SECTOR = 0
59 040336 012737 177376 001414      MOV      #-258.,RMWCO    ;2 * 256. WORDS (2'S COMP)
60 040344 012737 010000 001444      MOV      #FMT16,RMOFO    ;16 BIT FORMAT
61 040352 012737 110446 001416      MOV      #MFGFIL,RMBAO   ;POINT TO MANUFACTURES FILE BUFFER
62
63 040360 012700 001553              MOV      #PUTINX,R0      ;R0 POINTS TO REGISTER INDEX TABLE
64 040364 112720 000006              MOV      #RMDA,(R0)+
65 040370 112720 000034              MOV      #RMDC,(R0)+
66 040374 112720 000002              MOV      #RMWC,(R0)+
67 040400 112720 000032              MOV      #RMOF,(R0)+
68 040404 112720 000004              MOV      #RMBA,(R0)+
69 040410 112720 000000              MOV      #RMCS1,(R0)+
70 040414 112720 000200              MOV      #200,(R0)+
71 040420 012600                    MOV      (SP)+,R0      ;;POP STACK INTO R0
72
73                                ;SET GET INDEX TABLE FOR READING STATUS
74 040422 3$:
75 040422 004737 042774              JSR      PC,GETSTS      ;SETUP GET INDEX REGISTER FOR STATUS
76 040426 004737 043060              JSR      PC,GET      ;GET REGISTERS
77 040432 000411                    BR      5$          ;BR IF NO ERROR
78 040434 000401                    BR      4$          ;JUMP OVER ERROR NUMBER
79 040436 000000                    .WORD    0          ;ERROR DEFINED BY GET SUB
80 040440 062716 000006              4$:      ADD      #6,(SP)      ;XFER ERROR TO USER AND
81 040444 113776 040436 000000      MOV      4$-2,a(SP)    ;GET ERROR NUMBER.
82 040452 000137 041246              JMP      42$
83
84 040456 013737 001400 042172      5$:      MOV      RMER2I,73$      ;GET RMER2 AND SAVE FOR LATER
85
86                                ;CLEAR THE DEVICE USING DRIVE CLEAR COMMAND
87 040464 012737 000011 001412      MOV      #DRVCLR!GO,RMCS10 ;LOAD COMMAND IN PUT BUFFER
88 040472 004737 043330              JSR      PC,PUT      ;OUTPUT COMMAND
89 040476 000411                    BR      8$          ;RETURN HERE IF NO ERROR
90 040500 000401                    BR      7$          ;GET AROUND ERROR #
91 040502 000000                    .WORD    0          ;ERROR # GOES HERE
92 040504 062716 000006              6$:      ADD      #6,(SP)      ;MOVE SP TO USERS ERROR CALL
93 040510 113776 040502 000000      7$:      MOV      6$,a(SP)    ;MOVE ERROR NUMBER TO USER
94 040516 000137 041246              JMP      42$
95
96 040522 004737 043672              8$:      JSR      PC,TIMOUT    ;WAIT FOR COMPLETION
97 040526 004737 043060              JSR      PC,GET      ;GO GET STATUS
98 040532 000411                    BR      11$         ;RETURN HERE IF NO ERROR
99 040534 000401                    BR      10$         ;GET AROUND ERROR #
100 040536 000000                    .WORD    0          ;ERROR # GOES HERE
101 040540 062716 000006              9$:      ADD      #6,(SP)      ;MOVE SP TO USERS ERROR CALL
102 040544 113776 040536 000000      10$:     MOV      9$,a(SP)    ;MOVE ERROR # TO USERS ERROR CALL
103 040552 000137 041246              JMP      42$
104
105 040556 004737 055770              11$:     JSR      PC,DRVSTS    ;GO VERIFY DRIVE CLEAR COMMAND
106 040562 000412                    BR      14$         ;RETURN HERE IF NO ERROR
107 040564 000401                    BR      13$         ;GET AROUND ERROR
108 040566 000000                    .WORD    0          ;ERROR # GOES HERE
109 040570 005726                    12$:     TST      (SP)+      ;STRIP RETURN TO SUBROUTINE
110 040572 062716 000006              13$:     ADD      #6,(SP)      ;MOVE SP TO USERS ERROR CALL
111 040576 113776 040566 000000      MOV      12$,a(SP)    ;MOVE ERROR # TO USER CALL
112 040604 000137 041246              JMP      42$
113

```



```

114 ;ISSUE A PACK ACKNOWLEDGE IF VOLUME VALID IS RESET
115 040610 14$:
116 040610 032737 000100 001350 BIT #VV,RMDSI ;IS VV RESET ??
117 040616 001052 BNE 23$ ;NO ..
118
119 040620 012737 000023 001412 MOV #PACACK.GO,RMCS10 ;LOAD COMMAND
120 040626 004737 043330 JSR PC,PUT ;GO PUT COMMAND TO DRIVE
121 040632 000411 BR 17$ ;RETURN HERE IF NO OUTPUT ERROR
122 040634 000401 BR 16$ ;GET AROUND ERROR #
123 040636 000000 15$: .WORD 0 ;ERROR # GOES HERE
124 040640 062716 000006 16$: ADD #6,(SP) ;MOVE SP TO USERS ERROR CALL
125 040644 113776 040636 000000 MOV 15$,a(SP) ;MOVE ERROR # TO ERROR CALL
126 040652 000137 041246 JMP 42$
127
128 040656 004737 043672 17$: JSR PC,TIMOUT ;WAIT FOR COMPLETION
129 040662 004737 043060 JSR PC,GET ;GO GET STATUS FOR PACK ACK
130 040666 000411 BR 20$ ;RETURN HERE IF NO ERROR
131 040670 000401 BR 19$ ;GET AROUND ERROR #
132 040672 000000 18$: .WORD 0 ;ERROR # GOES HERE
133 040674 062716 000006 19$: ADD #6,(SP) ;MOVE SP TO USERS ERROR CALL
134 040700 113776 040672 000000 MOV 18$,a(SP) ;MOVE ERROR # TO CALL
135 040706 000137 041246 JMP 42$
136
137 040712 004737 053432 20$: JSR PC,ACKSTS ;GO VERIFY ACKNOWLEDGE STATUS
138 040716 000412 BR 23$ ;RETURN HERE IF NO ERROR
139 040720 000401 BR 22$ ;GET AROUND ERROR #
140 040722 000000 21$: .WORD 0 ;ERROR # GOES HERE
141 040724 005726 22$: TST (SP)+ ;STRIP RETURN TO SUBROUTINE
142 040726 062716 000006 ADD #6,(SP) ;MOVE SP TO USERS ERROR CALL
143 040732 113776 040722 000000 MOV 21$,a(SP) ;MOVE ERROR # TO USERS ERROR CALL
144 040740 000137 041246 JMP 42$
145
146 ;RECALIBRATE THE DRIVE IF 'SKI' OR 'PIP' IS SET
147 040744 23$:
148 040744 032737 040000 042172 BIT #SKI,73$ ;WAS SKI SET ?
149 040752 001004 BNE 24$ ;YES, GO RECALIBRATE
150 040754 032737 020000 001350 BIT #PIP,RMDSI ;IS PIP SET ??
151 040762 001452 BEQ 32$ ;NO !!
152
153 040764 012737 000007 001412 24$: MOV #RECAL!GO,RMCS10 ;LOAD RECALIBRATE COMMAND
154 040772 004737 043330 JSR PC,PUT ;PUT THE RECAL COMMAND
155 040776 000411 BR 26$ ;RETURN HERE IF NO ERROR
156 041000 000401 BR 25$ ;GET AROUND ERROR #
157 041002 000000 .WORD 0 ;ERROR # GOES HERE
158 041004 062716 000006 25$: ADD #6,(SP) ;MOVE SP TO USERS ERROR CALL
159 041010 113776 041002 000000 MOV 25$-2,a(SP) ;MOVE ERROR # TO USERS CALL
160 041016 000137 041246 JMP 42$
161
162 041022 004737 043672 26$: JSR PC,TIMOUT ;WAIT FOR RECALIBRATE TO COMPLETE
163 041026 004737 043060 JSR PC,GET ;GO GET RECAL STATUS
164 041032 000411 BR 29$ ;RETURN HERE IF NO ERROR
165 041034 000401 BR 28$ ;GET AROUND ERROR #
166 041036 000000 27$: .WORD 0 ;ERROR # GOES HERE
167 041040 062716 000006 28$: ADD #6,(SP) ;MOVE SP TO USERS ERROR CALL
168 041044 113776 041036 000000 MOV 27$,a(SP) ;MOVE ERROR TO USERS CALL
169 041052 000137 041246 JMP 42$
170

```

```

171 041056 004737 054226      29$: JSR    PC,RCLSTS      ;GO VERIFY RECALIBRATE STATUS
172 041062 000412              BR      32$      ;RETURN HERE IF NO ERROR
173 041064 000401              BR      31$      ;GET AROUND ERROR #
174 041066 000000      30$: .WORD    0          ;ERROR # GOES HERE
175 041070 005726      31$: TST      (SP)+      ;STRIP RETURN TO SUBROUTINE
176 041072 062716 000006      ADD      #6,(SP)    ;MOVE SP TO USERS ERROR CALL
177 041076 113776 041066 000000      MOVB    30$,a(SP) ;MOVE ERROR # TO USERS CALL
178 041104 000137 041246      JMP      42$
179
180      ;READ THE SECTOR IDENTIFIED BY RMDAO, INCLUDING HEADER AND DATA
181 041110      32$: MOV      #RH!GO,RMCS10    ;LOAD READ HEADER AND DATA COMMAND
182 041110 012737 000073 001412      JSR      PC,PUT      ;PUT COMMAND
183 041116 004737 043330      BR      35$      ;RETURN HERE IF NO ERROR
184 041122 000411              BR      34$      ;GET AROUND ERROR #
185 041124 000401              .WORD    0          ;ERROR # GOES HERE
186 041126 000000      33$: ADD      #6,(SP)    ;MOVE SP TO USERS ERROR CALL
187 041130 062716 000006      34$: MOVB    33$,a(SP) ;MOVE ERROR # TO USERS ERROR CALL
188 041134 113776 041126 000000      JMP      42$
189 041142 000137 041246
190
191 041146 004737 043672      35$: JSR      PC,TIMOUT    ;WAIT FOR READ OPERATION TO COMPLETE
192 041152 004737 043060      JSR      PC,GET      ;GO GET STATUS FOR READ OPERATION
193 041156 000411              BR      38$      ;RETURN HERE IF NO ERROR
194 041160 000401              BR      37$      ;GET AROUND ERROR #
195 041162 000000      36$: .WORD    0          ;ERROR # GOES HERE
196 041164 062716 000006      37$: ADD      #6,(SP)    ;MOVE SP TO USERS ERROR CALL
197 041170 113776 041162 000000      MOVB    36$,a(SP) ;MOVE ERROR # TO CALL
198 041176 000137 041246      JMP      42$
199
200 041202 004737 056572      38$: JSR      PC,DTASTS    ;GO VERIFY RESULTS OF READ OPERATION
201 041206 000412              BR      41$      ;RETURN HERE IF NO ERROR
202 041210 000401              BR      40$      ;GET AROUND ERROR #
203 041212 000000      39$: .WORD    0          ;ERROR # GOES HERE
204 041214 005726      40$: TST      (SP)+      ;STRIP RETURN ADDRESS TO SUBROUTINE
205 041216 062716 000006      ADD      #6,(SP)    ;MOVE SP TO USERS ERROR CALL
206 041222 113776 041212 000000      MOVB    39$,a(SP) ;MOVE ERROR # TO USERS CALL
207 041230 000137 041246      JMP      42$
208
209 041234 032737 040000 001336 41$: BIT      #TRE,RMCS11    ;ANY CONTROLLER ERRORS ?
210 041242 001001              BNE      42$      ;BR IF YES
211 041244 000446              BR      48$      ;NO ERRORS DETECTED
212
213      ;*****
214      ;AN ERROR HAS BEEN DETECTED IN TRYING TO READ THE BAD SECTOR FILE.
215      ;THE SECTOR WILL BE RETRIED IF POSSIBLE.
216
217 041246 005337 042160      42$: DEC      68$          ;YES, DECREMENT RETRY COUNT AND
218 041252 100026              BPL      45$          ;RETRY IF COUNT NOT NEGATIVE.
219
220      ;THE RETRY COUNT HAS EXPIRED - SEE IF THE ERROR IS MEDIA RELATED
221
222 041254 032737 100720 001352 43$: BIT      #DCK.HCRC!HCE.FER!ECH,RMER1I ;ANY MEDIA RELATED ERRORS ?
223 041262 001004              BNE      44$          ;YES, GO TRY NEXT AVAILABLE SECTOR
224
225 041264 032737 100000 001400      BIT      #BSE,RMER2I    ;ANY MEDIA RELATED ERRORS ?
226 041272 001422              BEQ      46$          ;NO, EXIT AND REPORT ERROR ON RETURN
227

```



```

228 ;THE ERRORS DETECTED WHILE TRYING TO RECOVER THE BAD SECTOR FILE ARE
229 ;DUE TO THE MEDIA. SEE IF THE BAD SECTOR FILE CAN BE RECOVERED FROM
230 ;ANOTHER AREA ON THE LAST TRACK
231
232 041274 062737 000002 001420 44$: ADD #2,RMDAO ;ADVANCE SECTOR ADDRESS BY 2
233 041302 122737 000012 001420 CMPB #10.,RMDAO ;QUIT IF ALL MFG SECTORS HAVE BEEN
234 041310 001413 BEQ 46$ ;TRIED.
235 041312 122737 000040 001420 CMPB #32.,RMDAO ;QUIT IF ALL USER SECTORS HAVE BEEN
236 041320 001407 BEQ 46$ ;TRIED.
237
238 041322 012737 000003 042160 MOV #3,68$ ;REINSTATE RETRY COUNT FOR THIS SECTOR
239 041330 162716 000006 45$: SUB #6,(SP) ;MOVE SP BACK TO NO ERROR
240 041334 000137 040422 JMP 3$ ;RETRY THE READ OPERATION
241
242 ;THE BAD SECTOR FILE CANNOT BE READ
243
244 041340 000240 46$: NOP
245 041342 032777 020000 137604 BIT #SW13,@SWR ;INHIBIT MESSAGE ?
246 041350 001002 BNE 47$ ;YES
247 041352 162716 000004 SUB #4,(SP) ;MOVE SP TO ERROR RETURN
248 041356 000137 042154 47$: JMP 67$ ;GO TO MODULE EXIT
249
250 ;THE SECTOR WAS RECOVERED WITHOUT ERROR - READ THE USER FILE IF
251 ;THIS IS THE MFG FILE OR ELSE DONE.
252
253 041362 022737 111454 001416 48$: CMP #USRFIL,RMBAD ;WAS THE USER FILE READ ??
254 041370 001446 BEQ 52$ ;YES - READ IS COMPLETE
255 041372 112737 000012 001420 MOVB #10.,RMDAO ;READ THE USER FILE LAST TRACK, SECTOR 10.
256 041400 012737 111454 001416 MOV #USRFIL,RMBAD ;POINT TO USERS FILE BUFFER
257
258 041406 012737 000003 042160 MOV #3,68$ ;RELOAD THE RETRY COUNT FOR THIS SECTOR
259 041414 000137 040422 JMP 3$ ;GO READ THE USER FILE
260
261 ;DUMMY THE BAD SECTOR FILES
262 041420 49$:
263 041420 010046 MOV R0,-(SP) ;:PUSH R0 ON STACK
264 041422 010146 MOV R1,-(SP) ;:PUSH R1 ON STACK
265 041424 012701 000374 MOV #252.,R1 ;R1 = NUMBER OF ENTRIES IN FILES
266 041430 012700 000014 MOV #14,R0 ;R0 = ADDRESS INDEX TO FILE STORAGE
267 041434 012760 177777 110446 50$: MOV #-1,MFGFIL(R0) ;ENTER ALL ONES IN MFG FILE
268 041442 012760 177777 111454 MOV #-1,USRFIL(R0) ;ENTER ALL ONES IN USER FILE
269 041450 005720 TST (R0)+ ;ADVANCE ADDRESS
270 041452 005301 DEC R1 ;DECREMENT COUNT
271 041454 001367 BNE 50$ ;CONTINUE IF NOT DONE
272
273 041456 012701 000006 MOV #6.,R1 ;CLEAR HEADER, CLEAR ID & SERIAL NUMBERS
274 041462 005000 CLR R0
275 041464 005060 110446 51$: CLR MFGFIL(R0)
276 041470 005060 111454 CLR USRFIL(R0)
277 041474 005720 TST (R0)+ ;ADVANCE ADDRESS
278 041476 005301 DEC R1
279 041500 001371 BNE 51$
280 041502 012601 MOV (SP)+,R1 ;:POP STACK INTO R1
281 041504 012600 MOV (SP)+,R0 ;:POP STACK INTO R0
282
283 ;SET MEDIA ENABLE AND RESTORE THE USERS PUT BUFFER
284 52$
  
```

```

282 041506 010046      MOV      R0,-(SP)          ;;PUSH R0 ON STACK
283 041510 005000      CLR      R0              ;;R0 IS REGISTER INDEX
284 041512 012737 177777 001512      MOV      #-1,MEDENB
285 041520 016060 106436 001412 53$:  MOV      BUFFER(R0),PUTBUF(R0)
286 041526 062700 000002      ADD      #2,R0          ;;ADVANCE R0
287 041532 022700 000046      CMP      #46,R0         ;;DONE ??
288 041536 103370      BNEIS    53$
289 041540 012600      MOV      (SP)+,R0          ;;POP STACK INTO R0
290
291      ;;*****
292      ;;VERIFY THAT THE DESIRED SECTOR IS NOT IN THE MFG BAD SECTOR FILE
293      ;;AND NOT IN THE USERS BAD SECTOR FILE. ASSIGN A NEW SECTOR IF THE
294      ;;SECTOR IS IN ANY OF THE FILES.
295      ;;*****
296      ;;LOAD INITIAL VARIABLES AND COMPUTE THE NUMBER OF SECTORS
297 041542 54$:
041542 010046      MOV      R0,-(SP)          ;;PUSH R0 ON STACK
041544 010146      MOV      R1,-(SP)          ;;PUSH R1 ON STACK
041546 010246      MOV      R2,-(SP)          ;;PUSH R2 ON STACK
298 041550 013737 001446 001514      MOV      RMDCO,ASNDC      ;;LOAD REQUESTED CYLINDER, TRACK,
299 041556 013737 001420 001516      MOV      RMDAO,ASNDA      ;;AND SECTOR ADDRESS IN ASSIGNED STORAGE
300 041564 005002      CLR      R2              ;;R2 = NUMBER OF SECTORS
301 041566 013700 001414      MOV      RMCWO,R0          ;;R0 = WORD COUNT
302 041572 005400      NEG      R0              ;;MAKE NUMBER POSITIVE
303 041574 012701 000400      MOV      #256.,R1         ;;R1 = NUMBER OF WORDS PER SECTOR
304 041600 032737 000002 001412      BIT      #BIT1,RMCS10      ;;IS THIS A HEADER AND DATA COMMAND ??
305 041606 001402      BEQ      55$            ;;NO !!
306 041610 012701 000402      MOV      #258.,R1         ;;CHANGE WORDS PER SECTOR
307 041614 020100 55$:      CMP      R1,R0          ;;IS THERE A FULL SECTOR ??
308 041616 101404      BLOS    56$            ;;YES !!
309 041620 005700      TST      R0              ;;IS R0 ZERO ??
310 041622 001405      BEQ      57$            ;;YES !!
311 041624 005202      INC      R2              ;;INCREMENT FOR PARTIAL SECTOR
312 041626 000403      BR       57$
313 041630 160100 56$:      SUB      R1,R0          ;;SUBTRACT ONE SECTOR FROM WORD COUNT
314 041632 005202      INC      R2              ;;INCREMENT SECTOR COUNT
315 041634 000767      BR       55$
316 041636 010237 042160 57$:      MOV      R2,68$          ;;SAVE SECTOR COUNT
317
318      ;;LOAD PARAMETERS AND SEARCH THE MFG/USER SECTOR FILE FOR THE
319      ;;ASSIGNED SECTOR. ALSO, SEARCH THE ADJACENT SECTORS IF THE
320      ;;SECTOR COUNT IS MORE THAN ONE.
321
322 041642 012737 110462 042170      MOV      #MFGFIL+14,72$      ;;THE STARTING ADDRESS OF MFG FILE
323      ;;TO BASE ADDRESS STORAGE.
324 041650 004737 041672      JSR      PC,58$          ;;GO SEARCH FILE
325 041654 012737 111470 042170      MOV      #USRFIL+14,72$      ;;LOAD STARTING ADDRESS OF USR FILE
326      ;;TO BASE ADDRESS STORAGE.
327 041662 004737 041672      JSR      PC,58$          ;;GO SEARCH FILE
328 041666 000137 042132      JMP      66$            ;;DONE WITH ALL FILE SEARCHES .!
329
330 041672 013737 001514 042164 58$:  MOV      ASNDC,70$          ;;LOAD COMPARING CYLINDER ADDRESS
331 041700 013737 001516 042166      MOV      ASNDA,71$          ;;LOAD COMPARING TRACK, SECTOR ADDRESS
332 041706 013737 042160 042162      MOV      68$,69$      ;;LOAD NUMBER OF SECTORS TO CONFIRM
333
334      ;;SETUP FOR A BINARY SEARCH OF THE CURRENT FILE FOR THE COMPARING

```

```

335 ;CYLINDER, TRACK AND SECTOR ADDRESS
336 041714 59$:
337 041714 013700 042170 MOV 72$,RO ;LOAD THE BASE ADDRESS IN RO
338 041720 022710 177777 60$: CMP #-1,(RO) ;IS THIS FILE TERMINATOR ?
339 041724 001446 BEQ 64$ ;BR IF YES
340 041726 021037 042164 CMP (RO),70$ ;DOES TABLE ENTRY = COMPARING CYLINDER ?
341 041732 001010 BNE 62$ ;BR IF NO
342
343 ;FILE ENTRY EQUALS COMPARING CYLINDER. SEE IF THE NEXT ENTRY EQUALS
344 ;THE COMPARING TRACK, AND SECTOR.
345 041734 61$:
346 041734 126037 000003 042167 CMPB 3(RO),71$+1 ;DOES TABLE ENTRY = COMPARING TRACK ?
347 041742 001004 BNE 62$ ;BR IF NO
348
349 041744 126037 000002 042166 CMPB 2(RO),71$ ;DOES TABLE ENTRY - COMPARING SECTOR ?
350 041752 001402 BEQ 63$ ;BR IF YES
351
352 041754 022020 62$: CMP (RO)+,(RO)+ ;NO, ADJUST CYLINDER POINTER IN BAD FILE
353 041756 000760 BR 60$ ;AND CONTINUE SEARCH.
354
355 ;THE COMPARING CYLINDER, TRACK AND SECTOR IS IN THE BAD SECTOR FILE.
356 ;ADVANCE THE ASSIGNED SECTOR AND START THE SEARCH ALL OVER.
357 041760 63$:
358 041760 105237 001516 INCB ASNDA ;INCREMENT SECTOR
359 041764 122737 000037 001516 CMPB #31.,ASNDA ;SECTOR OK ??
360 041772 103337 BHIS 58$ ;YES !!
361 041774 105037 001516 CLRB ASNDA ;CLEAR SECTOR AND ADVANCE TRACK
362 042000 105237 001517 INCB ASNDA+1
363 042004 123737 001335 001517 CMPB LSTRK+1,ASNDA+1 ;TRACK OK ?
364 042012 103327 BHIS 58$ ;YES !!
365 042014 005037 001516 CLR ASNDA ;CLEAR TRACK AND SECTOR
366 042020 005237 001514 INC ASNDC ;INCREMENT CYLINDER
367 042024 022737 001466 001514 CMP #822.,ASNDC ;CYLINDER OK ??
368 042032 103317 BHIS 58$ ;YES !!
369 042034 005037 001514 CLR ASNDC ;START AT CYLINDER 0
370 042040 000714 BR 58$ ;SEARCH NEXT SECTOR
371
372 ;THE COMPARING SECTOR IS NOT IN THE BAD SECTOR FILES. DECREMENT THE
373 ;NUMBER OF SECTORS TO COMPARE AND SEARCH THE NEXT SECTOR IF THE NUMBER
374 ;IS NOT ZERO.
375 042042 64$:
376 042042 005337 042162 DEC 69$ ;DECREMENT NUMBER OF SECTORS TO COMPARE
377 042046 001442 BEQ 67$ ;DONE IF ZERO
378
379 042050 105237 042166 INCB 71$ ;INCREMENT THE COMPARING SECTOR
380 042054 122737 000037 042166 CMPB #31.,71$ ;SECTOR OK ??
381 042062 103022 BHIS 65$ ;YES !!
382 042064 105037 042166 CLRB 71$ ;CLEAR SECTOR
383 042070 105237 042167 INCB 71$+1 ;INCREMENT TRACK
384 042074 123737 001335 042167 CMPB LSTRK+1,71$+1 ;TRACK OK ??
385 042102 103012 BHIS 65$ ;YES !!
386 042104 005037 042166 CLR 71$ ;CLEAR SECTOR TRACK
387 042110 005237 042164 INC 70$ ;INCREMENT CYLINDER
388 042114 022737 001466 042164 CMP #822.,70$ ;CYLINDER OK ??
389 042122 103002 BHIS 65$ ;YES !!
390 042124 005037 042164 CLR 70$ ;START AT CYLINDER 0
391

```

```

392 042130 000671      65$:  BR      59$      ;SEARCH NEXT SECTOR
393
394      ;ASSIGN THE SECTOR AND RETURN TO USER
395 042132      66$:
396 042132 013737 001514 001446      MOV      ASNDC,RMDCO      ;LOAD CYLINDER
397 042140 013737 001516 001420      MOV      ASNDA,RMDAO      ;LOAD TRACK AND SECTOR
398 042146 012602      MOV      (SP)+,R2      ;;POP STACK INTO R2
      042150 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
      042152 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
399
400 042154 000240      67$:  NOP
401 042156 000207      RTS      PC
402
403      ;THE FOLLOWING ARE STORAGE LOCATIONS FOR THE MODULE
404
405 042160 000000      68$:  .WORD 0      ;RETRY COUNT/ NUMBER OF SECTORS REQUIRED
406 042162 000000      69$:  .WORD 0      ;NUMBER OF SECTORS TO COMPARE
407 042164 000000      70$:  .WORD 0      ;COMPARING CYLINDER
408 042166 000000      71$:  .WORD 0      ;COMPARING TRACK AND SECTOR
409 042170 000000      72$:  .WORD 0      ;BASE ADDRESS OF BAD SECTOR FILE BEING SEARCHED
410 042172 000000      73$:  .WORD 0      ;CONTAINS RMER2
  
```

```

1      .SBTTL  BUFFER GENERATOR SUBROUTINE
2
3      ;THIS SUBROUTINE GENERATES A DATA BUFFER FOR WRITE COMMANDS.  THE
4      ;BUFFER STARTS AT RMBA AND IS RMWC WORDS LONG.  THE BUFFER
5      ;CONTAINS A REPETITIVE DATA PATTERN CONSISTING OF $TMP1 WORDS
6      ;FROM THE DATA PATTERN TABLE, BEGINNING AT ADDRESS $TMP0.
7      ;HEADER INFORMATION FOR THE BUFFER IS EXTRACTED FROM RMDC,
8      ;RMDA AND RMOF.
9
10     ;R0 = ADDRESS OF DATA BUFFER
11     ;R1 = LENGTH OF DATA BUFFER
12     ;R2 = ADDRESS OF DATA PATTERN
13     ;R3 = LENGTH OF DATA PATTERN
14     ;R4 = SECTOR COUNT
15
16     ;CALL:
17     ;(1)   JSR      PC,GENBUF
18     ;(2)   ??
19
20     GENBUF:
21     042174 010046      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
22     042176 010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
23     042200 010246      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
24     042202 010346      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
25     042204 010446      MOV      R4,-(SP)      ;;PUSH R4 ON STACK
26     042206 013700 001416  MOV      RMBAO,R0      ;LOAD DATA BUFFER ADDRESS
27     042212 013701 001414  MOV      RMWCO,R1      ;LOAD WORD COUNT
28     042216 013737 001446 042426  MOV      RMDCO,60$      ;LOAD STARTING CYLINDER ADDRESS
29     042224 013737 001420 042430  MOV      RMDAO,65$      ;LOAD STARTING TRACK,SECTOR ADDRESS
30
31     042232 032737 000002 001412 10$:  BIT      #BIT1,RMCS10      ;WRITE HEADER & DATA??
32     042240 001445          BEQ      25$          ;NO!!
33     042242 013710 042426      MOV      60$,(R0)      ;WRITE HEADER WORD #1
34     042246 052710 140000      BIS      #MSE!USE,(R0)  ;SET BAD SECTOR FLAGS FOR GOOD SECTOR
35     042252 012702 000035      MOV      #29.,R2      ;R2 = MAXIMUM SECTOR ADDRESS (29.)
36
37     042256 032737 010000 001444      BIT      #FMT16,RMOFO      ;18 BIT FORMAT??
38     042264 001404          BEQ      15$          ;YES !!
39     042266 052710 010000      BIS      #FMT16,(R0)      ;SET 16 FORMAT BIT IN HEADER
40     042272 012702 000037      MOV      #31.,R2      ;CHANGE MAXIMUM SECTOR ADDRESS (31.)
41
42     042276 005201          15$:  INC      R1          ;INCREMENT WORD COUNT
43     042300 001443          BEQ      50$          ;EXIT IF DONE
44
45     042302 005720          TST      (R0)+          ;MOVE R0 TO HEADER WORD #2
46     042304 013720 042430      MOV      65$,(R0)+      ;WRITE HEADER WORD #2
47     042310 005201          INC      R1          ;INCREMENT WORD COUNT AND
48     042312 001436          BEQ      50$          ;EXIT IF DONE
49     042314 012703 042430      MOV      #65$,R3      ;ADVANCE SECTOR ADDRESS
50     042320 105213          INCB     (R3)
51     042322 120213          CMPB     R2,(R3)          ;SECTOR OVERFLOW ??
52     042324 103013          BHIS     25$          ;NO !!
53     042326 105013          CLRB     (R3)          ;YES - CLEAR SECTOR ADDRESS
54     042330 105263 000001      INCB     1(R3)          ;ADVANCE TRACK ADDRESS
55     042334 123763 001335 000001      CMPB     LSTRK+1,1(R3)      ;TRACK OVERFLOW ??
56     042342 103004          BHIS     25$          ;NO !!
57     042344 105063 000001      CLRB     1(R3)          ;YES - CLEAR TRACK ADDRESS

```

BUFFER GENERATOR SUBROUTINE

53	042350	105237	042426		INCB	60\$		:ADVANCE CYLINDER ADDRESS
54	042354	012704	000400	25\$:	MOV	#256,R4		:LOAD SECTOR DATA COUNT
55	042360	013702	001174	30\$:	MOV	\$TMP0,R2		:LOAD PATTERN ADDRESS
56	042364	013703	001176		MOV	\$TMP1,R3		:LOAD PATTERN COUNT
57	042370	012220		40\$:	MOV	(R2)+,(R0)+		:WRITE DATA PATTERN
58	042372	005201			INC	R1		:INCREMENT WORD COUNT AND
59	042374	001405			BEQ	50\$		:EXIT IF DONE
60	042376	005304			DEC	R4		:DECREMENT SECTOR COUNT
61	042400	001714			BEQ	10\$		:START NEXT SECTOR IF 0
62	042402	005303			DEC	R3		:DECREMENT PATTERN COUNT
63	042404	001765			BEQ	30\$		:RESTART PATTERN IF 0
64	042406	000770			BR	40\$		:CONTINUE DATA PATTERN
65	042410			50\$:				
	042410	012604			MOV	(SP)+,R4		::POP STACK INTO R4
	042412	012603			MOV	(SP)+,R3		::POP STACK INTO R3
	042414	012602			MOV	(SP)+,R2		::POP STACK INTO R2
	042416	012601			MOV	(SP)+,R1		::POP STACK INTO R1
	042420	012600			MOV	(SP)+,R0		::POP STACK INTO R0
66	042422	000240			NOP			
67	042424	000207			RTS	PC		
68								
69	042426	000000		60\$:	.WORD			:CYLINDER ADDRESS STORAGE
70	042430	000000		65\$:	.WORD			:TRACK, SECTOR ADDRESS STORAGE


```

1      .SBTTL COMPARE BUFFER SUBROUTINE
2
3      ;THIS SUBROUTINE COMPARES THE CONTENTS OF BUFONE AND BUFTWO,
4      ;ASSUMING THAT BUFONE IS THE BUFFER FROM WHICH DATA WAS WRITTEN
5      ;AND BUFTWO IS THE BUFFER TO WHICH DATA WAS READ. ERRORS IN BUFFER
6      ;COMPARISON ARE REPORTED TO THE USER VIA THE USER'S ERROR CALL.
7
8      ;CALL:
9      ;(1) JSR PC,CMPBUF
10     ;(2) .WORD WRITE BUFFER ADDRESS
11     ;(3) .WORD READ BUFFER ADDRESS
12     ;(4) BR ?? RETURN HERE IF NO ERROR
13     ;(5) NOP RETURN HERE IF ERROR
14     ;(6) ERROR ERROR DEFINED BY SUBROUTINE
15     ;(7) ???
16
17     CMPBUF:
18     MOV R0,-(SP) ;;PUSH R0 ON STACK
19     MOV R1,-(SP) ;;PUSH R1 ON STACK
20     MOV R2,-(SP) ;;PUSH R2 ON STACK
21     MOV R3,-(SP) ;;PUSH R3 ON STACK
22     CLR 150$ ;;CLEAR CORRECTION FLAG
23
24     ;DETERMINE IF DATA SHOULD BE CORRECTED
25     BIT EC1,RMOFI ;;WAS ECC CORRECTION ALLOWED ??
26     BNE 80$ ;;NO !!
27     BIT #DCK,RMER1I ;;WAS THERE A DATA CHECK ??
28     BEQ 80$ ;;NO !!
29     BIT #ECH,RMER1I ;;IS ERROR CORRECTION HARD SET ?
30     BNE 80$ ;;YES !!
31     BIT #FMT16,RMOFI ;;IS THIS 16 BIT FORMAT ??
32     BEQ 80$ ;;NO !!
33
34     ;CORRECT DATA USING ECC INFORMATION
35     MOV RMBAD,R0 ;;R0 = STARTING BUFFER ADDRESS
36     MOV RMECI,R1 ;;R1 = ECC POSITION
37     BIS #BIT15,150$ ;;SET CORRECTION FLAG
38
39     ;MOVE R0 TO WORD BOUNDARY OF ERROR BURST
40     10$: CMP #16.,R1 ;;IS BIT POSITION > 1 WORD
41     BHIS 20$ ;;NO !!
42     SUB #16.,R1 ;;SUBTRACT 1 WORDS WORTH
43     TST (R0)+ ;;ADVANCE BUFFER ADDRESS 1 WORD
44     BR 10$
45
46     20$: MOV #1,R2 ;;R2 = BIT POINTER
47     MOV R2,R3 ;;R3 = BIT NUMBER
48
49     ;MOVE R2 TO STARTING BIT OF ERROR BURST
50     30$: CMP R3,R1 ;;IS R3 SAME AS R1 ??
51     BEQ 35$ ;;YES !!
52     ASL R2 ;;SHIFT BIT POINTER
53     INC R3 ;;INCREMENT BIT NUMBER
54
55     35$: MOV #11.,R3 ;;R3 = LENGTH OF ERROR BURST
56
57     ;CORRECT THE ERROR BURST
58     40$: BIT R2,RMECI ;;IS THIS BIT SET IN ECC PATTERN ??

```

```

54 042572 001405      BEQ      60$      ;NO - DO NOT CORRECT THIS BIT
55 042574 030210      BIT      R2,(R0)  ;IS THE BIT PRESENTLY SET ??
56 042576 001402      BEQ      50$      ;NO
57 042600 040210      BIC      R2,(R0)  ;RESET THE BIT
58 042602 000401      BR       60$
59 042604 050210      50$:  BIS      R2,(R0)  ;SET THE BIT
60 042606 006302      60$:  ASL      R2      ;SHIFT TO NEXT BIT
61 042610 001003      BNE      70$
62 042612 012702 000001  MOV      #1,R2      ;CONTINUE WITH FIRST BIT OF NEXT WORD
63 042616 005720      TST      (R0)+
64 042620 005303      70$:  DEC      R3      ;END OF BURST ??
65 042622 001361      BNE      40$      ;NO !!
66
67      ;COMPARE WRITE BUFFER TO READ BUFFER
68 042624 017600 000010  80$:  MOV      @10(SP),R0      ;R0 = WRITE BUFFER
69 042630 062766 000002 000010  ADD      #2,10(SP)      ;MOVE SP TO READ ADDRESS
70 042636 017601 000010      MOV      @10(SP),R1      ;R1 = READ BUFFER
71 042642 062766 000002 000010  ADD      #2,10(SP)      ;MOVE SP TO RETURN ADDRESS
72 042650 013702 001340      MOV      RMWC1,R2      ;R2 = NUMBER OF WORDS TRANSFER
73 042654 163702 001414      SUB      RMWC0,R2
74 042660 022021      90$:  CMP      (R0)+,(R1)+      ;COMPARE DATA WORDS
75 042662 001003      BNE      100$      ;EXIT IF NOT EQUAL
76 042664 005302      DEC      R2      ;DECREMENT WORD COUNT
77 042666 001374      BNE      90$      ;CONTINUE IF NOT DONE
78 042670 000433      BR       110$      ;DONE COMPARE - NO ERROR
79
80      ;DATA COMPARE FAILED
81 042672 014037 001140  100$:  MOV      -(R0),$GDDAT      ;STORE GOOD DATA FOR TYPEOUT
82 042676 014137 001142      MOV      -(R1),$BDDAT      ;STORE BAD DATA FOR TYPEOUT
83 042702 010037 001134      MOV      R0,$GDADR      ;STORE ADDRESS OF GOOD DATA
84 042706 010137 001136      MOV      R1,$BDADR      ;STORE ADDRESS OF BAD DATA
85 042712 010237 001174      MOV      R2,$TMP0      ;STORE WORD COUNT OF ERROR
86 042716 062766 000004 000010  ADD      #4,10(SP)      ;MOVE SP TO USER'S ERROR CALL
87 042724 112776 000336 000010  MOVB     #336,@10(SP)      ;WRITE ERROR NUMBER IN CALL
88
89      ;CHANGE ERROR NUMBER IF ECC CORRECTION FAILED
90 042732 032737 100000 042772  BIT      #BIT15,150$      ;WAS ECC CORRECTION USED ??
91 042740 001403      BEQ      105$      ;NO !!
92 042742 112776 000163 000010  MOVB     #163,@10(SP)      ;ECC CORRECTION FAILED
93 042750 162766 000002 000010  105$:  SUB      #2,10(SP)      ;MOVE SP TO RETURN IF ERROR
94 042756 000240      NOP
95 042760      110$:
96      MOV      (SP)+,R3      ;;POP STACK INTO R3
97      MOV      (SP)+,R2      ;;POP STACK INTO R2
98      MOV      (SP)+,R1      ;;POP STACK INTO R1
99      MOV      (SP)+,R0      ;;POP STACK INTO R0
00      RTS      PC      ;RETURN TO USER
01
02      150$:  .WORD      ;ECC CORRECTION FLAG
  
```



```

1      .SBTTL  GET STATUS SUBROUTINE
2
3      ;THIS SUBROUTINE SETS UP THE 'GET INDEX TABLE' AND THE 'GET
4      ;BUFFER' FOR READING ALL SUBSYSTEM REGISTERS VIA THE GET SUBROUTINE
5      ;AND THEN RETURNS TO THE USER.
6
7      ;CALL:  JSR      PC,GETSTS
8      ;      ???
9
10     GETSTS:
11         MOV      R0,-(SP)      ;;PUSH R0 ON STACK
12         MOV      R1,-(SP)      ;;PUSH R1 ON STACK
13         MOV      R2,-(SP)      ;;PUSH R2 ON STACK
14         MOV      #GETINX,R0    ;R0 = ADDRESS OF INDEX TABLE
15         MOV      #RMEC2I+2,R1  ;R1 = ADDRESS OF GET BUFFER
16         MOV      #RMEC2,R2     ;R2 = REGISTER INDE
17     2$:  MOV      R2,(R0)+      ;WRITE REGISTER INDEX IN TABLE
18         CLR      -(R1)         ;CLEAR CORRESPONDING LOCATION
19     3$:  SUB      #2,R2         ;DECREMENT TO NEXT INDEX
20         BMI      4$           ;BRANCH OUT IF DONE
21         CMP      #RMDB,R2     ;DONT WRITE RMDB INDEX
22         BNE      2$
23         CLR      -(R1)
24         BR       3$
25     4$:  MOV      #200,(R0)+    ;WRITE TERMINATOR
26         MOV      (SP)+,R2      ;;POP STACK INTO R2
27         MOV      (SP)+,R1      ;;POP STACK INTO R1
28         MOV      (SP)+,R0      ;;POP STACK INTO R0
29         NOP
30         RTS      PC
  
```

```

1      .SBTTL  GET SUBROUTINE
2
3      ;THIS SUBROUTINE READS THE REGISTERS WHICH ARE LISTED IN THE
4      ;'GET INDEX TABLE' AND STORES THEIR VALUES IN THE CORRESPONDING
5      ;LOCATION IN THE 'GET REGISTER BUFFER'.  FOR EXAMPLE, AN
6      ;ENTRY OF 04 IN THE TABLE WILL CAUSE THE SUBROUTINE TO
7      ;READ 'RMB4' AND STORE ITS CONTENTS AT THE LOCATION IN
8      ;THE BUFFER ASSIGNED TO THAT REGISTER.  THE NUMBER OF
9      ;REGISTERS TO BE READ IS VARIABLE FROM 1 TO 22; THE INDEX
10     ;TABLE MUST BE TERMINATED WITH A CONTROL BYTE (200)
11     ;WHICH SHOULD FOLLOW THE LAST ENTRY.
12
13     ;SUBROUTINE CALL:
14     ;(1)  'GET INDEX TABLE' HAS BEEN LOADED WITH REGISTER INDEX
15     ;      VALUES AND TERMINATED WITH A CONTROL BYTE
16     ;(2)  'GET INPUT BUFFER' IS AVAILABLE FOR USE.  (NOTE THAT
17     ;      UNUSED LOCATIONS, I.E., ENTRIES IN BUFFER CORRESPONDING
18     ;      TO REGISTERS NOT READ, ARE NOT CHANGED.)
19     ;(3)  JSR      PC,GET
20     ;      BR      ???          RETURN HERE IF NO ERROR FOUND
21     ;      NOP          RETURN HERE IF ANY ERROR FOUND
22     ;      ERROR          SUB DEFINES ERROR NUMBER
23     ;      ???
24
25     ;R0 = REGISTER BASE ADDRESS
26     ;R1 = REGISTER ADDRESS
27     ;R2 = BUFFER BASE ADDRESS
28     ;R3 = BUFFER ADDRESS
29     ;R4 = POINTER TO REGISTER INDEX
30
31     GET:  NOP
32           ADD      #4,(SP)          ;CLEAR ERROR NUMBER IN USER'S
33           CLR      @ (SP)          ;ERROR CALL
34           SUB      #4,(SP)
35           MOV      R0,-(SP)        ;;PUSH R0 ON STACK
36           MOV      R1,-(SP)        ;;PUSH R1 ON STACK
37           MOV      R2,-(SP)        ;;PUSH R2 ON STACK
38           MOV      R3,-(SP)        ;;PUSH R3 ON STACK
39           MOV      R4,-(SP)        ;;PUSH R4 ON STACK
40           MOV      ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
41           MOV      ERRVEC+2,-(SP)  ;;PUSH ERRVEC+2 ON STACK
42           MOV      $BASE,R0
43           MOV      #GETBUF,R2
44           MOV      #GETINX,R4
45           MOV      #5$,ERRVEC      ;SETUP FOR TIMEOUT
46           MOV      #PR6,ERRVEC+2
47           1$:  MOV      RMCS2(R0),$TMP0  ;GET 'NED' STATUS
48           MOV      RMCS1(R0),$TMP1  ;GET 'DVA' STATUS
49           BIT      #DVA,$TMP1      ;DEVICE AVAILABLE??
50           BNE      3$              ;YES!!
51           ADD      #4,16(SP)        ;WRITE ERROR NUMBER IN USER'S
52           MOVB     #112,@16(SP)    ;ERROR CALL
53           BR      7$
54           3$:  TSTB     (R4)          ;DONE??
55           BMI      9$              ;YES!!
56           MOVB     (R4),R1          ;R1 = REGISTER ADDRESS
57           BIC      #^CIDXMSK,R1    ;CLEAR ANY SIGN EXTENSION

```

```

31 043060 000240
32 043062 062716 000004
33 043066 105076 000000
34 043072 162716 000004
35 043076 010046
   043100 010146
   043102 010246
   043104 010346
   043106 010446
   043110 013746 000004
   043114 013746 000006
36 043120 013700 001276
37 043124 012702 001336
38 043130 012704 001524
39 043134 012737 043242 000004
40 043142 012737 000300 000006
41 043150 016037 000010 001174
42 043156 016037 000000 001176
43 043164 032737 004000 001176
44 043172 001007
45 043174 062766 000004 000016
46 043202 112776 000112 000016
47 043210 000423
48 043212 105714
49 043214 100433
50 043216 111401
51 043220 042701 177700

```

52	043224	060001			ADD	R0,R1	
53	043226	112403			MOVB	(R4)+,R3	;R3 - STORAGE ADDRESS FOR REGISTER
54	043230	042703	177700		BIC	#^CIDXMSK,R3	;CLEAR ANY SIGN EXTENSION
55	043234	060203			ADD	R2,R3	
56	043236	011113			MOV	(R1),(R3)	;READ REGISTER
57	043240	000764			BR	3\$	
58							
59	043242	022626		5\$:	CMP	(SP)+,(SP)+	;RESTORE STACK
60	043244	062766	000004	000016	ADD	#4,16(SP)	;WRITE ERROR NUMBER IN
61	043252	112776	000007	000016	MOVB	#7,216(SP)	;USER'S ERROR CALL
62	043260	162766	000002	000016	SUB	#2,16(SP)	
63	043266	105714		8\$:	TSTB	(R4)	;DONE CLEARING??
64	043270	100405			BMI	9\$	;YES.!
65	043272	005003			CLR	R3	;CLEAR REMAINING STORAGE
66	043274	112403			MOVB	(R4)+,R3	;LOCATIONS
67	043276	060203			ADD	R2,R3	
68	043300	005013			CLR	(R3)	
69	043302	000771			BR	8\$	
70	043304			9\$:			
	043304	012637	000006		MOV	(SP)+,ERRVEC+2	::POP STACK INTO ERRVEC+2
	043310	012637	000004		MOV	(SP)+,ERRVEC	::POP STACK INTO ERRVEC
	043314	012604			MOV	(SP)+,R4	::POP STACK INTO R4
	043316	012603			MOV	(SP)+,R3	::POP STACK INTO R3
	043320	012602			MOV	(SP)+,R2	::POP STACK INTO R2
	043322	012601			MOV	(SP)+,R1	::POP STACK INTO R1
	043324	012600			MOV	(SP)+,R0	::POP STACK INTO R0
71	043326	000207			RTS	PC	;RETURN

```

1      .SBTTL PUT SUBROUTINE
2
3      ;THIS SUBROUTINE WRITES THE REGISTERS WHICH ARE LISTED IN THE
4      ;'PUT INDEX TABLE' WITH THE CONTENTS OF THE CORRESPONDING
5      ;LOCATION IN THE 'PUT REGISTER BUFFER'. THE NUMBER OF
6      ;REGISTERS WRITTEN IS VARIABLE; THE INDEX TABLE MUST
7      ;BE TERMINATED WITH A CONTROL BYTE (200) WHICH SHOULD
8      ;FOLLOW THE LAST ENTRY.
9
10     ;SUBROUTINE CALL:
11
12     (1) 'PUT INDEX TABLE' HAS BEEN LOADED WITH INDEX VALUES
13         OF REGISTERS TO BE WRITTEN.
14     (2) 'PUT REGISTER BUFFER' CONTAINS CONTENTS OF EACH
15         REGISTER TO BE WRITTEN.
16     (3) JSR      PC,PUT
17         BR      ???          RETURN HERE IF NO ERROR FOUND
18         NOP
19         ERROR          RETURN HERE IF ANY ERROR FOUND
20         ???          SUB DEFINES ERROR NUMBER
21
22     ;R0 = REGISTER BASE ADDRESS
23     ;R1 = REGISTER ADDRESS
24     ;R2 = BUFFER BASE ADDRESS
25     ;R3 = BUFFER ADDRESS
26     ;R4 = POINTER TO REGISTER INDEX
27
28     PUT:  NOP
29           MOV      R0,-(SP)      ;;PUSH R0 ON STACK
30           MOV      R1,-(SP)      ;;PUSH R1 ON STACK
31           MOV      R2,-(SP)      ;;PUSH R2 ON STACK
32           MOV      R3,-(SP)      ;;PUSH R3 ON STACK
33           MOV      R4,-(SP)      ;;PUSH R4 ON STACK
34           MOV      ERRVEC,-(SP)  ;;PUSH ERRVEC ON STACK
35           MOV      ERRVEC+2,-(SP) ;;PUSH ERRVEC+2 ON STACK
36           MOV      $BASE,R0
37           MOV      #PUTBUF,R2
38           MOV      #PUTINX,R4
39           MOV      #5$,ERRVEC    ;SETUP FOR TIMEOUT
40           MOV      #PR6,ERRVEC+2
41           1$: MOV      RMCS2(R0),$TMP0 ;GET 'NED' STATUS
42           MOV      RMCS1(R0),$TMP1 ;GET 'DVA' STATUS
43           BIT      #DVA,$TMP1      ;DEVICE AVAILABLE??
44           BNE      3$             ;YES!!
45           ADD      #4,16(SP)       ;WRITE ERROR NUMBER IN
46           MOVB     #112,@16(SP)   ;USER'S ERROR CALL
47           BR      7$
48           3$:  TSTB     (R4)       ;DONE??
49           BMI      9$             ;YES!!
50           MOVB     (R4),R1        ;R1 = REGISTER ADDRESS
51           BIC      #^CIDXMSK,R1   ;CLEAR ANY SIGN EXTENSION
52           ADD      R0,R1
53           MOVB     (R4),R3        ;R3 = STORAGE ADDRESS
54           BIC      #^CIDXMSK,R3   ;CLEAR ANY SIGN EXTENSION
55           ADD      R2,R3
56           MOV      (R3),(R1)      ;WRITE REGISTER
57           4$:  TSTB     (R4)+     ;ADJUST REGISTER POINTER

```

```

52 043476 000763      BR      3$
53
54 043500 022626      5$:    CMP      (SP)+,(SP)+      ;ADJUST STACK
55 043502 062766 000004 000016      ADD      #4,16(SP)      ;WRITE ERROR NUMBER IN
56 043510 112776 000007 000016      MOVB     #7,216(SP)      ;USER'S ERROR CALL
57 043516 162766 000002 000016      7$:    SUB      #2,16(SP)
58
59 043524      9$:    MOV      (SP)+,ERRVEC+2      ;;POP STACK INTO ERRVEC+2
   043524 012637 000006      MOV      (SP)+,ERRVEC      ;;POP STACK INTO ERRVEC
   043530 012637 000004      MOV      (SP)+,R4      ;;POP STACK INTO R4
   043534 012604      MOV      (SP)+,R3      ;;POP STACK INTO R3
   043536 012603      MOV      (SP)+,R2      ;;POP STACK INTO R2
   043540 012602      MOV      (SP)+,R1      ;;POP STACK INTO R1
   043542 012601      MOV      (SP)+,R0      ;;POP STACK INTO R0
   043544 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
60 043546 000207      RTS      PC      ;RETURN
  
```

				.SBTTL SIZE CLOCK SUBROUTINE	
1					
2					
3	043550			SIZCLK:	
4	043550	013746	000004	MOV	ERRVEC, -(SP) ;:PUSH ERRVEC ON STACK
5	043554	013746	000006	MOV	ERRVEC+2, -(SP) ;:PUSH ERRVEC+2 ON STACK
6	043560	012737	043616 000004	MOV	#1\$,ERRVEC ;SET UP FOR BUS TIMEOUT
7	043566	012737	000300 000006	MOV	#PR6,ERRVEC+2
8	043574	012737	177546 001520	MOV	#177546,CLKADR ;LOAD ADDRESSES FOR KW11-L
9	043602	012737	000100 001522	MOV	#100,CLKVCT
10	043610	005777	135704	TST	@CLKADR ;TEST FOR KW11-L PRESENT
11	043614	000421		BR	3\$;YES - KW11-L IS PRESENT
12	043616	022626		1\$: CMP	(SP)+, (SP)+ ;RESTORE SP
13	043620	012737	043650 000004	MOV	#2\$,ERRVEC ;SET UP FOR BUS TIMEOUT
14	043626	012737	172540 001520	MOV	#172540,CLKADR ;LOAD ADDRESSES FOR KW11-P CLOCK
15	043634	012737	000104 001522	MOV	#104,CLKVCT
16	043642	005777	135652	TST	@CLKADR ;TEST FOR KW11-P PRESENT
17	043646	000404		BR	3\$;YES - KW11-P IS PRESENT
18	043650	022626		2\$: CMP	(SP)+, (SP)+ ;RESTORE SP
19	043652	062766	000002 000004	ADD	#2,4(SP) ;MOVE RETURN TO ERROR
20	043660			3\$: MOV	(SP)+,ERRVEC+2 ;:POP STACK INTO ERRVEC+2
21	043664	012637	000006	MOV	(SP)+,ERRVEC ;:POP STACK INTO ERRVEC
22	043670	000207		RTS	PC ;RETURN TO USER


```

1      .SBTTL TIMEOUT SUBROUTINE
2
3      ;THIS SUBROUTINE WAITS FOR RDY 1 AND GO = 0 OR FOR A TIMEOUT
4      ;GREATER THAN APPROX. 500 MSEC AND THEN RETURNS.
5
6      :CALL: JSR      PC,TIMOUT
7      :      ???
8      :      RETURN HERE
9      :TIMOUT:
10     043672 010046      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
11     043674 010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
12     043676 010246      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
13     043700 013746 000004      MOV      ERRVEC,-(SP)      ;;PUSH ERRVEC ON STACK
14     043704 013746 000006      MOV      ERRVEC+2,-(SP)      ;;PUSH ERRVEC+2 ON STACK
15     043710 012737 044012 000004      MOV      #4$,ERRVEC      ;SETUP FOR BUS TIMEOUT - 04 TRAP
16     043716 012737 000300 000006      MOV      #PR6,ERRVEC+2
17     043724 013700 001276      MOV      $BASE,R0      ;R0=BASE ADDRESS
18     043730 013701 001520      MOV      CLKADR,R1      ;R1=CLOCK ADDRESS
19     043734 012702 000036      MOV      #30,,R2      ;R2=NUMBER OF CLOCK CYCLES
20     043740 020127 172540      1$: CMP      R1,#172540      ;KW11-P CLOCK??
21     043744 001003      BNE      2$      ;NO!!
22     043746 012761 000001 000002      MOV      #1,2(R1)      ;SET COUNTER
23     043754 012711 000005      2$: MOV      #BIT2!BIT0,(R1) ;START COUNTER
24
25     043760 016046 000000      3$: MOV      RMCS1(R0),-(SP) ;GET STATUS
26     043764 042716 177576      BIC      #^C<RDY!GO>,(SP)
27     043770 022726 000200      CMP      #RDY,(SP)+      ;RDY=1,GO=0??
28     043774 001420      BEQ      5$      ;YES!!
29     043776 032711 000200      BIT      #BIT7,(R1)      ;TIMER DONE??
30     044002 001766      BEQ      3$      ;NO!!
31     044004 005302      DEC      R2      ;DEC NUMBER OF CYCLES
32     044006 001354      BNE      1$      ;CONTINUE IF NOT DONE
33     044010 000412      BR       5$      ;'RDY' DID NOT SET OR 'GO' DID NOT RESET
34
35     044012 022626      4$: CMP      (SP)+,(SP)+      ;WITHIN 500 MSEC AFTER THE COMMAND WAS ISSUED.
36     044014 062766 000004 000012      ADD      #4,12(SP)      ;ADJUST STACK
37     044022 112776 000007 000012      MOVB     #7,212(SP)      ;MOVE SP TO USER'S CALL
38     044030 162766 000002 000012      SUB      #2,12(SP)      ;WRITE ERROR NUMBER
39
40     044036      5$: MOV      (SP)+,ERRVEC+2      ;;POP STACK INTO ERRVEC+2
41     044036 012637 000006      MOV      (SP)+,ERRVEC      ;;POP STACK INTO ERRVEC
42     044042 012637 000004      MOV      (SP)+,R2      ;;POP STACK INTO R2
43     044046 012602      MOV      (SP)+,R1      ;;POP STACK INTO R1
44     044050 012601      MOV      (SP)+,R0      ;;POP STACK INTO R0
45     044052 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
46     044054 000207      RTS      PC      ;RETURN TO USER

```

```
1      .SBTTL PRIMARY ERROR CHECK SUBROUTINE
2
3      ;THE PURPOSE OF THIS SUBROUTINE IS TO VERIFY THAT STATUS IS VALID AND
4      ;THAT FURTHER ERROR AND STATUS CHECKING SHOULD BE PERFORMED. THE
5      ;FOLLOWING CHECKS ARE MADE:
6
7      ;.CORRECT UNIT IS SELECTED, I.E., THE UNIT SELECT BITS OF RMCS2
8      ;(BITS 0-2) EQUAL THE UNIT BEING TESTED;
9
10     ;.SELECTED UNIT IS AVAILABLE, I.E., DVA (BIT 11 OF RMCS1) IS SET
11     ;AND NED (BIT 12 OF RMCS2) IS RESET;
12
13     ;.LAST COMMAND WAS COMPLETED, I.E., THE MASSBUS CONTROLLER IS
14     ;READY (BIT 7 OF RMCS1) AND THE GO BIT IS RESET (BIT 0 OF RMCS1) OR THE
15     ;DRIVE READY BIT (BIT 7 OF RMD5) IS SET.
16     ;.NO PARITY ERROR OCCURRED WHEN READING REMOTE REGISTERS,
17     ;I.E., MCPE = 0.
18     ;.NO PARITY ERROR OCCURRED WHEN WRITING REMOTE REGISTERS,
19     ;I.E., PAR = 0, OR, PAR = DPE = 1
20
21     ;
22     ;THE SUBROUTINE ASSUMES THAT:
23
24     ;.STATUS HAS BEEN STORED IN THE REGISTER INPUT BUFFER,
25     ;IN PARTICULAR, RMCS1, RMCS2 AND RMD5 HAVE BEEN STORED IN THEIR
26     ;CORRESPONDING LOCATIONS OF THE "GET" BUFFER.
27
28     ;.($UNIT) CONTAINS THE DRIVE NUMBER
29
30     ;THE SUBROUTINE IS CALLED AS FOLLOWS:
31
32     ;(1)   JSR      PC,PRIERR
33     ;       BR       ???
34     ;       NOP
35     ;       ERROR
36     ;       JSR      PC,@(SP)+
37     ;       ???
38
39     ;RETURN HERE IF NO ERROR
40     ;RETURN HERE TO REPORT AN ERROR
41     ;ERROR NUMBER DEFINED BY SUB
42     ;GO BACK TO SUB FOR MORE ERROR CHECKS
43     ;RETURN HERE IF NO MORE ERRORS
44
45     PRIERR:
46
47     ;CLEAR USER'S ERROR CALL
48     ADD     #4,(SP)
49     CLRB    @ (SP)
50     SUB     #4,(SP)
51
52     ;MOVE (SP) TO ERROR CALL
53     ;CLEAR ERROR NUMBER
54     ;MOVE (SP) TO NO ERROR RETURN
55
56     ;REPORT AN ERROR IF THE WRONG UNIT IS SELECTED
57     MOV     RMCS2I,$BDDAT
58     BIC     #^CUNTMSK,$BDDAT
59     MOV     $UNIT,$GDDAT
60     BIC     #^CUNTMSK,$GDDAT
61     CMPB    $GDDAT,$BDDAT
62
63     ;CORRECT UNIT SELECTED??
64     ;GOOD DATA FOR TYPEOUT
65     ;COMPARE EXPECTED AND RECEIVED
66     ;DRIVE NUMBERS
67     ;YES..
68
69     BEQ     1$
70     ADD     #4,(SP)
71     MOVB    #1,@(SP)
72     SUB     #2,(SP)
73     JSR     PC,@(SP)+
74
75     ;ERROR 1
76     ;MOVE SP TO RETURN FOR ERROR
77     ;REPORT WRONG UNIT SELECTED
```



```
58 044152 162716 000010          SUB    #10,(SP)          ;RESTORE (SP)
59 044156 000240          NOP
60 044160 000137 044700          JMP     10$              ;SKIP OTHER CHECKS
61 044164          1$:
62
63          ;REPORT AN ERROR IF THE DEVICE IS NOT AVAILABLE OR IF
64          ;THE DEVICE IS NONEXISTANT
65 044164 032737 004000 001336    BIT     #DVA,RMCS1I      ;DEVICE AVAILABLE??
66 044172 001045          BNE     5$                  ;YES!!
67 044174 013737 001336 001140    MOV     RMCS1I,$GDDAT    ;EXPECTED STATUS
68 044202 052737 004000 001140    BIS     #DVA,$GDDAT
69 044210 013737 001336 001142    MOV     RMCS1I,$BDDAT    ;RECEIVED STATUS
70 044216 062716 000004          ADD     #4,(SP)
71 044222 112776 000002 000000    MOV     #2,@(SP)      ;ERROR #2
72 044230 032737 010000 001346    BIT     #NED,RMCS2I      ;WAS NED SET??
73 044236 001414          BEQ     2$                  ;NO!!
74 044240 013737 001346 001140    MOV     RMCS2I,$GDDAT    ;EXPECTED STATUS
75 044246 013737 001346 001142    MOV     RMCS2I,$BDDAT    ;RECEIVED STATUS
76 044254 042737 010000 001140    BIC     #NED,$GDDAT
77 044262 112776 000003 000000    MOV     #3,@(SP)      ;YES - CHANGE ERROR NUMBER
78 044270 162716 000002          2$: SUB     #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
79 044274 004736          JSR     PC,@(SP)+             ;REPORT DEVICE NOT AVAILABLE
80 044276 162716 000010          SUB     #10,(SP)      ;RESTORE (SP)
81 044302 000240          NOP
82 044304 000575          BR      10$              ;SKIP OTHER CHECKS
83 044306          5$:
84
85          ;REPORT AN ERROR IF MASSBUS CONTROLLER IS NOT READY
86 044306 032737 000200 001336    BIT     #RDY,RMCS1I      ;CONTROLLER READY??
87 044314 001030          BNE     7$                  ;YES!!
88 044316 013737 001336 001140    MOV     RMCS1I,$GDDAT    ;EXPECTED STATUS
89 044324 052737 000200 001140    BIS     #RDY,$GDDAT
90 044332 042737 160001 001140    BIC     #SC!TRE!MCPE!GO,$GDDAT
91 044340 013737 001336 001142    MOV     RMCS1I,$BDDAT    ;RECEIVED STATUS
92 044346 062716 000004          ADD     #4,(SP)
93 044352 112776 000004 000000    MOV     #4,@(SP)      ;ERROR #4
94 044360 162716 000002          SUB     #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
95 044364 004736          JSR     PC,@(SP)+             ;REPORT CONTROLLER NOT READY
96 044366 162716 000010          SUB     #10,(SP)      ;RESTORE (SP)
97 044372 000240          NOP
98 044374 000541          BR      10$              ;SKIP OTHER CHECKS
99 044376          7$:
100
101          ;REPORT AN ERROR IF GO IS NOT ZERO AND DRY IS NOT ONE
102 044376 032737 000001 001336    BIT     #GO,RMCS1I      ;GO RESET??
103 044404 001431          BEQ     8$                  ;YES!!
104 044406 032737 000200 001350    BIT     #DRY,RMDSI      ;DRIVE READY??
105 044414 001025          BNE     8$                  ;YES!!
106 044416 013737 001336 001140    MOV     RMCS1I,$GDDAT    ;EXPECTED STATUS
107 044424 042737 160001 001140    BIC     #SC!TRE!MCPE!GO,$GDDAT
108 044432 013737 001336 001142    MOV     RMCS1I,$BDDAT    ;RECEIVED STATUS
109 044440 062716 000004          ADD     #4,(SP)
110 044444 112776 000005 000000    MOV     #5,@(SP)      ;ERROR #5
111 044452 162716 000002          SUB     #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
112 044456 004736          JSR     PC,@(SP)+             ;REPORT DRIVE NOT READY
113 044460 162716 000010          SUB     #10,(SP)      ;RESTORE (SP)
114 044464 000240          NOP
```

```
115 044466 000504          BR      10$
116 044470          8$:
117
118          ;REPORT AN ERROR IF THE RH CONTROLLER DETECTED BAD
119          ;PARITY ON THE MASSBUS CONTROL BUS
120 044470 032737 020000 001336  BIT      #MCPE,RMCS1I      ;PARITY ERROR ??
121 044476 001425          BEQ      9$                  ;NO!!
122 044500 013737 001336 001140  MOV      RMCS1I,$GDDAT      ;EXPECTED STATUS
123 044506 042737 160001 001140  BIC      #SC!TRE!MCPE!GO,$GDDAT
124 044514 013737 001336 001142  MOV      RMCS1I,$BDDAT      ;RECEIVED STATUS
125 044522 062716 000004          ADD      #4,(SP)          ;MOVE STACK TO USER'S ERROR
126 044526 112776 000013 000000  MOVB     #13,@(SP)      ;ERROR #13
127 044534 162716 000002          SUB      #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
128 044540 004736          JSR      PC,@(SP)+              ;REPORT ERROR VIA USER
129 044542 162716 000010          SUB      #10,(SP)         ;RESTORE STACK
130 044546 000240          NOP
131 044550 000453          BR      10$
132 044552          9$:
133
134          ;REPORT AN ERROR IF DETECTED A CONTROL BUS PARITY ERROR
135 044552 032737 000010 001352  BIT      #PAR,RMER1I      ;WAS THERE A PARITY ERROR??
136 044560 001451          BEQ      11$                  ;NO!!
137 044562 032737 000010 001400  BIT      #DPE,RMER2I      ;WAS IT THE CONTROL BUS??
138 044570 001045          BNE      11$                  ;NOT SURE!!
139 044572 032737 000010 001426  BIT      #PAR,RMER10      ;DID TEST SET PAR ??
140 044600 001413          BEQ      93$                  ;NO!!
141 044602 010046          MOV      R0,-(SP)              ;PUSH R0 ON STACK
142 044604 012700 001553          MOV      #PUTINX,R0       ;R0 POINTS TO INDEX TABLE
143 044610 122710 000014          91$: CMPB     #RMER1,(R0)   ;SEARCH TABLE FOR RMER1
144 044614 001002          BNE      92$
145 044616 012600          MOV      (SP)+,R0              ;POP STACK INTO R0
146 044620 000431          BR      11$                  ;PAR WAS SET BY TEST
147 044622 105720          92$: TSTB     (R0)+              ;END OF TABLE??
148 044624 100371          BPL      91$                  ;NO!!
149 044626 012600          MOV      (SP)+,R0              ;POP STACK INTO R0
150 044630 013737 001352 001140  93$: MOV      RMER1I,$GDDAT      ;EXPECTED STATUS
151 044636 042737 000010 001140  BIC      #PAR,$GDDAT
152 044644 013737 001352 001142  MOV      RMER1I,$BDDAT      ;RECEIVED STATUS
153 044652 062716 000004          ADD      #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
154 044656 112776 000050 000000  MOVB     #50,@(SP)        ;WRITE THE ERROR NUMBER
155 044664 162716 000002          SUB      #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
156 044670 004736          JSR      PC,@(SP)+              ;REPORT THE ERROR
157 044672 162716 000010          SUB      #10,(SP)         ;MOVE SP TO NO ERROR RETURN
158 044676 000240          NOP
159 044700 062716 000010          10$: ADD      #10,(SP)      ;RETURN TO ERROR
160 044704 000240          11$: NOP                      ;RETURN TO NO ERROR
161 044706 000207          RTS      PC
```

```

1      .SBTTL  SECONDARY ERROR CHECK SUBROUTINE
2
3      ;THE ERROR CHECK SUBROUTINE PROVIDES DETECTION OF SECONDARY ERRORS
4      ;SUCH AS UNEXPECTED ERRORS AND UNEXPECTED REGISTER CONTENTS. THESE
5      ;ERRORS ARE DEEMED SECONDARY IN THAT THEY ARE NOT NECESSARILY
6      ;ASSOCIATED WITH THE OPERATION BEING PERFORMED.
7      ;WHEN THE SUBROUTINE IDENTIFIES SUCH AN ERROR, IT MOVES THE ERROR
8      ;NUMBER TO THE ERROR CALL IN THE TEST ROUTINE AND THEN RETURNS
9      ;TO THE TEST ROUTINE WHICH MAKES THE ERROR CALL. AFTER THE TEST ROUTINE
10     ;MAKES THE ERROR CALL, IT RETURNS TO THE SUBROUTINE WHICH THEN LOOKS FOR
11     ;OTHER ERRORS. WHEN ALL ERRORS HAVE BEEN REPORTED, THE SUBROUTINE
12     ;RETURNS TO THE ADDRESS FOLLOWING THE SUBROUTINE CALL.
13
14     :CALL:  JSR      PC,SECERR
15             BR       ???          RETURN HERE IF NO ERROR
16             NOP      RETURN HERE TO REPORT AN ERROR
17             ERROR    ERROR NUMBER DEFINED BY SUB
18             JSR      PC,@(SP)+    GO BACK TO SUB FOR MORE ERROR CHECKS
19             ???      RETURN HERE IF NO MORE ERRORS
20
21     ;NOTE: THE SUBROUTINE ASSUMES THAT REGISTERS HAVE BEEN STORED AT THE
22     ;INPUT REGISTER BUFFER.
23
24     044710 SECERR:
25
26     ;*****
27     ;STORE FUNCTION CODE AND CLEAR USER'S ERROR NUMBER
28     044710 013737 001412 050544 MOV     RMCS10,515$ ;STORE FUNCTION CODE
29     044716 042737 177701 050544 BIC     #^C<F0!F1!F2.F3!F4>,515$
30     044724 062716 000004 ADD      #4,(SP) ;MOVE (SP) TO ERROR CALL
31     044730 105076 000000 CLRB     @ (SP) ;CLEAR ERROR NUMBER
32     044734 162716 000004 SUB      #4,(SP) ;MOVE (SP) TO NO ERROR RETURN
33
34     ;*****
35     ;CHECK SECONDARY ERRORS COMMON TO ALL COMMANDS
36
37     ;REPORT ERROR IF DRIVE IS NOT READY, I.E., IF DRY = 0
38     044740 032737 000200 001350 BIT     #DRY,RMDSI ;DRIVE READY??
39     044746 001024 BNE      5$ ;YES!!
40     044750 013737 001350 001142 MOV     RMDSI,$BDDAT ;BAD DATA FOR TYPEOUT
41     044756 042737 177577 001142 BIC     #^CDRY,$BDDAT
42     044764 012737 000200 001140 MOV     #DRY,$GDDAT ;GOOD DATA FOR TYPEOUT
43     044772 062716 000004 ADD      #4,(SP)
44     044776 112776 000010 000000 MOVB    #10,@(SP) ;ERROR NUMBER
45     045004 162716 000002 SUB      #2,(SP) ;MOVE SP TO RETURN FOR ERROR
46     045010 004736 JSR      PC,@(SP)+ ;REPORT NOT READY
47     045012 162716 000010 SUB      #10,(SP) ;RESTORE (SP) TO ERROR N
48     045016 000240 NOP
49
50     ;REPORT ERROR IF GO BIT IS NOT RESET
51     045020 032737 000001 001336 5$: BIT     #GO,RMCS11 ;GO BIT RESET??
52     045026 001423 BEQ      10$ ;YES..
53     045030 013737 001336 001142 MOV     RMCS11,$BDDAT ;BAD DATA FOR TYPEOUT
54     045036 042737 177776 001142 BIC     #^CGO,$BDDAT
55     045044 005037 001140 CLR      $GDDAT ;GOOD DATA FOR TYPEOUT
56     045050 062716 000004 ADD      #4,(SP)
57     045054 112776 000011 000000 MOVB    #11,@(SP) ;ERROR NUMBER

```

58	045062	162716	000002		SUB	#2,(SP)	;MOVE SP TO RETURN FOR ERROR
59	045066	004736			JSR	PC,@(SP)+	;REPORT DEVICE NOT AVAILABLE
60	045070	162716	000010		SUB	#10,(SP)	;RESTORE (SP)
61	045074	000240			NOP		
62							
63							
64	045076	013737	001336	001142	10\$:	MOV	RMCS1I,\$BDDAT ;IS FUNCTION CODE CORRECT??
65	045104	042737	177701	001142		BIC	#^C76,\$BDDAT
66	045112	013737	050544	001140		MOV	515\$,\$GDDAT ;EXPECTED FUNCTION CODE
67	045120	023737	001142	001140		CMP	\$BDDAT,\$GDDAT
68	045126	001413				BEQ	15\$;YES.!
69	045130	062716	000004			ADD	#4,(SP)
70	045134	112776	000012	000000		MOVB	#12,@(SP) ;ERROR NUMBER
71	045142	162716	000002			SUB	#2,(SP) ;MOVE SP TO RETURN FOR ERROR
72	045146	004736				JSR	PC,@(SP)+ ;REPORT WRONG FUNCTION CODE
73	045150	162716	000010			SUB	#10,(SP) ;RESTORE (SP)
74	045154	000240				NOP	
75	045156				15\$:		
76							
77							
78							
79	045156	005037	001140			CLR	\$GDDAT ;EXPECT 'ERR'-- 0
80	045162	005737	001352			TST	RMER1I ;IS RMER1 = 0??
81	045166	001003				BNE	20\$;NO!!
82	045170	005737	001400			TST	RMER2I ;IS RMERZ = 0??
83	045174	001403				BEQ	25\$;YES.!
84	045176	052737	040000	001140	20\$:	BIS	#ERR,\$GDDAT ;'ERR' SOULD BE SET
85	045204	013737	001350	001142	25\$:	MOV	RMDSI,\$BDDAT
86	045212	042737	137777	001142		BIC	#^CERR,\$BDDAT
87	045220	023737	001140	001142		CMP	\$GDDAT,\$BDDAT ;IS 'ERR' OK??
88	045226	001412				BEQ	30\$;YES!!
89	045230	062716	000004			ADD	#4,(SP) ;MOVE SP TO USER'S ERROR
90	045234	112776	000047	000000		MOVB	#47,@(SP) ;WRITE ERROR NUMBER
91	045242	162716	000002			SUB	#2,(SP) ;MOVE SP TO ERROR RETURN
92	045246	004736				JSR	PC,@(SP)+ ;REPORT INVALID COMP ERROR
93	045250	162716	000010			SUB	#10,(SP)
94							
95							
96							
97							
98	045254	005037	001140		30\$:	CLR	\$GDDAT ;EXPECT 'TRE' - 0
99	045260	013746	001346			MOV	RMCS2I,-(SP) ;WAS DLT, WCE, UPE, NED, NEM
100	045264	042726	000377			BIC	#377,(SP)+ ;PGE, MXF OR MDPE SET
101	045270	001010				BNE	35\$;YES.!
102	045272	032737	040000	001350		BIT	#ERR,RMDSI ;WAS EXCEPTION RECEIVED??
103	045300	001407				BEQ	40\$;NO..
104	045302	022737	000030	050544		CMP	#SEARCH,515\$;WAS DATA TRANSFERRED??
105	045310	103003				BHIS	40\$;NO.!
106	045312	052737	040000	001140	35\$:	BIS	#TRE,\$GDDAT ;'TRE' SHOULD BE SET
107	045320	013737	001336	001142	40\$:	MOV	RMCS1I,\$BDDAT ;BAD DATA FOR TYPEOUT
108	045326	042737	137777	001142		BIC	#^CTRE,\$BDDAT
109	045334	023737	001140	001142		CMP	\$GDDAT,\$BDDAT ;IS 'TRE' OK??
110	045342	001413				BEQ	45\$;YES.!
111	045344	042716	000004			ADD	#4,(SP) ;MOVE SP TO USER'S ERROR CALL
112	045350	112776	000014	000000		MOVB	#14,@(SP) ;WRITE ERROR NUMBER
113	045356	162716	000002			SUB	#2,(SP) ;MOVE SP TO RETURN FOR ERROR
114	045362	004736				JSR	PC,@(SP)+ ;REPORT TRE ERROR

```

115 045364 162716 000010          SUB      #10,(SP)          ;RESTORE (SP)
116 045370 000240          NOP
117 045372          45$:
118
119          ;*****
120          ;USING THE FUNCTION CODE TABLE, CHECK FOR THE FOLLOWING ERRORS:
121          ;.STATUS BITS NOT SET THAT SHOULD BE SET, E.G., ATA AND ILF
122          ;.STATUS BITS SET THAT SHOULD NOT BE SET, E.G., WCE AND ECH
123          ;NOTE THAT SOME ERROR BITS ARE CONDITIONAL ON THE COMMAND AND OTHER
124          ;STATUS CONDITIONS, E.G., WRITE LOCK ERROR SHOULD ONLY BE SET IF
125          ;WRITE LOCK IS ON AND THE COMMAND IS A WRITE.
126
127          ;GET AND STORE THE ENTRY FROM THE FUNCTION CODE TABLE
128 045372 010046          MOV      R0,-(SP)          ;;PUSH R0 ON STACK
129 045374 013700 050544      MOV      515$,R0          ;GET FUNCTION CODE
130 045400 016037 071646 050536      MOV      FNCDTB(R0),500$ ;STORE ENTRY
131 045406 012600          MOV      (SP)+,R0          ;;POP STACK INTO R0
132
133          ;REPORT AN ERROR IF AN UNEXPECTED ATTENTION OCCURRED OR IF
134          ;ATA IS NOT SET AND SHOULD BE SET.
135 045410 013737 050536 001140      MOV      500$,$GDDAT          ;GET EXPECTED ATA STATUS
136 045416 032737 040000 001350      BIT      #ERR,RMDSI          ;IS COMPOSITE ERROR SET ??
137 045424 001403          BEQ      50$              ;NO !!
138 045426 052737 100000 001140      BIS      #ATA,$GDDAT          ;EXPECT AN ATTENTION
139 045434 042737 077777 001140 50$: BIC      #^CATA,$GDDAT          ;STRIP DONT CARES
140 045442 013737 001350 001142      MOV      RMDSI,$BDDAT          ;GET RECEIVED ATA
141 045450 042737 077777 001142      BIC      #^CATA,$BDDAT          ;STRIP DONT CARES
142 045456 023737 001140 001142      CMP      $GDDAT,$BDDAT          ;IS ATA OK ??
143 045464 001413          BEQ      55$              ;YES !!
144 045466 062716 000004          ADD      #4,(SP)          ;MOVE SP TO USERS ERROR CALL
145 045472 112776 000006 000000      MOVSB   #6,@(SP)          ;LOAD ERROR # IN CALL
146 045500 162716 000002          SUB      #2,(SP)          ;MOVE SP TO ERROR RETURN
147 045504 004736          JSR      PC,@(SP)+          ;REPORT ERROR
148 045506 162716 000010          SUB      #10,(SP)         ;RESTORE SP
149 045512 000240          NOP
150 045514          55$:
151
152          ;REPORT ERROR IF ILF IS INCORRECT, I.E., IF ILF DOES NOT COMPARE
153          ;WITH FUNCTION CODE TABLE
154 045514 013737 050536 001140      MOV      500$,$GDDAT          ;GET EXPECTED ILF
155 045522 042737 177776 001140      BIC      #^CILF,$GDDAT          ;CLEAR ALL OTHER BITS
156 045530 013737 001352 001142      MOV      RMERR1I,$BDDAT          ;GET RECEIVED ILF
157 045536 042737 177776 001142      BIC      #^CILF,$BDDAT          ;CLEAR ALL OTHER BITS
158 045544 023737 001140 001142      CMP      $GDDAT,$BDDAT          ;IS ILF OK ??
159 045552 001412          BEQ      60$              ;YES !!
160 045554 062716 000004          ADD      #4,(SP)          ;MOVE SP TO USERS ERROR CALL
161 045560 112776 000254 000000      MOVSB   #254,@(SP)          ;WRITE ERROR NUMBER IN CALL
162 045566 162716 000002          SUB      #2,(SP)          ;MOVE SP TO ERROR RETURN
163 045572 004736          JSR      PC,@(SP)+          ;REPORT ERROR AND RETURN
164 045574 162716 000010          SUB      #10,(SP)         ;MOVE SP TO NO ERROR
165 045600 005037 001140 60$: CLR      $GDDAT          ;CLEAR EXPECTED STATUS
166
167          ;REPORT AN ERROR IF WCE IS SET AND SHOULD NOT BE SET
168 045604 013746 050536          MOV      500$,-(SP)          ;GET WCE STATUS ENABLE
169 045610 052716 137777          BIS      #^CWCE,(SP)          ;SET ALL OTHER BITS
170 045614 013737 001346 001142      MOV      RMCS2I,$BDDAT          ;RECEIVED STATUS
171 045622 042637 001142          BIC      (SP)+,$BDDAT          ;CLEAR WCE IF ENABLED
    
```

172	045626	001412			BEQ	90\$		;BRANCH IF WCE OK
173	045630	062716	000004		ADD	#4,(SP)		;MOVE SP TO USER'S ERROR CALL
174	045634	112776	000026	000000	MOVB	#26,@(SP)		;WRITE ERROR NUMBER
175	045642	162716	000002		SUB	#2,(SP)		;MOVE SP TO ERROR RETURN
176	045646	004736			JSR	PC,@(SP)+		;REPORT ERROR
177	045650	162716	000010		SUB	#10,(SP)		;RESTORE ERROR
178	045654					90\$:		
179								
180								;REPORT ERROR IF OPI STATUS IS SET AND SHOULD NOT BE SFT
181	045654	013746	050536		MOV	500\$,-(SP)		;GET OPI STATUS ENABLE
182	045660	052716	157777		BIS	#^COPI,(SP)		;SET ALL OTHER BITS
183	045664	013737	001352	001142	MOV	RMER1I,\$BDDAT		;GET RECEIVED STATUS
184	045672	042637	001142		BIC	(SP)+,\$BDDAT		;CLEAR OPI IF ENABLED
185	045676	001412			BEQ	100\$		;BRANCH IF OPI OK
186	045700	062716	000004		ADD	#4,(SP)		;MOVE SP TO USER'S ERROR CALL
187	045704	112776	000164	000000	MOVB	#164,@(SP)		;WRITE ERROR NUMBER IN CALL
188	045712	162716	000002		SUB	#2,(SP)		;MOVE SP TO ERROR RETURN
189	045716	004736			JSR	PC,@(SP)+		;REPORT ERROR
190	045720	162716	000010		SUB	#10,(SP)		;RESTORE SP
191	045724					100\$:		
192								
193								;REPORT ERROR IF IVC IS SET AND IS NOT ENABLED OR IF IVC IS
194								;SET AND VV IS NOT RESET
195	045724	013746	050536		MOV	500\$,-(SP)		;GET IVC STATUS ENABLE
196	045730	032737	000100	001350	BIT	#VV,RMDSI		;IS VV SET
197	045736	001402			BEQ	105\$		;NO !!
198	045740	042716	010000		BIC	#IVC,(SP)		;YES - IVC SHOULD BE 0
199	045744	052716	167777		BIS	#^CIVC,(SP)		;SET ALL OTHER BITS
200	045750	013737	001400	001142	MOV	RMER2I,\$BDDAT		;GET RECEIVED STATUS
201	045756	042637	001142		BIC	(SP)+,\$BDDAT		;CLEAR IVC IF ENABLED
202	045762	001412			BEQ	110\$		;BRANCH IF IVC OK
203	045764	062716	000004		ADD	#4,(SP)		;MOVE SP TO USERS ERROR CALL
204	045770	112776	000165	000000	MOVB	#165,@(SP)		;WRITE ERROR NUMBER IN CALL
205	045776	162716	000002		SUB	#2,(SP)		;MOVE SP TO ERROR RETURN
206	046002	004736			JSR	PC,@(SP)+		;REPORT ERROR
207	046004	162716	000010		SUB	#10,(SP)		;RESTORE SP TO NO ERROR
208	046010					110\$:		
209								
210								;BIT 11 (WLE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
211								;ALL WRITE ERRORS, I.E.,
212								;RMER1 - WLE, WCF
213								;RMER2 - DPE
214								;RMCS2 - UPE.
215								;EACH OF THESE ERRORS IS CHECKED TO SEE IF AN ERROR IS SET WHEN THE
216								;WRITE ERROR ENABLE BIT IS RESET.
217								
218								;REPORT AN ERROR IF WLE IS SET AND WRITE ERRORS ARE NOT ENABLED, OR IF
219								;THE DRIVE IS NOT WRITE PROTECTED
220	046010	012746	177777		MOV	#-1,-(SP)		;ASSUME WRITE ERRORS ENABLED
221	046014	032737	004000	050536	BIT	#WLF,500\$		;ARE WRITE ERRORS FNABLED ??
222	046022	001404			BEQ	115\$		;NO !!
223	046024	032737	004000	001350	BIT	#WRL,RMDSI		;IS THE DRIVE WRITE PROTECTED ??
224	046032	001002			BNE	120\$		;YES !!
225	046034	042716	004000		BIC	#WLE,(SP)		;RESET WLE ENABLE
226	046040	013737	001352	001142	MOV	RMER1I,\$BDDAT		;GET RECEIVED STATUS
227	046046	042637	001142		BIC	(SP)+,\$BDDAT		;CLEAR WLE IF ENABLED
228	046052	001412			BEQ	125\$		;BRANCH IF WLE OK


```

229 046054 062716 000004      ADD    #4,(SP)      ;MOVE SP TO USERS ERROR CALL
230 046060 112776 000023 000000  MOV    #23,@(SP)    ;WRITE ERROR NUMBER IN CALL
231 046066 162716 000002      SUB    #2,(SP)      ;MOVE SP TO ERROR RETURN
232 046072 004736      JSR    PC,@(SP)+  ;REPORT ERROR AND RETURN
233 046074 162716 000010      SUB    #10,(SP)     ;RESTORE SP TO NO ERROR
234 046100      125$:
235
236      ;REPORT ERROR IF WCF IS SET AND WRITE ERRORS ARE NOT ENABLED
237 046100 012746 177777      MOV    #-1,-(SP)    ;ASSUME WRITE ERRORS ENABLED
238 046104 032737 004000 050536  BIT    #WLE,500$    ;ARE WRITE ERRORS ENABLED ??
239 046112 001002      BNE    130$      ;YES !!
240 046114 042716 000040      BIC    #WCF,(SP)    ;DISABLE WCF ERROR
241 046120 013737 001352 001142 130$: MOV    RMER1I,$BDDAT ;GET RECEIVED STATUS
242 046126 042637 001142      BIC    (SP)+,$BDDAT ;RESET WCF IF ENABLED
243 046132 001412      BEQ    135$      ;BRANCH IF WCF OK
244 046134 062716 000004      ADD    #4,(SP)      ;MOVE SP TO USERS ERROR CALL
245 046140 112776 000025 000000  MOV    #25,@(SP)    ;WRITE ERROR NUMBER IN CALL
246 046146 162716 000002      SUB    #2,(SP)      ;MOVE SP TO ERROR RETURN
247 046152 004736      JSR    PC,@(SP)+  ;REPORT ERROR
248 046154 162716 000010      SUB    #10,(SP)     ;RESTORE SP TO NO ERROR
249 046160      135$:
250
251      ;REPORT ERROR IF DPE IS SET AND WRITE ERRORS ARE NOT ENABLED
252 046160 012746 177777      MOV    #-1,-(SP)    ;ASSUME WRITE ERRORS ARE ENABLED
253 046164 032737 004000 050536  BIT    #WLE,500$    ;ARE WRITE ERRORS ENABLED ??
254 046172 001002      BNE    140$      ;YES !!
255 046174 042716 000010      BIC    #DPE,(SP)    ;RESET DPE ENABLE
256 046200 013737 001400 001142 140$: MOV    RMER2I,$BDDAT ;GET RECEIVED STATUS
257 046206 042637 001142      BIC    (SP)+,$BDDAT ;RESET DPE IF ENABLED
258 046212 001412      BEQ    145$      ;BRANCH IF DPE OK
259 046214 062716 000004      ADD    #4,(SP)      ;MOVE SP TO USERS ERROR CALL
260 046220 112776 000040 000000  MOV    #40,@(SP)    ;WRITE ERROR NUMBER IN CALL
261 046226 162716 000002      SUB    #2,(SP)      ;MOVE SP TO ERROR RETURN
262 046232 004736      JSR    PC,@(SP)+  ;REPORT ERROR
263 046234 162716 000010      SUB    #10,(SP)     ;RESTORE SP TO NO ERROR
264 046240      145$:
265
266      ;REPORT AN ERROR IF UPE IS SET AND WRITE ERRORS ARE NOT ENABLED
267 046240 012746 177777      MOV    #-1,-(SP)    ;ASSUME WRITE ERRORS ARE ENABLED
268 046244 032737 004000 050536  BIT    #WLE,500$    ;ARE WRITE ERRORS ENABLED ??
269 046252 001002      BNE    150$      ;YES !!
270 046254 042716 020000      BIC    #UPE,(SP)    ;DISABLE UPE ERROR
271 046260 013737 001346 001142 150$: MOV    RMCS2I,$BDDAT ;GET RECEIVED STATUS
272 046266 042637 001142      BIC    (SP)+,$BDDAT ;RESET UPE IF ENABLED
273 046272 001412      BEQ    155$      ;BRANCH IF UPE OK
274 046274 062716 000004      ADD    #4,(SP)      ;MOVE SP TO USERS ERROR CALL
275 046300 112776 000024 000000  MOV    #24,@(SP)    ;WRITE ERROR NUMBER IN CALL
276 046306 162716 000002      SUB    #2,(SP)      ;MOVE SP TO ERROR RETURN
277 046312 004736      JSR    PC,@(SP)+  ;REPORT ERROR AND RETURN
278 046314 162716 000010      SUB    #10,(SP)     ;MOVE SP TO NO ERROR
279 046320      155$:
280
281      ;REPORT AN ERROR IF IAE IS SET AND IS NOT ENABLED
282 046320 013746 050536      MOV    500$,-(SP)    ;GET IAE ENABLE
283 046324 052716 175777      BIS    #^CIAE,(SP)  ;SET ALL OTHER BITS
284 046330 013737 001352 001142      MOV    RMER1I,$BDDAT ;GET RECEIVED STATUS
285 046336 042637 001142      BIC    (SP)+,$BDDAT ;CLEAR IAE IF ENABLED

```

```

286 046342 001412          BEQ      160$          ;BRANCH IF IAE IS OK
287 046344 062716 000004    ADD      #4,(SP)        ;MOVE SP TO USERS ERROR CALL
288 046350 112776 000166 000000 MOV     #166,@(SP)    ;WRITE ERROR NUMBER
289 046356 162716 000002    SUB      #2,(SP)        ;MOVE SP TO ERROR RETURN
290 046362 004736          JSR      PC,@(SP)+      ;REPORT ERROR AND RETURN
291 046364 162716 000010    SUB      #10,(SP)       ;MOVE SP TO NO ERROR
292 046370          160$:
293
294          ;BIT 09 (AOE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
295          ; ALL READ/WRITE ERRORS, I.E.,
296          ; RMCS1 - TRE
297          ; RMCS2 - DLT,NEM,MXF
298          ; RMDS - LBT
299          ; RMER1 - AOE
300          ;NOTE:
301          ; LBT IS NOT CHECKED BECAUSE IT ONLY RESETS WHEN THE DESIRED
302          ;CYLINDER REGISTER IS WRITTEN
303          ;NOTE:
304          ; AOE CANNOT BE SET IF LBT IS NOT ALSO SET
305          ;NOTE:
306          ; TRE IS CHECKED AS A FUNCTION OF OTHER ERROR CONDITONS ABOVE
307
308
309          ;REPORT AN ERROR IF DLT IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
310 046370 012746 177777    MOV      #-1,-(SP)        ;ASSUME ERRORS ARE ENABLED
311 046374 032737 001000 050536 BIT      #AOE,500$      ;ARE ERRORS ENABLED ??
312 046402 001002          BNE      165$          ;YES !!
313 046404 042716 100000    BIC      #DLT,(SP)        ;RESET DLT ENABLE
314 046410 013737 001346 001142 165$: MOV     RMCS2I,$BDDAT ;GET RECEIVED STATUS
315 046416 042637 001142    BIC      (SP)+,$BDDAT    ;CLEAR DLT IF ENABLED
316 046422 001412          BEQ      170$          ;BRANCH IF DLT IS OK
317 046424 062716 000004    ADD      #4,(SP)        ;MOVE SP TO USERS ERROR CALL
318 046430 112776 000032 000000 MOV     #32,@(SP)    ;WRITE ERROR NUMBER IN CALL
319 046436 162716 000002    SUB      #2,(SP)        ;MOVE SP TO ERROR RETURN
320 046442 004736          JSR      PC,@(SP)+      ;REPORT ERROR AND RETURN
321 046444 162716 000010    SUB      #10,(SP)       ;MOVE SP TO NO ERROR
322 046450          170$:
323
324          ;REPORT ERROR IF NEM IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
325 046450 012746 177777    MOV      #-1,-(SP)        ;ASSUME ERRORS ARE ENABLED
326 046454 032737 001000 050536 BIT      #AOE,500$      ;ARE ERRORS ENABLED ??
327 046462 001002          BNE      175$          ;YES !!
328 046464 042716 004000    BIC      #NEM,(SP)        ;DISABLE NEM
329 046470 013737 001346 001142 175$: MOV     RMCS2I,$BDDAT ;GET RECEIVED STATUS
330 046476 042637 001142    BIC      (SP)+,$BDDAT    ;CLEAR NEM IF ENABLED
331 046502 001412          BEQ      180$          ;BRANCH IF NEM IS OK
332 046504 062716 000004    ADD      #4,(SP)        ;MOVE SP TO USERS ERROR CALL
333 046510 112776 000167 000000 MOV     #167,@(SP)    ;WRITE ERROR NUMBER IN CALL
334 046516 162716 000002    SUB      #2,(SP)        ;MOVE SP TO ERROR RETURN
335 046522 004736          JSR      PC,@(SP)+      ;REPORT ERROR AND RETURN
336 046524 162716 000010    SUB      #10,(SP)       ;MOVE SP TO NO ERROR
337 046530          180$:
338
339          ;REPORT ERROR IF MXF IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
340 046530 012746 177777    MOV      #-1,-(SP)        ;ASSUME ERRORS ARE ENABLED
341 046534 032737 001000 050536 BIT      #AOE,500$      ;ARE DATA ERRORS ENABLED ??
342 046542 001002          BNE      185$          ;YES !!
    
```



```

343 046544 042716 001000      BIC      #MXF,(SP)      ;DISABLE MXF ERROR
344 046550 013737 001346 001142 185$: MOV      RMCS2I,$BDDAT    ;GET RECEIVED STATUS
345 046556 042637 001142      BIC      (SP)+,$BDDAT    ;CLEAR MXF IF ENABLED
346 046562 001412      BEQ      190$      ;BRANCH IF MXF IS OK
347 046564 062716 000004      ADD      #4,(SP)      ;MOVE SP TO USERS ERROR CALL
348 046570 112776 000033 000000  MOVB     #33,@(SP)    ;WRITE ERROR NUMBER IN CALL
349 046576 162716 000002      SUB      #2,(SP)      ;MOVE SP TO ERROR RETURN
350 046602 004736      JSR      PC,@(SP)+    ;REPORT ERROR AND RETURN
351 046604 162716 000010      SUB      #10,(SP)     ;MOVE SP TO NO ERROR
352 046610      190$:
353
354      ;REPORT ERROR IF AOE IS SET AND DATA ERRORS ARE NOT ENABLED
355 046610 012746 177777      MOV      #-1,-(SP)    ;ASSUME DATA ERRORS ARE ENABLED
356 046614 032737 001000 050536  BIT      #AOE,500$    ;ARE DATA ERRORS EANBLED ??
357 046622 001404      BEQ      191$      ;NO !!
358 046624 032737 002000 001350  BIT      #LBT,RMDSI    ;IS LBT ALSO SET ??
359 046632 001002      BNE      195$      ;YES !!
360 046634 042716 001000 191$: BIC      #AOE,(SP)    ;DISABLE AOE
361 046640 013737 001352 001142 195$: MOV      RMER1I,$BDDAT  ;GET RECEIVED STATUS
362 046646 042637 001142      BIC      (SP)+,$BDDAT  ;CLEAR AOE IF ENABLED
363 046652 001412      BEQ      200$      ;BRANCH IF AOE IS OK
364 046654 062716 000004      ADD      #4,(SP)      ;MOVE SP TO USERS ERROR CALL
365 046660 112776 000020 000000  MOVB     #20,@(SP)    ;WRITE ERROR NUMBER
366 046666 162716 000002      SUB      #2,(SP)      ;MOVE SP TO ERROR RETURN
367 046672 004736      JSR      PC,@(SP)+    ;REPORT ERROR AND RETURN
368 046674 162716 000010      SUB      #10,(SP)     ;MOVE SP TO NO ERROR
369 046700      200$:
370
371      ;BIT 07 (HCE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
372      ;HEADER ERRORS, I.E.,
373      ;      RMER1 - HCRC,HCE,FER
374      ;      RMER2 - BSE
375
376      ;RESET THE ENABLING BIT (HCE) IF HEADER COMPARE INHIBIT IS SET
377 046700 032737 002000 001370  BIT      #HCI,RMOFI    ;IS HCI SET ??
378 046706 001403      BEQ      201$      ;NO !
379 046710 042737 000200 050536  BIC      #HCE,500$    ;YES - DISABLE ALL HEADER ERRORS
380 046716      201$:
381
382      ;REPORT AN ERROR IF HCRC IS SET AND HEADER ERRORS ARE NOT ENABLED
383 046716 012746 177777      MOV      #-1,-(SP)    ;ASSUME ERRORS ENABLED
384 046722 032737 000200 050536  BIT      #HCE,500$    ;ARE HEADER ERRORS ENABLED ??
385 046730 001002      BNE      205$      ;YES !!
386 046732 042716 000400      BIC      #HCRC,(SP)    ;DISABLE HCRC
387 046736 013737 001352 001142 205$: MOV      RMER1I,$BDDAT  ;GET RECEIVED STATUS
388 046744 042637 001142      BIC      (SP)+,$BDDAT  ;RESET HCRC IF ENABLED
389 046750 001412      BEQ      210$      ;BRANCH IF HCRC IS OK
390 046752 062716 000004      ADD      #4,(SP)      ;MOVE SP TO USERS ERROR CALL
391 046756 112776 000035 000000  MOVB     #35,@(SP)    ;WRITE ERROR NUMBER IN CALL
392 046764 162716 000002      SUB      #2,(SP)      ;MOVE SP TO ERROR RETURN
393 046770 004736      JSR      PC,@(SP)+    ;REPORT ERROR AND RETURN
394 046772 162716 000010      SUB      #10,(SP)     ;MOVE SP TO NO ERROR
395 046776      210$:
396
397      ;REPORT ERROR IF HCE IS SET AND HEADER ERRORS ARE NOT ENABLED
398 046776 012746 177777      MOV      #-1,-(SP)    ;ASSUME ERRORS ENABLED
399 047002 032737 000200 050536  BIT      #HCE,500$    ;ARE ERRORS ENABLED ??

```

```

400 047010 001002      BNE      215$      ;YES !!
401 047012 042716 C00200 BIC      #HCE,(SP)      ;DISABLE HCE
402 047016 013737 001352 001142 215$ MOV      RMER1I,$BDDAT      ;GET RECEIVED STATUS
403 047024 042637 001142      BIC      (SP)+,$BDDAT      ;CLEAR HCE IF ENABLED
404 047030 001412      BEQ      220$      ;BRANCH IF HCE IS OK
405 047032 062716 000004      ADD      #4,(SP)      ;MOVE SP TO USERS ERROR CALL
406 047036 112776 000036 000000      MOVB      #36,@(SP)      ;WRITE ERROR NUMBER IN CALL
407 047044 162716 000002      SUB      #2,(SP)      ;MOVE SP TO ERROR RETURN
408 047050 004736      JSR      PC,@(SP)+      ;REPORT ERROR AND RETURN
409 047052 162716 000010      SUB      #10,(SP)      ;MOVE SP TO NO ERROR
410 047056      220$:
411
412      ;REPORT ERROR IF FER IS SET AND HEADER ERRORS ARE NOT ENABLED
413 047056 012746 177777      MOV      #-1,-(SP)      ;ASSUME FER IS ENABLED
414 047062 032737 000200 050536      BIT      #HCE,500$      ;ARE HEADER ERRORS ENABLED ??
415 047070 001002      BNE      225$      ;YES !!
416 047072 042716 000020      BIC      #FER,(SP)      ;DISABLE FER
417 047076 013737 001352 001142 225$: MOV      RMER1I,$BDDAT      ;GET RECEIVED STATUS
418 047104 042637 001142      BIC      (SP)+,$BDDAT      ;RESET FER IF ENABLED
419 047110 001412      BEQ      230$      ;BRANCH IF FER OK
420 047112 062716 000004      ADD      #4,(SP)      ;MOVE SP TO USERS ERROR CALL
421 047116 112776 000037 000000      MOVB      #37,@(SP)      ;WRITE ERROR NUMBER IN CALL
422 047124 162716 000002      SUB      #2,(SP)      ;MOVE SP TO ERROR RETURN
423 047130 004736      JSR      PC,@(SP)+      ;REPORT ERROR AND RETURN
424 047132 162716 000010      SUB      #10,(SP)      ;MOVE SP TO NO ERROR
425 047136      230$:
426
427      ;REPORT ERROR IF BSE IS SET AND HEADER ERRORS ARE NOT EVABLED
428 047136 012746 177777      MOV      #-1,-(SP)      ;ASSUME ERRORS ENABLED
429 047142 032737 000200 050536      BIT      #HCE,500$      ;ARE THEY ENABLED ??
430 047150 001002      BNE      235$      ;YES !!
431 047152 042716 100000      BIC      #BSE,(SP)      ;DISABLE BSE
432 047156 013737 001400 001142 235$: MOV      RMER2I,$BDDAT      ;GET RECEIVED STATUS
433 047164 042637 001142      BIC      (SP)+,$BDDAT      ;CLEAR BSE IF ENABLED
434 047170 001412      BEQ      240$      ;BRANCH IF BSE OK
435 047172 062716 000004      ADD      #4,(SP)      ;MOVE SP TO USERS ERROR CALL
436 047176 112776 000354 000000      MOVB      #354,@(SP)      ;WRITE ERROR NUMBER
437 047204 162716 000002      SUB      #2,(SP)      ;MOVE SP TO ERROR RETURN
438 047210 004736      JSR      PC,@(SP)+      ;REPORT ERROR AND RETURN
439 047212 162716 000010      SUB      #10,(SP)      ;MOVE SP TO NO ERROR
440 047216      240$:
441
442      ;BIT 06 OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR DATA
443      ;FIELD ERRORS, I.E.,
444      ;      RMCS2 - MDPE
445      ;      RMER1 - DCK,ECH
446      ;NOTE:
447      ;      ECH CANNOT SET UNLESS IT IS ENABLED AND ECI IS RESET AND
448      ;      DCK IS SET.
449
450      ;REPORT ERROR IF MDPE IS SET AND IS NOT ENABLED
451 047216 012746 177777      MOV      #-1,-(SP)      ;ASSUME ENABLED
452 047222 032737 000100 050536      BIT      #ECH,500$      ;ARE DATA FIELD ERRORS ENABLED ??
453 047230 001002      BNE      245$      ;YES !!
454 047232 042716 000400      BIC      #MDPE,(SP)      ;DISBALE MDPE
455 047236 013737 001346 001142 245$: MOV      RMCS2I,$BDDAT      ;GET RECEIVED STATUS
456 047244 042637 001142      BIC      (SP)+,$BDDAT      ;CLEAR MDPE IF ENABLED

```

```

457 047250 001412          BEQ      250$          ;BRANCH IF MDPE OK
458 047252 062716 000004          ADD      #4,(SP)          ;MOVE SP TO USERS ERROR CALL
459 047256 112776 000027 000000          MOVVB #27,@(SP)          ;WRITE ERROR NUMBER IN CALL
460 047264 162716 000002          SUB      #2,(SP)          ;MOVE SP TO ERROR RETURN
461 047270 004736          JSR      PC,@(SP)+          ;REPORT ERROR AND RETURN
462 047272 162716 000010          SUB      #10,(SP)          ;MOVE SP TO NO ERROR
463 047276          250$:
464
465          ;REPORT ERROR IF DCK IS SET AND DATA FIELD ERRORS ARE NOT ENABLED
466 047276 012746 177777          MOV      #-1,-(SP)          ;ASSUME ENABLED
467 047302 032737 000100 050536          BIT      #ECH,500$          ;ARE THEY ENABLED ??
468 047310 001002          BNE      255$          ;YES !!
469 047312 042716 100000          BIC      #DCK,(SP)          ;DISABLE DCK
470 047316 013737 001352 001142 255$: MOV      RMER11,$BDDAT          ;GET RECEIVED STATUS
471 047324 042637 001142          BIC      (SP)+,$BDDAT          ;CLEAR DCK IF ENABLED
472 047330 001412          BEQ      260$          ;BRANCH IF DCK IS OK
473 047332 062716 000004          ADD      #4,(SP)          ;MOVE SP TO USERS ERROR CALL
474 047336 112776 000030 000000          MOVVB #30,@(SP)          ;WRITE ERROR NUMBER IN CALL
475 047344 162716 000002          SUB      #2,(SP)          ;MOVE SP TO ERROR RETURN
476 047350 004736          JSR      PC,@(SP)+          ;REPORT ERROR AND RETURN
477 047352 162716 000010          SUB      #10,(SP)          ;MOVE SP TO NO ERROR
478 047356          260$:
479
480          ;REPORT ERROR IF ECH IS SET AND,
481          ;DATA FIELD ERRORS ARE NOT ENABLED, OR
482          ;ECI IS SET, OR
483          ;DCK IS NOT SET.
484 047356 012746 177777          MOV      #-1,-(SP)          ;ASSUME ENABLED
485 047362 032737 000100 050536          BIT      #ECH,500$          ;ARE ERRORS ENABLED ??
486 047370 001410          BEQ      265$          ;NO !!
487 047372 032737 004000 001370          BIT      #ECI,RMOF1          ;IS ECI SET ??
488 047400 001004          BNE      265$          ;YES !!
489 047402 032737 100000 001352          BIT      #DCK,RMER11          ;IS DCK ALSO SET ??
490 047410 001002          BNE      270$          ;YES !!
491 047412 042716 000100 265$: BIC      #ECH,(SP)          ;DISABLE ECH
492 047416 013737 001352 001142 270$: MOV      RMER11,$BDDAT          ;GET RECEIVED STATUS
493 047424 042637 001142          BIC      (SP)+,$BDDAT          ;CLEAR ECH IF ENABLED
494 047430 001412          BEQ      275$          ;BRANCH IF ECH IS OK
495 047432 062716 000004          ADD      #4,(SP)          ;MOVE SP TO USERS ERROR CALL
496 047436 112776 000031 000000          MOVVB #31,@(SP)          ;WRITE ERROR NUMBER IN CALL
497 047444 162716 000002          SUB      #2,(SP)          ;MOVE SP TO ERROR RETURN
498 047450 004736          JSR      PC,@(SP)+          ;REPORT ERROR AND RETURN
499 047452 162716 000010          SUB      #10,(SP)          ;MOVE SP TO NO ERROR
500 047456          275$:

```

```

1
2
3
4
5 047436 022737 000030 050544      CMP      #SEARCH,515$      ;WAS DATA TRANSFERRED ?
6 047464 103402      BLO      280$      ;BR IF YES
7 047466 000137 050510      JMP      355$      ;NO - EXIT
8
9      ;REPORT ERROR IF RMWC NOT ZERO AND TRE IS ZERO
10 047472 013737 001340 001142 280$: MOV      RMWCI,$BDDAT      ;WORD COUNT ZERO??
11 047500 001421      BEQ      285$      ;YES
12 047502 032737 040000 001336      BIT      #TRE,RMCS1I      ;TRANSFER ERROR DETECTED??
13 047510 001015      BNE      285$      ;YES.!!
14 047512 062716 000004      ADD      #4,(SP)
15 047516 112776 000015 000000      MOV      #15,@(SP)      ;ERROR NUMBER
16 047524 005037 001140      CLR      $GDDAT      ;GOOD DATA FOR TYPEOUT
17 047530 162716 000002      SUB      #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
18 047534 004736      JSR      PC,@(SP)+      ;REPORT WORD COUNT NOT ZERO
19 047536 162716 000010      SUB      #10,(SP)      ;RESTORE (SP)
20 047542 000240      NOP
21
22      ;REPORT ERROR IF RMBA IS NOT CORRECT
23 047544 013737 001340 001140 285$: MOV      RMWCI,$GDDAT      ;GET WORD COUNT AT END OF TRANSFER AND
24 047552 163737 001414 001140      SUB      RMWCO,$GDDAT      ;SUBTRACT STARTING WORD COUNT.
25 047560 006337 001140      ASL      $GDDAT      ;* 2
26 047564 063737 001416 001140      ADD      RMBA0,$GDDAT      ;ADD STARTING BUS ADDRESS
27
28 047572 032737 000010 001346      BIT      #BAI,RMCS2I      ;WAS BUS ADDRESS INHIBIT (BAI) SET ??
29 047600 001403      BEQ      290$      ;NO !!
30 047602 013737 001416 001140      MOV      RMBA0,$GDDAT      ;ADDRESS SHOULD NOT HAVE CHANGED
31
32 047610 023737 001140 001342 290$: CMP      $GDDAT,RMBAI      ;BUS ADDRESS OK??
33 047616 001416      BEQ      295$      ;YES!!
34 047620 013737 001342 001142      MOV      RMBAI,$BDDAT      ;BAD DATA FOR TYPEOUT
35 047626 062716 000004      ADD      #4,(SP)
36 047632 112776 000016 000000      MOV      #16,@(SP)      ;ERROR NUMBER
37 047640 162716 000002      SUB      #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
38 047644 004736      JSR      PC,@(SP)+      ;REPORT UNEXPECTED ADDRESS
39 047646 162716 000010      SUB      #10,(SP)      ;RESTORE (SP)
40 047652 000240      NOP
41
42      ;COMPUTE NUMBER OF SECTORS TRANSFERRED FROM WORD COUNT
43 047654 005046 295$: CLR      -(SP)      ;NUMBER OF SECTORS TRANSFERRED
44 047656 013746 001340      MOV      RMWCI,-(SP)      ;GET WORD COUNT AT END OF TRANSFER AND
45 047662 163716 001414      SUB      RMWCO,(SP)      ;SUBTRACT STARTING WORD COUNT.
46
47 047666 012746 000400      MOV      #256,-(SP)      ;ASSUME 256. WORDS PER SECTOR
48 047672 032737 000002 001412      BIT      #BIT1,RMCS10      ;HEADER & DATA COMMAND ??
49 047700 001402      BEQ      300$      ;NO !!
50 047702 062716 000002      ADD      #2,(SP)      ;CHANGE TO 258. WORDS PER SECTOR
51
52 047706 005266 000004 300$: INC      4(SP)      ;INCREMENT SECTOR COUNT
53 047712 161666 000002      SUB      (SP),2(SP)      ;SUBTRACT ONE SECTOR'S WORTH
54 047716 003373      BGT      300$      ;CONTINUE IF NOT DONE
55 047720 022626      CMP      (SP)+,(SP)+      ;RESTORE STACK
56
57      ;COMPUTE EXPECTED SECTOR, TRACK AND CYLINDER ADDRESS FROM

```

```

58                                     ;NUMBER OF SECTORS
59 047722 013737 001446 050536      MOV     RMDCO,500$      ;STORE ORIGINAL CYLINDER
60 047730 013737 001420 050540      MOV     RMDAO,505$      ;STORE ORIGINAL TRACK
61 047736 013737 001420 050542      MOV     RMDAO,510$      ;STORE ORIGINAL SECTOR
62 047744 013737 001334 050546      MOV     LSTRK,520$      ;STORE LAST TRACK,
63 047752 000337 050546              SWAB     520$          ;GET TRACK ADDRESS TO LO BYTE AND
64 047756 005237 050546              INC      520$          ;INCREMENT TO GET TOTL # OF TRACKS.
65
66 047762 042737 000377 050540      BIC      #^C<TADMSK>,505$      ;SAVE TRACK ADDRESS BITS AND
67 047770 000337 050540              SWAB     505$          ;SWAP TRACK ADDRESS TO LOW BYTE.
68 047774 042737 177400 050542      BIC      #^C<SADMSK>,510$      ;SAVE SECTOR ADDRESS BITS
69 050002 062637 050542              ADD      (SP)+,510$
70
71 050006 023727 050542 000040 310$: CMP     510$,#32.        ;SECTOR OVEFLOWED??
72 050014 103406                    BLO      315$            ;NO..
73 050016 005237 050540              INC      505$          ;INCREMENT TRACK
74 050022 162737 000040 050542      SUB      #32.,510$      ;ADJUST SECTOR
75 050030 000766                    BR       310$            ;TRY AGAIN
76
77 050032 023737 050540 050546 315$: CMP     505$,520$      ;TRACK OVEFLOWED??
78 050040 103407                    BLO      320$            ;NO..
79 050042 005237 050536              INC      500$          ;INCREMENT CYLINDER
80 050046 163737 050546 050540      SUB      520$,505$      ;ADJUST TRACK
81 050054 000766                    BR       315$            ;TRY AGAIN
82 050056 000240                    NOP
83
84                                     ;REPORT ERROR IF 'LBT' IS NOT CORRECT
85 050060 005037 001140 001466 320$: CLR      $GDDAT          ;SET GOOD DATA FOR LBT - 0
86 050060 023727 050536 001466      CMP      500$,#822.        ;SHOULD LBT BE SET??
87 050064 101407                    BLOS     325$            ;NO!!
88 050072 032737 002000 001352      BIT      #IAE,RMER11      ;WAS IAE SET ??
89 050074 001003                    BNE      325$            ;YES - LBT SHOULD NOT BE SET
90 050102 012737 002000 001140      MOV      #LBT,$GDDAT      ;SET GOOD DATA FOR LBT - 1
91 050104 013737 001350 001142 325$: MOV      RMDSI,$BDDAT      ;BAD DATA FOR TYPEOUT
92 050112 042737 175777 001142      BIC      #^CLBT,$BDDAT
93 050120 023737 001140 001142      CMP      $GDDAT,$BDDAT      ;IS LBT CORRECT??
94 050126 001413                    BEQ      330$            ;YES!!
95 050134 062716 000004              ADD      #4,(SP)
96 050136 112776 000017 000000      MOVB    #17,a(SP)      ;ERROR NUMBER
97 050142 162716 000002              SUB      #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
98 050150 004736                    JSR      PC,a(SP)+      ;REPORT LBT IS WRONG
99 050154 162716 000010              SUB      #10,(SP)      ;RESTORE (SP)
100 050156 000240                    NOP
101
102                                     ;REPORT ERROR IF 'AOE' IS INCORRECT
103 050164 005037 001140 001352 330$: CLR      $GDDAT          ;SET FOR AOE = 0
104 050170 032737 002000 001352      BIT      #IAE,RMER11      ;WAS 'IAE' DETECTED??
105 050176 001031                    BNE      340$            ;YES-'AOE' SHOULD BE ZERO
106 050200 023727 050536 001466      CMP      500$,#822.        ;SHOULD AOE BE SET??
107 050206 101425                    BLOS     340$            ;NO!!
108 050210 005737 050540              TST      505$          ;MAYBE
109 050214 001012                    BNE      335$            ;YES
110 050216 005737 050542              TST      510$          ;YES !!
111 050222 001007                    BNE      335$            ;YES !!
112 050224 032737 000010 050544      BIT      #F2,515$      ;WAS THIS READ OR WRITE CHECK ??
113 050232 001413                    BEQ      340$            ;NO ..

```

```

115 050234 005737 001340      TST      RMWCI      ;WAS ALL DATA TRANSFERRED ??
116 050240 001410      BEQ      340$      ;YES !!
117 050242 012737 001000 001140 335$: MOV      #AOE,$GDDAT ;SET FOR AOE = 1
118 050250 005037 050540      CLR      505$      ;CLEAR EXPECTED TRACK
119 050254 012737 000001 050542      MOV      #1,510$     ;EXPECT SECTOR = 1
120 050262 013737 001352 001142 340$: MOV      RMER1I,$BDDAT ;BAD DATA FOR TYPEOUT
121 050270 042737 176777 001142      BIC      #^CAOE,$BDDAT
122 050276 023737 001140 001142      CMP      $GDDAT,$BDDAT ;IS AOE CORRECTY??
123 050304 001413      BEQ      345$      ;YES!!
124 050306 062716 000004      ADD      #4,(SP)
125 050312 112776 000020 000000      MOV      #20,@(SP) ;ERROR NUMBER
126 050320 162716 000002      SUB      #2,(SP) ;MOVE SP TO RETURN FOR ERROR
127 050324 004736      JSR      PC,@(SP)+ ;REPORT AOE IS WRONG
128 050326 162716 000010      SUB      #10,(SP) ;RESTORE (SP)
129 050332 000240      NOP
130
131      ;REPORT ERROR IF RMDA IS NOT CORRECT
132 050334 032737 002000 001352 345$: BIT      #IAE,RMER1I ;WAS THERE AN IAE ERROR ??
133 050342 001062      BNE      355$      ;YES - DONT CHECK RMDA,RMDC
134 050344 013737 050540 001140      MOV      505$,$GDDAT ;SETUP EXPECTED DISK ADDRESS
135 050352 000337 001140      SWAB      $GDDAT
136 050356 113737 050542 001140      MOV      510$,$GDDAT
137 050364 013737 001344 001142      MOV      RMDAI,$BDDAT ;SETUP RECEIVED DISK ADDRESS
138 050372 023737 001140 001142      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED & RECEIVED
139 050400 001413      BEQ      350$      ;BRANCH IF EQUAL
140 050402 062716 000004      ADD      #4,(SP)
141 050406 112776 000021 000000      MOV      #21,@(SP) ;ERROR NUMBER
142 050414 162716 000002      SUB      #2,(SP) ;MOVE SP TO RETURN FOR ERROR
143 050420 004736      JSR      PC,@(SP)+ ;REPORT BAD DISK ADDRESS
144 050422 162716 000010      SUB      #10,(SP) ;RESTORE (SP)
145 050426 000240      NOP
146
147      ;REPORT ERROR IF RMDC IS INCORRECT
148 050430 013737 050536 001140 350$: MOV      500$,$GDDAT ;SETUP EXPECTED CYLINDER
149 050436 042737 176000 001140      BIC      #^C1777,$GDDAT
150 050444 013737 001372 001142      MOV      RMDCI,$BDDAT ;SETUP RECEIVED CYLINDER
151 050452 023737 001140 001142      CMP      $GDDAT,$BDDAT ;COMPARE CYLINDERS
152 050460 001413      BEQ      355$      ;BRANCH IF EQUAL
153 050462 062716 000004      ADD      #4,(SP)
154 050466 112776 000022 000000      MOV      #22,@(SP) ;ERROR NUMBER
155 050474 162716 000002      SUB      #2,(SP) ;MOVE SP TO RETURN FOR ERROR
156 050500 004736      JSR      PC,@(SP)+ ;REPORT BAD CYLINDER
157 050502 162716 000010      SUB      #10,(SP) ;RESTORE (SP)
158 050506 000240      NOP
159
160 050510 062716 000004      355$: ADD      #4,(SP) ;MOVE (SP) TO ERROR CALL
161 050514 105776 000000      TSTB      @(SP) ;WAS ERROR FOUND??
162 050520 001403      BEQ      360$
163 050522 062716 000004      ADD      #4,(SP) ;MOVE (SP) TO ERROR RETURN
164 050526 000402      BR      365$
165 050530 162716 000004      360$: SUB      #4,(SP) ;MOVE (SP) TO NO ERROR RETURN
166 050534 000207      365$: RTS      PC
167
168 050536 000000      500$: .WORD      0 ;CYLINDER
169 050540 000000      505$: .WORD      0 ;TRACK
170 050542 000000      510$: .WORD      0 ;SECTOR
171 050544 000000      515$: .WORD      0 ;FUNCTION CODE

```


050546 000000

5208: .WORD 0

;TOTAL # OF TRACKS = LAST TRACK +1

;

```

1      .SBTTL  COMPOSITE ERROR CHECK SUBROUTINE
2
3      ;THIS SUBROUTINE CHECKS THE STORED CONTENTS OF RMER1 AND
4      ;RMER2 AFTER MASKING EACH REGISTER WORD WITH THE USER'S STATUS
5      ;MASKS AND REPORTS AN ERROR IF ANY BITS ARE LEFT ON AFTER
6      ;THE MASKS ARE APPLIED.
7
8      ;CALL:
9      ;(1)  JSR      PC, CMPERRSTS
10             .WORD      MASK FOR ERROR REGISTER 1
11             .WORD      MASK FOR ERROR REGISTER 2
12             BR        ???      RETURN HERE IF NO ERROR
13             NOP        RETURN HERE TO REPORT AN ERROR
14             ERROR      ERROR NUMBER DEFINED BY SUB
15             JSR      PC,@(SP)+  GO BACK TO SUB FOR MORE ERROR CHECKS
16             ???      RETURN HERE IF NO MORE ERRORS
17
18      ;NOTE: BITS TO BE MASKED SHOULD BE ONE; BITS TO BE TESTED SHOULD
19      ;BE ZERO
20
21      CMPERRSTS:
22
23      ;MASK AND STORE THE CONTENTS OF RMER1 AND RMER2
24      050550 013737 001352 001176      MOV      RMER1I,$TMP1      ;STORE RMER1 AT TEMP STORAGE
25      050556 047637 000000 001176      BIC      @(SP),$TMP1      ;MASK RMER1
26      050564 062716 000002             ADD      #2,(SP)      ;MOVE SP TO NEXT MASK
27      050570 013737 001400 001200      MOV      RMER2I,$TMP2      ;STORE RMER2 AT TEMP STORAGE
28      050576 047637 000000 001200      BIC      @(SP),$TMP2      ;MASK RMER2
29
30
31      ;CLEAR USER'S ERROR CALL
32      050604 062716 000006             ADD      #6,(SP)      ;MOVE SP TO USER'S ERROR CALL
33      050610 105076 000000             CLRB     @(SP)      ;CLEAR ERROR NUMBER
34      050614 162716 000004             SUB      #4,(SP)      ;LEAVE SP AT NO ERROR RETURN
35
36      ;SEE IF THERE WERE ANY ERRORS IN RMER1, I.E., $TMP1
37      050620 005737 001176             TST      $TMP1      ;ANY ERRORS TO REPORT??
38      050624 001420             BEQ      5$      ;NO !!
39      050626 013737 001176 001142      MOV      $TMP1,$BDDAT      ;RECEIVED STATUS FOR TYPEOUT
40      050634 005037 001140             CLR      $GDDAT      ;EXPECTED STATUS FOR TYPEOUT
41      050640 062716 000004             ADD      #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
42      050644 112776 000066 000000      MOVVB   #66,@(SP)      ;CORRECTABLE DATA CHECK ERROR #
43      050652 162716 000002             SUB      #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
44      050656 004736             JSR      PC,@(SP)+  ;REPORT ERROR VIA USER
45      050660 162716 000010             SUB      #10,(SP)     ;MOVE SP BACK TO BRANCH
46      050664 000240             NOP
47      050666      5$:
48
49
50      ;SEE IF THERE ARE ANY ERRORS TO REPORT IN RMER2 ($TMP2)
51      050666 005737 001200             TST      $TMP2      ;ANY ERRORS IN RMER2?
52      050672 001420             BEQ      10$      ;NO!!
53
54      050674 013737 001200 001142      MOV      $TMP2,$BDDAT      ;RECEIVED STATUS FOR TYPEOUT
55      050702 005037 001140             CLR      $GDDAT      ;EXPECTED STATUS FOR TYPEOUT
56      050706 062716 000004             ADD      #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
57      050712 112776 000067 000000      MOVVB   #67,@(SP)      ;WRITE ERROR NUMBER IN USER'S CALL

```



```

58 050720 162716 000002      SUB      #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
59 050724 004736              JSR      PC,@(SP)+    ;REPORT ERROR VIA USER
60 050726 162716 000010      SUB      #10,(SP)     ;MOVE SP TO NO ERROR RETURN
61 050732 000240              NOP
62 050734              10$:
63
64              ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
65 050734 062716 000004      ADD      #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
66 050740 105776 000000      TSTB     @ (SP)      ;WAS THERE AN ERROR CALLED??
67 050744 001403              BEQ      20$         ;NO!!
68 050746 062716 000004      ADD      #4,(SP)      ;YES - MOVE SP TO ERROR RETURN
69 050752 000402              BR       30$
70 050754 162716 000004      20$:    SUB      #4,(SP)  ;MOVE SP TO NO ERROR RETURN
71 050760 000207      30$:    RTS      PL          ;RETURN TO USER

```

.SBTTL DEVICE SELECT SUBROUTINE

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

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

DEVSEL:

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

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

;SELECT DEVICE AND VERIFY THAT DEVICE IS AVAILABLE
 MOVB (R1),RMCS2(R0) ;WRITE UNIT SELECT BITS
 MOV RMCS1(R0),\$TMP1 ;GET 'DVA' STATUS
 MOV RMCS2(R0),\$TMP0 ;GET 'NED' STATUS

BIT #NED,\$TMP0 ;IS DEVICE NONEXISTENT ?
 BEQ 10\$;NO..
 ADD #4,10(SP) ;MOVE SP TO USERS ERROR CALL
 MOVB #111,@10(SP) ;WRITE ERROR NUMBER
 BR 30\$

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

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

;RESTORE USERS DATA AND RETURN TO ADDRESS ON STACK
 35\$:

MOV (SP)+,R1 ;:POP STACK INTO R1
 MOV (SP)+,R0 ;:POP STACK INTO R0
 MOV (SP)+,ERRVEC+2 ;:POP STACK INTO ERRVEC+2

12 050762
 15 050762 062716 000004
 16 050766 105076 000000
 17 050772 162716 000004
 20 050776 013746 000004
 051002 013746 000006
 051008 010046
 051010 010146
 21 051012 012737 051132 000004
 22 051020 012737 000300 000006
 23 051026 013700 001276
 24 051032 013701 001466
 27 051036 111160 000010
 28 051042 016037 000000 001176
 29 051050 016037 000010 001174
 31 051056 032737 010000 001174
 32 051064 001407
 33 051066 062766 000004 000010
 34 051074 112776 000111 000010
 35 051102 000422
 37 051104 032737 004000 001176 10\$:
 38 051112 001021
 39 051114 062766 000004 000010
 40 051122 112776 000112 000010
 41 051130 000407
 44 051132 022626
 45 051134 062766 000004 000010
 46 051142 112776 000113 000010
 47 051150 162766 000002 000010
 51 051156
 051156 012601
 051160 012600
 051162 012637 000006

051166 012637 000004
051172 000207

MOV
RTS

(SP)+,ERRVEC
PC

::POP STACK INTO ERRVEC
::EXIT

```

1  .SBTTL  SEEK STATUS CHECK SUBROUTINE
2
3  ;THIS SUBROUTINE VERIFIES THE RESULTS OF SEEK TESTS USING STATUS
4  ;STORED IN THE GET BUFFER AND TEST PARAMETERS STORED IN THE PUT BUFFER.
5
6  ;THE SUBROUTINE RETURNS TO THE CALLING ROUTINE IF AN ERROR IS DETECTED
7  ;AFTER HAVING LOADED THE APPROPRIATE ERROR NUMBER IN THE 'ERROR' TRAP
8  ;OF THE CALLING ROUTINE.  SEEK STATUS IS CHECKED AS FOLLOWS:
9
10 ;CALL:
11 ;(1)  JSR    PC,SEKSTS
12 ;      BR     ???
13 ;      NOP
14 ;      ERROR
15 ;      JSR    PC,@(SP)+
16 ;      ???
17
18 ;SEKSTS:
19 051174
20
21 ;CLEAR USER'S ERROR CALL
22 051174 000240      NOP
23 051176 062716 000004  ADD     #4,(SP)      ;MOVE (SP) TO ERROR CALL
24 051202 105076 000000  CLRB    @ (SP)      ;CLEAR ERROR NUMBER
25 051206 162716 000004  SUB     #4,(SP)      ;MOVE (SP) TO NO ERROR RETURN
26 051212 005037 052432  CLR     300$      ;CLEAR ERROR FLAGS
27
28 ;TEST FOR MASSBUS CONTROL BUS PARITY ERROR WHEN WRITING
29 ;LOCAL REGISTERS, I.E., 'PAR' 1 AND 'DPE' - 0
30 051216 032737 000010 001352  BIT     #PAR,RMER1I  ;WAS PARITY ERROR DETECTED??
31 051224 001424 000000 001352  BEQ     1$          ;NO!!
32 051226 032737 000010 001400  BIT     #DPE,RMER2I  ;WAS IT DUE TO CONTROL BUS??
33 051234 001020 000000 001400  BNE     1$          ;NOT SURE!!
34
35 ;REPORT REGISTER PARITY ERROR VIA USER'S ERROR CALL
36 051236 005037 001140 001142  CLR     $GDDAT      ;EXPECTED STATUS
37 051242 013737 001352 001142  MOV     RMER1I,$BDDAT  ;RECEIVED STATUS
38 051250 062716 000004 000000  ADD     #4,(SP)      ;MOVE STACK TO USER'S ERROR
39 051254 112776 000050 000000  MOV     #50,@(SP)    ;ERROR #50
40 051262 162716 000002 000000  SUB     #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
41 051266 004736 000010 000000  JSR     PC,@(SP)+
42 051270 162716 000010 000000  SUB     #10,(SP)     ;RESTORE STACK
43 051274 000437 000000 000000  BR      3$          ;IAE SHOULD BE ZERO
44
45 ;DETERMINE THE VALUE OF 'IAE' STATUS BASED ON TRACK, SECTOR AND CYLINDER
46 ;ALSO, SET 'SKI' IF CYLINDER ADDRESS IS TOO LARGE.
47 051276 012737 002000 001140 1$: MOV     #IAE,$GDDAT  ;SETUP FOR IAE - 1
48 051304 052737 040000 052432  BIS     #SKI,300$    ;SETUP FOR SKI = 1
49 051312 023727 001446 001466  CMP     RMDCO,#822.  ;GREATER THAN LAST CYLINDER ?
50 051320 101025 000000 001466  BHI     3$          ;YES - CYLINDER IS INVALID
51 051322 042737 040000 052432  BIC     #SKI,300$    ;CLEAR SKI ERROR FLAG
52
53 051330 123737 001421 001335  CMP     RMDAO+1,LSTRK+1 ;GREATER THAN LAST TRACK ?
54 051336 101016 000000 001335  BHI     3$          ;YES - TRACK IS INVALID
55
56 051340 123727 001420 000035  CMP     RMDAO,#29.    ;SECTOR > 29. ?
57 051346 101410 000000 000035  BLOS    2$          ;BR IF NO

```

```

58 051350 032737 010000 001444      BIT    #FMT16,RMOFO    ;18 BIT FORMAT ?
59 051356 001406                BEQ     3$              ;YES - SECTOR IS INVALID FOR 18 BIT MODE
60 051360 123727 001420 000037      CMPB   RMDAO,#31.    ;SECTOR > 31. ?
61 051366 101002                BHI     3$              ;YES - SECTOR IS INVALID
62
63 051370 005037 001140      2$:      CLR     $GDDAT        ;'IAE' SHOULD = 0
64
65                                ;COMPARE EXPECTED AND RECIEVED 'IAE' STATUS
66 051374 013737 001352 001142      3$:      MOV     RMFR1I,$BDDAT    ;IS IAE OK??
67 051402 042737 175777 001142      BIC     #^CIAE,$BDDAT    ;SAVE IAE BIT FOR COMPARE
68 051410 023737 001140 001142      CMP     $GDDAT,$BDDAT    ;CORRECT 'IAE' STATUS ?
69 051416 001004                BNE     35$             ;BR IF NO
70 051420 042737 040000 052432      BIC     #SKI,300$      ;CLEAR SKI FLAG
71 051426 000413                BR      5$              ;GO CHECK NEXT ERROR
72 051430
73                                35$:
74 051430 062716 000004      ;REPORT INCORRECT 'IAE' STATUS VIA USER'S ERROR CALL
75 051434 112776 000051 000000      ADD     #4,(SP)
76 051442 162716 000002      MOV     #51,@(SP)      ;ERROR 51
77 051446 004736                SUB     #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
78 051450 162716 000010      JSR     PC,@(SP)+      ;REPORT INCORRECT IAE
79 051454 000240                SUB     #10,(SP)      ;RESTORE (SP)
80 051456
81                                5$:
82                                ;REPORT ANY IVC ERROR AS
83                                ; IVC ERROR WITH VOLUME VALID ZERO
84                                ; ERRONEOUS IVC ERROR, VOLUME VALID IS SET
85 051456 032737 010000 001400      BIT     #IVC,RMER2I    ;IVC ERROR??
86 051464 001427                BEQ     52$             ;NO!!
87 051466 005037 001140      CLR     $GDDAT        ;EXPECTED STATUS
88 051472 013737 001400 001142      MOV     RMER2I,$BDDAT    ;RECEIVED STATUS
89 051500 062716 000004      ADD     #4,(SP)      ;MOVE SP TO USER'S ERROR
90 051504 112776 000060 000000      MOV     #60,@(SP)      ;ERROR 60 IF VV - 0
91 051512 032737 000100 001350      BIT     #VV,RMDSI
92 051520 001403                BEQ     51$             ;ERROR 61 IF VV - 1
93 051522 112776 000061 000000      MOV     #61,@(SP)      ;MOVE SP TO RETURN FOR ERROR
94 051530 162716 000002      51$:      SUB     #2,(SP)      ;REPORT ERROR VIA USER
95 051534 004736                JSR     PC,@(SP)+      ;RESTORE SP
96 051536 162716 000010      SUB     #10,(SP)
97 051542 000240                NOP
98
99 051544 013737 001400 001142      52$:      MOV     RMER2I,$BDDAT    ;RECEIVED STATUS
100 051552 042737 137777 001142      BIC     #^CSKI,$BDDAT    ;CLEAR DONT CARES
101 051560 013737 052432 001140      MOV     300$,$GDDAT    ;GET EXPECTED SKI STATUS
102 051566 042737 137777 001140      BIC     #^CSKI,$GDDAT    ;CLEAR DONT CARES
103 051574 001417                BEQ     53$             ;BRANCH IF 0 EXPECTED
104
105                                ;REPORT ERROR IF SKI IS NOT SET (IAE WAS NOT DETECTED)
106 051576 032737 040000 001142      BIT     #SKI,$BDDAT    ;WAS SKI DETECTED ??
107 051604 001032                BNE     54$             ;YES !!
108 051606 062716 000004      ADD     #4,(SP)      ;MOVE SP TO USERS ERROR CALL
109 051612 112776 000267 000000      MOV     #267,@(SP)    ;WRITE ERROR NUMBER
110 051620 162716 000002      SUB     #2,(SP)      ;MOVE SP TO ERROR RETURN
111 051624 004736                JSR     PC,@(SP)+      ;REPORT ERROR AND RETURN
112 051626 162716 000010      SUB     #10,(SP)      ;MOVE SP TO NO ERROR
113 051632 000443                BR      6$              ;GO TO NEXT ERROR CHECK
114 051634                                53$:

```

```

115
116
117 051634 032737 040000 001142 ;REPORT ERROR IF SKI IS SET
118 051642 001413 BIT #SKI,$BDDAT ;IS SKI SET ??
119 051644 062716 000004 BEQ 54$ ;NO - SKI IS OK
120 051650 112776 000054 000000 ADD #4,(SP) ;MOVE (SP) TO ERROR
121 051656 162716 000002 MOVB #54,@(SP) ;LOAD ERROR NUMBER
122 051662 004736 JSR PC,@(SP)+ ;MOVE SP TO RETURN FOR ERROR
123 051664 162716 000010 SUB #10,(SP) ;REPORT SEEK ERROR
124 051670 000240 NOP ;RESTORE (SP)
125
126 ;REPORT ANY DEVICE CHECK
127 051672 032737 000200 001400 54$: BIT #DVC,RMER2I ;WAS THERE DVC DURING SEEK??
128 051700 001420 BEQ 6$ ;NO!!
129 051702 005037 001140 CLR $GDDAT ;EXPECTED STATUS
130 051706 013737 001400 001142 MOV RMER2I,$BDDAT ;RECEIVED STATUS
131 051714 062716 000004 ADD #4,(SP)
132 051720 112776 000055 000000 MOVB #55,@(SP) ;ERROR #55
133 051726 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
134 051732 004736 JSR PC,@(SP)+ ;REPORT ERROR VIA USER
135 051734 162716 000010 SUB #10,(SP) ;RESTORE SP
136 051740 000240 NOP
137
138 ;REPORT ANY 'OPI' ERROR AS OPI WITH MOL 0, OR OPI
139 ;BECAUSE ON CYLINDER LATCH DIDN'T RESET
140 051742 032737 020000 001352 6$: BIT #OPI,RMER1I ;'OPI' ERROR??
141 051750 001427 BEQ 8$ ;NO!!
142 051752 005037 001140 CLR $GDDAT ;EXPECTED STATUS
143 051756 013737 001352 001142 MOV RMER1I,$BDDAT ;RECEIVED STATUS
144 051764 062716 000004 ADD #4,(SP) ;MOVE (SP) TO ERROR
145 051770 112776 000052 000000 MOVB #52,@(SP) ;LOAD ERROR NUMBER
146 051776 032737 010000 001350 BIT #MOL,RMDSI ;WAS MEDIUM ON LINE??
147 052004 001403 BEQ 7$ ;NO!!
148 052006 112776 000053 000000 MOVB #53,@(SP) ;YES - CHANGE ERROR NUMBER
149 052014 162716 000002 7$: SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
150 052020 004736 JSR PC,@(SP)+ ;REPORT 'OPI' ERROR
151 052022 162716 000010 SUB #10,(SP) ;RESTORE (SP)
152 052026 000240 NOP
153
154 ;SEE IF 'PIP' = 0, AND 'ATA', 'MOL' AND 'VV' = 1
155 052030 013746 001350 8$: MOV RMDSI,-(SP)
156 052034 042716 047677 BIC #^C<ATA!PIP.MOL!VV>,(SP)
157 052040 022726 110100 CMP #ATA!MOL!VV,(SP)+
158 052044 001002 BNE 9$ ;ERROR IN RMDS
159 052046 000137 052402 JMP 14$ ;RMDS IS OK
160
161 ;REPORT ERROR IF MOL = 0 AND OPI = 0
162 052052 032737 010000 001350 9$: BIT #MOL,RMDSI ;IS MOL RESET??
163 052060 001030 BNE 10$ ;NO - MOL IS SET
164 052062 032737 020000 001352 BIT #OPI,RMER1I ;WAS OPI SET
165 052070 001024 BNE 10$ ;YES - DONT REPORT ERROR
166 052072 013737 001350 001140 MOV RMDSI,$GDDAT ;EXPECTED STATUS
167 052100 052737 010000 001140 BIS #MOL,$GDDAT
168 052106 013737 001350 001142 MOV RMDSI,$BDDAT ;RECEIVED STATUS
169 052114 062716 000004 ADD #4,(SP)
170 052120 112776 000062 000000 MOVB #62,@(SP)
171 052126 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR

```



```

172 052132 004736      JSR      PL,@(SP)+      :REPORT ERROR VIA USER
173 052134 162716 000010 SUB      #10,(SP)
174 052140 000240      NOP
175
176      :REPORT AN ERROR IF 'PIP' IS STIL SET AND SKI NOT SET
177 052142 032737 020000 001350 10$: BIT      #PIP,RMDSI      :IS 'PIP' STILL SET??
178 052150 001430      BEQ      11$      :NO!!
179 052152 032737 040000 001400 BIT      #SKI,RMER2I      :WAS 'SKI' SET??
180 052160 001024      BNE      11$      :YES-DONT REPORT PIP
181 052162 013737 001350 001140 MOV      RMDSI,$GDDAT      :EXPECTED STATUS
182 052170 042737 020000 001142 BIC      #PIP,$BDDAT
183 052176 013737 001350 001142 MOV      RMDSI,$BDDAT      :RECEIVED STATUS
184 052204 062716 000004      ADD      #4,(SP)      :MOVE (SP) TO ERROR
185 052210 112776 000056 000000 MOVB     #56,@(SP)      :LOAD ERROR NUMBER
186 052216 162716 000002      SUB      #2,(SP)      :MOVE SP TO RETURN FOR ERROR
187 052222 004736      JSR      PC,@(SP)+      :REPORT 'PIP' SET AFTER SEEK
188 052224 162716 000010      SUB      #10,(SP)      :RESTORE (SP)
189 052230 000240      NOP
190
191      :REPORT AN ERROR IF 'ATA' IS NOT SET
192 052232 032737 100000 001350 11$: BIT      #ATA,RMDSI      :WAS 'ATA' SET ??
193 052240 001024      BNE      13$      :YES!!
194 052242 013737 001350 001140 MOV      RMDSI,$GDDAT      :EXPECTED STATUS
195 052250 052737 110600 001140 BIS      #ATA!MOL!DPR!DRY,$GDDAT
196 052256 013737 001350 001142 MOV      RMDSI,$BDDAT      :RECEIVED STATUS
197 052264 062716 000004      ADD      #4,(SP)      :MOVE (SP) TO ERROR
198 052270 112776 000057 000000 MOVB     #57,@(SP)      :LOAD ERROR NUMBER
199 052276 162716 000002      SUB      #2,(SP)      :MOVE SP TO RETURN FOR ERROR
200 052302 004736      JSR      PC,@(SP)+      :REPORT ATTENTION NOT SET DURING
201      :SEEK TEST
202 052304 162716 000010      SUB      #10,(SP)      :RESTORE (SP)
203 052310 000240      NOP
204
205      :REPORT ERROR IF VOLUME VALID IS RESET AND IVC IS ZERO
206 052312 032737 000100 001350 13$: BIT      #VV,RMDSI      :IS VV = 0 ??
207 052320 001030      BNE      14$      :NO!!
208 052322 032737 010000 001400 BIT      #IVC,RMER2I      :IS IVC ALSO 0 ??
209 052330 001024      BNE      14$      :NO - IVC IS SET
210 052332 013737 001350 001140 MOV      RMDSI,$GDDAT      :EXPECTED STATUS
211 052340 052737 000100 001140 BIS      #VV,$GDDAT
212 052346 013737 001350 001142 MOV      RMDSI,$BDDAT      :RECEIVED STATUS
213 052354 062716 000004      ADD      #4,(SP)
214 052360 112776 000064 000000 MOVB     #64,@(SP)      :ERROR #64
215 052366 162716 000002      SUB      #2,(SP)      :MOVE SP TO RETURN FOR ERROR
216 052372 004736      JSR      PC,@(SP)+
217 052374 162716 000010      SUB      #10,(SP)
218 052400 000240      NOP
219 052402
220
221      14$:
222      :MODIFY THE RETURN ADDRESS IF AN ERROR WAS DETECTED
223 052402 000240      NOP
224 052404 062716 000004      ADD      #4,(SP)      :MOVE (SP) TO ERROR CALL
225 052410 105776 000000      TSTB     @ (SP)      :WAS ERROR CALLED??
226 052414 001403      BEQ      15$      :NO!!
227 052416 062716 000004      ADD      #4,(SP)      :MOVE TO ERROR RETURN
228 052422 000402      BR       16$
  
```

SEEK STATUS CHECK SUBROUTINE

229 052424 162716 000004
230 052430 000207
231
232 052432 000000

15\$: SUB #4,(SP)
16\$: RTS PC
300\$: .WORD 0

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


```
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12 052434  
052434 010046  
052436 010146  
052440 013746 000004  
052444 013746 000006  
13 052450 012737 052510 000004  
14 052456 012737 000300 000006  
15 052464 013700 001276  
16 052470 012760 000040 000010  
17 052476 013701 001466  
18 052502 111160 000010  
19 052506 000412  
20  
21 052510 022626  
22 052512 062766 000004 000010  
23 052520 112776 000007 000010  
24 052526 162766 000002 000010  
25 052534  
052534 012637 000006  
052540 012637 000004  
052544 012601  
052546 012600  
26 052550 000207
```

```
.SBTTL CONTROLLER CLEAR SUBROUTINE  
:THIS SUBROUTINE CLEARS THE MASSBUS CONTROLLER, MASSBUS ADAPTERS,  
:AND DRIVES, THEN SELECTS THE DRIVE.  
:CALL: JSR PC,CNTCLR  
:      BR ??? RETURN HERE IF NO ERROR FOUND  
:      NOP RETURN HERE IF ANY ERROR FOUND  
:      ERROR SUB DEFINES ERROR NUMBER  
:      ???  
CNTCLR: MOV R0,-(SP) ;;PUSH R0 ON STACK  
MOV R1,-(SP) ;;PUSH R1 ON STACK  
MOV ERRVEC,-(SP) ;;PUSH ERRVEC ON STACK  
MOV ERRVEC+2,-(SP) ;;PUSH ERRVEC+2 ON STACK  
MOV #10$,ERRVEC ;SETUP FOR BUS TIMEOUT  
MOV #PR6,ERRVEC+2  
MOV $BASE,R0 ;R0 = UNIBUS ADDRESS  
MOV #CLR,RMCS2(R0) ;CLEAR MASSBUS  
MOV TSTQUE,R1 ;GET DEVICE UNDER TEST  
MOVB (R1),RMCS2(R0) ;SELECT DEVICE  
BR 20$  
10$: CMP (SP)+,(SP)+ ;ADJUST STACK  
ADD #4,10(SP) ;MOVE SP TO USER'S ERROR CALL  
MOVB #7,210(SP) ;WRITE THE ERROR NUMBER  
SUB #2,10(SP) ;ADJUST SP TO RETURN TO ERROR  
20$: MOV (SP)+,ERRVEC+2 ;;POP STACK INTO ERRVEC+2  
MOV (SP)+,ERRVEC ;;POP STACK INTO ERRVEC  
MOV (SP)+,R1 ;;POP STACK INTO R1  
MOV (SP)+,R0 ;;POP STACK INTO R0  
RTS PC
```

```

1      .SBTTL  CONTROLLER CLEAR STATUS CHECK SUBROUTINE
2
3      ;THIS SUBROUTINE VERIFIES THAT THE SUBSYSTEM IS INITIALIZED BASED ON
4      ;STATUS STORED IN THE GET BUFFER. THIS SUBROUTINE SHOULD ONLY BE
5      ;USED FOLLOWING A CONTROLLER CLEAR OPERATION, I.E., WRITING A 1 IN BIT
6      ;5 OF RMCS2, BECAUSE THE ERROR MESSAGES ARE BASED ON THAT CONDITION.
7
8      ;STATUS PERTINENT TO THE DEVICE IS NOT CHECKED. IN PARTICULAR, THE
9      ;FOLLOWING STATUS BITS ARE NOT CHECKED:
10
11      :
12      :       ATA,ERR,PIP,MOL,WRL,LBT,PGM,VV,OM,UNS,SKI,DVC
13      :
14
15      :CALL:
16      : (1)  JSR      PC,CLRSTS
17      :       BR      ???
18      :       NOP
19      :       ERROR
20      :       JSR      PC,@(SP)+
21      :       ???
22
23      CLRSTS:
24
25      ;CLEAR USER'S ERROR CALL
26      ADD      #4,(SP)
27      CLPB     @ (SP)
28      SUB      #4,(SP)
29
30      ;REPORT ERROR IF RMCS1 NOT INITIALIZED
31      4$: MOV     RMCS1I,$BDDAT
32      BIC      #SC,$BDDAT
33      MOV      #DVA!RDY,$GDDAT
34      CMP      $GDDAT,$BDDAT
35      BEQ      5$
36      ADD      #4,(SP)
37      MOVB     #126,@(SP)
38      SUB      #2,(SP)
39      JSR      PC,@(SP)+
40      SUB      #10,(SP)
41      NOP
42
43      ;REPORT ERROR IF RMBA NOT RESET
44      5$: CLR      $GDDAT
45      MOV      RMBAI,$BDDAT
46      BEQ      7$
47      ADD      #4,(SP)
48      MOVB     #127,@(SP)
49      SUB      #2,(SP)
50      JSR      PC,@(SP)+
51      SUB      #10,(SP)
52      NOP
53
54      ;REPORT ERROR IF RMCS2 NOT INITIALIZED
55      7$: MOV      RMCS2I,$BDDAT
56      MOV      R1,-(SP)
57      CLR      -(SP)
58      MOV      TSTQUE,R1
59      MOVB     (R1),(SP)
60      BIS      #IR,(SP)
61      MOV      (SP)+,$GDDAT

```

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

;MOVE SP TO ERROR
 ;CLEAR ERROR NUMBER
 ;MOVE SP BACK TO NO ERROR
 ;VERIFY RMCS1
 ;IGNORE SPECIAL CONDITION
 ;EXPECT DVA & RDY
 ;COMPARE EXPECTED, RECEIVED
 ;BRANCH IF EQUAL
 ;MOVE SP TO USER'S ERROR CALL
 ;WRITE ERROR NUMBER IN CALL
 ;MOVE SP TO RETURN FOR ERROR
 ;REPORT ERROR VIA USER
 ;MOVE SP BACK TO NO ERROR
 ;VERIFY RMBA IS ZERO
 ;BRANCH IF ZERO
 ;MOVE SP TO USER'S ERROR CALL
 ;WRITE ERROR NUMBER IN CALL
 ;MOVE SP TO RETURN FOR ERROR
 ;REPORT ERROR VIA USER
 ;MOVE SP BACK TO NO ERROR

;VERIFY RMCS2
 ;PUSH R1 ON STACK
 ;EXPECT IR & UNIT NUMBER
 ;R1 = ADDRESS OF TEST QUE

58	052740	012601			MOV	(SP)+,R1	::POP STACK INTO R1
59	052742	023737	001140	001142	CMP	\$GDDAT,\$BDDAT	::COMPARE EXPECTED & RECEIVED
60	052750	001413			BEQ	9\$	::BRANCH IF EQUAL
61	052752	062716	000004		ADD	#4,(SP)	::MOVE SP TO USER'S ERROR CALL
62	052756	112776	000130	000000	MOVB	#130,@(SP)	::WRITE ERROR NUMBER IN CALL
63	052764	162716	000002		SUB	#2,(SP)	::MOVE SP TO RETURN FOR ERROR
64	052770	004736			JSR	PC,@(SP)+	::REPORT ERROR VIA USER
65	052772	162716	000010		SUB	#10,(SP)	::MOVE SP BACK TO NO ERROR
66	052776	000240			NOP		
67					:REPORT ERROR IF RMER1 NOT RESET-IGNORE UNS		
68	053000	005037	001140		9\$: CLR	\$GDDAT	::VERIFY RMER1
69	053004	013737	001352	001142	MOV	RMER1I,\$BDDAT	
70	053012	042737	040000	001142	BIC	#UNS,\$BDDAT	::IGNORE UNSAFE
71	053020	001413			BEQ	13\$	::BRANCH IF ZERO
72	053022	062716	000004		ADD	#4,(SP)	::MOVE SP TO USER'S ERROR CALL
73	053026	112776	000131	000000	MOVB	#131,@(SP)	::WRITE ERROR NUMBER IN CALL
74	053034	162716	000002		SUB	#2,(SP)	::MOVE SP TO RETURN FOR ERROR
75	053040	004736			JSR	PC,@(SP)+	::REPORT ERROR VIA USER
76	053042	162716	000010		SUB	#10,(SP)	::MOVE SP BACK TO NO ERROR
77	053046	000240			NOP		
78					:REPORT ERROR IF RMMR1 NOT INITIALIZED-IGNORE WC,LS,LST		
79	053050	013737	001362	001142	13\$: MOV	RMMR1I,\$BDDAT	::VERIFY RMMR
80	053056	042737	000046	001142	BIC	#WC!LS!LST,\$BDDAT	::IGNORE WORD CLOCK, SCT, TRK
81	053064	012737	000010	001140	MOV	#MWD,\$GDDAT	::EXPECT WRITE DATA BIT
82	053072	023737	001140	001142	CMP	\$GDDAT,\$BDDAT	::COMPARE EXPECTED AND RECEIVED
83	053100	001413			BEQ	17\$	::BRANCH IF 0
84	053102	062716	000004		ADD	#4,(SP)	::MOVE SP TO USER'S ERROR CALL
85	053106	112776	000133	000000	MOVB	#133,@(SP)	::WRITE ERROR NUMBER IN CALL
86	053114	162716	000002		SUB	#2,(SP)	::MOVE SP TO RETURN FOR ERROR
87	053120	004736			JSR	PC,@(SP)+	::REPORT ERROR VIA USER
88	053122	162716	000010		SUB	#10,(SP)	::MOVE SP BACK TO NO ERROR
89	053126	000240			NOP		
90					:REPORT AN ERROR IF RMEC2 IS NOT RESET		
91	053130	005037	001140		17\$: CLR	\$GDDAT	::EXPECT ZEROS
92	053134	013737	001404	001142	MOV	RMEC2I,\$BDDAT	::VERIFY RMEC2-0
93	053142	001413			BEQ	19\$	
94	053144	062716	000004		ADD	#4,(SP)	::MOVE SP TO USER'S ERROR CALL
95	053150	112776	000135	000000	MOVB	#135,@(SP)	::WRITE ERROR NUMBER IN CALL
96	053156	162716	000002		SUB	#2,(SP)	::MOVE SP TO RETURN FOR ERROR
97	053162	004736			JSR	PC,@(SP)+	::REPORT ERROR VIA USER
98	053164	162716	000010		SUB	#10,(SP)	::MOVE SP BACK TO NO ERROR
99	053170	000240			NOP		
100					:REPORT ERROR IF RMMR2 NOT INITIALIZED-IGNORE RQA,RQB		
101	053172	013737	001376	001142	19\$: MOV	RMMR2I,\$BDDAT	::VERIFY RMMR2
102	053200	042737	140000	001142	BIC	#RQA!RQB,\$BDDAT	
103	053206	012737	011777	001140	MOV	#TST!1777,\$GDDAT	::EXPECT TEST BIT ON
104	053214	023737	001140	001142	CMP	\$GDDAT,\$BDDAT	
105	053222	001413			BEQ	21\$	
106	053224	062716	000004		ADD	#4,(SP)	::MOVE SP TO USER'S ERROR CALL
107	053230	112776	000136	000000	MOVB	#136,@(SP)	::WRITE ERROR NUMBER IN CALL
108	053236	162716	000002		SUB	#2,(SP)	::MOVE SP TO RETURN FOR ERROR
109	053242	004736			JSR	PC,@(SP)+	::REPORT ERROR VIA USER
110	053244	162716	000010		SUB	#10,(SP)	::MOVE SP BACK TO NO ERROR
111	053250	000240			NOP		
112					:REPORT ERROR IF RMER2 NOT RESET-IGNORE SKI,DVC		
113	053252	005037	001140		21\$: CLR	\$GDDAT	::EXPECT ALL ZEROS
114	053256	013737	001400	001142	MOV	RMER2I,\$BDDAT	::VERIFY RMER2

115	053264	042737	040200	001142	BIC	#SKI!DVC,\$BDDAT	;IGNORE DEVICE ERRORS
116	053272	001413			BEQ	215\$	;BRANCH IF OTHER BITS 0
117	053274	062716	000004		ADD	#4,(SP)	;MOVE SP TO USER'S ERROR CALL
118	053300	112776	000174	000000	MOVB	#174,@(SP)	;WRITE ERROR NUMBER IN CALL
119	053306	162716	000002		SUB	#2,(SP)	;MOVE SP TO RETURN FOR ERROR
120	053312	004736			JSR	PC,@(SP)+	;REPORT ERROR VIA USER
121	053314	162716	000010		SUB	#10,(SP)	;MOVE SP BACK TO NO ERROR
122	053320	000240			NOP		
123							
124	053322	013737	001350	001142	215\$: MOV	RMDSI,\$BDDAT	;TEST DRIVE STATUS REGISTER
125	053330	042737	77177	001142	BIC	#^C<DRY!DPR>,\$BDDAT	
126	053336	012737	000600	001140	MOV	#DPR.DRY,\$GDDAT	;EXPECTED DRIVE STATUS
127	053344	023737	001140	001142	CMP	\$GDDAT,\$BDDAT	;COMPARE EXPECTED & RECEIVED
128	053352	001413			BEQ	22\$	;BRANCH IF EQUAL
129	053354	062716	000004		ADD	#4,(SP)	;MOVE SP TO USER'S ERROR CALL
130	053360	112776	000134	000000	MOVB	#134,@(SP)	;WRITE ERROR NUMBER
131	053366	162716	000002		SUB	#2,(SP)	;MOVE SP TO RETURN FOR ERROR
132	053372	004736			JSR	PC,@(SP)+	;REPORT ERROR TO USER
133	053374	162716	000010		SUB	#10,(SP)	;MOVE SP BACK TO NO ERROR
134	053400	000240			NOP		
135	053402	062716	000004		22\$: ADD	#4,(SP)	;MOVE SP TO ERROR CALL
136	053406	105776	000000		TSTB	@(SP)	;WAS AN ERROE DETECTED??
137	053412	001403			BEQ	23\$	;NO!
138	053414	062716	000004		ADD	#4,(SP)	;YES - MOVE TO ERROR RETURN
139	053420	000402			BR	24\$	
140	053422	162716	000004		23\$: SUB	#4,(SP)	;MOVE SP TO NO ERROR RETURN
141	053426	000240			24\$: NOP		
142	053430	000207			RTS	PC	

```

1      .SBTTL  PACK ACKNOWLEDGE STATUS CHECK
2
3      ;THIS SUBROUTINE CHECKS THE RESULTS OF A PACK ACKNOWLEDGE
4      ;COMMAND USING THE STATUS STORED IN THE GET BUFFER.  ERRORS ARE
5      ;REPORTED TO THE USER VIA THE USER'S ERROR CALL.
6
7      ;CALL:
8      ;(1)  JSR      PC,ACKSTS
9      ;      BR      ???
10     ;      NOP
11     ;      ERROR
12     ;      JSR      PC,@(SP)+
13     ;      ???
14
15     ACKSTS:
16
17     ;CLEAR USER'S ERROR CALL
18     ADD     #4,(SP)      ;MOVE SP TO ERROR CALL
19     CLRB    @ (SP)      ;CLEAR LOW ORDER BYTE
20     SUB     #4,(SP)      ;MOVE SP BACK
21
22     ;REPORT AN ERROR IF 'VV' IS 0
23     BIT     #VV,RMDSI    ;IS VOLUME VALID SET??
24     BNE     1$           ;YES!!
25     MOV     RMDSI,$GDDAT ;EXPECTED STATUS
26     BIS     #VV,$GDDAT
27     MOV     RMDSI,$BDDAT ;RECEIVED STATUS
28     ADD     #4,(SP)      ;MOVE SP TO ERROR CALL
29     MOVB    #155,@(SP)   ;WRITE NUMBER IN ERROR CALL
30     SUB     #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
31     JSR     PC,@(SP)+    ;REPORT THE ERROR
32     SUB     #10,(SP)     ;MOVE SP BACK TO BRANCH
33     NOP
34
35     1$:
36
37     ;REPORT AN ERROR IF 'MOL' IS 0
38     BIT     #MOL,RMDSI   ;IS MOL SET??
39     BNE     2$           ;YES!!
40     MOV     RMDSI,$GDDAT ;EXPECTED STATUS
41     BIS     #MOL,$GDDAT
42     MOV     RMDSI,$BDDAT ;RECEIVED STATUS
43     ADD     #4,(SP)      ;MOVE SP TO ERROR CALL
44     MOVB    #41,@(SP)    ;WRITE NUMBER OF ERROR IN CALL
45     SUB     #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
46     JSR     PC,@(SP)+    ;REPORT TH ERROR
47     SUB     #10,(SP)     ;MOVE SP TO BRANCH
48     NOP
49
50     2$:
51
52     ;SEE IF 'UNS','OPI','RMR','ILR', OR 'ILF' IS SET
53     BIT     #UNS,OPI,RMR,ILR,ILF,RMER11
54     BEQ     7$
55
56     ;REPORT AN ERROR IF 'UNS' IS SET
57     BIT     #UNS,RMER11   ;WAS UNS SET??
58     BEQ     3$           ;NO..
59     MOV     RMER11,$BDDAT ;RECEIVED STATUS

```

58	053634	013737	001352	001140	MOV	RMER11,\$GDDAT	:EXPECTED STATUS
59	053642	042737	040000	001140	BIC	#UNS,\$GDDAT	
60	053650	062716	000004		ADD	#4,(SP)	:MOVE SP TO ERROR CALL
61	053654	112776	000042	000000	MOVB	#42,@(SP)	:WRITE NUMBER OF ERROR IN CALL
62	053662	162716	000002		SUB	#2,(SP)	:MOVE SP TO RETURN FOR ERROR
63	053666	004736			JSR	PC,@(SP)+	:REPORT THE ERROR VIA USER
64	053670	162716	000010		SUB	#10,(SP)	:MOVE SP TO NO ERROR RETURN
65	053674	000240			NOP		
66	053676						
67							
68					:REPORT	ANY OPI ERROR	
69	053676	032737	020000	001352	BIT	#OPI,RMER11	:WAS OPI SET ??
70	053704	001424			BEQ	4\$	:NO..
71	053706	013737	001352	001142	MOV	RMER11,\$BDDAT	:RECEIVED STATUS
72	053714	013737	001352	001140	MOV	RMER11,\$GDDAT	:EXPECTED STATUS
73	053722	042737	020000	001140	BIC	#OPI,\$GDDAT	
74	053730	062716	000004		ADD	#4,(SP)	:MOVE SP TO ERROR CALL
75	053734	112776	000043	000000	MOVB	#43,@(SP)	:WRITE NUMBER OF ERROR IN CALL
76	053742	162716	000002		SUB	#2,(SP)	:MOVE SP TO RETURN FOR ERROR
77	053746	004736			JSR	PC,@(SP)+	:REPORT THE ERROR VIA USER
78	053750	162716	000010		SUB	#10,(SP)	:MOVE SP TO NO ERROR RETURN
79	053754	000240			NOP		
80	053756						
81					4\$:		
82					:REPORT	ANY RMR ERROR	
83	053756	032737	000004	001352	BIT	#RMR,RMER11	:WAS RMR SET??
84	053764	001424			BEQ	5\$	:NO..
85	053766	013737	001352	001142	MOV	RMER11,\$BDDAT	:RECEIVED STATUS
86	053774	013737	001352	001140	MOV	RMER11,\$GDDAT	:EXPECTED STATUS
87	054002	042737	000004	001140	BIC	#RMR,\$GDDAT	
88	054010	062716	000004		ADD	#4,(SP)	:MOVE SP TO ERROR CALL
89	054014	112776	000044	000000	MOVB	#44,@(SP)	:WRITE NUMBER OF ERROR IN CALL
90	054022	162716	000002		SUB	#2,(SP)	:MOVE SP TO RETURN FOR ERROR
91	054026	004736			JSR	PC,@(SP)+	:REPORT THE ERROR VIA USER
92	054030	162716	000010		SUB	#10,(SP)	:MOVE SP TO NO ERROR RETURN
93	054034	000240			NOP		
94	054036						
95					5\$:		
96					:REPORT	ANY ILR ERROR	
97	054036	032737	000002	001352	BIT	#ILR,RMER11	:WAS ILR SET??
98	054044	001424			BEQ	6\$	:NO..
99	054046	013737	001352	001142	MOV	RMER11,\$BDDAT	:RECEIVED STATUS
100	054054	013737	001352	001140	MOV	RMER11,\$GDDAT	:EXPECTED STATUS
101	054062	042737	000002	001140	BIC	#ILR,\$GDDAT	
102	054070	062716	000004		ADD	#4,(SP)	:MOVE SP TO ERROR CALL
103	054074	112776	000045	000000	MOVB	#45,@(SP)	:WRITE NUMBER OF ERROR IN CALL
104	054102	162716	000002		SUB	#2,(SP)	:MOVE SP TO RETURN FOR ERROR
105	054106	004736			JSR	PC,@(SP)+	:REPORT THE ERROR VIA USER
106	054110	162716	000010		SUB	#10,(SP)	:MOVE SP TO NO ERROR RETURN
107	054114	000240			NOP		
108	054116				6\$:		
109					:REPORT	ANY ILF ERROR	
110							
111	054116	032737	000001	001352	BIT	#ILF,RMER11	:WAS ILF SET??
112	054124	001424			BEQ	7\$	:NO..
113	054126	013737	001352	001142	MOV	RMER11,\$BDDAT	:RECEIVED STATUS
114	054134	013737	001352	001140	MOV	RMER11,\$GDDAT	:EXPECTED STATUS

115	054142	042737	000001	001140	BIC	#ILF,\$GDDAT	
116	054150	062716	000004		ADD	#4,(SP)	;MOVE SP TO ERROR CALL
117	054154	112776	000046	000000	MOVB	#46,@(SP)	;WRITE NUMBER OF ERROR IN CALL
118	054162	162716	000002		SUB	#2,(SP)	;MOVE SP TO RETURN FOR ERROR
119	054166	004736			JSR	PC,@(SP)+	;REPORT THE ERROR VIA USER
120	054170	162716	000010		SUB	#10,(SP)	;MOVE SP TO NO ERROR RETURN
121	054174	000240			NOP		
122	054176			7\$:			
123							
124							
125	054176	062716	000004		ADD	#4,(SP)	;MOVE SP TO ERROR CALL
126	054202	105776	000000		TSTB	@(SP)	;WAS ERROR FOUND??
127	054206	001403			BEQ	8\$	;NO!
128	054210	062716	000004		ADD	#4,(SP)	;YES - MOVE TO ERROR RETURN
129	054214	000402			BR	9\$	
130	054216	162716	000004	8\$:	SUB	#4,(SP)	;MOVE SP TO NO ERROR RETURN
131	054222	000240		9\$:	NOP		
132	054224	000207			RTS	PC	

```

1      .SBTTL RECALIBRATE STATUS CHECK SUBROUTINE
2
3      ;THIS SUBROUTINE CHECKS THE RESULTS OF A RECALIBRATE OPERATION
4      ;USING THE STATUS STORED IN THE GET BUFFER.
5
6      ;CALL:
7
8      ;(1) JSR PC,RCLSTS ;CALL SUBROUTINE
9      ;    BR ???       ;RETURN HERE IF NO ERROR
10     ;    NOP          ;RETURN HERE TO REPORT AN ERROR
11     ;    ERROR        ;ERROR NUMBER DEFINED BY SUB
12     ;    JSR PC,@(SP)+ ;GO BACK TO SUB FOR MORE ERROR CHECKS
13     ;    ???          ;RETURN HERE IF NO MORE ERRORS
14
15 054226 RCLSTS:
16
17 ;CLEAR USER'S ERROR NUMBER
18 054226 062716 000004 ADD #4,(SP)
19 054232 105076 000000 CLRB @ (SP) ;CLEAR USER'S ERROR CALL
20 054236 162716 000004 SUB #4,(SP) ;MOVE SP BACK TO BRANCH
21
22
23 ;SEE IF 'PAR' OR 'ILF' OR 'OPI' OR 'IAE' IS SET
24 054242 032737 022011 001352 BIT #OPI.PAR.ILF.IAE,RMER1I
25 054250 001553 BEQ 4$ ;NONE ARE SET - GO TO NEXT CHECK
26
27 ;REPORT ANY MASSBUS CONTROL BUS PARITY ERROR, I.E.,
28 ;'PAR' 1 AND 'DPE' - 0
29 054252 032737 000010 001352 BIT #PAR,RMER1I ;WAS 'PAR' SET??
30 054260 001430 BEQ 1$ ;NO!!
31 054262 032737 000010 001400 BIT #DPE,RMER2I ;WAS 'DPE' SET??
32 054270 001024 BNE 1$ ;YES - NOT A REGISTER ERROR
33 054272 013737 001352 001140 MOV RMER1I,$GDDAT ;EXPECTED STATUS
34 054300 042737 000010 001140 BIC #PAR,$GDDAT
35 054306 013737 001352 001142 MOV RMER1I,$BDDAT ;RECEIVED STATUS
36 054314 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
37 054320 112776 000050 000000 MOVB #50,@(SP) ;WRITE ERROR NUMBER IN CALL
38 054326 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
39 054332 004736 JSR PC,@(SP)+ ;GO REPORT ERROR
40 054334 162716 000010 SUB #10,(SP) ;MOVE SP BACK TO BRANCH
41 054340 000240 NOP
42 054342 1$:
43
44 ;REPORT ANY 'ILF' ERROR
45 054342 032737 000001 001352 BIT #ILF,RMER1I ;WAS 'ILF' SET??
46 054350 001424 BEQ 2$ ;NO!!
47 054352 013737 001352 001140 MOV RMER1I,$GDDAT ;EXPECTED STATUS
48 054360 042737 000001 001140 BIC #ILF,$GDDAT
49 054366 013737 001352 001142 MOV RMER1I,$BDDAT ;RECEIVED STATUS
50 054374 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
51 054400 112776 000071 000000 MOVB #71,@(SP) ;WRITE ERROR NUMBER IN CALL
52 054406 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
53 054412 004736 JSR PC,@(SP)+ ;REPORT ERROR VIA USER
54 054414 162716 000010 SUB #10,(SP) ;MOVE SP BACK TO BRANCH
55 054420 000240 NOP
56 054422 2$:
57

```



```

58      ;REPORT ANY 'OPI' ERROR AS
59      :
60      : . OPI DUE TO 'MOL' 0
61      : . OPI BECAUSE ON CYLINDER LATCH DIDN'T RESET
62      :
63      054422 032737 020000 001352      BIT      #OPI,RMER11      ;WAS OPI SET??
64      054430 001433                      BEQ      31$              ;NO!!
65      054432 013737 001352 001140      MOV      RMER11,$GDDAT      ;EXPECTED STATUS
66      054440 042737 020000 001140      BIC      #OPI,$GDDAT
67      054446 013737 001352 001142      MOV      RMER11,$BDDAT      ;RECEIVED STATUS
68      054454 062716 000004              ADD      #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
69      054460 112776 000072 000000      MOV      #72,@(SP)        ;WRITE ERROR NUMBER IN USER'S CALL
70      054466 032737 010000 001350      BIT      #MOL,RMDSI      ;WAS 'MOL' = 0??
71      054474 001403                      BEQ      3$              ;YES!!
72      054476 112776 000073 000000      MOV      #73,@(SP)        ;NO - CHANGE ERROR NUMBER
73      054504 162716 000002      3$:      SUB      #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
74      054510 004736                      JSR      PC,@(SP)+        ;REPORT ERROR VIA USER
75      054512 162716 000010                      SUB      #10,(SP)      ;MOVE SP BACK TO BRANCH
76      054516 000240                      NOP
77      054520      31$:
78      ;REPORT AN ERROR IF 'IAE' IS SET
79      054526 001424                      BIT      #IAE,RMER11      ;IS 'IAE' SET??
80      054530 013737 001352 001140      BEQ      4$              ;NO!!
81      054536 042737 002000 001140      MOV      RMER11,$GDDAT      ;EXPECTED STATUS
82      054544 013737 001352 001142      BIC      #IAE,$GDDAT
83      054552 062716 000004              MOV      RMER11,$BDDAT      ;RECEIVED STATUS
84      054556 112776 000070 000000      ADD      #4,(SP)          ;MOVE SP TO ERROR CALL
85      054564 162716 000002              MOV      #70,@(SP)        ;WRITE ERROR NUMBER IN USER'S CALL
86      054570 004736                      SUB      #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
87      054572 162716 000010              JSR      PC,@(SP)+        ;REPORT ERROR
88      054576 000240                      SUB      #10,(SP)      ;MOVE SP BACK TO NO ERROR RETURN
89      054600      4$:      NOP
90
91      ;SEE IF 'SKI' OR 'IVC' OR 'DVC' IS SET
92      054600 032737 050200 001400      BIT      #SKI!IVC!DVC,RMER21
93      054606 001517                      BEQ      8$              ;NONE OF THE BITS ARE SET
94
95
96      ;REPORT ANY 'IVC' ERROR AS
97      :
98      : . IVC WITH VV = 0
99      : . ERRONEOUS IVC ERROR
100     :
101     054610 032737 010000 001400      BIT      #IVC,RMER21      ;WAS IVC SET??
102     054616 001433                      BEQ      6$              ;NO!!
103     054620 013737 001400 001140      MOV      RMER21,$GDDAT      ;EXPECTED STATUS
104     054626 042737 010000 001140      BIC      #IVC,$GDDAT
105     054634 013737 001400 001142      MOV      RMER21,$BDDAT      ;RECEIVED STATUS
106     054642 062716 000004              ADD      #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
107     054646 112776 000074 000000      MOV      #74,@(SP)        ;WRITE ERROR NUMBER IN CALL
108     054654 032737 000100 001350      BIT      #VV,RMDSI      ;WAS VV = 0??
109     054662 001403                      BEQ      5$              ;YES!!
110     054664 112776 000075 000000      MOV      #75,@(SP)        ;NO - CHANGE ERROR NUMBER
111     054672 162716 000002      5$:      SUB      #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
112     054676 004736                      JSR      PC,@(SP)+        ;REPORT ERROR VIA USER
113     054700 162716 000010                      SUB      #10,(SP)      ;MOVE SP BACK TO BRANCH
114     054706 000240                      NOP
115     054706      6$:
  
```

```

115 ;REPORT A
116 054706 032737 040000 001400 BI R21 ;WAS SKI SET??
117 054714 001424 RE ;NO..
118 054716 013737 001400 001140 MOV RM, $GDDAT ;EXPECTED STATUS
119 054724 042737 040000 001140 BIC #SKI, $DDAT
120 054732 013737 001400 001142 MOV RMER2, $BDDAT ;RECEIVED STATUS
121 054740 0627 6 000004 ADD #, (SP) ;MOVE SP TO USER'S ERROR CALL
122 054744 112776 000076 000000 MOVW #76, @ (SP) ;WRITE ERROR NUMBER
123 054752 162716 000002 SUB #2, (SP) ;MOVE SP TO RETURN FOR ERROR
124 054756 004736 JSR PC, @ (SP)+ ;REPORT ERROR VIA USER
125 054760 162716 000010 SUB #10, (SP) ;MOVE SP TO BRANCH
126 054764 000240 NOP
127 054766 7$:
128
129 ;REPORT ANY 'DVC' ERROR
130 054766 032737 000200 001400 BIT #DVC, RMER2I ;WAS 'DVC' SET??
131 054774 001424 BEQ 8$ ;NO!!
132 054776 013737 001400 001140 MOV RMER2I, $GDDAT ;EXPECTED STATUS
133 055004 042737 000200 001140 BIC #DVC, $GDDAT
134 055012 013737 001400 001142 MOV RMER2I, $BDDAT ;RECEIVED STATUS
135 055020 062716 000004 ADD #4, (SP) ;MOVE SP TO USER'S ERROR CALL
136 055024 112776 000077 000000 MOVW #77, @ (SP) ;WRITE ERROR NUMBER
137 055032 162716 000002 SUB #2, (SP) ;MOVE SP TO RETURN FOR ERROR
138 055036 004736 JSR PC, @ (SP)+ ;REPORT ERROR VIA USER
139 055040 162716 000010 SUB #10, (SP) ;MOVE SP TO USER'S BRANCH
140 055044 000240 NOP
141 055046 8$:
142
143 ;SEE IF 'PIP' AND 'OM' ARE 0, AND 'ATA', 'MOL' AND 'VV' ARE 1
144 055046 013746 001350 MOV RMDSI, - (SP) ;PUT RMDS ON STACK
145 055052 042716 047676 BIC #^C<PIP!MOL!VV!OM!ATA>, (SP)
146 055056 022726 110100 CMP #ATA!MOL!VV, (SP)+
147 055062 001007 BNE 85$
148 055064 000137 055500 JMP 13$
149 055070 85$:
150
151 ;REPORT AN ERROR IF MOL = 0 AND OPI = 0, I.E., MEDIUM WENT OFF
152 ;LINE AFTER RECALIBRATE WAS INITIATED
153 055070 032737 010000 001350 BIT #MOL, RMDSI ;DID MOL DROP??
154 055076 001030 BNE 9$ ;NO!!
155 055100 032737 020000 001352 BIT #OPI, RMER1I ;WAS OPI ERROR REPORTED??
156 055106 001024 BNE 9$ ;YES - DON'T REPORT MOL 0
157 055110 013737 001350 001140 MOV RMDSI, $GDDAT ;EXPECTED STATUS
158 055116 052737 010000 001140 BIS #MOL, $GDDAT
159 055124 013737 001350 001142 MOV RMDSI, $BDDAT ;RECEIVED STATUS
160 055132 062716 000004 ADD #4, (SP) ;MOVE SP TO USER'S ERROR CALL
161 055136 112776 000100 000000 MOVW #100, @ (SP) ;WRITE ERROR NUMBER
162 055144 162716 000002 SUB #2, (SP) ;MOVE SP TO RETURN FOR ERROR
163 055150 004736 JSR PC, @ (SP)+ ;REPORT ERROR VIA USER
164 055152 162716 000010 SUB #10, (SP) ;MOVE SP BACK TO USER'S BRANCH
165 055156 000240 NOP
166 055160 9$:
167
168 ;REPORT AN ERROR IF 'VV' = 0 AND 'IVC' = 0
169 055160 032737 000100 001350 BIT #VV, RMDSI ;DID 'VV' DROP??
170 055166 001030 BNE 10$ ;NO!!
171 055170 032737 010000 001400 BIT #IVC, RMER2I ;WAS THERE A IVC ERROR??

```

172	055176	001024			BNE	10\$	:YES - DONT REPORT VV-0
173	055200	013737	001350	001140	MOV	RMDSI,\$GDDAT	:EXPECTED STATUS
174	055206	013737	001350	001142	MOV	RMDSI,\$BDDAT	:RECEIVED STATUS
175	055214	052737	000100	001140	BIS	#VV,\$GDDAT	
176	055222	062716	000004		ADD	#4,(SP)	:MOVE SP TO USER'S ERROR CALL
177	055226	112776	000101	000000	MOVB	#101,@(SP)	:WRITE ERROR NUMBER IN CALL
178	055234	162716	000002		SUB	#2,(SP)	:MOVE SP TO RETURN FOR ERROR
179	055240	004736			JSR	PC,@(SP)+	
180	055242	162716	000010		SUB	#10,(SP)	:MOVE SP BACK TO USER'S BRANCH
181	055246	000240			NOP		
182	055250						
183							
184							
185	055250	032737	100000	001350			:REPORT AN ERROR IF ATA IS NOT SET
186	055256	001024			BIT	#ATA,RMDSI	:WAS ATA SET DURING RECALIBRATE??
187	055260	013737	001350	001140	BNE	11\$	:YES!!
188	055266	052737	100000	001140	MOV	RMDSI,\$GDDAT	:EXPECTED STATUS
189	055274	013737	001350	001142	BIS	#ATA,\$GDDAT	
190	055302	062716	000004		MOV	RMDSI,\$BDDAT	:RECEIVED STATUS
191	055306	112776	000102	000000	ADD	#4,(SP)	:MOVE SP TO USER'S ERROR CALL
192	055314	162716	000002		MOVB	#102,@(SP)	:WRITE ERROR NUMBER IN CALL
193	055320	004736			SUB	#2,(SP)	
194	055322	162716	000010		JSR	PC,@(SP)+	
195	055326	000240			SUB	#10,(SP)	:MOVE SP TO USER'S BRANCH
196					NOP		
197	055330						
198							
199							
200							
201	055330	032737	000001	001350			:REPORT AN ERROR IF 'OM' IS NOT ZERO BECAUSE RECALIBRATE SHOULD
202	055336	001424			:ALWAYS	CLEAR OFFSET MODE	
203	055340	013737	001350	001140	BIT	#OM,RMDSI	:WAS 'OM' RESET??
204	055346	042737	000001	001140	BEQ	12\$	:YES!!
205	055354	013737	001350	001142	MOV	RMDSI,\$GDDAT	:EXPECTED STATUS
206	055362	062716	000004		BIC	#OM,\$GDDAT	
207	055366	112776	000103	000000	MOV	RMDSI,\$BDDAT	:RECEIVED STATUS
208	055374	162716	000002		ADD	#4,(SP)	:MOVE SP TO USER'S ERROR CALL
209	055400	004736			MOVB	#103,@(SP)	:WRITE ERROR NUMBER
210	055402	162716	000010		SUB	#2,(SP)	:MOVE SP TO RETURN FOR ERROR
211	055406	000240			JSR	PC,@(SP)+	:REPORT ERROR VIA USER
212	055410				SUB	#10,(SP)	:MOVE SP TO USER'S BRANCH
213					NOP		
214							
215							
216	055410	032737	020000	001350			:REPORT AN ERROR IF 'PIP' IS STIL ON, I.E., DRIVE NOT ON
217	055416	001430			:CYLINDER		
218	055420	032737	040000	001400	BIT	#PIP,RMDSI	:IS DRIVE OFF CYLINDER??
219	055426	001024			BEQ	13\$	:NO!!
220	055430	013737	001350	001140	BIT	#SKI,RMER2I	:WAS 'SKI' DETECTED??
221	055436	042737	020000	001140	BNE	13\$	:YES-DONT REPORT 'PIP'
222	055444	013737	001350	001142	MOV	RMDSI,\$GDDAT	:EXPECTED STATUS
223	055452	062716	000004		BIC	#PIP,\$GDDAT	
224	055456	112776	000104	000000	MOV	RMDSI,\$BDDAT	:RECEIVED STATUS
225	055464	162716	000002		ADD	#4,(SP)	:MOVE SP TO USER'S ERROR CALL
226	055470	004736			MOVB	#104,@(SP)	:WRITE ERROR NUMBER
227	055472	162716	000010		SUB	#2,(SP)	:MOVE SP TO RETURN FOR ERROR
228	055476	000240			JSR	PC,@(SP)+	
					SUB	#10,(SP)	:MOVE SP BACK TO USER'S BRANCH
					NOP		

```

229 055500      13$:
230
231      ;SEE IF 'ILR' OR 'RMR' OR 'UNS' IS SET
232 055500 032737 040006 001352      BIT    #ILR,RMER11
233 055506 001514      BEQ    16$
234
235      ;REPORT AN ERROR IF 'ILR' IS SET
236 055510 032737 000002 001352      BIT    #ILR,RMER11      ;WAS ILR SET DURING RECALIBRATE??
237 055516 001424      BEQ    14$      ;NO!!
238 055520 013737 001352 001140      MOV    RMER11,$GDDAT      ;EXPECTED STATUS
239 055526 042737 000002 001140      BIC    #ILR,$GDDAT
240 055534 013737 001352 001142      MOV    RMER11,$BDDAT      ;RECEIVED STATUS
241 055542 062716 000004      ADD    #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
242 055546 112776 000105 000000      MOV    #105,a(SP)      ;WRITE ERROR NUMBER IN CALL
243 055554 162716 000002      SUB    #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
244 055560 004736      JSR    PC,a(SP)+
245 055562 162716 000010      SUB    #10,(SP)      ;MOVE SP TO USER'S BRANCH
246 055566 000240      NOP
247 055570
248
249      14$:
250      ;REPORT AN ERROR IF 'RMR' IS SET
251 055570 032737 000004 001352      BIT    #RMR,RMER11      ;WAS RMR SET??
252 055576 001424      BEQ    15$      ;NO!!
253 055600 013737 001352 001140      MOV    RMER11,$GDDAT      ;EXPECTED STATUS
254 055606 042737 000004 001140      BIC    #RMR,$GDDAT
255 055614 013737 001352 001142      MOV    RMER11,$BDDAT      ;RECEIVED STATUS
256 055622 062716 000004      ADD    #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
257 055626 112776 000106 000000      MOV    #106,a(SP)      ;WRITE ERROR NUMBER IN USER'S CALL
258 055634 162716 000002      SUB    #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
259 055640 004736      JSR    PC,a(SP)+
260 055642 162716 000010      SUB    #10,(SP)      ;REPORT ERROR VIA USER
261 055646 000240      NOP      ;MOVE SP TO USER'S BRANCH
262 055650
263
264      15$:
265      ;REPORT AN ERROR IF 'UNS' IS SET AND 'DVC' IS 0
266 055650 032737 040000 001352      BIT    #UNS,RMER11      ;WAS UNSAFE ON??
267 055656 001430      BEQ    16$      ;NO!!
268 055660 032737 000200 001400      BIT    #DVC,RMER21      ;WAS THERE A DEVICE CHECK??
269 055666 001024      BNE    16$      ;YES - DON'T REPORT UNSAFE
270 055670 013737 001352 001140      MOV    RMER11,$GDDAT      ;EXPECTED STATUS
271 055676 042737 040000 001140      BIC    #UNS,$GDDAT
272 055704 013737 001352 001142      MOV    RMER11,$BDDAT      ;RECEIVED STATUS
273 055712 062716 000004      ADD    #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
274 055716 112776 000107 000000      MOV    #107,a(SP)      ;WRITE ERROR NUMBER
275 055724 162716 000002      SUB    #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
276 055730 004736      JSR    PC,a(SP)+
277 055732 162716 000010      SUB    #10,(SP)      ;REPORT ERROR VIA USER
278 055736 000240      NOP      ;MOVE SP BACK TO USER'S BRANCH
279 055740
280
281      16$:
282      ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
283 055740 062716 000004      ADD    #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
284 055744 105776 000000      TSTB    a(SP)      ;WAS AN ERROR REPORTED??
285 055750 001403      BEQ    17$      ;NO!!
286 055752 062716 000004      ADD    #4,(SP)      ;YES - AUGMENT SP RETURN
287 055756 000402      BR    18$
288 055760 162716 000004      SUB    #4,(SP)      ;NO ERROR - RETURN SP TO BRANCH
  
```

286 055764 000240
287 055766 000207

18\$:

NOP
RTS

PC

;STATUS CHECK IS COMPLETE

```

1      .SBTTL  DRIVE CLEAR STATUS CHECK SUBROUTINE
2
3      :      BR      ???      RETURN HERE IF NO ERROR
4      :      NOP      RETURN HERE TO REPORT AN ERROR
5      :      ERROR    ERROR NUMBER DEFINED BY SUB
6      :      JSR      PC,@(SP)+ GO BACK TO SUB FOR MORE ERROR CHECKS
7      :      ???      RETURN HERE IF NO MORE ERRORS
8
9 055770 DRVSTS:
10
11      ;CLEAR USER'S ERROR CALL
12 055770 062716 000004      ADD      #4,(SP)      ;MOVE SP TO ERROR CALL
13 055774 105076 000000      CLRB      @(SP)      ;CLEAR ERROR CALL
14 056000 162716 000004      SUB      #4,(SP)      ;MOVE SP TO USER'S BRANCH
15
16 056004 013737 001336 001142 ;REPORT ERROR IF RMCS1 NOT INITIALIZED
17 056012 042737 173700 001142 4$: MOV      RMCS1,$BDDAT      ;CHECK RMCS1
18 056020 012737 004010 001140      BIC      #^C<DVA!FNCMSK>,$BDDAT ;CLEAR DONT CARES
19 056026 023737 001140 001142      MOV      #DVA!DRVCLR,$GDDAT      ;EXPECT DVA
20 056034 001443      CMP      $GDDAT,$BDDAT      ;COMPARE EXPECTED & RECEIVED
21 056036 062716 000004      BEQ      6$      ;BRANCH IF EQUAL
22 056042 112776 000141 000000      ADD      #4,(SP)      ;MOVE SP TO ERROR CALL
23 056050 162716 000002      MOVB      #141,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
24 056054 004736      SUB      #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
25 056056 162716 000010      JSR      PC,@(SP)+      ;REPORT THE ERROR VIA USER
26 056062 000240      SUB      #10,(SP)      ;MOVE SP TO NO ERROR RETURN
27      NOP
28 056064 013737 001350 001142 ;REPORT ERROR IF RMD5 NOT INITIALIZED
29 056072 042737 021101 001142 5$: MOV      RMD5,$BDDAT      ;CHECK RMD5
30 056100 012737 010600 001140      BIC      #PGM!OM!VV!PIP,$BDDAT ;CLEAR DONT CARES
31 056106 023737 001140 001142      MOV      #MOL!DPR!DRY,$GDDAT      ;EXPECT DRY & DPR
32 056114 001413      CMP      $GDDAT,$BDDAT      ;COMPARE EXPECTED & RECEIVED
33 056116 062716 000004      BEQ      6$      ;BRANCH IF EQUAL
34 056122 112776 000142 000000      ADD      #4,(SP)      ;MOVE SP TO ERROR CALL
35 056130 162716 000002      MOVB      #142,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
36 056134 004736      SUB      #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
37 056136 162716 000010      JSR      PC,@(SP)+      ;REPORT THE ERROR VIA USER
38 056142 000240      SUB      #10,(SP)      ;MOVE SP TO NO ERROR RETURN
39      NOP
40 056144 005037 001140      ;REPORT ERROR IF RMER1 NOT INITIALIZED
41 056150 013737 001352 001142 6$: CLR      $GDDAT      ;EXPECT 0'S
42 056156 001413      MOV      RMER1,$BDDAT      ;CHECK RMER1
43 056160 062716 000004      BEQ      8$      ;BRANCH IF EQUAL
44 056164 112776 000143 000000      ADD      #4,(SP)      ;MOVE SP TO ERROR CALL
45 056172 162716 000002      MOVB      #143,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
46 056176 004736      SUB      #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
47 056200 162716 000010      JSR      PC,@(SP)+      ;REPORT THE ERROR VIA USER
48 056204 000240      SUB      #10,(SP)      ;MOVE SP TO NO ERROR RETURN
49      NOP
50 056206 013737 001354 001142 ;REPORT ERROR IF ATA NOT INITIALIZED
51 056214 010146      8$: MOV      RMAS1,$BDDAT      ;CHECK ATTENTION BIT
52 056216 010246      MOV      R1,-(SP)      ;PUSH R1 ON STACK
53 056220 013701 001466      MOV      R2,-(SP)      ;PUSH R2 ON STACK
54 056224 116102 000001      MOV      TSTQUE,R1
55 056230 042702 177400      MOVB      1(R1),R2
56 056234 005102      BIC      #^CATNMSK,R2
57 056236 040237 001142      COM      R2
      BIC      R2,$BDDAT
  
```



```

58 056242 012602      MOV      (SP)+,R2      ;;POP STACK INTO R2
59 056244 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
60 056246 005737 001142  TST      $BDDAT      ;;IS ATTENTION CLEARED??
61 056252 001413      BEQ      9$          ;;BRANCH IF ATTENTION CLEARED
62 056254 062716 000004  ADD      #4,(SP)      ;;MOVE SP TO ERROR CALL
63 056260 112776 000144 000000  MOVB    #144,@(SP)      ;;WRITE NUMBER OF ERROR IN CALL
64 056266 162716 000002  SUB      #2,(SP)      ;;MOVE SP TO RETURN FOR ERROR
65 056272 004736      JSR      PC,@(SP)+    ;;REPORT THE ERROR VIA USER
66 056274 162716 000010  SUB      #10,(SP)     ;;MOVE SP TO NO ERROR RETURN
67 056300 000240      NOP
68                                ;:REPORT ERROR IF RMMR1 NOT INITIALIZED
69 056302 013737 001362 001142 9$:      MOV      RMMR1I,$BDDAT      ;;CHECK RMMR
70 056310 042737 000046 001142      BIC      #WC!LS!LST,$BDDAT      ;;CLEAR DONT CARES
71 056316 012737 000010 001140      MOV      #MWD,$GDDAT      ;;EXPECT WRITE DATA ON
72 056324 023737 001140 001142      CMP      $GDDAT,$BDDAT      ;;COMPARE EXPECTED AND RECEIVED
73 056332 001413      BEQ      11$          ;;BRANCH IF ZERO
74 056334 062716 000004 000000  ADD      #4,(SP)      ;;MOVE SP TO ERROR CALL
75 056340 112776 000145 000000  MOVB    #145,@(SP)      ;;WRITE NUMBER OF ERROR IN CALL
76 056346 162716 000002  SUB      #2,(SP)      ;;MOVE SP TO RETURN FOR ERROR
77 056352 004736      JSR      PC,@(SP)+    ;;REPORT THE ERROR VIA USER
78 056354 162716 000010  SUB      #10,(SP)     ;;MOVE SP TO NO ERROR RETURN
79 056360 000240      NOP
80                                ;:REPORT ERROR IF RMMR2 NOT INITIALIZED
81 056362 013737 001376 001142 11$:     MOV      RMMR2I,$BDDAT      ;;CHECK RMMR2
82 056370 042737 140000 001142      BIC      #RQA!RQB,$BDDAT      ;;CLEAR RQA, RQB
83 056376 012737 011777 001140      MOV      #TST!1777,$GDDAT      ;;EXPECT TEST BIT ON
84 056404 023737 001140 001142      CMP      $GDDAT,$BDDAT      ;;COMPARE EXPECTED & RECEIVED
85 056412 001413      BEQ      15$          ;;BRANCH IF EQUAL
86 056414 062716 000004 000000  ADD      #4,(SP)      ;;MOVE SP TO ERROR CALL
87 056420 112776 000146 000000  MOVB    #146,@(SP)      ;;WRITE NUMBER OF ERROR IN CALL
88 056426 162716 000002  SUB      #2,(SP)      ;;MOVE SP TO RETURN FOR ERROR
89 056432 004736      JSR      PC,@(SP)+    ;;REPORT THE ERROR VIA USER
90 056434 162716 000010  SUB      #10,(SP)     ;;MOVE SP TO NO ERROR RETURN
91 056440 000240      NOP
92 056442 005037 001140 15$:      CLR      $GDDAT      ;;EXPECT ZEROS
93                                ;:REPORT ERROR IF RMEC2 NOT RESET
94 056446 013737 001404 001142      MOV      RMEC2I,$BDDAT      ;;CHECK RMEC2
95 056454 001413      BEQ      17$          ;;BRANCH IF 0
96 056456 062716 000004 000000  ADD      #4,(SP)      ;;MOVE SP TO ERROR CALL
97 056462 112776 000150 000000  MOVB    #150,@(SP)      ;;WRITE NUMBER OF ERROR IN CALL
98 056470 162716 000002  SUB      #2,(SP)      ;;MOVE SP TO RETURN FOR ERROR
99 056474 004736      JSR      PC,@(SP)+    ;;REPORT THE ERROR VIA USER
100 056476 162716 000010  SUB      #10,(SP)     ;;MOVE SP TO NO ERROR RETURN
101 056502 000240      NOP
102                                ;:REPORT ERROR IF RMER2 NOT RESET
103 056504 013737 001400 001142 17$:     MOV      RMER2I,$BDDAT      ;;CHECK RMER2
104 056512 001413      BEQ      18$          ;;BRANCH IF NO ERROR
105 056514 062716 000004 000000  ADD      #4,(SP)      ;;MOVE SP TO ERROR CALL
106 056520 112776 000147 000000  MOVB    #147,@(SP)      ;;WRITE NUMBER OF ERROR IN CALL
107 056526 162716 000002  SUB      #2,(SP)      ;;MOVE SP TO RETURN FOR ERROR
108 056532 004736      JSR      PC,@(SP)+    ;;REPORT THE ERROR VIA USER
109 056534 162716 000010  SUB      #10,(SP)     ;;MOVE SP TO NO ERROR RETURN
110 056540 000240      NOP
111 056542      18$:
112
113 056542      19$:
114

```



```

1      .SBTTL DATA TRANSFER COMMAND STATUS CHECK SUBROUTINE
2
3      ;THIS SUBROUTINE VERIFIES THE RESULTS OF ALL DATA TRANSFER COMMANDS
4      ;USING STATUS STORED IN THE GET BUFFER AND TEST PARAMETERS
5      ;STORED IN THE PUT BUFFER.  ERRORS ARE REPORTED BY WRITING
6      ;THE ERROR NUMBER IN THE USERS ERROR CALL.
7
8      ;USER'S SUBROUTINE CALL:
9      ;(1) JSR PC,DTASTS
10     ;(2) BR ??
11     ;(3) NOP
12     ;(4) ERROR
13     ;(5) JSR PC,@(SP)+
14     ;(6) ??
15
16     DTASTS:
17 056572
18
19     ;CLEAR USER'S ERROR CALL AND ERROR FLAGS
20 056572 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR
21 056576 105076 000000 CLR B @ (SP) ;CLEAR LOW ORDER BYTE OF TRAP
22 056602 162716 000004 SUB #4,(SP) ;RESTORE SP TO NO ERROR
23 056606 005037 062166 CLR 500$ ;CLEAR ERROR FLAGS
24
25     ;REPORT ANY CONTROL BUS PARITY ERROR WHILE READING REMOTE REGISTERS,
26     ;I.E., MCPE = 1
27 056612 032737 020000 001336 BIT #MCPE,RMCS1I ;WAS THERE A PARITY ERROR??
28 056620 001422 BEQ 10$ ;NO!!
29 056622 013737 001336 001140 MOV RMCS1I,$GDDAT ;EXPECTED STATUS
30 056630 042737 020000 001140 BIC #MCPE,$GDDAT
31 056636 013737 001336 001142 MOV RMCS1I,$BDDAT ;RECEIVED STATUS
32 056644 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
33 056650 112776 000013 000000 MOVB #13,@(SP) ;WRITE ERROR NUMBER
34 056656 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN IF ERROR
35 056662 004736 JSR PC,@(SP)+ ;REPORT ERROR AND RETURN
36 056664 000466 BR 30$
37 056666
38
39     ;REPORT ANY CONTROL BUS PARITY ERROR WHILE WRITTING REMOTE REGISTERS,
40     ;I.E., PAR = 1 AND DPE = 0
41 056666 032737 000010 001352 BIT #PAR,RMER1I ;WAS THERE A PARITY ERROR??
42 056674 001435 BEQ 20$ ;NO!!
43 056676 032737 000010 001400 BIT #DPE,RMER2I ;DATA PARITY ERROR ?
44 056704 001031 BNE 20$ ;YES!!
45 056706 013737 001352 001140 MOV RMER1I,$GDDAT ;EXPECTED STATUS
46 056714 042737 000010 001140 BIC #PAR,$GDDAT
47 056722 013737 001352 001142 MOV RMER1I,$BDDAT ;RECEIVED STATUS
48 056730 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR
49 056734 112776 000050 000000 MOVB #50,@(SP) ;WRITE ERROR NUMBER
50 056742 032737 001000 001346 BIT #MXF,RMCS2I ;DID MXF GET SET??
51 056750 001003 BNE 15$ ;YES!!
52 056752 112776 000274 000000 MOVB #274,@(SP) ;NO - CHANGE ERROR NUMBER
53 056760 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN IF ERROR
54 056764 004736 JSR PC,@(SP)+ ;REPORT ERROR AND RETURN
55 056766 000425 BR 30$
56
57     ;LOOK FOR ANY ERRORS WHICH MAY HAVE OCCURRED DURING COMMAND INITIATION OR
  
```

```

58      ;MECHANICAL POSITIONING
59
60      ;FIRST TEST MXF WHICH WOULD INDICATE COMPOSITE ERROR SET WHEN FUNCTION
61      ;CODE AND GO BIT WERE LOADED
62      20$:
63      056770 032737 001000 001346      BIT      #MXF,RMCS2I      ;WAS 'MISSED TRANSFER' SET??
64      056776 001425                      BEQ      40$              ;NO..
65      057000 013737 001346 001140      MOV      RMCS2I,$GDDAT      ;EXPECTED STATUS
66      057006 042737 001000 001 40      BIC      #MXF,$GDDAT
67      057014 013737 001346 001142      MOV      RMCS2I,$BDDAT      ;RECEIVED STATUS
68      057022 062716 000004              ADD      #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
69      057026 112776 000275 000000      MOVB     #275,@(SP)        ;WRITE ERROR NUMBER
70      057034 162716 000002              SUB      #2,(SP)          ;MOVE SP TO RETURN IF ERROR
71      057040 004736                      JSR      PC,@(SP)+        ;REPORT ERROR AND RETURN
72      057042
73
74      30$:
75      057042 162716 000010              SUB      #10,(SP)        ;MOVE SP TO NO ERROR
76      057046 000137 062140              JMP      380$          ;SKIP TO END OF SUB
77
78      40$:
79
80      ;REPORT AN ERROR IF 'OPI' ERROR OCCURRED DUE TO 'MOL' - 0, OR IF 'OPI'
81      ;AND 'MOL' ARE SET, BUT 'VV' IS RESET, INDICATING AN INTERMITTENT
82      ;'MOL'
83      057052 032737 020000 001352      BIT      #OPI,RMER1I      ;IS 'OPI' SET??
84      057060 001447                      BEQ      60$              ;NO..
85      057062 013737 001352 001140      MOV      RMER1I,$GDDAT      ;EXPECTED STATUS
86      057070 042737 020000 001140      BIC      #OPI,$GDDAT
87      057076 013737 001352 001142      MOV      RMER1I,$BDDAT      ;RECEIVED STATUS
88      057104 032737 010000 001350      BIT      #MOL,RMDSI      ;WAS MEDIUM OFF LINE??
89      057112 001404                      BEQ      45$              ;YES!!
90      057114 032737 000100 001350      BIT      #VV,RMDSI        ;WAS 'MOL' INTERMITTENT??
91      057122 001013                      BNE      50$              ;NO!!
92      057124 062716 000004              ADD      #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
93      057130 112776 000276 000000      MOVB     #276,@(SP)        ;WRITE ERROR NUMBER IN CALL
94      057136 162716 000002              SUB      #2,(SP)          ;MOVE SP TO RETURN IF ERROR
95      057142 004736                      JSR      PC,@(SP)+        ;REPORT ERROR AND RETURN
96      057144 162716 000010              SUB      #10,(SP)        ;RESTORE SP TO NO ERROR
97      057150 000413                      BR       60$
98      057152
99
100     50$:
101     ;REPORT 'OPI' ERROR, WHICH IS DUE TO 'ON CYLINDER' NOT DROPPING OR
102     ;'RUN' TIMEOUT (20 MS) OR SEARCH TIMEOUT (50 MS)
103     057152 062716 000004              ADD      #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
104     057156 112776 000277 000000      MOVB     #277,@(SP)        ;WRITE ERROR NUMBER IN CALL
105     057164 162716 000002              SUB      #2,(SP)          ;MOVE SP TO RETURN IF ERROR
106     057170 004736                      JSR      PC,@(SP)+        ;REPORT ERROR AND RETURN
107     057172 162716 000010              SUB      #10,(SP)        ;RESTORE SP TO NO ERROR
108     057176 000240                      NOP
109
110     60$:
111     ;LOOK FOR 'IVC' ERROR DURING COMMAND INITIATION
112     057200 032737 010000 001400      BIT      #IVC,RMER2I      ;WAS THERE AN 'IVC' ERROR??
113     057206 001432                      BEQ      70$              ;NO..
114     057210 013737 001400 001140      MOV      RMER2I,$GDDAT      ;EXPECTED STATUS

```

```

115 057216 042737 010000 001140 BIC #IVC,$GDDAT
116 057224 013737 001400 001142 MOV RMER2I,$BDDAT ;RECEIVED STATUS
117 057232 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR
118 057236 112776 000300 000000 MOVB #300,@(SP) ;WRITE ERROR NUMBER IN CALL
119 057244 032737 000100 001350 BIT #VV,RMDSI ;WAS VOLUME VALID??
120 057252 001403 BEQ 65$ ;NO.
121 057254 112776 000301 000000 MOVB #301,@(SP) ;CHANGE ERROR NUMBER
122 057262 162716 000002 65$: SUB #2,(SP) ;MOVE SP TO RETURN IF ERROR
123 057266 004736 JSR PC,@(SP)+ ;REPORT 'IVC' ERROR AND RETURN
124 057270 162716 000010 SUB #10,(SP) ;RESTORE SP TO NO ERROR
125 057274 70$:
126
127 ;SEE IF 'ILF' OR 'RMR' IS SET
128 057274 032737 000007 001352 BIT #ILR,ILF.RMR,RMER1I
129 057302 001510 BEQ 100$ ;NO ERRORS DETECTED
130 ;REPORT AN ERROR IF 'ILR' IS SET
131 057304 032737 000002 001352 BIT #ILR,RMER1I ;WAS 'ILR' DETECTED??
132 057312 001424 BFQ 80$ ;NO.
133 057314 013737 001352 001140 MOV RMER1I,$GDDAT ;EXPECTED STATUS
134 057322 042737 000002 001140 BIC #ILR,$GDDAT
135 057330 013737 001352 001142 MOV RMER1I,$BDDAT ;RECEIVED STATUS
136 057336 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
137 057342 112776 000302 000000 MOVB #302,@(SP) ;WRITE ERROR NUMBER IN CALL
138 057350 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN IF ERROR
139 057354 004736 JSR PC,@(SP)+ ;REPORT ERROR AND RETURN
140 057356 162716 000010 SUB #10,(SP) ;RESTORE SP TO NO ERROR
141 057362 000240 NOP
142 057364 80$:
143
144 ;REPORT AN ERROR IF 'ILF' IS SET
145 057364 032737 000001 001352 BIT #ILF,RMER1I ;WAS 'ILF' DETECTED??
146 057372 001424 BEQ 90$ ;NO.
147 057374 013737 001352 001140 MOV RMER1I,$GDDAT ;EXPECTED STATUS
148 057402 042737 000001 001140 BIC #ILF,$GDDAT
149 057410 013737 001352 001142 MOV RMER1I,$BDDAT ;RECEIVED STATUS
150 057416 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
151 057422 112776 000303 000000 MOVB #303,@(SP) ;WRITE ERROR NUMBER IN CALL
152 057430 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN IF ERROR
153 057434 004736 JSR PC,@(SP)+ ;REPORT ERROR AND RETURN
154 057436 162716 000010 SUB #10,(SP) ;RESTORE SP TO NO ERROR
155 057442 000240 NOP
156 057444 90$:
157
158 ;REPORT AN ERROR IF 'RMR' IS SET
159 057444 032737 000004 001352 BIT #RMR,RMER1I ;WAS 'RMR' DETECTED??
160 057452 001424 BEQ 100$ ;NO.
161 057454 013737 001352 001140 MOV RMER1I,$GDDAT ;EXPECTED STATUS
162 057462 042737 000004 001140 BIC #RMR,$GDDAT
163 057470 013737 001352 001142 MOV RMER1I,$BDDAT ;RECEIVED STATUS
164 057476 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
165 057502 112776 000304 000000 MOVB #304,@(SP) ;WRITE ERROR NUMBER IN CALL
166 057510 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN IF ERROR
167 057514 004736 JSR PC,@(SP)+ ;REPORT ERROR AND RETURN
168 057516 162716 000010 SUB #10,(SP) ;RESTORE SP TO NO ERROR
169 057522 000240 NOP
170 057524 100$:
171 ;DETERMINE WHETHER OR NOT 'IAE' SHOULD BE SET AND CHECK FOR ERROR

```

```

172 057524 012737 002000 001140      MOV      #IAE,$GDDAT      ;SETUP FOR 'IAE' - 1
173 057532 052737 040000 062166      BIS      #SKI,500$      ;SETUP FOR 'SKI' - 1
174 057540 023727 001446 001466      CMP      RMDCO,#822.      ;GREATER THAN LAST CYLINDER ?
175 057546 101025          BHJ      110$      ;YES - CYLINDER IS INVALID
176 057550 042737 040000 062166      BIC      #SKI,500$      ;RESET SKI FLAG
177
178 057556 123737 001421 001335      CMPB     RMDAO+1,LSTRK+1 ;GREATER THAN LAST TRACK ?
179 057564 101016          BHJ      110$      ;YES - TRACK IS INVALID
180
181 057566 123727 001420 000035      CMPB     RMDAO,#29.      ;IS SECTOR > 29. ?
182 057574 101410          BLOS     105$      ;NO
183 057576 032737 010000 001444      BIT      #FMT16,RMOFO    ;18 BIT FORMAT ?
184 057604 001406          BEQ      110$      ;YES - SECTOR IS INVALID FOR 18 BIT MODE
185 057606 123727 001420 000037      CMPB     RMDAO,#31.      ;IS SECTOR > 31. ?
186 057614 101002          BHJ      110$      ;YES - SECTOR IS INVALID
187 057616 005037 001140          CLR      $GDDAT      ;'IAE' SHOULD = 0
188
189 057622 013737 001352 001142 110$: MOV      RMER1I,$BDDAT      ;GET RECEIVED STATUS
190 057630 042737 175777 001142      BIC      #^CIAE,$BDDAT
191 057636 023737 001140 001142      CMP      $GDDAT,$BDDAT  ;IS 'IAE' STATUS OK??
192 057644 001004          BNE      115$      ;NO!
193 057646 042737 040000 062166      BIC      #SKI,500$      ;IAE OK - SKI SHOULD BE 0
194 057654 000412          BR       120$
195 057656 062716 000004          ADD      #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
196 057662 112776 000305 000000 115$: MOVB     #305,@(SP)      ;WRITE ERROR NUMBER
197 057670 162716 000002          SUB      #2,(SP)      ;MOVE SP TO RETURN IF ERROR
198 057674 004736          JSR      PC,@(SP)+    ;REPORT ERROR AND RETURN
199 057676 162716 000010          SUB      #10,(SP)     ;MOVE SP TO NO ERROR
200 057702          120$:
201
202      ;REPORT AN ERROR IF 'SKI' IS SET AND 'IAE' STATUS WAS OK
203 057702 013737 001400 001142      MOV      RMER2I,$BDDAT  ;RECEIVED STATUS
204 057710 042737 137777 001142      BIC      #^CSKI,$BDDAT
205 057716 013737 062166 001140      MOV      500$,$GDDAT    ;EXPECTED STATUS
206 057724 042737 137777 001140      BIC      #^CSKI,$GDDAT
207 057732 032737 040000 001400      BIT      #SKI,RMER2I    ;WAS 'SKI' SET??
208 057740 001417          BEQ      140$      ;NO!!
209 057742 032737 040000 062166      BIT      #SKI,500$      ;WAS SKI CAUSED BY IAE - 0??
210 057750 001032          BNE      150$      ;YES - DON'T REPORT SKI
211 057752 062716 000004          ADD      #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
212 057756 112776 000306 000000 140$: MOVB     #306,@(SP)      ;WRITE ERROR NUMBER
213 057764 162716 000002          SUB      #2,(SP)      ;MOVE SP TO RETURN IF ERROR
214 057770 004736          JSR      PC,@(SP)+    ;REPORT ERROR AND RETURN
215 057772 162716 000010          SUB      #10,(SP)     ;MOVE SP TO NO ERROR
216 057776 000417          BR       150$
217
218 060000          140$:
219
220      ;REPORT AN ERROR IF SKI 0 AND IAE WAS NOT DETECTED
221 060000 032737 040000 062166      BIT      #SKI,500$      ;SHOULD SKI BE SET??
222 060006 001413          BEQ      150$      ;NO!
223 060010 062716 000004          ADD      #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
224 060014 112776 000307 000000 150$: MOVB     #307,@(SP)      ;WRITE ERROR NUMBER IN CALL
225 060022 162716 000002          SUB      #2,(SP)      ;MOVE SP TO RETURN IF ERROR
226 060026 004736          JSR      PC,@(SP)+    ;REPORT ERROR AND RETURN
227 060030 162716 000010          SUB      #10,(SP)     ;RESTORE SP TO NO ERROR
228 060034 000240          NOP

```

```

229 060036      150$:
230
231      ;LOOK FOR 'LSC' OR 'LBC' OR 'DVC' IN ERROR REGISTER #2
232 060036 032737 006200 001400      BIT      #LSC!LBC!DVC,RMER2I
233 060044 001512      BEQ      180$      ;NO ERRORS SET
234
235      ;REPORT ANY DEVICE FAULT, I.E., 'DVC' = 1
236 060046 032737 000200 001400      BIT      #DVC,RMER2I      ;IS 'DVC' = 1??
237 060054 001424      BEQ      160$      ;NO!!
238 060056 013737 001400 001140      MOV      RMER2I,$GDDAT      ;EXPECTED STATUS
239 060064 042737 000200 001140      BIC      #DVC,$GDDAT
240 060072 013737 001400 001142      MOV      RMER2I,$BDDAT      ;RECEIVED STATUS
241 060100 062716 000004      ADD      #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
242 060104 112776 000310 000000      MOVB     #310,a(SP)      ;WRITE ERROR NUMBER IN CALL
243 060112 162716 000002      SUB      #2,(SP)      ;MOVE SP TO RETURN IF ERROR
244 060116 004736      JSR      PC,a(SP)+      ;REPORT ERROR AND RETURN
245 060120 162716 000010      SUB      #10,(SP)      ;RESTORE SP TO NO ERROR
246 060124 000240      NOP
247 060126
248
249      160$:
250      ;REPORT LOSS OF BIT CLOCK, I.E., 'LBC' = 1, IF 'MOL' = 1
251 060126 032737 002000 001400      BIT      #LBC,RMER2I      ;IS LBC SET??
252 060134 001430      BEQ      170$      ;NO!!
253 060136 032737 010000 001350      BIT      #MOL,RMDSI      ;WAS LBC ERROR BY MOL = 0
254 060144 001424      BEQ      170$      ;YES!!
255 060146 013737 001400 001140      MOV      RMER2I,$GDDAT      ;EXPECTED STATUS
256 060154 042737 002000 001140      BIC      #LBC,$GDDAT
257 060162 013737 001400 001142      MOV      RMER2I,$BDDAT      ;RECEIVED STATUS
258 060170 062716 000004      ADD      #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
259 060174 112776 000311 000000      MOVB     #311,a(SP)      ;WRITE ERROR NUMBER IN CALL
260 060202 162716 000002      SUB      #2,(SP)      ;MOVE SP TO RETURN IF ERROR
261 060206 004736      JSR      PC,a(SP)+      ;REPORT ERROR AND RETURN
262 060210 162716 000010      SUB      #10,(SP)      ;RESTORE SP TO NO ERROR
263 060214 000240      NOP
264 060216
265      170$:
266      ;REPORT LOS OF SYSTEM CLOCK, I.E., 'LSC' = 1
267 060216 032737 004000 001400      BIT      #LSC,RMER2I      ;IS 'LSC' = 1??
268 060224 001422      BEQ      180$      ;NO!!
269 060226 013737 001400 001140      MOV      RMER2I,$GDDAT      ;EXPECTED STATUS
270 060234 042737 004000 001140      BIC      #LSC,$GDDAT
271 060242 013737 001400 001142      MOV      RMER2I,$BDDAT      ;RECEIVED STATUS
272 060250 062716 000004      ADD      #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
273 060254 112776 000312 000000      MOVB     #312,a(SP)      ;WRITE ERROR NUMBER
274 060262 004736      JSR      PC,a(SP)+      ;REPORT ERROR AND RETURN
275 060264 162716 000010      SUB      #10,(SP)      ;RESTORE SP TO NO ERROR
276 060270 000240      NOP
277 060272
278      180$:
279      ;LOOK FOR 'UNS' OR 'DTE' OR 'WLE' IN ERROR REGISTER #1
280 060272 032737 054000 001352      BIT      #UNS!DTE!WLE,RMER1I
281 060300 001527      BEQ      220$      ;NO BITS SET
282      ;REPORT 'UNS' ERROR IF 'DVC' = 0
283 060302 032737 040000 001352      BIT      #UNS,RMER1I      ;IS 'UNS' SET??
284 060310 001427      BEQ      190$      ;NO!!
285 060312 032737 000200 001400      BIT      #DVC,RMER2I      ;WAS 'UNS' CAUSED BY 'DVC'?
286 060320 001023      BNE      190$      ;YES..

```

```

286 060322 013737 001352 001140      MOV    RMER11,$GDDAT      ;EXPECTED STATUS
287 060330 042737 040000 001140      BIC    #UNS,$GDDAT
288 060336 013737 001352 001142      MOV    RMER11,$BDDAT      ;RECEIVED STATUS
289 060344 062716 000004                ADD    #4,(SP)          ;MOVE SP TO USERS ERROR CALL
290 060350 112776 000313 000000      MOV    #313,@(SP)        ;WRITE ERROR NUMBER
291 060356 162716 000002                SUB    #2,(SP)          ;MOVE SP TO RETURN IF ERROR
292 060362 004736                JSR    PC,@(SP)+      ;REPORT ERROR AND RETURN
293 060364 162716 000010                SUB    #10,(SP)         ;RESTORE SP TO NO ERROR
294 060370
295
296                                190$:
297 060370 032737 010000 001352      ;REPORT ANY DRIVE TIMING ERROR, I.E., 'DTE' = 1
298 060376 001423                BIT    #DTE,RMER11      ;IS DTE SET??
299 060400 013737 001352 001140      BEQ    200$          ;NO!!
300 060406 042737 010000 001140      MOV    RMER11,$GDDAT      ;EXPECTED STATUS
301 060414 013737 001352 001142      BIC    #DTE,$GDDAT
302 060422 062716 000004                MOV    RMER11,$BDDAT      ;RECEIVED STATUS
303 060426 112776 000314 000000      ADD    #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
304 060434 162716 000002                MOV    #314,@(SP)        ;WRITE ERROR NUMBER IN CALL
305 060440 004736                SUB    #2,(SP)          ;MOVE SP TO RETURN IF ERROR
306 060442 162716 000010                JSR    PC,@(SP)+      ;REPORT ERROR AND RETURN
307 060446                SUB    #10,(SP)         ;MOVE SP TO NO ERROR
308
309                                200$:
310                                ;REPORT AN ERROR IF WRITE LOCK ERROR IS SET. SEE IF DRIVE IS NOT
311 060446 032737 004000 001352      ;WRITE PROTECTED, OR IF FUNCTION WAS NOT A WRITE
312 060454 001441                BIT    #WLE,RMER11      ;WAS 'WLE' SET??
313 060456 013737 001352 001142      BEQ    220$          ;NO!!
314 060464 013737 001352 001140      MOV    RMER11,$BDDAT      ;RECEIVED STATUS
315 060472 052737 004000 001140      MOV    RMER11,$GDDAT      ;EXPECTED STATUS
316 060500 062716 000004                BIS    #WLE,$GDDAT
317 060504 112776 000315 000000      ADD    #4,(SP)          ;MOVE SP TO USERS ERROR CALL
318 060512 032737 004000 001350      MOV    #315,@(SP)        ;WRITE ERROR NUMBER IN CALL
319 060520 001404                BIT    #WRL,RMDS1      ;WAS DRIVE WRITE PROTECTED??
320 060522 032737 000010 001412      BEQ    205$          ;NO!!
321 060530 001406                BIT    #BIT3,RMCS10      ;WAS COMMAND A WRITE??
322 060532 112776 000316 000000      BEQ    210$          ;YES!!
323 060540 042737 004000 001140      MOV    #316,@(SP)        ;CHANGE ERROR NUMBER
324 060546 162716 000002                BIC    #WLE,$GDDAT
325 060552 004736                SUB    #2,(SP)          ;MOVE SP TO RETURN IF ERROR
326 060554 162716 000010                JSR    PC,@(SP)+      ;REPORT ERROR AND RETURN
327                                SUB    #10,(SP)         ;MOVE SP TO NO ERROR
328                                210$:
329                                220$:
330                                ;OMIT DATA ERROR CHECKS IF ANY PREVIOUS ERRORS HAVE BEEN DETECTED
331 060560 062716 000004                ADD    #4,(SP)          ;MOVE SP TO USER'S ERROR
332 060564 105776 000000                TSTB    @(SP)          ;WAS ERROR DETECTED??
333 060570 001404                BEQ    225$          ;NO - DO DATA CHECKS
334 060572 162716 000004                SUB    #4,(SP)          ;RESTORE SP
335 060576 000137 061600                JMP    340$          ;SKIP DATA CHECKS
336 060602 162716 000004                SUB    #4,(SP)          ;RESTORE SP
337
338                                225$:
339                                ;CHECK HEADER ERRORS IF FUNCTION WAS NOT WRITE HEADER AND DATA, AND
340 060606 013737 001412 062170      ;IF HEADER COMPARE IS NOT INHIBITED
341 060614 042737 177700 062170      MOV    RMCS10,510$      ;STRIP AND STORE FUNCTION CODE
342 060622 022737 000063 062170      BIC    #^CFNCMSK,510$
      CMP    #WH!GO,510$      ;WAS FUNCTION WRITE HEADER & DATA??

```



```

343 060630 001512      BEQ      250$      ;YES - SKIP HEADER CHECKS
344 060632 032737 002000 001370      BIT      #HCI,RMOFI      ;WAS HCI SET??
345 060640 001106      BNE      250$      ;YES - SKIP HEADER CHECKS
346
347      ;SEE IF ANY HEADER ERRORS ARE SET, I.E., 'FER' OR 'HCRC' OR 'HCE'
348 060642 032737 000620 001352      BIT      #HCRC!FER,HCE,RMER11
349 060650 001533      BEQ      270$      ;NO ERRORS SET
350
351      ;REPORT HEADER CRC ERROR IF SET
352 060652 032737 000400 001352      BIT      #HCRC,RMER11      ;WAS HCRC SET??
353 060660 001422      BEQ      230$      ;NO!!
354 060662 013737 001352 001140      MOV      RMER11,$GDDAT      ;EXPECTED STATUS
355 060670 042737 000400 001140      BIC      #HCRC,$GDDAT
356 060676 013737 001352 001142      MOV      RMER11,$BDDAT      ;RECEIVED STATUS
357 060704 062716 000004      ADD      #4,(SP)      ;MOVE SP TO USERS ERROR
358 060710 112776 000317 000000      MOV      #317,@(SP)      ;WRITE ERROR NUMBER
359 060716 162716 000002      SUB      #2,(SP)      ;MOVE SP TO RETURN IF ERROR
360 060722 004736      JSR      PC,@(SP)+      ;REPORT ERROR AND RETURN
361 060724 000501      BR
362 060726      230$:
363
364      ;REPORT FORMAT ERROR IF SET
365 060726 032737 000020 001352      BIT      #FER,RMER11      ;WAS 'FER' SET??
366 060734 001422      BEQ      240$      ;NO!!
367 060736 013737 001352 001140      MOV      RMER11,$GDDAT      ;EXPECTED STATUS
368 060744 042737 000020 001140      BIC      #FER,$GDDAT
369 060752 013737 001352 001142      MOV      RMER11,$BDDAT      ;RECEIVED STATUS
370 060760 062716 000004      ADD      #4,(SP)      ;MOVE SP TO USERS ERROR
371 060764 112776 000320 000000      MOV      #320,@(SP)      ;WRITE ERROR NUMBER
372 060772 162716 000002      SUB      #2,(SP)      ;MOVE SP TO RETURN IF ERROR
373 060776 004736      JSR      PC,@(SP)+      ;REPORT ERROR AND RETURN
374 061000 000453      BR
375 061002      240$:
376
377      ;REPORT HEADER COMPARE ERROR IF SET
378 061002 032737 000200 001352      BIT      #HCE,RMER11      ;WAS 'HCE' SET??
379 061010 001453      BEQ      270$      ;NO!!
380 061012 013737 001352 001140      MOV      RMER11,$GDDAT      ;EXPECTED STATUS
381 061020 042737 000200 001140      BIC      #HCE,$GDDAT
382 061026 013737 001352 001142      MOV      RMER11,$BDDAT      ;RECEIVED STATUS
383 061034 062716 000004      ADD      #4,(SP)      ;MOVE SP TO USER'S ERROR
384 061040 112776 000321 000000      MOV      #321,@(SP)      ;WRITE ERROR NUMBER
385 061046 162716 000002      SUB      #2,(SP)      ;MOVE SP TO RETURN IF ERROR
386 061052 004736      JSR      PC,@(SP)+      ;REPORT ERROR AND RETURN
387 061054 000425      BR
388      ;THERE SHOULD BE NO HEADER ERRORS BECAUSE
389      ;.COMMAND WAS WRITE HEADER AND DATA, OR
390      ;.HEADER COMPARE INHIBIT WAS SET
391 061056 032737 000620 001352 250$: BIT      #HCE!FER!HCRC,RMER11
392 061064 001425      BEQ      270$      ;NO ERRORS WERE SET
393 061066 013737 001352 001140      MOV      RMER11,$GDDAT      ;EXPECTED STATUS
394 061074 042737 000620 001140      BIC      #HCE!FER!HCRC,$GDDAT
395 061102 013737 001352 001142      MOV      RMER11,$BDDAT      ;RECEIVED STATUS
396 061110 062716 000004      ADD      #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
397 061114 112776 000322 000000      MOV      #322,@(SP)      ;WRITE ERROR NUMBER
398 061122 162716 000002      SUB      #2,(SP)      ;MOVE SP TO RETURN IF ERROR
399 061126 004736      JSR      PC,@(SP)+      ;REPORT ERROR AND RETURN

```

```

400 061130 162716 000010      260$: SUB #10,(SP)      ;MOVE SP TO NO ERROR
401 061134 000137 061600      JMP 340$      ;OMIT FURTHER DATA CHECKS
402
403 061140      270$:
404
405      ;IF COMMAND WAS A WRITE COMMAND, GO DO WRITE ERROR CHECKS, OTHERWISE
406      ;DO READ ERROR CHECKS
407 061140 032737 000010 062170      BIT #BIT3,510$      ;WAS THIS A WRITE COMMAND?
408 061146 001002      BNE 275$      ;NO..
409 061150 000137 061366      JMP 310$      ;GO DO WRITE STATUS CHECK
410 061154
411
412      ;REPORT DATA CHECK IF SET
413 061154 032737 100000 001352      BIT #DCK,RMER11      ;DATA CHECK ERROR??
414 061162 001450      BEQ 290$      ;NO..
415 061164 013737 001352 001140      MOV RMER11,$GDDAT      ;EXPECTED STATUS
416 061172 042737 100000 001140      BIC #DCK,$GDDAT
417 061200 013737 001352 001142      MOV RMER11,$BDDAT      ;RECEIVED STATUS
418 061206 062716 000004      ADD #4,(SP)      ;MOVE SP TO USER'S ERROR
419 061212 112776 000323 000000      MOV #323,a(SP)      ;WRITE ERROR NUMBER
420 061220 032737 004000 001370      BIT #EC1,RMOFI      ;WAS ECC CORRECTION DISABLED??
421 061226 001021      BNE 280$      ;YES!!
422 061230 112776 000324 000000      MOV #324,a(SP)      ;CHANGE TO RECOVERABLE ERROR
423 061236 032737 000100 001352      BIT #ECH,RMER11      ;IS ERROR RECOVERABLE??
424 061244 001007      BNE 276$      ;NO !!
425      ;DO NOT REPORT RECOVERABLE ERROR IF READ COMMAND
426 061246 052737 000020 062170      BIT #BIT4,510$      ;WAS THIS A READ COMMAND ??
427 061254 001406      BEQ 280$      ;NO !!
428 061256 162716 000004      SUB #4,(SP)      ;RESTORE SP
429 061262 000410      BR 290$      ;SKIP ERROR - DATA WILL BE CORRECTED
430 061264 112776 000325 000000      MOV #325,a(SP)      ;CHANGE TO NON RECOVERABLE
431 061272 162716 000002      276$: SUB #2,(SP)      ;MOVE SP TO RETURN IF ERROR
432 061276 004736      280$: JSR PC,a(SP)+      ;REPORT ERROR AND RETURN
433 061300 162716 000010      SUB #10,(SP)      ;RESTORE SP TO NO ERROR
434
435 061304      290$:
436
437      ;REPORT DATA BUS PARITY ERROR IF SET, I.E., MDPE - 1
438 061304 032737 000400 001346      BIT #MDPE,RMCS21      ;PARITY ERROR SET??
439 061312 001423      BEQ 300$      ;NO!!
440 061314 013737 001346 001140      MOV RMCS21,$GDDAT      ;EXPECTED STATUS
441 061322 042737 000400 001140      BIC #MDPE,$GDDAT
442 061330 013737 001346 001142      MOV RMCS21,$BDDAT      ;RECEIVED STATUS
443 061336 062716 000004      ADD #4,(SP)      ;MOVE SP TO USER'S ERROR
444 061342 112776 000326 000000      MOV #326,a(SP)      ;WRITE ERROR NUMBER
445 061350 162716 000002      SUB #2,(SP)      ;MOVE SP TO RETURN IF ERROR
446 061354 004736      JSR PC,a(SP)+      ;REPORT ERROR AND RETURN
447 061356 162716 000010      SUB #10,(SP)      ;MOVE SP TO NO ERROR
448 061362 000137 061600      300$: JMP 340$      ;SKIP WRITE STATUS CHECK
449
450 061366      310$:
451
452      ;TEST TO SEE THAT OFFSET MODE WAS RESET; REPORT ERROR IF 'OM' 1
453 061366 032737 000001 001350      BIT #OM,RMDS1      ;IS OFFSET ON??
454 061374 001423      BEQ 320$      ;NO
455 061376 013737 001350 001140      MOV RMDS1,$GDDAT      ;EXPECTED STATUS
456 061404 042737 000001 001140      BIC #OM,$GDDAT

```



```

457 061412 013737 001350 001142      MOV      RMDSI,$BDDAT      ;RECEIVED STATUS
458 061420 062716 000004      ADD      #4,(SP)        ;MOVE SP TO USER'S ERROR CALL
459 061424 112776 000327 000000      MOVB    #327,a(SP)      ;WRITE ERROR NUMBER IN CALL
460 061432 162716 000002      SUB      #2,(SP)        ;MOVE SP TO RETURN IF ERROR
461 061436 004736      JSR      PC,a(SP)+    ;REPORT ERROR AND RETURN
462 061440 162716 000010      SUB      #10,(SP)      ;MOVE SP TO NO ERROR
463 061444      320$:
464
465      ;TEST FOR DATA BUS PARITY ERROR; REPORT ERROR IF 'DPE' = 1
466 061444 032737 000010 001400      BIT      #DPE,RMER2I    ;DATA PARITY ERROR??
467 061452 001423      BEQ      330$      ;NO!!
468 061454 013737 001400 001140      MOV      RMER2I,$GDDAT    ;EXPECTED STATUS
469 061462 042737 000010 001140      BIC      #DPE,$GDDAT
470 061470 013737 001400 001142      MOV      RMER2I,$BDDAT    ;RECEIVED STATUS
471 061476 062716 000004      ADD      #4,(SP)        ;MOVE SP TO USER'S ERROR CALL
472 061502 112776 000330 000000      MOVB    #330,a(SP)      ;WRITE ERROR NUMBER
473 061510 162716 000002      SUB      #2,(SP)        ;MOVE SP TO RETURN IF ERROR
474 061514 004736      JSR      PC,a(SP)+    ;REPORT ERROR AND RETURN
475 061516 162716 000010      SUB      #10,(SP)      ;MOVE SP TO NO ERROR
476 061522      330$:
477
478      ;TEST FOR WRITE CLOCK FAILURE; REPORT ERROR IF 'WCF' = 1
479 061522 032737 000040 001352      BIT      #WCF,RMER1I    ;IS 'WCF' SET??
480 061530 001423      BEQ      340$      ;NO!!
481 061532 013737 001352 001140      MOV      RMER1I,$GDDAT    ;EXPECTED STATUS
482 061540 042737 000040 001140      BIC      #WCF,$GDDAT
483 061546 013737 001352 001142      MOV      RMER1I,$BDDAT    ;RECEIVED STATUS
484 061554 062716 000004      ADD      #4,(SP)        ;MOVE SP TO USER'S ERROR CALL
485 061560 112776 000331 000000      MOVB    #331,a(SP)      ;WRITE ERROR NUMBER
486 061566 162716 000002      SUB      #2,(SP)        ;MOVE SP TO RETURN IF ERROR
487 061572 004736      JSR      PC,a(SP)+    ;REPORT ERROR AND RETURN
488 061574 162716 000010      SUB      #10,(SP)      ;MOVE SP TO NO ERROR
489 061600      340$:
490
491      ;REPORT 'DATA LATE' ERROR IF 'DLT' = 1
492 061600 032737 100000 001346      BIT      #DLT,RMCS2I    ;IS 'DLT' SET??
493 061606 001423      BEQ      350$      ;NO!!
494 061610 013737 001346 001140      MOV      RMCS2I,$GDDAT    ;EXPECTED STATUS
495 061616 042737 100000 001140      BIC      #DLT,$GDDAT
496 061624 013737 001346 001142      MOV      RMCS2I,$BDDAT    ;RECEIVED STATUS
497 061632 062716 000004      ADD      #4,(SP)        ;MOVE SP TO USER'S ERROR CALL
498 061636 112776 000332 000000      MOVB    #332,a(SP)      ;WRITE ERROR NUMBER
499 061644 162716 000002      SUB      #2,(SP)        ;MOVE SP TO RETURN IF ERROR
500 061650 004736      JSR      PC,a(SP)+    ;REPORT ERROR AND RETURN
501 061652 162716 000010      SUB      #10,(SP)      ;MOVE SP TO NO ERROR
502 061656      350$:
503
504      ;LOOK FOR UNEXPECTED CHANGES IN DRIVE STATUS
504 061656 013746 001350      MOV      RMDSI,-(SP)    ;STACK DRIVE STATUS
505 061662 042716 147c77      BIC      #^C<PIP!MOL!VV>,(SP) ;CLEAR DONT CAPES
506 061666 022726 010100      CMP      #MOL!VV,(SP)+    ;IS DRIVE STATUS OK??
507 061672 001522      BEQ      380$      ;YES!!
508
509      ;REPORT ERROR IF POSITIONING IN PROGRESS AND NO SEEK INCOMPLETE ERROR,
510      ;I.E. PIP = 1 AND SKI = 0
511 061674 032737 020000 001350      BIT      #PIP,RMDSI    ;IS 'PIP' SET??
512 061702 001430      BEQ      360$      ;NO!!
513 061704 032737 040000 001400      BIT      #SKI,RMER2I    ;WAS 'SKI' ERROR REPORTED??
  
```

```

514 061712 001024      BNE      360$      ;YES-DONT REPORT PIP
515 061714 013737 001350 001140      MOV      RMDSI,$GDDAT ;EXPECTED STATUS
516 061722 042737 020000 001140      BIC      #PIP,$GDDAT
517 061730 013737 001350 001142      MOV      RMDSI,$BDDAT ;RECEIVED STATUS
518 061736 062716 000004      ADD      #4,(SP) ;MOVE SP TO USERS ERROR CALL
519 061742 112776 000333 000000      MOV      #333,a(SP) ;WRITE ERROR NUMBER
520 061750 162716 000002      SUB      #2,(SP) ;MOVE SP TO RETURN IF ERROR
521 061754 004736      JSR      PC,a(SP)+ ;REPORT ERROR AND RETURN
522 061756 162716 000010      SUB      #10,(SP) ;MOVE SP TO NO ERROR
523 061762 000240      NOP
524 061764      360$:
525
526      ;REPORT ERROR IF MEDIUM IS NOT ON LINE AND OPI ERROR WAS NOT
527      ;REPORTED, I.E., MOL = OPI = 0
528 061764 032737 010000 001350      BIT      #MOL,RMDSI ;IS MEDIUM ON LINE??
529 061772 001027      BNE      370$      ;YES!!
530 061774 032737 020000 001352      BIT      #OPI,RMER11 ;WAS OPI ERROR REPORTED??
531 062002 001023      BNE      370$      ;YES..
532 062004 013737 001350 001140      MOV      RMDSI,$GDDAT ;EXPECTED STATUS
533 062012 052737 010000 001140      BIS      #MOL,$GDDAT
534 062020 013737 001350 001142      MOV      RMDSI,$BDDAT ;RECEIVED STATUS
535 062026 062716 000004      ADD      #4,(SP) ;MOVE SP TO USER'S ERROR
536 062032 112776 000334 000000      MOV      #334,a(SP) ;WRITE ERROR NUMBER
537 062040 162716 000002      SUB      #2,(SP) ;MOVE SP TO RETURN IF ERROR
538 062044 004736      JSR      PC,a(SP)+ ;REPORT ERROR AND RETURN
539 062046 162716 000010      SUB      #10,(SP) ;MOVE SP TO NO ERROR
540 062052      370$:
541
542      ;REPORT ERROR IF VOLUME IS NOT VALID AND "IVC" ERROR WAS NOT
543      ;REPORTED, I.E., VV = IVC = 0
544 062052 032737 000100 001350      BIT      #VV,RMDSI ;IS VOLUME VALID??
545 062060 001027      BNE      380$      ;YES!!
546 062062 032737 010000 001400      BIT      #IVC,RMER21 ;WAS IVC ERROR REPORTED??
547 062070 001033      BNE      390$      ;YES!!
548 062072 013737 001350 001140      MOV      RMDSI,$GDDAT ;EXPECTED STATUS
549 062100 052737 000100 001140      BIS      #VV,$GDDAT
550 062106 013737 001350 001142      MOV      RMDSI,$BDDAT ;RECEIVED STATUS
551 062114 062716 000004      ADD      #4,(SP) ;MOVE SP TO USERS ERROR CALL
552 062120 112776 000335 000000      MOV      #335,a(SP) ;WRITE ERROR NUMBER
553 062126 162716 000002      SUB      #2,(SP) ;MOVE SP TO RETURN IF ERROR
554 062132 004736      JSR      PC,a(SP)+ ;REPORT ERROR AND RETURN
555 062134 162716 000010      SUB      #10,(SP) ;MOVE SP TO NO ERROR
556 062140      380$:
557
558      ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS FOUND
559 062140 062716 000004      ADD      #4,(SP) ;MOVE SP TO ERROR CALL
560 062144 105776 000000      TSIB      a(SP) ;ANY ERROR??
561 062150 001403      BEQ      390$      ;NO!!
562 062152 062716 000004      ADD      #4,(SP) ;YES - MOVE SP TO ERROR RETURN
563 062156 000402      BR      400$
564 062160 162716 000004      390$: SUB      #4,(SP) ;MOVE SP TO NO ERROR RETURN
565
566 062164 000207      400$: RTS      PC ;RETURN TO USER
567
568 062166 000000      500$: .WORD ;ERROR FLAGS
569 062170 000000      510$: .WORD ;TEMPORARY STORAGE

```

```

1  .SMTL STATIC DRIVE STATUS CHECK SUBROUTINE
2
3  :THIS SUBROUTINE LOOKS FOR UNEXPECTED CHANGES IN DRIVE
4  :STATUS, SUCH AS THE DRIVE LOSING VOLUME VALID. THE SUBROUTINE
5  :CAN BE USED BY HOUSEKEEPING AND OTHER COMMANDS DURING WHICH THERE
6  :SHOULD NOT BE ANY DRIVE ERRORS OR CHANGES IN STATE.
7
8  :THE FOLLOWING CONDITIONS ARE TESTED AND REPORTED AS ERRORS
9  :IF TRUE:
10
11  :   .MOL = 0, INDICATES DRIVE WENT OFFLINE, NOTE
12  :   THAT MOL IS ASSUMED TO HAVE BEEN SET
13  :   .VV = 0, INDICATES THE DRIVE LOST VOLUME VALID
14  :   .PIP = 1, INDICATES THAT THE DRIVE IS OFF CYLINDER
15  :   .SKI = 1, INDICATES THE DRIVE HAS AN UNEXPECTED SKI ERROR
16  :   .DVC = 1, INDICATES AN UNEXPECTED DEVICE FAULT
17
18  :THE SUBROUTINE IS CALLED AFTER STORING STATUS IN THE GET BUFFER.
19
20  : (1) JSR PC,STCDRVSTS
21  :      BR      ??? RETURN HERE IF NO ERROR
22  :      NOP      RETURN HERE TO REPORT AN ERROR
23  :      ERROR    ERROR NUMBER DEFINED BY SUB
24  :      JSR PC,@(SP)+ GO BACK TO SUB FOR MORE ERROR CHECKS
25  :      ???      RETURN HERE IF NO MORE ERRORS
26
27  STCDRVSTS:
28
29  :CLEAR USER'S ERROR CALL
30  ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
31  CLRB @ (SP) ;CLEAR ERROR NUMBER
32  SUB #4,(SP) ;MOVE SP BACK TO NO ERROR RETURN
33
34  :SEE IF 'MOL' = 'VV' = 1, AND 'PIP' = 0
35  MOV RMDSI,-(SP) ;PUT DRIVE STATUS ON STACK
36  BIC #^C<PIP!MOL!VV>,(SP)
37  CMP #MOL!VV,(SP)+ ;ARE MOL,VV AND PIP O.K.??
38  BEQ 30$ ;YES!!
39
40  :REPORT AN ERROR IF MOL = 0 AND 'OPI' = 0
41  BIT #MOL,RMDSI ;IS MOL ON ??
42  BNE 10$ ;YES!!
43  BIT #OPI,RMER1I ;WAS 'OPI' SET??
44  BNE 10$ ;YES-DONT REPORT 'MOL' 0
45  MOV RMDSI,$GDDAT ;EXPECTED STATUS
46  BIS #MOL,$GDDAT
47  MOV RMDSI,$BDDAT ;RECEIVED STATUS
48  ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
49  MOVB #207,@(SP) ;WRITE ERROR NUMBER IN CALL
50  SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
51  JSR PC,@(SP)+ ;REPORT ERROR VIA USER
52  SUB #10,(SP) ;MOVE SP BACK TO NO ERROR RETURN
53  NOP
54
55  10$:
56
57  :REPORT AN ERROR IF VOLUME VALID IS NOW ZERO AND 'IVC' 0
58  BIT #VV,RMDSI ;IS 'VV' = 0??
59  BNE 20$ ;NO.

```

```

58 062324 032737 010000 001400      BIT      #IVC,RMER2I      ;WAS 'IVC' SET??
59 062332 001024                      BNE      20$          ;YES-DONT REPORT 'VV' 0
60 062334 013737 001350 001140      MOV      RMDSI,$GDDAT      ;EXPECTED STATUS
61 062342 052737 000100 001350      BIS      #VV,RMDSI
62 062350 013737 001350 001142      MOV      RMDSI,$BDDAT      ;RECEIVED STATUS
63 062356 062716 000004              ADD      #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
64 062362 112776 000210 000000      MOV      #210,a(SP)        ;WRITE ERROR NUMBER IN CALL
65 062370 162716 000002              SUB      #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
66 062374 004736                      JSR      PC,a(SP)+        ;REPORT ERROR VIA USER
67 062376 162716 000010              SUB      #10,(SP)         ;MOVE SP BACK TO NO ERROR
68 062402 000240                      NOP
69 062404
70
71                                     ;REPORT AN ERROR IF DRIVE IS OFF CYLINDER AND 'SKI' - 0
72 062404 032737 020000 001350      BIT      #PIP,RMDSI      ;IS DRIVE OFF CYLINDER??
73 062412 001430                      BEQ      30$          ;NO!!
74 062414 032737 040000 001400      BIT      #SKI,RMER2I      ;WAS 'SKI' SET??
75 062422 001024                      BNE      30$          ;YES-DONT REPORT 'PIP' - 1
76 062424 013737 001350 001140      MOV      RMDSI,$GDDAT      ;EXPECTED STATUS
77 062432 042737 020000 001140      BIC      #PIP,$GDDAT
78 062440 013737 001350 001142      MOV      RMDSI,$BDDAT      ;RECEIVED STATUS
79 062446 062716 000004              ADD      #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
80 062452 112776 000211 000000      MOV      #211,a(SP)        ;WRITE ERROR NUMBER IN USER'S CALL
81 062460 162716 000002              SUB      #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
82 062464 004736                      JSR      PC,a(SP)+        ;REPORT ERROR VIA USER
83 062466 162716 000010              SUB      #10,(SP)         ;MOVE SP TO NO ERROR RETURN
84 062472 000240                      NOP
85 062474
86
87                                     ;SEE IF 'SKI' = 'DVC' = 0
88 062474 013746 001400              MOV      RMER2I,-(SP)      ;PUT ERROR REG 2 ON STACK
89 062500 042726 137577              BIC      #^C<SKI!DVC>,(SP)+
90 062504 001460                      BEQ      60$          ;BRANCH IF NO ERROR
91 062506
92
93                                     ;REPORT AN ERROR IF THERE IS A DEVICE FAULT
94 062506 032737 000200 001400      BIT      #DVC,RMER2I      ;ANY DEVICE FAULT??
95 062514 001424                      BEQ      50$          ;NO!!
96 062516 013737 001400 001140      MOV      RMER2I,$GDDAT      ;EXPECTED STATUS
97 062524 042737 000200 001140      BIC      #DVC,$GDDAT
98 062532 013737 001400 001142      MOV      RMER2I,$BDDAT      ;RECEIVED STATUS
99 062540 062716 000004              ADD      #4,(SP)          ;MOVE SP TO USER'S CALL
100 062544 112776 000212 000000      MOV      #212,a(SP)        ;WRITE NUMBER OF ERROR IN CALL
101 062552 162716 000002              SUB      #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
102 062556 004736                      JSR      PC,a(SP)+        ;REPORT ERROR VIA USER
103 062560 162716 000010              SUB      #10,(SP)         ;MOVE SP BACK TO NO ERROR
104 062564 000240                      NOP
105 062566
106
107                                     ;REPORT AN ERROR IF 'SKI' = 1
108 062566 032737 040000 001400      BIT      #SKI,RMER2I      ;IS THERE A SEEK INCOMPLETE ERROR
109 062574 001424                      BEQ      60$          ;NO!!
110 062576 013737 001400 001140      MOV      RMER2I,$GDDAT      ;EXPECTED STATUS
111 062604 042737 040000 001140      BIC      #SKI,$GDDAT
112 062612 013737 001400 001142      MOV      RMER2I,$BDDAT      ;RECEIVED STATUS
113 062620 062716 000004              ADD      #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
114 062624 112776 000213 000000      MOV      #213,a(SP)        ;WRITE ERROR NUMBER IN USER'S ERROR CALL

```

115	062632	162716	000002	SUB	#2,(SP)	;MOVE SP TO RETURN FOR ERROR
116	062636	004736		JSR	PC,@(SP)+	;REPORT ERROR VIA USER
117	062640	162716	000010	SUB	#10,(SP)	;MOVE SP BACK TO NO ERROR
118	062644	000240		NOP		
119	062646			60\$:		
120						
121				;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED		
122	062646	062716	000004	ADD	#4,(SP)	;MOVE SP TO USER'S ERROR CALL
123	062652	105776	000000	TSTB	@(SP)	;WAS AN ERROR DETECTED??
124	062656	001403		BEQ	70\$	;NO!
125	062660	062716	000004	ADD	#4,(SP)	;YES - MOVE SP TO USER'S ERROR RETURN
126	062664	000402		BR	80\$	
127	062666	162716	000004	70\$:	SUB	#4,(SP)
128	062672	000240		80\$:	NOP	
129	062674	000207		RTS	PC	;RETURN TO USER

1 .SBTTL SAVE AND RESTORE R0-R5 ROUTINES

```

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

062676			\$SAVREG:		
062676	010046		MOV	R0,-(SP)	::PUSH R0 ON STACK
062700	010146		MOV	R1,-(SP)	::PUSH R1 ON STACK
062702	010246		MOV	R2,-(SP)	::PUSH R2 ON STACK
062704	010346		MOV	R3,-(SP)	::PUSH R3 ON STACK
062706	010446		MOV	R4,-(SP)	::PUSH R4 ON STACK
062710	010546		MOV	R5,-(SP)	::PUSH R5 ON STACK
062712	016646	000022	MOV	22(SP),-(SP)	::SAVE PS OF MAIN FLOW
062716	016646	000022	MOV	22(SP),-(SP)	::SAVE PC OF MAIN FLOW
062722	016646	000022	MOV	22(SP),-(SP)	::SAVE PS OF CALL
062726	016646	000022	MOV	22(SP),-(SP)	::SAVE PC OF CALL
062732	000002		RTI		

*RESTORE R0-R5

*CALL:

* RESREG

\$RESREG:

062734					
062734	012666	000022	MOV	(SP)+,22(SP)	::RESTORE PC OF CALL
062740	012666	000022	MOV	(SP)+,22(SP)	::RESTORE PS OF CALL
062744	012666	000022	MOV	(SP)+,22(SP)	::RESTORE PC OF MAIN FLOW
062750	012666	000022	MOV	(SP)+,22(SP)	::RESTORE PS OF MAIN FLOW
062754	012605		MOV	(SP)+,R5	::POP STACK INTO R5
062756	012604		MOV	(SP)+,R4	::POP STACK INTO R4
062760	012603		MOV	(SP)+,R3	::POP STACK INTO R3
062762	012602		MOV	(SP)+,R2	::POP STACK INTO R2
062764	012601		MOV	(SP)+,R1	::POP STACK INTO R1
062766	012600		MOV	(SP)+,R0	::POP STACK INTO R0
062770	000002		RTI		

.SBTTL BINARY TO ASCII AND TYPE ROUTINE

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

*(CALL:

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

062772	010146			\$TYPBN:	MOV	R1, -(SP)	;;SAVE R1 ON THE STACK
062774	016601	000006			MOV	6(SP), R1	;;GET THE INPUT NUMBER
063000	000261				SEC		;;SET 'C' SO CAN KEEP TRACK OF THE NUMBER OF BITS
063002	112737	000060	063044	1\$:	MOVB	#'0, \$BIN	;;SET CHARACTER TO AN ASCII '0'.
063010	006101				ROL	R1	;;GET THIS BIT
063012	001406				BEQ	2\$	;;DONE?
063014	105537	063044			ADCB	\$BIN	;;NO--SET THE CHARACTER EQUAL TO THIS BIT
063020	104401	063044			TYPE	,\$BIN	;;GO TYPE THIS BIT
063024	000241				CLC		;;CLEAR 'C' SO CAN KEEP TRACK OF BITS
063026	000765				BR	1\$	;;GO DO THE NEXT BIT
063030	012601			2\$:	MOV	(SP)+, R1	;;POP THE STACK INTO R1
063032	016666	000002	000004		MOV	2(SP), 4(SP)	;;ADJUST THE STACK
063040	012616				MOV	(SP)+, (SP)	
063042	000002				RTI		;;RETURN TO USER
063044	000	000		\$BIN:	.BYTE	0,0	;;STORAGE FOR ASCII CHAR. AND TERMINATOR

.SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

```

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

$TYPDS:
MOV      R0,-(SP)      ::PUSH R0 ON STACK
MOV      R1,-(SP)      ::PUSH R1 ON STACK
MOV      R2,-(SP)      ::PUSH R2 ON STACK
MOV      R3,-(SP)      ::PUSH R3 ON STACK
MOV      R5,-(SP)      ::PUSH R5 ON STACK
MOV      #20200,-(SP)    ::SET BLANK SWITCH AND SIGN
MOV      20(SP),R5      ::GET THE INPUT NUMBER
BPL      1$            ::BR IF INPUT IS POS.
NEG      R5            ::MAKE THE BINARY NUMBER POS.
MOVB     #'-,1(SP)      ::MAKE THE ASCII NUMBER NEG.
1$:      CLR      R0      ::ZERO THE CONSTANTS INDEX
MOV      #SDBLK,R3      ::SETUP THE OUTPUT POINTER
MOVB     #'',(R3)+      ::SET THE FIRST CHARACTER TO A BLANK
2$:      CLR      R2      ::CLEAR THE BCD NUMBER
MOV      $DTBL(R0),R1    ::GET THE CONSTANT
3$:      SUB      R1,R5    ::FORM THIS BCD DIGIT
BLT      4$            ::BR IF DONE
INC      R2            ::INCREASE THE BCD DIGIT BY 1
BR       3$
4$:      ADD      R1,R5    ::ADD BACK THE CONSTANT
TST      R2            ::CHECK IF BCD DIGIT 0
BNE      5$            ::FALL THROUGH IF 0
TSTB     (SP)          ::STILL DOING LEADING 0'S?
BMI      7$            ::BR IF YES
5$:      ASLB     (SP)      ::MSD?
BCC      6$            ::BR IF NO
MOVB     1(SP),-1(R3)    ::YES--SET THE SIGN
6$:      BIS      #'0,R2    ::MAKE THE BCD DIGIT ASCII
7$:      BIS      #' ,R2    ::MAKE IT A SPACE IF NOT ALREADY A DIGIT
MOVB     R2,(R3)+      ::PUT THIS CHARACTER IN THE OUTPUT BUFFER
TST      (R0)+          ::JUST INCREMENTING
CMP      R0,#10        ::CHECK THE TABLE INDEX
BLT      2$            ::GO DO THE NEXT DIGIT
BGJ      8$            ::GO TO EXIT
MOV      R5,R2          ::GET THE LSD
BR       6$            ::GO CHANGE TO ASCII
8$:      TSTB     (SP)+    ::WAS THE LSD THE FIRST NON-ZERO?
BPL      9$            ::BR IF NO
MOVB     -1(SP),-2(R3)   ::YES--SET THE SIGN FOR TYPING
9$:      CLRB     (R3)      ::SET THE TERMINATOR
MOV      (SP)+,R5        ::POP STACK INTO R5
MOV      (SP)+,R3        ::POP STACK INTO R3
MOV      (SP)+,R2        ::POP STACK INTO R2
MOV      (SP)+,R1        ::POP STACK INTO R1

```

```

063046
063046 010046
063050 010146
063052 010246
063054 010346
063056 010546
063060 012746 020200
063064 016605 000020
063070 100004
063072 005405
063074 112766 000055 000001
063102 005000
063104 012703 063262
063110 112723 000040
063114 005002
063116 016001 063252
063122 160105
063124 002402
063126 005202
063130 000774
063132 060105
063134 005702
063136 001002
063140 105716
063142 100407
063144 106316
063146 103003
063150 116663 000001 177777
063156 052702 000060
063162 052702 000040
063166 110223
063170 005720
063172 020027 000010
063176 002746
063200 003002
063202 010502
063204 000764
063206 105726
063210 100003
063212 116663 177777 177776
063220 105013
063222 012605
063224 012603
063226 012602
063230 012601

```


063232	012600			MOV	(SP)+,R0	::POP STACK INTO R0
063234	104401	063262		TYPE	,SDBLK	::NOW TYPE THE NUMBER
063240	016666	000002	000004	MOV	2(SP),4(SP)	::ADJUST THE STACK
063246	012616			MOV	(SP)+,(SP)	
063250	000002			RTI		::RETURN TO USER
063252	023420			\$DTBL:	10000.	
063254	001750				1000.	
063256	000144				100.	
063260	000012				10.	
063262				\$DBLK:	.BLKW 4	

.SRITL BINARY TO OCTAL (ASCII) AND TYPE

```

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

```

```

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

```

```

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

```

```

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

```

```

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

```

063272	017646	000000		\$TYPOS: MOV	@(SP),-(SP)	;;PICKUP THE MODE
063276	116637	000001	063515	MOVB	1(SP), \$OFILL	;;LOAD ZERO FILL SWITCH
063304	112637	063517		MOVB	(SP)+, \$OMODE+1	;;NUMBER OF DIGITS TO TYPE
063310	062716	000002		ADD	#2,(SP)	;;ADJUST RETURN ADDRESS
063314	000406			BR	\$TYPON	
063316	112737	000001	063515	\$TYPOC: MOVB	#1, \$OFILL	;;SET THE ZERO FILL SWITCH
063324	112737	000006	063517	MOVB	#6, \$OMODE+1	;;SET FOR SIX(6) DIGITS
063332	112737	000005	063514	\$TYPON: MOVB	#5, \$OCNT	;;SET THE ITERATION COUNT
063340	010346			MOV	R3, -(SP)	;;SAVE R3
063342	010446			MOV	R4, -(SP)	;;SAVE R4
063344	010546			MOV	R5, -(SP)	;;SAVE R5
063346	113704	063517		MOVB	\$OMODE+1, R4	;;GET THE NUMBER OF DIGITS TO TYPE
063352	005404			NEG	R4	
063354	062704	000006		ADD	#6, R4	;;SUBTRACT IT FOR MAX ALLOWED
063360	110437	063516		MOVB	R4, \$OMODE	;;SAVE IT FOR USE
063364	113704	063515		MOVB	\$OFILL, R4	;;GET THE ZERO FILL SWITCH
063370	016605	000012		MOV	12(SP), R5	;;PICKUP THE INPUT NUMBER
063374	005003			CLR	R3	;;CLEAR THE OUTPUT WORD
063376	006105		1\$:	ROL	R5	;;ROTATE MSB INTO 'C'
063400	000404			BR	3\$	;;GO DO MSB
063402	006105		2\$:	ROL	R5	;;FORM THIS DIGIT
063404	006105			ROL	R5	
063406	006105			ROL	R5	
063410	010503			MOV	R5, R3	
063412	006103		3\$:	ROL	R3	;;GET LSB OF THIS DIGIT
063414	105337	063516		DECB	\$OMODE	;;TYPE THIS DIGIT?
063420	100016			BPL	7\$	;;BR IF NO
063422	042703	177770		BIC	#177770, R3	;;GET RID OF JUNK
063426	001002			BNE	4\$	;;TEST FOR 0
063430	005704			TST	R4	;;SUPPRESS THIS 0?
063432	001403			BEQ	5\$	;;BR IF YES
063434	005204		4\$:	INC	R4	;;DON'T SUPPRESS ANYMORE 0'S

063436	052703	000060		BIS	#'0,R3	::MAKE THIS DIGIT ASCII
063442	052703	000040	5\$:	BIS	#' ,R3	::MAKE ASCII IF NOT ALREADY
063446	110337	063512		MOVB	R3,8\$	::SAVE FOR TYPING
063452	104401	063512		TYPE	,8\$	::GO TYPE THIS DIGIT
063456	105337	063514	7\$:	DECB	\$OCNT	::COUNT BY 1
063462	003347			BGT	2\$	::BR IF MORE TO DO
063464	002402			BLT	0\$	::BR IF DONE
063466	005204			INC	R4	::INSURE LAST DIGIT ISN'T A BLANK
063470	000744			BR	2\$	::GO DO THE LAST DIGIT
063472	012605		6\$:	MOV	(SP)+,R5	::RESTORE R5
063474	012604			MOV	(SP)+,R4	::RESTORE R4
063476	012603			MOV	(SP)+,R3	::RESTORE R3
063500	016666	000002 000004		MOV	2(SP),4(SP)	::SET THE STACK FOR RETURNING
063506	012616			MOV	(SP)+,(SP)	
063510	000002			RTI		::RETURN
063512	000		8\$:	.BYTE	0	::STORAGE FOR ASCII DIGIT
063513	000			.BYTE	0	::TERMINATOR FOR TYPE ROUTINE
063514	000		\$OCNT:	.BYTE	0	::OCTAL DIGIT COUNTER
063515	000		\$OFILL:	.BYTE	0	::ZERO FILL SWITCH
063516	000000		\$OMODE:	.WORD	0	::NUMBER OF DIGITS TO TYPE

.SBTIL TYPE ROUTINE

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

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

063520	105737	001173	\$TYPE:	TSTB	\$TPFLG	::IS THERE A TERMINAL?
063524	100002			BPL	1\$	::BR IF YES
063526	000000			HALT		::HALT HERE IF NO TERMINAL
063530	000430			BR	3\$	::LEAVE
063532	010046		1\$:	MOV	R0,-(SP)	::SAVE R0
063534	017600	000002		MOV	@2(SP),R0	::GET ADDRESS OF ASCIZ STRING
063540	122737	000001 .001242		CMPIB	#APTENV,\$ENV	::RUNNING IN APT MODE
063546	001011			BNE	62\$	::NO,GO CHECK FOR APT CONSOLE
063550	132737	000100 001243		BITB	#APTSPOOL,\$ENV	::SPOOL MESSAGE TO APT
063556	001405			BEQ	62\$	::NO,GO CHECK FOR CONSOLE
063560	010037	063570		MOV	R0,61\$	::SETUP MESSAGE ADDRESS FOR APT
063564	004737	070166		JSR	PC,\$ATY3	::SPOOL MESSAGE TO APT
063570	000000		61\$:	.WORD	0	::MESSAGE ADDRESS
063572	132737	000040 001243	62\$:	BITB	#APTCSUP,\$ENV	::APT CONSOLE SUPPRESSED
063600	001003			BNE	60\$	::YES,SKIP TYPE OUT
063602	112046		2\$:	MOVB	(R0)+,-(SP)	::PUSH CHARACTER TO BE TYPED ONTO STACK
063604	001005			BNE	4\$	::BR IF IT ISN'T THE TERMINATOR
063606	005726			TST	(SP)+	::IF TERMINATOR POP IT OFF THE STACK
063610	012600		60\$:	MOV	(SP)+,R0	::RESTORE R0
063612	062716	000002	3\$:	ADD	#2,(SP)	::ADJUST RETURN PC
063616	000002			RTI		::RETURN
063620	122716	000011	4\$:	CMPIB	#HT,(SP)	::BRANCH IF <HT>
063624	001430			BEQ	8\$	
063626	122716	000200		CMPIB	#CRLF,(SP)	::BRANCH IF NOT <CRLF>
063632	001006			BNE	5\$	
063634	005726			TST	(SP)+	::POP <CR><LF> EQUIV
063636	104401			TYPE		::TYPE A CR AND LF
063640	001217			\$CRLF		
063642	105037	064050		CLRB	\$CHARCNT	::CLEAR CHARACTER COUNT
063646	000755			BR	2\$	::GET NEXT CHARACTER
063650	004737	063732	5\$:	JSR	PC,\$TYPE	::GO TYPE THIS CHARACTER
063654	123726	001172	6\$:	CMPIB	\$FILLC,(SP)+	::IS IT TIME FOR FILLER CHARS.?
063660	001350			BNE	2\$	::IF NO GO GET NEXT CHAR.
063662	013746	001170		MOV	\$NULL,-(SP)	::GET # OF FILLER CHARS. NEEDED
						::AND THE NULL CHAR.
063666	105366	000001	7\$:	DECB	1(SP)	::DOES A NULL NEED TO BE TYPED?
063672	002770			BLT	6\$	::BR IF NO--GO POP THE NULL OFF OF STACK
063674	004737	063732		JSR	PC,\$TYPE	::GO TYPE A NULL
063700	105337	064050		DECB	\$CHARCNT	::DO NOT COUNT AS A COUNT
063704	000770			BR	7\$	::LOOP

;HORIZONTAL TAB PROCESSOR

```
063706 112716 000040      8$:      MOVB      #' , (SP)      ;;REPLACE TAB WITH SPACE
063712 004737 063732      9$:      JSR      PC, $TYPEC      ;;TYPE A SPACE
063716 132737 000007 064050      BITB      #7, $CHARCNT      ;;BRANCH IF NOT AT
063724 001372      BNE      9$      ;;TAB STOP
063726 005726      TST      (SP)+      ;;POP SPACE OFF STACK
063730 000724      BR      2$      ;;GET NEXT CHARACTER
063732      $TYPEC:      TSTB      @ $TKS      ;;CHAR IN KYBD BUFFER?
063732 105777 115222      BPL      10$      ;;BR IF NOT
063736 100022      MOV      @ $TKB, -(SP)      ;;GET CHAR
063740 017746 115216      BIC      #177600, (SP)      ;;STRIP EXTRANEIOUS BITS
063744 042716 177600      CMPB      # $XOFF, (SP)      ;;WAS CHAR XOFF
063750 122716 000023      BNE      102$      ;;BR IF NOT
063754 001012      101$:      TSTB      @ $TKS      ;;WAIT FOR CHAR
063756 105777 115176      BPL      101$      ;;GET CHAR
063762 100375      MOVB      @ $TKB, (SP)      ;;STRIP IT
063764 117716 115172      BIC      #177600, (SP)      ;;WAS IT XON?
063770 042716 177600      CMPB      # $XON, (SP)      ;;BR IF NOT
063774 122716 000021      BNE      101$      ;;FIX STACK
064000 001366      102$:      TST      (SP)+      ;;WAIT UNTIL PRINTER IS READY
064002      10$:      TSTB      @ $TPS      ;;LOAD CHAR TO BE TYPED INTO DATA REG.
064004 105777 115154      BPL      10$      ;;IS CHARACTER A CARRIAGE RETURN?
064010 100375      MOVB      2(SP), @ $TPB      ;;BRANCH IF NO
064012 116677 000002 115146      CMPB      #CR, 2(SP)      ;;YES--CLEAR CHARACTER COUNT
064020 122766 000015 000002      BNE      1$      ;;EXIT
064026 001003      CLRB      $CHARCNT      ;;IS CHARACTER A LINE FEED?
064030 105037 064050      BR      $TYPEX      ;;BRANCH IF YES
064034 000406      CMPB      #LF, 2(SP)      ;;COUNT THE CHARACTER
064036 122766 000012 000002 1$:      BEQ      $TYPEX      ;;CHARACTER COUNT STORAGE
064044 001402      INCB      (PC)+
064046 105227      $CHARCNT: .WORD 0
064050 000000      $TYPEX: RTS      PC
064052 000207
```

.SBTTL SCOPE HANDLER ROUTINE

```

*****
*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
*SW14=1      LOOP ON TEST
*SW11=1      INHIBIT ITERATIONS
*SW09=1      LOOP ON ERROR
*SW08=1      LOOP ON TEST IN SWR<7:0>
*CALL
*          SCOPE          ;;SCOPE-IOT

```

```

064054          $SCOPE:
064054 104410          CKSWR          ;;TEST FOR CHANGE IN SOFT-SWR
064056 004737 064612  JSR          PC_STOP
064062 032777 040000 115064 1$:  BIT      #BIT14,@SWR          ;;LOOP ON PRESENT TEST?
064070 001402          BEQ          9$          ;;NO IF SW14=0
064072 000137 064522  JMP          $OVER          ;;JUMP OVER SCOPE ROUTINE
064076          9$:
;#####START OF CODE FOR THE XOR TESTER#####
$XTSTR: BR      6$          ;;IF RUNNING ON THE 'XOR' TESTER CHANGE
                                ;;THIS INSTRUCTION TO A 'NOP' (NOP=240)
064100 013746 000004          MOV      @#ERRVEC,-(SP)          ;;SAVE THE CONTENTS OF THE ERROR VECTOR
064104 012737 064124 000004  MOV      #5$,@#ERRVEC          ;;SET FOR TIMEOUT
064112 005737 177060          TST      @#177060          ;;TIME OUT ON XOR?
064116 012637 000004          MOV      (SP)+,@#ERRVEC          ;;RESTORE THE ERROR VECTOR
064122 000561          BR      $SVLAD          ;;GO TO THE NEXT TEST
064124 022626          5$:  CMP      (SP)+,(SP)+          ;;CLEAR THE STACK AFTER A TIME OUT
064126 012637 000004          MOV      (SP)+,@#ERRVEC          ;;RESTORE THE ERROR VECTOR
064132 000521          BR      7$          ;;LOOP ON THE PRESENT TEST
064134          6$:;#####END OF CODE FOR THE XOR TESTER#####
064134 032777 000400 115012  BIT      #BIT08,@SWR          ;;LOOP ON SPEC. TEST?
064142 001421          BEQ          2$          ;;BR IF NO
064144 005046          CLR      -(SP)          ;;CLEAR A TEMP. LOCATION
064146 117716 115002          MOVB    @SWR,(SP)          ;;PICKUP THE DESIRED TEST NUMBER
064152 001414          BEQ          8$          ;;BRANCH IF BAD TEST NUMBER IN SWR
064154 022716 000025          CMP      #25,(SP)          ;;CHECK THE NUMBER IN THE SWR
064160 002411          BLT      8$          ;;BRANCH IF TEST NUMBER IS OUT OF RANGE
064162 011637 001116          MOV      (SP),$TSTNM          ;;UPDATE THE TEST NUMBER
064166 005316          DEC      (SP)          ;;BACKUP BY ONE
064170 006316          ASL      (SP)          ;;SCALE THE TEST NUMBER AS AN INDEX
064172 062716 064540          ADD      #$$SW08TBL,(SP)          ;;FORM THE ADDRESS OF TEST POINTER
064176 013637 001122          MOV      @ (SP)+,$LPADR          ;;SET LOOP ADDRESS TO DESIRED TEST
064202 000547          BR      $OVER          ;;GO LOOP ON THE TEST
064204 005726          8$:  TST      (SP)+          ;;CLEAN THE BAD TEST NUMBER OFF OF THE STACK
064206 105737 001117 2$:  TSTB     $ERFLG          ;;HAS AN ERROR OCCURRED?
064212 001502          BEQ          3$          ;;BR IF NO
064214 022737 177777 065230  CMP      #-1,CPSAVE          ;;SEE IF TIMEOUT WAS PREVIOUSLY RECORDED
064222 001455          BEQ          2003$          ;;KICK AROUND ROUTINE IF SO
064224 013746 000004          MOV      ERRVEC,-(SP)          ;;SAVE CONTENTS OF ERROR VECTOR
064230 012737 064246 000004  MOV      #2000$,@ERRVEC          ;;SETUP 'TRAP' RETURN ADDRESS
064236 013737 177766 065230  MOV      177766,CPSAVE          ;;MOVE CPU ERROR REGISTER TO CPSAVE FOR TEST
064244 000406          BR      2001$
064246 012737 177777 065230 2000$: MOV      #-1,CPSAVE          ;;SET CPU ERROR REGISTER TIMEOUT INDICATOR
064254 012716 064262          MOV      #2001$,(SP)          ;;SETUP RETURN ADDRESS

```



```

SCOPE HANDLER ROUTINE

064260 000002 RTI
064262 012637 000004 2001$: MOV (SP)+,ERRVEC ;;RESTORE CONTENTS OF ERROR VECTOR

064266 022737 177777 065230 2002$: CMP #-1,CPSAVE ;;SEE IF CPSAVE HAS CPU ERR REG TIMEOUT INDICATION
064274 001430 BEQ 2003$ ;;BRANCH IF SO
064276 032737 000001 065230 BIT #BIT00,CPSAVE ;;SEE IF THE POWER MONITOR BIT IS ON
064304 001424 BEQ 2003$ ;;BRANCH TO CONTINUE ROUTINE IF CLEAR
064306 042737 000001 177766 BIC #BIT00,177766 ;;CLEAR THE BIT FOUND TO BE SET
064314 013746 001154 MOV SWR,-(SP) ;;SAVE SWR ADDRESS
064320 017646 000000 MOV @ (SP),-(SP) ;;SAVE SWR VALUE
064324 012737 000176 001154 MOV #176,SWR ;;GET SOFTWARE SWR ADDRESS
064332 011677 114616 MOV (SP),@SWR ;;GET CURRENT SWR VALUE
064336 042777 001000 114610 BIC #BIT09,@SWR ;;DON'T ALLOW LOOP ON ERROR ON THIS ERROR
064344 104177 EMT 177 ;;CALL SPECIAL POWER FAIL BIT ERROR CALL
064346 012676 000000 MOV (SP)+,@(SP) ;;RESTORE SWR TO ORIGINAL VALUE
064352 012637 001154 MOV (SP)+,SWR ;;RESTORE SWR ADDRESS
064356 2003$: CMPB $ERMAX,$ERFLG ;;MAX. ERRORS FOR THIS TEST OCCURRED?
064356 123737 001131 001117 BHI 3$ ;;BR IF NO
064364 101015 BIT #BIT09,@SWR ;;LOOP ON ERROR?
064366 032777 001000 114560 BEQ 4$ ;;BR IF NO
064374 001404 MOV $LPERR,$LPADR ;;SET LOOP ADDRESS TO LAST SCOPE
064376 013737 001124 001122 7$: BR $OVER
064404 000446 4$: CLRB $ERFLG ;;ZERO THE ERROR FLAG
064406 105037 001117 CLR $TIMES ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
064412 005037 001206 BR 1$ ;;ESCAPE TO THE NEXT TEST
064416 000415 BIT #BIT11,@SWR ;;INHIBIT ITERATIONS?
064420 032777 004000 114526 3$: BNE 1$ ;;BR IF YES
064426 001011 TST $PASS ;;IF FIRST PASS OF PROGRAM
064430 005737 001230 BEQ 1$ ;; INHIBIT ITERATIONS
064434 001406 INC $ICNT ;;INCREMENT ITERATION COUNT
064436 005237 001120 CMP $TIMES,$ICNT ;;CHECK THE NUMBER OF ITERATIONS MADE
064442 023737 001206 001120 BGE $OVER ;;BR IF MORE ITERATION REQUIRED
064450 002024 1$: MOV #1,$ICNT ;;REINITIALIZE THE ITERATION COUNTER
064452 012737 000001 001120 MOV $MXCNT,$TIMES ;;SET NUMBER OF ITERATIONS TO DO
064460 013737 064536 001206 $SVLAD: INCB $TSTNM ;;COUNT TEST NUMBERS
064466 105237 001116 MOV $TSTNM,$TSTNM ;;SET TEST NUMBER IN APT MAILBOX
064472 113737 001116 001226 MOV (SP),$LPADR ;;SAVE SCOPE LOOP ADDRESS
064500 011637 001122 MOV (SP),$LPERR ;;SAVE ERROR LOOP ADDRESS
064504 011637 001124 CLR $ESCAPE ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
064510 005037 001210 MOV #1,$ERMAX ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
064514 112737 000001 001131 $OVER: MOV $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
064522 013777 001116 114426 MOV $LPADR,(SP) ;;FUDGE RETURN ADDRESS
064530 013716 001122 RTI ;;FIXES PS
064534 000002 $MXCNT: 5. ;;MAX. NUMBER OF ITERATIONS
064536 000005 $SW08TBL:
064540 .REPT $IN-1
064540 010070 .WORD TST1+2 ;;STARTING ADDRESS OF TEST 1
064542 010270 .WORD TST2+2 ;;STARTING ADDRESS OF TEST 2
064544 011276 .WORD TST3+2 ;;STARTING ADDRESS OF TEST 3
064546 012254 .WORD TST4+2 ;;STARTING ADDRESS OF TEST 4
064550 013450 .WORD TST5+2 ;;STARTING ADDRESS OF TEST 5
064552 014456 .WORD TST6+2 ;;STARTING ADDRESS OF TEST 6
064554 015434 .WORD TST7+2 ;;STARTING ADDRESS OF TEST 7
064556 016632 .WORD TST10+2 ;;STARTING ADDRESS OF TEST 10
064560 017636 .WORD TST11+2 ;;STARTING ADDRESS OF TEST 11
064562 021062 .WORD TST12+2 ;;STARTING ADDRESS OF TEST 12

```

064564	022270	.WORD	TST13+2	:: STARTING ADDRESS OF TEST 13
064566	023312	.WORD	TST14+2	:: STARTING ADDRESS OF TEST 14
064570	025160	.WORD	TST15+2	:: STARTING ADDRESS OF TEST 15
064572	026564	.WORD	TST16+2	:: STARTING ADDRESS OF TEST 16
064574	027604	.WORD	TST17+2	:: STARTING ADDRESS OF TEST 17
064576	030522	.WORD	TST20+2	:: STARTING ADDRESS OF TEST 20
064600	031560	.WORD	TST21+2	:: STARTING ADDRESS OF TEST 21
064602	032564	.WORD	TST22+2	:: STARTING ADDRESS OF TEST 22
064604	033572	.WORD	TST23+2	:: STARTING ADDRESS OF TEST 23
064606	034600	.WORD	TST24+2	:: STARTING ADDRESS OF TEST 24
064610	035626	.WORD	TST25+2	:: STARTING ADDRESS OF TEST 25
2				
3				
4				
5	064612			
	064612	012746	000140	
	064616	012746	064624	
	064622	000002		
	064624			
6				
7				
8				
9	064624	012746	000240	
	064630	012746	064636	
	064634	000002		
	064636			
10	064636	000207		

				; DROP PRIORITY TO ALLOW CONSOLE INTERRUPT
STOP:				
	MOV	#PR3,-(SP)		:: PUT NEW PS ON STACK
	MOV	#64\$,-(SP)		:: PUT NEW PC ON STACK
	RTI			:: POP NEW PC AND PS
64\$:				
				; RAISE PRIORITY TO INHIBIT CONSOLE INTERRUPT
	MOV	#PR5,-(SP)		:: PUT NEW PS ON STACK
	MOV	#65\$,-(SP)		:: PUT NEW PC ON STACK
	RTI			:: POP NEW PC AND PS
65\$:				
	RTS	PC		; RETURN

1

.SBTTL ERROR HANDLER ROUTINE

```

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

```

*      ERROR      N      ;;ERROR=EMT AND N=ERROR ITEM NUMBER

064640 105037 065232 $ERROR: CLRB      IBSAVE      ;;CLEAR THE ITEM BYTE SAVE LOCATION
064644 104410          CKSWR      ;;TEST FOR CHANGE IN SOFT-SWR
064646 105237 001117 7$:      INCB      $ERFLG      ;;SET THE ERROR FLAG
064652 001775          BEQ      7$      ;;DON'T LET THE FLAG GO TO ZERO
064654 013777 001116 114274 MOV      $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
064662 032777 002000 114264 BIT      #BIT10,@SWR      ;;BELL ON ERROR?
064670 001402          BEQ      1$      ;;NO - SKIP
064672 104401 001212          TYPE      $BELL      ;;RING BELL
064676 005237 001126 1$:      INC      $ERTTL      ;;COUNT THE NUMBER OF ERRORS
064702 011637 001132          MOV      (SP), $ERRPC ;;GET ADDRESS OF ERROR INSTRUCTION
064706 162737 000002 001132 SUB      #2, $ERRPC
064714 117737 114212 001130 MOV      @ $ERRPC, $ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
064722 032777 001000 114224 BIT      #BIT09,@SWR      ;;SEE IF LOOP ON ERROR IS SET
064730 001060          BNE      1004$      ;;BRANCH AROUND ROUTINE IF SO
064732 122737 000177 001130 CMP      #177, $ITEMB ;;SEE IF THIS IS THE POWER FAIL CALL
064740 001454          BEQ      1004$      ;;BRANCH AROUND ROUTINE IF IT IS
064742 105737 065232          TSTB      IBSAVE      ;;SEE IF THIS IS THE 2ND ERROR CALL IN THIS ROUTINE
064746 001047          BNE      1003$      ;;BRANCH IF SO
064750 022737 177777 065230 CMP      #-1, CPSAVE      ;;SEE IF CPSAVE HAS CPU ERR REG TIMEOUT INDICATION
064756 001445          BEQ      1004$      ;;BRANCH IF SO
064760 013746 000004          MOV      ERRVEC, -(SP) ;;SAVE CONTENTS OF ERROR VECTOR
064764 012737 065002 000004 MOV      #1000$, ERRVEC      ;;SETUP 'TRAP' RETURN ADDRESS
064772 013737 177766 065230 MOV      177766, CPSAVE      ;;MOVE CPU ERROR REGISTER TO CPSAVE FOR TEST
065000 000406          BR      1001$
065002 012737 177777 065230 1000$: MOV      #-1, CPSAVE      ;;SET CPU ERROR REGISTER TIMEOUT INDICATOR
065010 012716 065016          MOV      #1001$, (SP) ;;SETUP RETURN ADDRESS
065014 000002          RTI
065016 012637 000004          1001$: MOV      (SP)+, ERRVEC      ;;RESTORE CONTENTS OF ERROR VECTOR

065022 022737 177777 065230 1002$: CMP      #-1, CPSAVE      ;;SEE IF CPSAVE HAS CPU ERR REG TIMEOUT INDICATION
065030 001420          BEQ      1004$      ;;BRANCH IF SO
065032 032737 000001 065230 BIT      #BIT00, CPSAVE      ;;SEE IF POWER MONITOR BIT IS SET IN CPU ERR REG
065040 001414          BEQ      1004$      ;;BRANCH IF OK
065042 042737 000001 177766 BIC      #BIT00, 177766      ;;CLEAR THE BIT FOUND SET
065050 113737 001130 065232 MOV      $ITEMB, IBSAVE      ;;MAKE IBSAVE NON-ZERO FOR DUAL ERROR CALL
065056 112737 000177 001130 MOV      #177, $ITEMB      ;;SET $ITEMB TO SPECIAL POWER FAIL POINTER
065064 000402          BR      1004$      ;;BRANCH OVER IBSAVE CLEARING

065066 105037 065232          1003$: CLRB      IBSAVE      ;;CLEAR IBSAVE SO 2ND TIME THROUGH EXITS
065072          1004$:
065072 032777 020000 114054 BIT      #BIT13,@SWR      ;;SKIP TYPEOUT IF SET
065100 001004          BNE      20$      ;;SKIP TYPEOUTS
065102 004737 065234          JSR      PC, ERRTP      ;;GO TO USER ERROR ROUTINE
```

```

065106 104401 001217 TYPE ,SCRLF
065112 122737 000001 001242 20$: CMPB #APTENV,$ENV ;;RUNNING IN APT MODE
065120 001007 BNE 2$ ;;NO,SKIP APT ERROR REPORT
065122 113737 001130 065134 MOVB $ITEMB,21$ ;;SET ITEM NUMBER AS ERROR NUMBER
065130 004737 070176 JSR PC,$ATY4 ;;REPORT FATAL ERROR TO APT
065134 000 21$: .BYTE 0
065135 000 .BYTE 0
065136 000777 22$: BR 22$ ;;APT ERROR LOOP
065140 105737 065232 2$: TSTB IBSAVE ;;SEE IF IBSAVE IS LOADED
065144 001005 BNE 3$ ;;BRANCH IF NOT - NO HALT ON PWR MON BIT ERROR
065146 005777 114002 TST @SWR ;;HALT ON ERROR
065152 100002 BPL 3$ ;;SKIP IF CONTINUE
065154 000000 HALT ;;HALT ON ERROR.
065156 104410 CKSWR ;;TEST FOR CHANGE IN SOFT-SWR
065160 032777 001000 113766 3$: BIT #BIT09,@SWR ;;LOOP ON ERROR SWITCH SET?
065166 001402 BEQ 4$ ;;BR IF NO
065170 013716 001124 MOV $LPERR,(SP) ;;FUDGE RETURN FOR LOOPING
065174 005737 001210 4$: TST $ESCAPE ;;CHECK FOR AN ESCAPE ADDRESS
065200 001402 BEQ 5$ ;;BR IF NONE
065202 013716 001210 MOV $ESCAPE,(SP) ;;FUDGE RETURN ADDRESS FOR ESCAPE
065206 022737 037242 000042 5$: CMP #SENDAD,@#42 ;;ACT-11 AUTO-ACCEPT?
065214 001001 BNE 6$ ;;BRANCH IF NO
065216 000000 HALT ;;YES
065220 105737 065232 6$: TSTB IBSAVE ;;SEE IF ITEM BYTE SAVE LOCATION HAS AN ERROR CALL
065224 001210 BNE 7$ ;;BRANCH BACK TO CALL ORIGINAL ERROR
065226 000002 RTI ;;RETURN
065230 000000 CPSAVE: .WORD 0 ;;LOCATION TO SAVE CPU ERROR REG CONTENTS
065232 000000 IBSAVE: .WORD 0 ;;LOCATION TO SAVE ITEM BYTE

```

```

      .SBTTL  ERROR TIMEOUT ROUTINE

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

13 065234 104414
14 065236 032777 020000 113710
15 065244 001402
16 065246 000137 066074

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

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

23
24          ;TYPE 'DRIVE TYPE' RM05, RM03 OR RM02 FOR UNIT UNDER TEST
25 065272 013700 001276
26 065276 016000 000026
27 065302 042700 177740
28 065306 012737 071453 065360
29 065314 022700 000024
30 065320 001414
31
32 065322 012737 071446 065360
33 065330 022700 000025
34 065334 001406
35
36 065336 012737 071460 065360
37 065344 022700 000027
38 065350 001004
39 065352 104401 066145
40 065356 104401
41 065360 000000
42
43          ;TYPE TEST NUMBER, ERROR NUMBER AND PROGRAM COUNTER
44 065362 005037 066100
45 065366 013737 001226 066100
46 065374 104401 066115
47 065400 013746 066100

              MOV     $BASE,R0        ;GET RM BASE ADDRESS
              MOV     RMDT(R0),R0     ;GET DRIVE TYPE REGISTER
              BIC     #177740,R0      ;SAVE DRIVE TYPE BITS AND
              MOV     #$RM03,3$       ;GET ASCII DRIVE TYPE
              CMP     #24,R0          ;IS DEVICE AN RM03 ?
              BEQ     2$              ;YES !!

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

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

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

      065404 104403
      065406 003
      065407 000
48 065410 005037 066102
49 065414 113737 001130 066102
  
```

50	065422	001406		BEQ	5\$	:SKIP IF NO ERROR CALLED
51	065424	104401	066125	TYPE	,ERTY02	:TYPE 'ERR'
52	065430	013746	066102	MOV	ERRNMB,-(SP)	:SAVE ERRNMB FOR TYPEOUT
	065434	104403		TYPOS		:TYPE ERROR NUMBER
	065436	003		.BYTE	3	:GO TYPE--OCTAL ASCII
	065437	000		.BYTE	0	:TYPE 3 DIGIT(S)
53	065440	104401	066134	5\$: TYPE	,ERTY03	:SUPPRESS LEADING ZEROS
54	065444	013746	001132	MOV	\$ERRPC,-(SP)	:TYPE 'PC='
	065450	104403				:SAVE \$ERRPC FOR TYPEOUT
	065452	006		TYPOS		:TYPE PROGRAM COUNTER
	065453	001		.BYTE	6	:GO TYPE--OCTAL ASCII
				.BYTE	1	:TYPE 6 DIGIT(S)
						:TYPE LEADING ZEROS
55						
56						:GENERATE POINTER TO ERROR TABLE UNLESS ERROR NUMBER IS 0
57	065454	005737	066102	6\$: TST	ERRNMB	:WAS AN ERROR CALLED?
58	065460	001002		BNE	7\$	:BR IF YES
59	065462	000137	066074	JMP	27\$	:NO--EXIT
60						
61	065466	104401	001217	7\$: TYPE	,\$CRLF	:YES--TYPE CRLF
62	065472	105037	066106	CLRB	BOTFLG	:CLEAR BOT FLAG
63	065476	105037	066107	CLRB	CHRCNT	:CLEAR CHARACTER COUNTER
64	065502	013700	066102	MOV	ERRNMB,R0	:R0 POINTS TO FIRST OF
65	065506	122700	000177	CMPS	#177,R0	:SEE IF THIS ERROR CALL IS SPECIAL POWER FAIL CALL
66	065512	001003		BNE	8\$	:BRANCH IF NOT
67	065514	012700	066152	MOV	#PFERR,R0	:MOVE POWER FAIL ERROR CALL TABLE TO R0
68	065520	000405		BR	9\$	
69	065522	006300		8\$: ASL	R0	:FOUR ENTRIES IN ERROR
70	065524	006300		ASL	R0	:TABLE
71	065526	006300		ASL	R0	
72	065530	062700	001572	ADD	#ERRTB-8.,R0	
73	065534	011001		9\$: MOV	(R0),R1	:R1 POINTS TO ERROR MESSAGE
74						:TABLE
75	065536	001507		BEQ	19\$	:BRANCH IF NO ERROR MESSAGE
76						
77						:TYPE THE ERROR MESSAGE
78	065540	012102		10\$: MOV	(R1)+,R2	:R2-ADDRESS OF MESSAGE STRING
79	065542	001505		BEQ	19\$	:BRANCH IF END OF MESSAGE
80	065544	010237	065712	MOV	R2,18\$	:LOAD ADDRESS OF STRING
81	065550	005037	066104	CLR	BOTADR	:CLEAR BOT ADDRESS
82	065554	112203		11\$: MOVB	(R2)+,R3	:END OF STRING??
83	065556	001454		BEQ	17\$	:YES!!
84	065560	122703	000015	CMPS	#CR,R3	:CARRIAGE RETURN??
85	065564	001003		BNE	12\$	:NO!!
86	065566	105037	066107	CLRB	CHRCNT	:YES--CLEAR CHAR COUNT
87	065572	000770		BR	11\$	:GET NEXT CHARACTER
88	065574	122703	000012	12\$: CMPS	#LF,R3	:LINE FEED??
89	065600	001765		BEQ	11\$	:YES--GET NEXT CHARACTER
90	065602	122703	000011	CMPS	#HT,R3	:HORIZONTAL TAB??
91	065606	001007		BNE	14\$	:NO..
92	065610	105237	066107	13\$: INCB	CHRCNT	:ADJUST CHARACTER COUNT
93	065614	132737	000007	BITB	#7,CHRCNT	
94	065622	001372		BNE	13\$	
95	065624	000407		BR	15\$	
96	065626	105237	066107	14\$: INCB	CHRCNT	:INCREMENT CHARACTER COUNT
97	065632	122703	000040	CMPS	#',R3	:SPACE??
98	065636	001002		BNE	15\$	:NO.

99	065640	010237	066104		MOV	R2,BOTADR	:SAVE ADDRESS OF SPACE
100	065644	122737	000100	066107	15\$:	CMPS	#64,CHRCNT
101	065652	103340			BHIS	11\$	:END OF LINE??
102	065654	013704	066104		MOV	BOTADR,R4	:GET ADDRESS OF LAST SPACE
103	065660	001007			BNE	16\$	:BRANCH IF SPACE DETECTED
104	065662	104401	001217		TYPE	,SCLRF	:TYPE CRLF
105	065666	105037	066107		CLRB	CHRCNT	:CLEAR CHARACTER COUNT
106	065672	013702	065712		MOV	18\$,R2	:SET UP R2 FOR TESTING
107	065676	000726			BR	11\$	
108	065700	105044			CLRB	-(R4)	:REPLACE SPACE
109	065702	112737	177777	066106	16\$:	MOVB	#-1,BOTFLG
110	065710	104401			17\$:	TYPE	
111	065712	000000			18\$:	.WORD	
112	065714	105737	066106		TSTB	BOTFLG	:WAS STRING TRUNCATED??
113	065720	001707			BEQ	10\$	:NO!!
114	065722	104401	001217		TYPE	,SCLRF	:YES-TYPE CRLF
115	065726	105037	066106		CLRB	BOTFLG	:CLEAR BOT FLAG
116	065732	105037	066107		CLRB	CHRCNT	:CLEAR CHARACTER COUNT
117	065736	013702	066104		MOV	BOTADR,R2	:SETUP R2 FOR TESTING
118	065742	010237	065712		MOV	R2,18\$	:SETUP 18\$ FOR TYPING
119	065746	112742	000040		MOVB	#1,-(R2)	:RESTORE SPACE
120	065752	105722			TSTB	(R2)+	:RESTORE R2
121	065754	000677			BR	11\$	:TYPE REST OF STRING
122							
123							:TYPE ERROR HEADER AND ERROR DATA
124	065756	016001	000002		19\$:	MOV	2(R0),R1
125	065762	001444			BEQ	27\$	:R1 POINTS TO ERROR HEADER TABLE
126	065764	104401	001217		TYPE	,SCLRF	:BRANCH IF NO HEADER
127	065770	016002	000004		MOV	4(R0),R2	:R2 POINTS TO DATA ADDRESS TABLE
128	065774	016003	000006		MOV	6(R0),R3	:R3 POINTS TO FORMAT TABLE
129	066000	012137	066010		20\$:	MOV	(R1)+,21\$
130	066004	001433			BEQ	27\$	:PUT HEADER ADDRESS FOR TYPE
131							:BRANCH IF END OF HEADERS
132	066006	104401			TYPE		:R4 POINTS TO DATA ADDRESS
133	066010	000000			21\$:	.WORD	0
134	066012	104401	001217		TYPE	,SCLRF	:R5 POINTS TO FORMAT
135	066016	005702			TST	R2	:WHAT KIND OF DATA??
136	066020	001767			BEQ	20\$	:DATA WITH HEADER??
137	066022	012204			MOV	(R2)+,R4	:NO!!
138	066024	012305			MOV	(R3)+,R5	:R4 POINTS TO DATA ADDRESS
139	066026	105725			22\$:	TSTB	(R5)+
140	066030	100407			BMI	24\$	:BINARY
141	066032	001403			BEQ	23\$	:OCTAL
142	066034	013446			MOV	@(R4)+,-(SP)	:DECIMAL
143	066036	104405			TYPDS		
144	066040	000405			BR	25\$	
145	066042	013446			23\$:	MOV	@(R4)+,-(SP)
146	066044	104402			TYPDC		
147	066046	000402			BR	25\$	
148	066050	013446			24\$:	MOV	@(R4)+,-(SP)
149	066052	104406			TYPBN		
150	066054	005714			25\$:	TST	(R4)
151	066056	001403			BEQ	26\$	:MORE DATA??
152	066060	104401	066142		TYPE	,ERTY04	:YES-TYPE 2 SPACES
153	066064	000760			BR	22\$	:AND CONTINUE
154	066066	104401	001217		26\$:	TYPE	,SCLRF
155	066072	000742			BR	20\$	:TYPE ONE BLANK LINE
							:BEFORE NEXT HEADER

156	066074	1044'5			27\$:	RESREG		
157	066076	000207				RTS	PC	
158								
159	066100	000000			TSTNMB:	.WORD	0	;TEST NUMBER
160	066102	000000			ERRNMB:	.WORD	0	;ERROR NUMBER
161	066104	000000			BOTADR:	.WORD	0	;BEGINNING OF TEXT ADDRESS
162	066106	000			BOTFLG:	.BYTE	0	;BOT FLAG
163	066107	000			CHRCNT:	.BYTE	0	;CHARACTER COUNT
164								
165	066110	104	122	126	ERTY00:	.ASCIZ	@DV#@	
166	066115	054	040	124	ERTY01:	.ASCIZ	@, TEST#@	
167	066125	054	040	105	ERTY02:	.ASCIZ	@, ERR#@	
168	066134	054	040	120	ERTY03:	.ASCIZ	@, PC=@	
169	066142	040	040	000	ERTY04:	.ASCIZ	@ @	
170	066145	040	055	040	ERTY05:	.ASCIZ	@ - @	
171					.EVEN			
172	066152	066162	066250	066266	PFECH:	PFECH1,PFECH2,PFECH3,PFECH4		;WORDS DEFINING TABLES BELOW
173	066162	066166	000000		PFECH1:	.+4,0		
174	066166	120	117	127		.ASCIZ	?POWER MONITOR BIT IN CPU ERROR REGISTER FOUND SET?	
175						.EVEN		
176	066250	066254	000000		PFECH2:	.+4,0		
177	066254	103	120	125		.ASCIZ	?CPUERREG?	
178						.EVEN		
179	066266	066270			PFECH3:	.+2		
180	066270	065230	000000			.WORD	CPSAVE,0	
181	066274	066276			PFECH4:	.+2		
182	066276	000	000			.BYTE	0,0	

1

.SBTTL TTY INPUT ROUTINE

066300 000000
066302 000000
066304 000000
066306 066307

ENABL LSB
\$TKCNT: .WORD 0 ;:NUMBER OF ITEMS IN QUEUE
\$TKQIN: .WORD 0 ;:INPUT POINTER
\$TKQOUT: .WORD 0 ;:OUTPUT POINTER
\$TKQSR: .BLKB 1 ;:TTY KEYBOARD QUEUE
\$TKQEND=.
.EVEN

;*TK INITIALIZE ROUTINE
;*THIS ROUTINE WILL INITIA.LIZE THE TTY KEYBOARD INPUT QUFUE
;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT

CALL:
JSR PC,\$TKINT
RETURN

066310 005037 066300 \$TKINT: CLR \$TKCNT ;:CLEAR COUNT OF ITEMS IN QUEUE
066314 012737 066306 066302 MOV \$TKQSR,\$TKQIN ;:MOVE THE STARTING ADDRESS OF THE
066322 013737 066302 066304 MOV \$TKQIN,\$TKQOUT ;:QUEUE INTO THE INPUT & OUTPUT POINTERS.
066330 012737 066360 000060 MOV \$TKSRV,\$TKVEC ;:INITIALIZE THE KEYBOARD VECTOR
066336 012737 000200 000062 MOV #200,\$TKVEC+2 ;:'BR' LEVEL 4
066344 005777 112612 TST \$TKB ;:CLEAR DONE FLAG
066350 012777 000100 112602 MOV #100,\$TKS ;:ENABLE TTY KEYBOARD INTERRUPT
066356 000207 RTS PC ;:RETURN TO CALLER

;*K SERVICE ROUTINE
;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
;*IT IN THE QUEUE.
;*IF THE CHARACTER IS A 'CONTROL-C' (^C) \$TKINT IS CALLED AND
;*UPON RETURN EXIT IS MADE TO THE 'CONTROL-C' RESTART ADDRESS (SHUT2)

066360 117746 112576 \$TKSRV: MOVB \$TKB, -(SP) ;:PICKUP THE CHARACTER
066364 042716 177600 BIC #^C177, (SP) ;:STRIP THE JUNK
066370 021627 000021 CMP (SP),#\$XON ;:IS IT A RANDOM XON?
066374 001002 BNE 30\$;:BRANCH IF NO
066376 005726 TST (SP)+ ;:CLEAN RANDOM XON OFF STACK
066400 000002 RTI ;:RETURN
066402 30\$: CMP (SP),#3 ;:IS IT A CONTROL C?
066406 001007 BNE 1\$;:BRANCH IF NO
066410 104401 067506 TYPE ,SCNTLC ;:TYPE A CONTROL-C (^C)
066414 004737 066310 JSR PC,\$TKINT ;:INIT THE KEYBOARD
066420 005726 TST (SP)+ ;:CLEAN UP STACK
066422 000137 067550 JMP SHUT2 ;:CONTROL C RESTART
066426 021627 000007 1\$: CMP (SP),#7 ;:IS IT A CONTROL G?
066432 001004 BNE 2\$;:BRANCH IF NO
066434 022737 000176 001154 CMP #SWREG,SWR ;:IS SOFT-SWR SELECTED?
066442 001500 BEQ 6\$;:GO TO SWR CHANGE

066444 2\$: CMP #1,\$TKCNT ;:IS THE QUEUE FULL?
066444 022737 000001 066300 BNE 3\$;:BRANCH IF NO
066452 001004 TYPE ,SBELL ;:RING THE TTY BELL
066454 104401 001212

```

066460 005726      TST      (SP)+      ;;CLEAN CHARACTER OFF OF STACK
066462 000451      BR       5$          ;;EXIT
066464 021627 000023 3$:      CMP      (SP),#23      ;;IS IT A CONTROL-S?
066470 001021      BNE      32$          ;;BRANCH IF NO
066472 005077 112462      CLR      @STKS      ;;DISABLE TTY KEYBOARD INTERRUPTS
066476 005726      TST      (SP)+      ;;CLEAN CHAR OFF STACK
066500 105777 112454 31$:    TSTB     @STKS      ;;WAIT FOR A CHAR
066504 100375      BPL      31$          ;;LOOP UNTIL ITS THERE
066506 117746 112450      MOVB     @STKB,-(SP)      ;;GET THE CHARACTER
066512 042716 177600      BIC      #^C177,(SP)      ;;MAKE IT 7-BIT ASCII
066516 022627 000021      CMP      (SP)+,#21      ;;IS IT A CONTROL-Q?
066522 001366      BNE      31$          ;;BRANCH IF NO
066524 012777 000100 112426  MOV     #100,@STKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
066532 000002      RTI              ;;RETURN
066534 005237 066300 32$:    INC      $TKCNT      ;;COUNT THIS CHARACTER
066540 021627 000140      CMP      (SP),#140      ;;IS IT UPPER CASE?
066544 002405      BLT      4$          ;;BRANCH IF YES
066546 021627 000175      CMP      (SP),#175      ;;IS IT A SPECIAL CHAR?
066552 003002      BGT      4$          ;;BRANCH IF YES
066554 042716 000040      BIC      #40,(SP)          ;;MAKE IT UPPER CASE
066560 112677 177516 4$:    MOVB     (SP)+,@STKQIN      ;;AND PUT IT IN QUEUE
066564 005237 066302      INC      $TKQIN      ;;UPDATE THE POINTER
066570 023727 066302 066307  CMP     $TKQIN,$TKQEND      ;;GO OFF THE END?
066576 001003      BNE      5$          ;;BRANCH IF NO
066600 012737 066306 066302  MOV     $TKQSRT,$TKQIN      ;;RESET THE POINTER
066606 000002 5$:      RTI              ;;RETURN

```

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

```

066610 022737 000176 001154 $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
066616 001124      BNE      15$          ;;EXIT IF NOT
066620 105777 112334      TSTB     @STKS      ;;IS A CHAR WAITING?
066624 100121      BPL      15$          ;;IF NOT, EXIT
066626 117746 112330      MOVB     @STKB,-(SP)      ;;YES
066632 042716 177600      BIC      #^C177,(SP)      ;;MAKE IT 7-BIT ASCII
066636 021627 000007      CMP      (SP),#7          ;;IS IT A CONTROL-G?
066642 001300      BNE      2$          ;;IF NOT, PUT IT IN THE TTY QUEUE
                                ;;AND EXIT

```

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

```

066644 123727 001150 000001 6$:    CMPB     $AUTOB,#1      ;;ARE WE RUNNING IN AUTO-MODE?
066652 001674      BEQ      2$          ;;BRANCH IF YES
066654 005726      TST      (SP)+      ;;CLEAR CONTROL-G OFF STACK
066656 004737 066310      JSR      PC,$TKINT      ;;FLUSH THE TTY INPUT QUEUE
066662 005077 112272      CLR      @STKS      ;;DISABLE TTY KEYBOARD INTERRUPTS
066666 112737 000001 001151      MOVB     #1,$INTAG      ;;SET INTERRUPT MODE INDICATOR

066674 104401 067520      TYPE     , $CNTLG      ;;ECHO THE CONTROL-G (^G)
066700 104401 067525      TYPE     , $MSWR      ;;TYPE CURRENT CONTENTS
066704 013746 000176      MOV      SWREG,-(SP)      ;;SAVE SWREG FOR TYPEOUT
066710 104402      TYOC          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)

```


066712	104401	067536		TYPE	, \$MNFw	:: PROMPT FOR NEW SWR
066716	005046		19\$:	CLR	-(SP)	:: CLEAR COUNTER
066720	005046			CLR	-(SP)	:: THE NEW SWR
066722	105777	112232	7\$:	TSTB	@\$TKS	:: CHAR THERE?
066726	100375			BPL	7\$	:: IF NOT TRY AGAIN
066730	117746	112226		MOVB	@\$TKB, -(SP)	:: PICK UP CHAR
066734	042716	177600		BIC	#^C177, (SP)	:: MAKE IT 7-BIT ASCII
066740	021627	000003		CMP	(SP), #3	:: IS IT A CONTROL-C?
066744	001015			BNE	9\$	:: BRANCH IF NOT
066746	104401	067506		TYPE	, \$CNTLC	:: YES, ECHO CONTROL-C (^C)
066752	062706	000006		ADD	#6, SP	:: CLEAN UP STACK
066756	123727	001151	000001	CMPB	\$INTAG, #1	:: REENABLE TTY KEYBOARD INTERRUPTS?
066764	001003			BNE	8\$	:: BRANCH IF NO
066766	012777	000100	112164	MOV	#100, @\$TKS	:: ALLOW TTY KEYBOARD INTERRUPTS
066774	000137	067550	8\$:	JMP	SHUT2	:: CONTROL-C RESTART
067000	021627	000025	9\$:	CMP	(SP), #25	:: IS IT A CONTROL-U?
067004	001005			BNE	10\$	:: BRANCH IF NOT
067006	104401	067513		TYPE	, \$CNTLU	:: YES, ECHO CONTROL-U (^U)
067012	062706	000006	20\$:	ADD	#6, SP	:: IGNORE PREVIOUS INPUT
067016	000737			BR	19\$	:: LET'S TRY IT AGAIN
067020	021627	000015	10\$:	CMP	(SP), #15	:: IS IT A <CR>?
067024	001022			BNE	16\$	:: BRANCH IF NO
067026	005766	000004		TST	4(SP)	:: YES, IS IT THE FIRST CHAR?
067032	001403			BEQ	11\$	:: BRANCH IF YES
067034	016677	000002	112112	MOV	2(SP), @SWR	:: SAVE NEW SWR
067042	062706	000006	11\$:	ADD	#6, SP	:: CLEAN UP STACK
067046	104401	001217	14\$:	TYPE	, \$CRLF	:: ECHO <CR> AND <LF>
067052	123727	001151	000001	CMPB	\$INTAG, #1	:: RE-ENABLE TTY KBD INTERRUPTS?
067060	001003			BNE	15\$	:: BRANCH IF NOT
067062	012777	000100	112070	MOV	#100, @\$TKS	:: RE-ENABLE TTY KBD INTERRUPTS
067070	000002		15\$:	RTI		:: RETURN
067072	004737	063732	16\$:	JSR	PC, \$TYPEC	:: ECHO CHAR
067076	021627	000060		CMP	(SP), #60	:: CHAR < 0?
067102	002420			BLT	18\$	:: BRANCH IF YES
067104	021627	000067		CMP	(SP), #67	:: CHAR > 7?
067110	003015			BGT	18\$	:: BRANCH IF YES
067112	042726	000060		BIC	#60, (SP)+	:: STRIP-OFF ASCII
067116	005766	000002		TST	2(SP)	:: IS THIS THE FIRST CHAR
067122	001403			BEQ	17\$	:: BRANCH IF YES
067124	006316			ASL	(SP)	:: NO, SHIFT PRESENT
067126	006316			ASL	(SP)	:: CHAR OVER TO MAKE
067130	006316			ASL	(SP)	:: ROOM FOR NEW ONE.
067132	005266	000002	17\$:	INC	2(SP)	:: KEEP COUNT OF CHAR
067136	056616	177776		BIS	-2(SP), (SP)	:: SET IN NEW CHAR
067142	000667			BR	7\$	:: GET THE NEXT ONE
067144	104401	001216	18\$:	TYPE	, \$QUES	:: TYPE ?<CR><LF>
067150	000720			BR	20\$	:: SIMULATE CONTROL-U
				.DSABL	LSB	

::*****

```

      ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
      ;*CALL:
      ;*      RDCHR
      ;*      RETURN HERE
      ;*      ;:GET A CHARACTER FROM THE QUEUE
      ;*      ;:CHARACTER IS ON THE STACK
      ;*      ;:WITH PARITY BIT STRIPPED OFF
      ;*
      ;*
067152 011646      $RDCHR: MOV      (SP),-(SP)      ;:PUSH DOWN THE PC AND
067154 016666 000004 000002      MOV      4(SP),2(SP)      ;:THE PS
067162 005066 000004      CLR      4(SP)      ;:GET READY FOR A CHARACTER
067166 005046      CLR      -(SP)      ;:PUT NEW PS ON STACK
067170 012746 067176      MOV      #64$,-(SP)      ;:PUT NEW PC ON STACK
067174 000002      RTI      ;:POP NEW PC AND PS
067176
067176 005737 066300      64$:      TST      $TKCNT      ;:WAIT ON A CHARACTER
067202 001775      1$:      BEQ      1$
067204 005337 066300      DEC      $TKCNT      ;:DECREMENT THE COUNTER
067210 117766 177070 000004      MOV      @STKQOUT,4(SP)      ;:GET ONE CHARACTER
067216 005237 066304      INC      $TKQOUT      ;:UPDATE THE POINTER
067222 023727 066304 066307      CMP      $TKQOUT,$TKQEND      ;:DID IT GO OFF OF THE END?
067230 001003      BNE      2$      ;:BRANCH IF NO
067232 012737 066306 066304      MOV      $TKQSRRT,$TKQOUT      ;:RESET THE POINTER
067240 000002      2$:      RTI      ;:RETURN
      ;*****
      ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
      ;*CALL:
      ;*      RDLIN
      ;*      RETURN HERE
      ;*      ;:INPUT A STRING FROM THE TTY
      ;*      ;:ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
      ;*      ;:TERMINATOR WILL BE A BYTE OF ALL 0'S
      ;*
      ;*
067242 010346      $RDLIN: MOV      R3,-(SP)      ;:SAVE R3
067244 005046      CLR      -(SP)      ;:CLEAR THE RUBOUT KEY
067246 012703 067476      1$:      MOV      $TTYIN,R3      ;:GET ADDRESS
067252 022703 067506      2$:      CMP      $TTYIN+8.,R3      ;:BUFFER FULL?
067256 101456      BLOS      4$      ;:BR IF YES
067260 104411      RDCHR      ;:GO READ ONE CHARACTER FROM THE TTY
067262 112613      MOV      (SP)+,(R3)      ;:GET CHARACTER
067264 122713 000177      10$:      CMP      #177,(R3)      ;:IS IT A RUBOUT
067270 001022      BNE      5$      ;:BR IF NO
067272 005716      TST      (SP)      ;:IS THIS THE FIRST RUBOUT?
067274 001007      BNE      6$      ;:BR IF NO
067276 112737 000134 067474      MOV      #'\\,9$      ;:TYPE A BACK SLASH
067304 104401 067474      TYPE      ,9$
067310 012716 177777      MOV      #-1,(SP)      ;:SET THE RUBOUT KEY
067314 005303      6$:      DEC      R3      ;:BACKUP BY ONE
067316 020327 067476      CMP      R3,$TTYIN      ;:STACK EMP'Y?
067322 103434      BLO      4$      ;:BR IF YES
067324 111337 067474      MOV      (R3),9$      ;:SETUP TO TYPEOUT THE DELETED CHAR.
067330 104401 067474      TYPE      ,9$      ;:GO TYPE
067334 000746      BR      2$      ;:GO READ ANOTHER CHAR.
067336 005716      5$:      TST      (SP)      ;:RUBOUT KEY SET?
067340 001406      BEQ      7$      ;:BR IF NO
067342 112737 000134 067474      MOV      #'\\,9$      ;:TYPE A BACK SLASH
067350 104401 067474      TYPE      ,9$
067354 005016      CLR      (SP)      ;:CLEAR THE RUBOUT KEY
067356 122713 000025      7$:      CMP      #25,(R3)      ;:IS CHARACTER A CTRL U?
067362 001003      BNE      8$      ;:BR IF NO
  
```

067364	104401	067513		TYPE	,\$CNTLU	::TYPE A CONTROL 'U'	
067370	000726			BR	1\$	::GO START OVER	
067372	122713	00G022	8\$:	CMPB	#22,(R3)	::IS CHARACTER A '^R'?	
067376	001011			BNE	3\$	::BRANCH IF NO	
067400	105013			CLRB	(R3)	::CLEAR THE CHARACTER	
067402	104401	001217		TYPE	,\$CRLF	::TYPE A 'CR' & 'LF'	
067406	104401	067476		TYPE	,\$TTYIN	::TYPE THE INPUT STRING	
067412	000717			BR	2\$	::GO PICKUP ANOTHER CHACTER	
067414	104401	001216	4\$:	TYPE	,\$QUES	::TYPE A '?'	
067420	000712			BR	1\$	::CLEAR THE BUFFER AND LOOP	
067422	111337	067474	3\$:	MOVB	(R3),9\$	::ECHO THE CHARACTER	
067426	104401	067474		TYPE	,\$9\$		
067432	122723	000015		CMPB	#15,(R3)+	::CHECK FOR RETURN	
067436	001305			BNE	2\$	::LOOP IF NOT RETURN	
067440	105063	177777		CLRB	-1(R3)	::CLEAR RETURN (THE 15)	
067444	104401	001220		TYPE	,\$LF	::TYPE A LINE FEED	
067450	005726			TST	(SP)+	::CLEAN RUBOUT KEY FROM THE STACK	
067452	012603			MOV	(SP)+,R3	::RESTORE R3	
067454	011646			MOV	(SP),-(SP)	::ADJUST THE STACK AND PUT ADDRESS OF THE	
067456	016666	0C0004 000002		MOV	4(SP),2(SP)	::FIRST ASCII CHARACTER ON IT	
067464	012766	067476 000004		MOV	,\$TTYIN,4(SP)		
067472	000002			RTI		::RETURN	
067474	000		9\$:	.BYTE	0	::STORAGE FOR ASCII CHAR. TO TYPE	
067475	000			.BYTE	0	::TERMINATOR	
067476				,\$TTYIN:	.BLKB 8.	::RESERVE 8 BYTES FOR TTY INPUT	
067506	136	103	015	,\$CNTLC:	.ASCIZ /^C/<15><12>	::CONTROL 'C'	
067513	136	125	015	,\$CNTLU:	.ASCIZ /^U/<15><12>	::CONTROL 'U'	
067520	136	107	015	,\$CNTLG:	.ASCIZ /^G/<15><12>	::CONTROL 'G'	
067525	015	012	123	,\$MSWR:	.ASCIZ <15><12>/SWR - /		
067536	040	040	'16	,\$MNEW:	.ASCIZ / NEW = /		
				.EVEN			
3	067550	012737	177777	001326	SHUT2:	MOV # -1,CILFG	::SET THE CONTROL-C FLAG
4	067556	105737	001116		TSTB	,\$TSTNM	::DOING ANY TESTS ?
5	067562	001002			BNE	1\$	::BR IF YES
6	067564	000137	037262		JMP	SHUT	::NO--RESPOND TO ^C
7	067570	000002		1\$:	RTI		::EXIT FROM INTERRUPT

.SBTTL READ AN OCTAL NUMBER FROM THE TTY

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

*CALL:

* RDOCT
 * RETURN HERE
 *

:::READ AN OCTAL NUMBER
 :::LOW ORDER BITS ARE ON TOP OF THE STACK
 :::HIGH ORDER BITS ARE IN \$HI OCT

067572 011646
 067574 016666 000004 00000?
 067602 010046
 067604 010146
 067606 010246
 067610 104412
 067612 012600
 067614 005001
 067616 005002
 067620 112046
 067622 001412
 067624 006301
 067626 006102
 067630 006301
 067632 006102
 067634 006301
 067636 006102
 067640 042716 177770
 067644 062601
 067646 000764
 067650 005726
 067652 010166 000012
 067656 010237 067672
 067662 012602
 067664 012601
 067666 012600
 067670 000002
 067672 000000

\$RDOCT: MOV (SP),-(SP)
 MOV 4(SP),2(SP)
 MOV R0,-(SP)
 MOV R1,-(SP)
 MOV R2,-(SP)
 1\$: RDLIN
 MOV (SP)+,R0
 CLR R1
 CLR R2
 2\$: MOVB (R0)+,-(SP)
 BEQ 3\$
 ASL R1
 ROL R2
 ASL R1
 ROL R2
 ASL R1
 ROL R2
 BIC #^C7,(SP)
 ADD (SP)+,R1
 BR 2\$
 3\$: TST (SP)+
 MOV R1,12(SP)
 MOV R2,\$HI OCT
 MOV (SP)+,R2
 MOV (SP)+,R1
 MOV (SP)+,R0
 RTI
 \$HI OCT: .WORD 0

:::PROVIDE SPACE FOR THE
 :::INPUT NUMBER
 :::PUSH R0 ON STACK
 :::PUSH R1 ON STACK
 :::PUSH R2 ON STACK
 :::READ AN ASCII LINE
 :::GET ADDRESS OF 1ST CHARACTER
 :::CLEAR DATA WORD
 :::PICKUP THIS CHARACTER
 :::IF ZERO GET OUT
 :::*2
 :::*4
 :::*8
 :::STRIP THE ASCII JUNK
 :::ADD IN THIS DIGIT
 :::LOOP
 :::CLEAN TERMINATOR FROM STACK
 :::SAVE THE RESULT
 :::POP STACK INTO R2
 :::POP STACK INTO R1
 :::POP STACK INTO R0
 :::RETURN
 :::HIGH ORDER BITS GO HERE

.SBTTL TRAP DECODER

 ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE 'TRAP' INSTRUCTION
 ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
 ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
 ;*GO TO THAT ROUTINE.

067674	016646	000002	\$TRAP:	MOV	2(SP),-(SP)	::ASSUME THE STATUS OF
067700	042716	000020		BIC	#20,(SP)	:: THE CALLER--DO NOT ALLOW
067704	012746	067712		MOV	#1\$,-(SP)	:: T-BIT TRAPS
067710	000002			RTI		::SET THE NEW STATUS
067712	010046		1\$:	MOV	R0,-(SP)	::SAVE R0
067714	016600	000002		MOV	2(SP),R0	::GET TRAP ADDRESS
067720	005740			TST	-(R0)	::BACKUP BY 2
067722	111000			MOVB	(R0),R0	::GET RIGHT BYTE OF TRAP
067724	006300			ASL	R0	::POSITION FOR INDEXING
067726	016000	067746		MOV	\$TRPAD(R0),R0	::INDEX TO TABLE
067732	000200			RTS	R0	::GO TO ROUTINE

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

067734	011646		\$TRAP2:	MOV	(SP),-(SP)	::MOVE THE PC DOWN
067736	016666	000004		MOV	4(SP),2(SP)	::MOVE THE PSW DOWN
067744	000002	000002		RTI		::RESTORE THE PSW

.SBTTL TRAP TABLE

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

			:	ROUTINE	
			:	-----	
067746	067734		\$TRPAD:	.WORD	\$TRAP2
067750	063520			\$TYPE	::CALL=TYPE TRAP+1(104401) TTY TYPEOUT ROUTINE
067752	063316			\$TYPOC	::CALL=TYPOC TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
067754	063272			\$TYPOS	::CALL=TYPOS TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
067756	063332			\$TYPON	::CALL=TYPON TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
067760	063046			\$TYPDS	::CALL=TYPDS TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
067762	062772			\$TYPBN	::CALL=TYPBN TRAP+6(104406) TYPE BINARY (ASCII) NUMBER
067764	066700			\$GTSWR	::CALL=GTSWR TRAP+7(104407) GET SOFT-SWR SETTING
067766	066610			\$CKSWR	::CALL=CKSWR TRAP+10(104410) TEST FOR CHANGE IN SOFT-SWR
067770	067152			\$RDCHR	::CALL=RDCHR TRAP+11(104411) TTY TYPEIN CHARACTER ROUTINE
067772	067242			\$RDLIN	::CALL=RDLIN TRAP+12(104412) TTY TYPEIN STRING ROUTINE
067774	067572			\$RDOCT	::CALL=RDOCT TRAP+13(104413) READ AN OCTAL NUMBER FROM TTY
067776	062676			\$SAVREG	::CALL=SAVREG TRAP+14(104414) SAVE R0-R5 ROUTINE
070000	062734			\$RESREG	::CALL=RESREG TRAP+15(104415) RESTORE R0-R5 ROUTINE

.S\$TTL POWER DOWN AND UP ROUTINES

```

*****
:POWER DOWN ROUTINE
070002 012737 070142 000024 $PWRDN: MOV $ILLUP,@#PWRVEC ;;SET FOR FAST UP
070010 012737 000340 000026 MOV #340,@#PWRVEC+2 ;;PRIO:7
070016 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK
070020 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
070022 010246 MOV R2,-(SP) ;;PUSH R2 ON STACK
070024 010346 MOV R3,-(SP) ;;PUSH R3 ON STACK
070026 010446 MOV R4,-(SP) ;;PUSH R4 ON STACK
070030 010546 MOV R5,-(SP) ;;PUSH R5 ON STACK
070032 017746 111116 MOV @SWR,-(SP) ;;PUSH @SWR ON STACK
070036 010637 070146 MOV SP,$SAVR6 ;;SAVE SP
070042 012737 070054 000024 MOV $PWRUP,@#PWRVEC ;;SET UP VECTOR
070050 000000 HALT
070052 000776 BR -2 ;;HANG UP
*****
:POWER UP ROUTINE
070054 012737 070142 000024 $PWRUP: MOV $ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
070062 013706 070146 MOV $SAVR6,SP ;;GET SP
070066 005037 070146 CLR $SAVR6 ;;WAIT LOOP FOR THE TTY
070072 005237 070146 1$: INC $SAVR6 ;;WAIT FOR THE INC
070076 001375 BNE 1$ ;;OF WORD
070100 012677 111050 MOV (SP)+,@SWR ;;POP STACK INTO @SWR
070104 012605 MOV (SP)+,R5 ;;POP STACK INTO R5
070106 012604 MOV (SP)+,R4 ;;POP STACK INTO R4
070110 012603 MOV (SP)+,R3 ;;POP STACK INTO R3
070112 012602 MOV (SP)+,R2 ;;POP STACK INTO R2
070114 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
070116 012600 MOV (SP)+,R0 ;;POP STACK INTO R0
070120 012737 070002 000024 MOV $PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
070126 012737 000340 000026 MOV #340,@#PWRVEC+2 ;;PRIO:7
070134 104401 TYPE $POWER ;;REPORT THE POWER FAILURE
070136 070150 $PWRMG: .WORD $POWER ;;POWER FAIL MESSAGE POINTER
070140 000002 RTI
070142 000000 $ILLUP: HALT ;;THE POWER UP SEQUENCE WAS STARTED
070144 000776 BR -2 ;; BEFORE THE POWER DOWN WAS COMPLETE
070146 000000 $SAVR6: 0 ;;PUT THE SP HERE
070150 015 012 120 $POWER: .ASCIZ '<15><12>'POWER'
.EVEN

```


.SBTTL APT COMMUNICATIONS ROUTINE

```
*****
070160 112737 000001 070424 $ATY1: MOVB #1,$FFLG ;;TO REPORT FATAL ERROR
070166 112737 000001 070422 $ATY3: MOVB #1,$MFLG ;;TO TYPE A MESSAGE
070174 000403 BR $ATYC
070176 112737 000001 070424 $ATY4: MOVB #1,$FFLG ;;TO ONLY REPORT FATAL ERROR
070204 $ATYC:
070204 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK
070206 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
070210 105737 070422 TSTB $MFLG ;;SHOULD TYPE A MESSAGE?
070214 001450 BEQ 5$ ;;IF NOT: BR
070216 122737 000001 001242 CMPB #APTENV,$ENV ;;OPERATING UNDER APT?
070224 001031 BNE 3$ ;;IF NOT: BR
070226 132737 000100 001243 BITB #APTSPool,$ENVM ;;SHOULD SPOOL MESSAGES?
070234 001425 BEQ 3$ ;;IF NOT: BR
070236 017600 000004 MOV @4(SP),R0 ;;GET MESSAGE ADDR.
070242 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDR.
070250 005737 001222 1$: TST $MSGTYPE ;;SEE IF DONE W/ LAST XMISSION?
070254 001375 BNE 1$ ;;IF NOT: WAIT
070256 010037 001236 MOV R0,$MSGAD ;;PUT ADDR IN MAILBOX
070262 105720 2$: TSTB (R0)+ ;;FIND END OF MESSAGE
070264 001376 BNE 2$
070266 163700 001236 SUB $MSGAD,R0 ;;SUB START OF MESSAGE
070272 006200 ASR R0 ;;GET MESSAGE LGTH IN WORDS
070274 010037 001240 MOV R0,$MSGGLT ;;PUT LENGTH IN MAILBOX
070300 012737 000004 001222 MOV #4,$MSGCTYPE ;;TELL APT TO TAKE MSG.
070306 000413 BR 5$
070310 017637 000004 070334 3$: MOV @4(SP),4$ ;;PUT MSG ADDR IN JSR LINKAGE
070316 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDRESS
070324 013746 177776 MOV 177776,-(SP) ;;PUSH 177776 ON STACK
070330 004737 063520 JSR PC,$TYPE ;;CALL TYPE MACRO
070334 000000 4$: .WORD 0
070336 5$:
070336 105737 070424 10$: TSTB $FFLG ;;SHOULD REPORT FATAL ERROR?
070342 001416 BEQ 12$ ;;IF NOT: BR
070344 005737 001242 TST $ENV ;;RUNNING UNDER APT?
070350 001413 BEQ 12$ ;;IF NOT: BR
070352 005737 001222 11$: TST $MSGTYPE ;;FINISHED LAST MESSAGE?
070356 001375 BNE 11$ ;;IF NOT: WAIT
070360 017637 000004 001224 MOV @4(SP),$FATAL ;;GET ERROR #
070366 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDR.
070374 005237 001222 INC $MSGTYPE ;;TELL APT TO TAKE ERROR
070400 105037 070424 12$: CLRB $FFLG ;;CLEAR FATAL FLAG
070404 105037 070423 CLRB $LFLG ;;CLEAR LOG FLAG
070410 105037 070422 CLRB $MFLG ;;CLEAR MESSAGE FLAG
070414 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
070416 012600 MOV (SP)+,R0 ;;POP STACK INTO R0
070420 000207 RTS PC ;;RETURN
070422 000 $MFLG: .BYTE 0 ;;MESSG. FLAG
070423 000 $LFLG: .BYTE 0 ;;LOG FLAG
070424 000 $FFLG: .BYTE 0 ;;FATAL FLAG
          .EVEN
          APTSIZE = 200
          APTENV = 001
          APTSPool = 100
          APTCSUP = 040
000200
000001
000100
000040
```

.SBTTL CONSOLE MESSAGES

2					
3	070426	200	103	*01	SCMSG: .ASCIZ <CRLF>@CANNOT RECOVER THE BAD SECTOR FILES ON THIS DEVICE@
4	070512	075	000		EQUALS: .ASCIZ @=@
5	070514	101	114	114	ALL: .ASCIZ @ALL@<CRLF>
6	070521	040	077	040	QUES: .ASCIZ @ ? @
7	070525	054	040	000	COMMA: .ASCIZ @, @
8	070530	200	124	117	MSHELP: .ASCII <CRLF>@TO ENSURE THAT NO BAD HEADERS ARE LEFT ON THE DISK@
9	070616	200	120	101	.ASCII <CRLF>@PACK, THIS PROGRAM SHOULD BE HALTED BY TYPING A (^C)@
10	070704	200	103	117	.ASCII <CRLF>@CONTROL C. AS A RESULT, THE PROGRAM WILL BE HALTED@
11	070772	200	127	110	.ASCII <CRLF>@WHEN THE DRIVE UNDER TEST HAS COMPLETED TESTING.@<CRLF>
12	071054	200	124	131	.ASCII <CRLF>@TYPE HELP TEXT (L) N ? @
13	071105	200	122	115	CNSLO1: .ASCIZ <CRLF>@RMCS1=@
14	071115	040	114	111	CNSLO2: .ASCIZ @ LIMITS - LO= 160000, HI= 17XXXX@<CRLF>
15	071157	122	115	126	CNSLO3: .ASCIZ @RMVEC=@
16	071166	040	114	111	CNSLO4: .ASCIZ @ LIMITS - LO= 0, HI= 1000@<CRLF><LF>
17	071222	200	124	131	CNSLO7: .ASCII <CRLF>@TYPE 'A' TO TEST ALL DRIVES, OR TYPE DRIVE NUMBER(S)@
18	071307	200	101	116	.ASCIZ <CRLF>@AND TERMINATE INPUT WITH A CARRIAGE RETURN.@
19	071364	200			CNSLO8: .ASCII <CRLF>
20	071365	040	077	111	CNSLO9: .ASCIZ @ ?ILLEGAL INPUT@<CRLF>
21	071406	200	104	122	MSDRVS: .ASCIZ <CRLF>/DRIVE(S): /
22	071422	104	122	111	MSGDRV: .ASCIZ /DRIVE/
23	071430	200	125	116	SYSTAT: .ASCIZ <CRLF>/UNIT STATUS:/
24	071446	122	115	060	\$RM02: .ASCIZ /RM02/
25	071453	122	115	060	\$RM03: .ASCIZ /RM03/
26	071460	122	115	060	\$RM05: .ASCIZ /RM05/
27	071465	040	116	117	NOTRM: .ASCIZ @ NOT AN RM05/3/2@
28	071506	040	114	117	LODEV: .ASCIZ / LOAD DEVICE/
29	071523	040	116	117	NOTPRS: .ASCIZ / NOT PRESENT/
30	071540	040	116	117	NOTAVL: .ASCIZ / NOT AVAILABLE/
31	071557	040	117	106	UNTOFF: .ASCIZ / OFFLINE/
32	071570	040	117	116	UNTON: .ASCIZ / ONLINE/
33	071600	200	104	122	DRIVES: .ASCIZ <CRLF>/DRIVE(S) TO BE TESTED/
34	071627	116	117	116	NONE: .ASCIZ /NONE/
35	071634	116	000		N: .ASCIZ /N/
36	071636	131	000		Y: .ASCIZ /Y/
37	071640	040			BLNKS4: .ASCII / /
38	071641	040			BLNKS3: .ASCII / /
39	071642	040			BLNKS2: .ASCII / /
40	071643	040	000		BLNKS1: .ASCIZ / /
41					.EVEN

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

071646

071646 020000

.SBTTL FUNCTION CODE TABLE

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

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

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

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

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

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

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

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

; BIT 08 IS NOT USED.

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

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

; BIT 05 IS NOT USED.

; BIT 04 IS NOT USED.

; BIT 03 IS NOT USED.

; BIT 02 IS NOT USED.

; BIT 01 IS NOT USED.

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

FNCDTB:

;FUNCTION CODE TABLE

.WORD OPI

;NOP

58 071650 130001	.WORD OPI:ATA:ILF:IVC	; ILLEGAL FUNCTION (2)
59 071652 132000	.WORD ATA:OPI:IVC:IAE	; SEEK
60 071654 130000	.WORD ATA:OPI:IVC	; RECALIBRATE
61 071656 020000	.WORD OPI	; DRIVE CLEAR
62 071660 030000	.WORD OPI:IVC	; RELEASE
63 071662 130000	.WORD OPI:ATA:IVC	; OFFSET
64 071664 130000	.WORD OPI:ATA:IVC	; RETURN TO CENTERLINE
65 071666 020000	.WORD OPI	; READ IN PRESET
66 071670 020000	.WORD OPI	; PACK ACKNOWLEDGE
67 071672 130001	.WORD OPI:ATA:ILF:IVC	; ILLEGAL FUNCTION (24)
68 071674 130001	.WORD OPI:ATA:ILF:IVC	; ILLEGAL FUNCTION (26)
69 071676 132000	.WORD ATA:OPI:IVC:IAE	; SEARCH
70 071700 130001	.WORD OPI:ATA:ILF:IVC	; ILLEGAL FUNCTION (32)
71 071702 130001	.WORD OPI:ATA:ILF:IVC	; ILLEGAL FUNCTION (34)
72 071704 130001	.WORD OPI:ATA:ILF:IVC	; ILLEGAL FUNCTION (36)
73 071706 130001	.WORD OPI:ATA:ILF:IVC	; ILLEGAL FUNCTION (40)
74 071710 130001	.WORD OPI:ATA:ILF:IVC	; ILLEGAL FUNCTION (42)
75 071712 130001	.WORD OPI:ATA:ILF:IVC	; ILLEGAL FUNCTION (44)
76 071714 130001	.WORD OPI:ATA:ILF:IVC	; ILLEGAL FUNCTION (46)
77 071716 073300	.WORD WCE:OPI:IVC:IAE:AOE:HCE:ECH	; WRITE CHECK DATA
78 071720 073300	.WORD WCE:OPI:IVC:IAE:AOE:HCE:ECH	; WRITE CHECK HEADER AND DATA
79 071722 130001	.WORD OPI:ATA:ILF:IVC	; ILLEGAL FUNCTION (54)
80 071724 130001	.WORD OPI:ATA:ILF:IVC	; ILLEGAL FUNCTION (56)
81 071726 037200	.WORD OPI:IVC:WLE:IAE:AOE:HCE	; WRITE DATA
82 071730 037000	.WORD OPI:IVC:WLE:IAE:AOE	; WRITE HEADER AND DATA
83 071732 130001	.WORD OPI:ATA:ILF:IVC	; ILLEGAL FUNCTION (64)
84 071734 130001	.WORD OPI:ATA:ILF:IVC	; ILLEGAL FUNCTION (66)
85 071736 033300	.WORD OPI:IVC:IAE:AOE:HCE:ECH	; READ DATA
86 071740 033300	.WORD OPI:IVC:IAE:AOE:HCE:ECH	; READ HEADER AND DATA
87 071742 130001	.WORD OPI:ATA:ILF:IVC	; ILLEGAL FUNCTION (74)
88 071744 130001	.WORD OPI:ATA:ILF:IVC	; ILLEGAL FUNCTION (76)

ATTENTION (ATA) TABLE

1		
2		
3	071746	001
4	071747	002
5	071750	004
6	071751	010
7	071752	020
8	071753	040
9	071754	100
10	071755	200

.SBTTL ATTENTION (ATA) TABLE

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

1
 2
 3 071756
 4 071756
 5 071756 000000
 6 071760 000001
 7 071762 000003
 8 071764 000007
 9 071766 000017
 10 071770 000037
 11 071772 000077
 12 071774 000177
 13 071776 000377
 14 072000 000777
 15 072002 001777
 16 072004 003777
 17 072006 007777
 18 072010 017777
 19 072012 037777
 20 072014 077777
 21 072016 177777
 22 072020 177777
 23 072022 077777
 24 072024 037777
 25 072026 017777
 26 072030 007777
 27 072032 003777
 28 072034 001777
 29 072036 000777
 30 072040 000377
 31 072042 000177
 32 072044 000077
 33 072046 000037
 34 072050 000017
 35 072052 000007
 36 072054 000003
 37 072056 000001
 38 072060 000000
 39 072062 000000
 40 072064 000001
 41 072066 000002
 42 072070 000004
 43 072072 000010
 44 072074 000020
 45 072076 000040
 46 072100 000100
 47 072102 000200
 48 072104 000400
 49 072106 001000
 50 072110 002000
 51 072112 004000
 52 072114 010000
 53 072116 020000
 54 072120 040000
 55 072122 100000
 56 072124 100000
 57 072126 040000

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

58	072130	020000	.WORD	8192.
59	072132	010000	.WORD	4096.
60	072134	004000	.WORD	2048.
61	072136	002000	.WORD	1024.
62	072140	001000	.WORD	512.
63	072142	000400	.WORD	256.
64	072144	000200	.WORD	128.
65	072146	000100	.WORD	64.
66	072150	000040	.WORD	32.
67	072152	000020	.WORD	16.
68	072154	000010	.WORD	8.
69	072156	000004	.WORD	4.
70	072160	000002	.WORD	2.
71	072162	000001	.WORD	1.
72	072164	000000	.WORD	0.
73	072166	177777	.WORD	65535.
74	072170	177776	.WORD	65534.
75	072172	177774	.WORD	65532.
76	072174	177770	.WORD	65528.
77	072176	177760	.WORD	65520.
78	072200	177740	.WORD	65504.
79	072202	177700	.WORD	65472.
80	072204	177600	.WORD	65408.
81	072206	177400	.WORD	65280.
82	072210	177000	.WORD	65024.
83	072212	176000	.WORD	64512.
84	072214	174000	.WORD	63488.
85	072216	170000	.WORD	61440.
86	072220	160000	.WORD	57344.
87	072222	140000	.WORD	49152.
88	072224	100000	.WORD	32768.
89	072226	000000	.WORD	0.
90	072230	000000	.WORD	0.
91	072232	100000	.WORD	32768.
92	072234	140000	.WORD	49152.
93	072236	160000	.WORD	57344.
94	072240	170000	.WORD	61440.
95	072242	174000	.WORD	63488.
96	072244	176000	.WORD	64512.
97	072246	177000	.WORD	65024.
98	072250	177400	.WORD	65280.
99	072252	177600	.WORD	65408.
100	072254	177700	.WORD	65472.
101	072256	177740	.WORD	65504.
102	072260	177760	.WORD	65520.
103	072262	177770	.WORD	65528.
104	072264	177774	.WORD	65532.
105	072266	177776	.WORD	65534.
106	072270	177777	.WORD	65535.
107	072272	125252	.WORD	43690.
108	072274	152525	.WORD	43690./2
109	072276	125252	.WORD	43690.
110	072300	177777	.WORD	65535.
111	072302	177776	.WORD	65534.
112	072304	177775	.WORD	65533.
113	072306	177773	.WORD	65531.
114	072310	177767	.WORD	65527.

EARLY:

115	072312	177757	.WORD	65519.
116	072314	177737	.WORD	65503.
117	072316	177677	.WORD	65471.
118	072320	177577	.WORD	65407.
119	072322	177377	.WORD	65279.
120	072324	176777	.WORD	65023.
121	072326	175777	.WORD	64511.
122	072330	173777	.WORD	63487.
123	072332	167777	.WORD	61439.
124	072334	157777	.WORD	57343.
125	072336	137777	.WORD	49151.
126	072340	077777	.WORD	32767.
127	072342	077777	.WORD	32767.
128	072344	137777	.WORD	49151.
129	072346	157777	.WORD	57343.
130	072350	167777	.WORD	61439.
131	072352	173777	.WORD	63487.
132	072354	175777	.WORD	64511.
133	072356	176777	.WORD	65023.
134	072360	177377	.WORD	65279.
135	072362	177577	.WORD	65407.
136	072364	177677	.WORD	65471.
137	072366	177737	.WORD	65503.
138	072370	177757	.WORD	65519.
139	072372	177767	.WORD	65527.
140	072374	177773	.WORD	65531.
141	072376	177775	.WORD	65533.
142	072400	177776	.WORD	65534.
143	072402	177777	.WORD	65535.
144	072404			

ENRGDT:

1					.SBTTL	ERROR MESSAGE TABLE	
2							
3	072404	076772	000000		EMT1:	.WORD	EMS1,0
4	072410	077041	077056	000000	EMT2:	.WORD	EMS2,EMS3,0
5	072416	077041	077121	000000	EMT3:	.WORD	EMS2,EMS4,0
6	072424	077164	077214	000000	EMT4:	.WORD	EMS5,EMS6,0
7	072432	077164	077326	000000	EMT5:	.WORD	EMS5,EMS10,0
8	072440	103766	101205	000000	EMT6:	.WORD	EMS167,EMS64,0
9	072446	101753	104013	000000	EMT7:	.WORD	EMS110,EMS170,0
10	072454	077261	000000		EMT10:	.WORD	EMS7,0
11	072460	077326	000000		EMT11:	.WORD	EMS10,0
12	072464	077370	077401	000000	EMT12:	.WORD	EMS11,EMS12,0
13	072472	077442	077453	077464	EMT13:	.WORD	EMS13,EMS14,EMS15,EMS16,0
14	072504	077536	101205	000000	EMT14:	.WORD	EMS17,EMS64,0
15	072512	077370	077621	000000	EMT15:	.WORD	EMS11,EMS21,0
16	072520	077370	077644	077773	EMT16:	.WORD	EMS11,EMS22,EMS27,0
17	072530	077370	077660	100004	EMT17:	.WORD	EMS11,EMS23,EMS30,0
18	072540	077370	077706	100004	EMT20:	.WORD	EMS11,EMS24,EMS30,0
19	072550	077370	077735	077773	EMT21:	.WORD	EMS11,EMS25,EMS27,0
20	072560	077370	077752	077773	EMT22:	.WORD	EMS11,EMS26,EMS27,0
21	072570	077370	100014	100004	EMT23:	.WORD	EMS11,EMS31,EMS30,0
22	072600	077370	100043	100004	EMT24:	.WORD	EMS11,EMS32,EMS30,0
23	072610	077370	100072	100004	EMT25:	.WORD	EMS11,EMS33,EMS30,0
24	072620	077370	100120	100004	EMT26:	.WORD	EMS11,EMS34,EMS30,0
25	072630	077370	100171	100004	EMT27:	.WORD	EMS11,EMS35,EMS30,0
26	072640	077370	100220	100004	EMT30:	.WORD	EMS11,EMS36,EMS30,0
27	072650	077370	100247	100004	EMT31:	.WORD	EMS11,EMS37,EMS30,0
28	072660	077370	100275	100004	EMT32:	.WORD	EMS11,EMS40,EMS30,0
29	072670	077370	100324	100004	EMT33:	.WORD	EMS11,EMS41,EMS30,0
30	072700	077370	100352	100004	EMT34:	.WORD	EMS11,EMS42,EMS30,0
31	072710	077370	100401	100004	EMT35:	.WORD	EMS11,EMS43,EMS30,0
32	072720	077370	100430	100004	EMT36:	.WORD	EMS11,EMS44,EMS30,0
33	072730	077370	100503	100004	EMT37:	.WORD	EMS11,EMS45,EMS30,0
34	072740	101271	077576	000000	EMT40:	.WORD	EMS66,EMS20,0
35	072746	101457	103170	101465	EMT41:	.WORD	EMS75,EMS141,EMS76,0
36	072756	103261	103271	101413	EMT42:	.WORD	EMS144,EMS145,EMS72,EMS76,0
37	072770	100575	100711	101465	EMT43:	.WORD	EMS47,EMS53,EMS76,0
38	073000	101516	100711	101465	EMT44:	.WORD	EMS77,EMS53,EMS76,0
39	073010	101544	100711	101465	EMT45:	.WORD	EMS100,EMS53,EMS76,0
40	073020	101572	100711	101465	EMT46:	.WORD	EMS101,EMS53,EMS76,0
41	073030	101223	101205	000000	EMT47:	.WORD	EMS65,EMS64,0
42	073036	077442	077464	101157	EMT50:	.WORD	EMS13,EMS15,EMS63,0
43	073046	077370	100546	100004	EMT51:	.WORD	EMS11,EMS46,EMS30,EMS67,0
44	073060	100575	100711	101341	EMT52:	.WORD	EMS47,EMS53,EMS67,EMS115,EMS140,EMS141,0
45	073076	100575	100711	101341	EMT53:	.WORD	EMS47,EMS53,EMS67,EMS115,EMS141,EMS164,0
46	073114	100624	100711	101341	EMT54:	.WORD	EMS50,EMS53,EMS67,0
47	073124	103216	103233	100711	EMT55:	.WORD	EMS142,EMS143,EMS53,EMS67,0
48	073136	100653	101413	101341	EMT56:	.WORD	EMS51,EMS72,EMS67,EMS115,EMS50,EMS70,0
49	073154	103766	101423	101341	EMT57:	.WORD	EMS167,EMS73,EMS67,0
50	073164	100774	101341	102153	EMT60:	.WORD	EMS56,EMS67,EMS115,EMS150,EMS152,EMS70,0
51	073202	101400	100774	101341	EMT61:	.WORD	EMS71,EMS56,EMS67,EMS115,EMS150,EMS152,EMS72,0
52	073222	103147	101341	102153	EMT62:	.WORD	EMS140,EMS67,EMS115,EMS47,EMS70,0
53	073236	000000			EMT63:	.WORD	
54	073240	103354	103416	101366	EMT64:	.WORD	EMS150,EMS152,EMS70,EMS115,EMS56,EMS73,EMS67,0
55	073260	100701	104413	104574	EMT65:	.WORD	EMS52,EMS205,EMS214,EMS206,EMS115,EMS51,EMS72,0
56	073300	103725	101647	101413	EMT66:	.WORD	EMS165,EMS103,EMS72,EMS124,0
57	073312	103725	101647	101413	EMT67:	.WORD	EMS165,EMS103,EMS72,EMS171,0

58	073324	100546	077576	103620	EMT70:	.WORD	EMS46,EMS20,EMS163,0
59	073334	101400	101572	103620	EMT71:	.WORD	EMS71,EMS101,EMS163,0
60	073344	100575	101413	103620	EMT72:	.WORD	EMS47,EMS72,EMS163,EMS115,EMS140,EMS141,0
61	073362	100575	101413	103620	EMT73:	.WORD	EMS47,EMS72,EMS163,EMS115,EMS141,EMS72,0
62	073400	100774	100711	103620	EMT74:	.WORD	EMS56,EMS53,EMS163,0
63	073410	101400	100774	103620	EMT75:	.WORD	EMS71,EMS56,EMS163,EMS115,EMS150,EMS152,EMS72,0
64	073430	100624	100711	103620	EMT76:	.WORD	EMS50,EMS53,EMS163,0
65	073440	103216	103233	100711	EMT77:	.WORD	EMS142,EMS143,EMS53,EMS163,0
66	073452	101457	103170	103620	EMT100:	.WORD	EMS75,EMS141,EMS163,EMS115,EMS47,EMS70,0
67	073470	101457	103354	103620	EMT101:	.WORD	EMS75,EMS150,EMS163,EMS115,EMS56,EMS73,0
68	073506	103766	101423	103620	EMT102:	.WORD	EMS167,EMS73,EMS163,0
69	073516	103337	103372	101440	EMT103:	.WORD	EMS147,EMS151,EMS74,EMS163,0
70	073530	100653	101440	103620	EMT104:	.WORD	EMS51,EMS74,EMS163,EMS115,EMS50,EMS70,0
71	073546	103725	101544	103620	EMT105:	.WORD	EMS165,EMS100,EMS163,0
72	073556	103725	101516	103620	EMT106:	.WORD	EMS165,EMS77,EMS163,0
73	073566	103261	103271	100711	EMT107:	.WORD	EMS144,EMS145,EMS53,EMS163,EMS115,EMS143,EMS70,0
74	073606	101753	102013	102156	EMT110:	.WORD	EMS110,EMS112,EMS116,EMS111,0
75	073620	102077	077121	000000	EMT111:	.WORD	EMS113,EMS4,0
76	073626	102077	077056	000000	EMT112:	.WORD	EMS113,EMS3,0
77	073634	101753	102153	102156	EMT113:	.WORD	EMS110,EMS115,EMS116,EMS117,EMS114,0
78	073650	102077	102230	000000	EMT114:	.WORD	EMS113,EMS120,0
79	073656	102251	102711	102735	EMT115:	.WORD	EMS121,EMS132,EMS133,0
80	073666	102314	102711	102735	EMT116:	.WORD	EMS122,EMS132,EMS133,0
81	073676	102351	102711	102735	EMT117:	.WORD	EMS123,EMS132,EMS133,0
82	073706	102414	102711	102735	EMT120:	.WORD	EMS124,EMS132,EMS133,0
83	073716	102446	102711	102735	EMT121:	.WORD	EMS125,EMS132,EMS133,0
84	073726	102511	102711	102735	EMT122:	.WORD	EMS126,EMS132,EMS133,0
85	073736	103112	102711	102735	EMT123:	.WORD	EMS137,EMS132,EMS133,0
86	073746	102613	102711	102735	EMT124:	.WORD	EMS130,EMS132,EMS133,0
87	073756	102651	102711	102735	EMT125:	.WORD	EMS131,EMS132,EMS133,0
88	073766	102251	102711	102760	EMT126:	.WORD	EMS121,EMS132,EMS134,EMS123,0
89	074000	102314	102711	102760	EMT127:	.WORD	EMS122,EMS132,EMS134,EMS123,0
90	074012	102351	102711	102760	EMT130:	.WORD	EMS123,EMS132,EMS134,EMS123,0
91	074024	102414	102711	102760	EMT131:	.WORD	EMS124,EMS132,EMS134,EMS123,0
92	074036	102446	102711	102760	EMT132:	.WORD	EMS125,EMS132,EMS134,EMS123,0
93	074050	102511	102711	102760	EMT133:	.WORD	EMS126,EMS132,EMS134,EMS123,0
94	074062	103112	102711	102760	EMT134:	.WORD	EMS137,EMS132,EMS134,EMS123,0
95	074074	102613	102711	102760	EMT135:	.WORD	EMS130,EMS132,EMS134,EMS123,0
96	074106	102651	102711	102760	EMT136:	.WORD	EMS131,EMS132,EMS134,EMS123,0
97	074120	102251	102711	103022	EMT137:	.WORD	EMS121,EMS132,EMS135,0
98	074130	102351	102711	103022	EMT140:	.WORD	EMS123,EMS132,EMS135,0
99	074140	102251	102711	103065	EMT141:	.WORD	EMS121,EMS132,EMS136,0
100	074150	103112	102711	103065	EMT142:	.WORD	EMS137,EMS132,EMS136,0
101	074160	102414	102711	103065	EMT143:	.WORD	EMS124,EMS132,EMS136,0
102	074170	102446	102711	103065	EMT144:	.WORD	EMS125,EMS132,EMS136,0
103	074200	102511	102711	103065	EMT145:	.WORD	EMS126,EMS132,EMS136,0
104	074210	102651	102711	103065	EMT146:	.WORD	EMS131,EMS132,EMS136,0
105	074220	104042	102711	103065	EMT147:	.WORD	EMS171,EMS132,EMS136,0
106	074230	102613	102711	103065	EMT150:	.WORD	EMS130,EMS132,EMS136,0
107	074240	103147	102153	103170	EMT151:	.WORD	EMS140,EMS115,EMS141,EMS70,0
108	074252	103216	102153	103233	EMT152:	.WORD	EMS142,EMS115,EMS143,EMS72,0
109	074264	103261	103271	103320	EMT153:	.WORD	EMS144,EMS145,EMS146,EMS115,EMS143,EMS72,0
110	074302	103261	103271	077576	EMT154:	.WORD	EMS144,EMS145,EMS20,EMS115,EMS143,EMS70,0
111	074320	103354	103416	103464	EMT155:	.WORD	EMS150,EMS152,EMS154,EMS153,0
112	074332	103337	103372	103464	EMT156:	.WORD	EMS147,EMS151,EMS154,EMS155,0
113	074344	103337	103372	103520	EMT157:	.WORD	EMS147,EMS151,EMS156,EMS157,0
114	074356	103570	103520	103553	EMT160:	.WORD	EMS161,EMS156,EMS160,0

115	074366	077735	077773	103520	EMT161: .WORD	EMS25,EMS27,EMS156,EMS160,0
116	074400	077752	077773	103520	EMT162: .WORD	EMS26,EMS27,EMS156,EMS160,0
117	074412	100575	103620	103147	EMT163: .WORD	EMS47,EMS163,EMS140,0
118	074422	100575	077576	000000	EMT164: .WORD	EMS47,EMS20,0
119	074430	100774	077576	000000	EMT165: .WORD	EMS56,EMS20,0
120	074436	100546	077576	000000	EMT166: .WORD	EMS46,EMS20,0
121	074444	105114	077576	000000	EMT167: .WORD	EMS224,EMS20,0
122	074452	104352	103170	100004	EMT170: .WORD	EMS203,EMS141,EMS30,EMS202,0
123	074464	103766	103464	103536	EMT171: .WORD	EMS167,EMS154,EMS157,0
124	074474	103766	103464	103500	EMT172: .WORD	EMS167,EMS154,EMS155,0
125	074504	104042	102711	102735	EMT173: .WORD	EMS171,EMS132,EMS133,0
126	074514	104042	102711	102760	EMT174: .WORD	EMS171,EMS132,EMS134,EMS123,0
127	074526	102077	104215	000000	EMT175: .WORD	EMS113,EMS177,0
128	074534	104237	104254	000000	EMT176: .WORD	EMS200,EMS201,0
129	074542	000000			EMT177: .WORD	
130	074544	104352	100653	100004	EMT200: .WORD	EMS203,EMS51,EMS30,EMS202,0
131	074556	104352	104365	100004	EMT201: .WORD	EMS203,EMS204,EMS30,EMS202,0
132	074570	104352	100624	100004	EMT202: .WORD	EMS203,EMS50,EMS30,EMS202,0
133	074602	104352	103233	100004	EMT203: .WORD	EMS203,EMS143,EMS30,EMS202,0
134	074614	103354	101440	104322	EMT204: .WORD	EMS150,EMS74,EMS202,EMS115,EMS152,EMS72,0
135	074632	100624	101647	101413	EMT205: .WORD	EMS50,EMS103,EMS72,0
136	074642	101656	101423	104322	EMT206: .WORD	EMS104,EMS73,EMS202,0
137	074652	101457	103170	102153	EMT207: .WORD	EMS75,EMS141,EMS115,EMS140,0
138	074664	101457	103354	000000	EMT210: .WORD	EMS75,EMS150,0
139	074672	100653	101413	102153	EMT211: .WORD	EMS51,EMS72,EMS115,EMS50,EMS70,0
140	074706	103216	100711	102153	EMT212: .WORD	EMS142,EMS53,EMS115,EMS143,EMS72,0
141	074722	100624	101647	100711	EMT213: .WORD	EMS50,EMS103,EMS53,0
142	074732	100701	104413	103741	EMT214: .WORD	EMS52,EMS205,EMS166,EMS206,EMS115,EMS51,EMS72,0
143	074752	100701	102206	100774	EMT215: .WORD	EMS52,EMS117,EMS56,EMS163,0
144	074764	100774	100004	101205	EMT216: .WORD	EMS56,EMS30,EMS64,0
145	074774	100546	100004	101205	EMT217: .WORD	EMS46,EMS30,EMS64,0
146	075004	100014	100004	101205	EMT220: .WORD	EMS31,EMS30,EMS64,0
147	075014	100575	100004	101205	EMT221: .WORD	EMS47,EMS30,EMS64,0
148	075024	100701	102206	101516	EMT222: .WORD	EMS52,EMS117,EMS77,0
149	075034	100701	102206	101157	EMT223: .WORD	EMS52,EMS117,EMS63,0
150	075044	000000			EMT224: .WORD	
151	075046	000000			EMT225: .WORD	
152	075050	000000			EMT226: .WORD	
153	075052	000000			EMT227: .WORD	
154	075054	000000			EMT230: .WORD	
155	075056	000000			EMT231: .WORD	
156	075060	000000			EMT232: .WORD	
157	075062	000000			EMT233: .WORD	
158	075064	000000			EMT234: .WORD	
159	075066	000000			EMT235: .WORD	
160	075070	000000			EMT236: .WORD	
161	075072	000000			EMT237: .WORD	
162	075074	000000			EMT240: .WORD	
163	075076	000000			EMT241: .WORD	
164	075100	000000			EMT242: .WORD	
165	075102	000000			EMT243: .WORD	
166	075104	000000			EMT244: .WORD	
167	075106	000000			EMT245: .WORD	
168	075110	103766	102711	104452	EMT246: .WORD	EMS167,EMS132,EMS207,0
169	075120	103766	102711	104477	EMT247: .WORD	EMS167,EMS132,EMS210,EMS125,0
170	075132	103766	104510	104477	EMT250: .WORD	EMS167,EMS211,EMS210,EMS207,EMS206,0
171	075146	104526	104551	000000	EMT251: .WORD	EMS212,EMS213,0

172	075154	103725	104526	000000	EMT252: .WORD	EMS165,EMS212,0
173	075162	103337	103372	103520	EMT253: .WORD	EMS147,EMS151,EMS156,EMS210,EMS26,EMS27,0
174	075200	104352	101572	100004	EMT254: .WORD	EMS203,EMS101,EMS30,0
175	075210	104352	103766	100004	EMT255: .WORD	EMS203,EMS167,EMS30,0
176	075220	104352	101544	100004	EMT256: .WORD	EMS203,EMS100,EMS30,0
177	075230	077370	100546	100004	EMT257: .WORD	EMS11,EMS46,EMS30,EMS102,0
178	075242	100701	102206	100774	EMT260: .WORD	EMS52,EMS117,EMS56,EMS102,0
179	075254	100701	104413	104716	EMT261: .WORD	EMS52,EMS205,EMS220,EMS206,EMS115,EMS51,EMS72,0
180	075274	101656	101423	104322	EMT262: .WORD	EMS104,EMS73,EMS202,0
181	075304	100624	100711	101620	EMT263: .WORD	EMS50,EMS53,EMS102,0
182	075314	100774	100711	101620	EMT264: .WORD	EMS56,EMS53,EMS102,EMS115,EMS150,EMS152,EMS70,0
183	075334	101400	100774	101620	EMT265: .WORD	EMS71,EMS56,EMS102,EMS115,EMS150,EMS152,EMS72,0
184	075354	103216	103233	100711	EMT266: .WORD	EMS142,EMS143,EMS53,EMS102,0
185	075366	100624	103320	102153	EMT267: .WORD	EMS50,EMS146,EMS115,EMS52,EMS117,EMS46,0
186	075404	100575	100711	101620	EMT270: .WORD	EMS47,EMS53,EMS102,EMS115,EMS140,0
187	075420	100575	100711	101620	EMT271: .WORD	EMS47,EMS53,EMS102,EMS115,EMS141,EMS72,0
188	075436	101457	103170	101620	EMT272: .WORD	EMS75,EMS141,EMS102,EMS115,EMS47,EMS73,0
189	075454	100653	101440	101620	EMT273: .WORD	EMS51,EMS74,EMS102,EMS115,EMS50,EMS70,0
190	075472	101157	100711	101051	EMT274: .WORD	EMS63,EMS53,EMS57,EMS115,EMS41,EMS146,0
191	075510	102153	100711	100324	EMT275: .WORD	EMS116,EMS53,EMS41,EMS57,0
192	075522	100575	100711	101051	EMT276: .WORD	EMS47,EMS53,EMS57,EMS115,EMS140,0
193	075536	100575	100711	101051	EMT277: .WORD	EMS47,EMS53,EMS57,EMS115,EMS141,EMS72,0
194	075554	100774	100711	101051	EMT300: .WORD	EMS56,EMS53,EMS57,EMS115,EMS150,EMS152,EMS70,0
195	075574	101400	100774	100711	EMT301: .WORD	EMS71,EMS56,EMS53,EMS57,EMS115,EMS150,EMS152,EMS72,0
196	075616	103725	101544	101647	EMT302: .WORD	EMS165,EMS100,EMS103,EMS57,0
197	075630	103725	101572	101647	EMT303: .WORD	EMS165,EMS101,EMS103,EMS57,0
198	075642	103725	101516	101647	EMT304: .WORD	EMS165,EMS77,EMS103,EMS57,0
199	075654	100546	100004	101205	EMT305: .WORD	EMS46,EMS30,EMS64,EMS57,0
200	075666	100624	100711	101051	EMT306: .WORD	EMS50,EMS53,EMS57,0
201	075676	100624	103320	102153	EMT307: .WORD	EMS50,EMS146,EMS115,EMS52,EMS117,EMS46,EMS57,0
202	075716	103216	103233	100711	EMT310: .WORD	EMS142,EMS143,EMS53,EMS57,0
203	075730	101656	101647	100711	EMT311: .WORD	EMS104,EMS103,EMS53,EMS57,0
204	075742	101705	101647	100711	EMT312: .WORD	EMS105,EMS103,EMS53,EMS57,0
205	075754	103261	103271	101647	EMT313: .WORD	EMS144,EMS145,EMS103,EMS57,EMS115,EMS143,EMS70,0
206	075774	100352	101647	100711	EMT314: .WORD	EMS42,EMS103,EMS53,EMS57,0
207	076006	100014	101647	100711	EMT315: .WORD	EMS31,EMS103,EMS53,EMS57,0
208	076020	101400	100014	101647	EMT316: .WORD	EMS71,EMS31,EMS103,EMS57,0
209	076032	100401	101647	101051	EMT317: .WORD	EMS43,EMS103,EMS57,0
210	076042	100503	101647	101051	EMT320: .WORD	EMS45,EMS103,EMS57,0
211	076052	100430	101647	101051	EMT321: .WORD	EMS44,EMS103,EMS57,0
212	076062	101734	077576	000000	EMT322: .WORD	EMS106,EMS20,0
213	076070	100220	101647	101051	EMT323: .WORD	EMS36,EMS103,EMS57,0
214	076100	104115	100220	101647	EMT324: .WORD	EMS173,EMS36,EMS103,EMS57,0
215	076112	104075	100220	101647	EMT325: .WORD	EMS172,EMS36,EMS103,EMS57,0
216	076124	077442	104132	077464	EMT326: .WORD	EMS13,EMS174,EMS15,EMS35,EMS53,EMS175,0
217	076142	103337	103372	101440	EMT327: .WORD	EMS147,EMS151,EMS74,EMS175,0
218	076154	101271	100711	104140	EMT330: .WORD	EMS66,EMS53,EMS175,0
219	076164	100072	101647	100711	EMT331: .WORD	EMS33,EMS103,EMS53,EMS175,0
220	076176	100275	101647	100711	EMT332: .WORD	EMS40,EMS103,EMS53,EMS57,0
221	076210	100653	101440	101051	EMT333: .WORD	EMS51,EMS74,EMS57,EMS115,EMS50,EMS70,0
222	076226	101457	103170	101051	EMT334: .WORD	EMS75,EMS141,EMS57,EMS115,EMS47,EMS73,0
223	076244	101457	103354	103416	EMT335: .WORD	EMS75,EMS150,EMS152,EMS57,EMS115,EMS56,EMS73,0
224	076264	101077	101112	101141	EMT336: .WORD	EMS60,EMS61,EMS62,0
225	076274	102156	102206	100120	EMT337: .WORD	EMS116,EMS117,EMS34,0
226	076304	100120	100711	100723	EMT340: .WORD	EMS34,EMS53,EMS54,EMS111,0
227	076316	100745	100120	000000	EMT341: .WORD	EMS55,EMS34,0
228	076324	100701	102206	100774	EMT342: .WORD	EMS52,EMS117,EMS56,EMS57,0

229	076336	100701	102206	100503	EMT343: .WORD	EMS52,EMS117,EMS45,EMS57,0
230	076350	100701	102206	100430	EMT344: .WORD	EMS52,EMS117,EMS44,EMS57,0
231	076362	100701	102206	104736	EMT345: .WORD	EMS52,EMS117,EMS221,0
232	076372	101734	077576	102153	EMT346: .WORD	EMS106,EMS20,EMS115,EMS223,EMS72,0
233	076406	100701	104413	105006	EMT347: .WORD	EMS52,EMS205,EMS222,EMS206,0
234	076420	101457	103354	101620	EMT350: .WORD	EMS75,EMS150,EMS102,EMS115,EMS56,EMS73,0
235	076436	103766	101423	101620	EMT351: .WORD	EMS167,EMS73,EMS102,0
236	076446	104612	000000		EMT352: .WORD	EMS215,0
237	076452	104663	104352	104633	EMT353: .WORD	EMS217,EMS203,EMS216,0
238	076462	104736	077576	000000	EMT354: .WORD	EMS221,EMS20,0

1	076470	105166	105772	106047	EHT1:	.WORD	EH1,STSH1,STSH2,STSH4,0
2	076502	105772	106047	106174	EHT2:	.WORD	STSH1,STSH2,STSH4,0
3							
4	076512	105205	000000		EHT110:	.WORD	EH110,0
5	076516	105214	000000		EHT111:	.WORD	EH111,0
6							
7	076522	105233	000000		EHT114:	.WORD	EH114,0
8	076526	105262	105772	106047	EHT223:	.WORD	EH223,STSH1,STSH2,STSH4,0
9	076540	105310	105772	106047	EHT256:	.WORD	EH256,STSH1,STSH2,STSH4,0
10							
11	076552	105364	105772	106047	EHT336:	.WORD	EH336,STSH1,STSH2,STSH4,0
12	076564	105423	105772	106047	EHT337:	.WORD	EH337,STSH1,STSH2,STSH4,0
13	076576	105560	105772	106047	EHT344:	.WORD	EH344,STSH1,STSH2,STSH4,0
14							
15	076610	105716	000000		EHT353:	.WORD	EH353,0

1	076614	106232	106326	106344	EDT1:	.WORD	ED1,STSD1,STSD2,STSD4
2	076624	106326	106344	106376	EDT2:	.WORD	STSD1,STSD2,STSD4
3							
4	076632	106240			EDT110:	.WORD	ED110
5	076634	106244			EDT111:	.WORD	ED111
6							
7	076636	106252			EDT114:	.WORD	ED114
8	076640	106262	106326	106344	EDT223:	.WORD	ED223,STSD1,STSD2,STSD4
9							
10	076650	106272	106326	106344	EDT336:	.WORD	ED336,STSD1,STSD2,STSD4
11	076660	106304	106326	106344	EDT337:	.WORD	ED337,STSD1,STSD2,STSD4
12	076670	106304	106326	106344	EDT344:	.WORD	ED337,STSD1,STSD2,STSD4,0
13							
14	076702	106316			EDT353:	.WORD	ED353

1	076704	106411	106427	106427	EFT11: .WORD	EF111,STSF,STSF,STSF
2	076714	106427	106427	106427	EFT2: .WORD	STSF,STSF,STSF
3						
4	076722	106410			EFT110: .WORD	EF110
5	076724	106411			EFT111: .WORD	EF111
6						
7	076726	106413			EFT114: .WORD	EF114
8	076730	106413	106427	106427	EFT223: .WORD	EF114,STSF,STSF,STSF
9						
10	076740	106416	106427	106427	EFT336: .WORD	EF336,STSF,STSF,STSF
11	076750	106416	106427	106427	EFT337: .WORD	EF336,STSF,STSF,STSF
12	076760	106416	106427	106427	EFT344: .WORD	EF336,STSF,STSF,STSF
13						
14	076770	106413			EFT353: .WORD	EF114

				.SBTTL	ERROR MESSAGE STRINGS
1					
2					
3	076772	127	122	117	EMS1: .ASCIZ @WRONG UNIT SELECTED (RMCS2, BITS 0-2) @
4	077041	104	105	126	EMS2: .ASCIZ @DEVICE WENT @
5	077056	125	116	101	EMS3: .ASCIZ @UNAVAILABLE 'DVA' (RMCS1, BIT 11) @
6	077121	116	117	116	EMS4: .ASCIZ @NONEXISTENT 'NED' (RMCS2, BIT 12) @
7	077164	103	117	115	EMS5: .ASCIZ @COMMAND NOT COMPLETED, @
8	077214	103	117	116	EMS6: .ASCIZ @CONTROLLER NOT READY (RMCS1, BIT 7) @
9	077261	104	122	111	EMS7: .ASCIZ @DRIVE NOT READY 'DRY' (RMDS, BIT 7) @
10	077326	107	117	040	EMS10: .ASCIZ @GO NOT RESET 'GO' (RMCS1, BIT 0) @
11	077370	111	116	126	EMS11: .ASCIZ @INVALID @
12	077401	106	125	116	EMS12: .ASCIZ @FUNCTION CODE (RMCS1, BITS 1-5) @
13	077442	115	101	123	EMS13: .ASCIZ @MASSBUS @
14	077453	103	117	116	EMS14: .ASCIZ @CONTROL @
15	077464	102	125	123	EMS15: .ASCIZ @BUS PARITY ERROR @
16	077506	042	115	103	EMS16: .ASCIZ @'MCPE' (RMCS1, BIT 13) @
17	077536	124	122	101	EMS17: .ASCIZ @TRANSFER ERROR (RMCS1, BIT 14) @
18	077576	123	110	117	EMS20: .ASCIZ @SHOULD NOT BE SET @
19	077621	127	117	122	EMS21: .ASCIZ @WORD COUNT (RMWC) @
20	077644	102	125	123	EMS22: .ASCIZ @BUS (RMSA) @
21	077660	042	114	102	EMS23: .ASCIZ @'LBT' (RMDS, BIT 10) @
22	077706	042	101	117	EMS24: .ASCIZ @'AOE' (RMER1, BIT 09) @
23	077735	104	111	123	EMS25: .ASCIZ @DISK (RMDA) @
24	077752	103	131	114	EMS26: .ASCIZ @CYLINDER (RMDC) @
25	077773	101	104	104	EMS27: .ASCIZ @ADDRESS @
26	100004	123	124	101	EMS30: .ASCIZ @STATUS @
27	100014	042	127	114	EMS31: .ASCIZ @'WLE' (RMER1, BIT 11) @
28	100043	042	125	120	EMS32: .ASCIZ @'UPE' (RMCS2, BIT 13) @
29	100072	042	127	103	EMS33: .ASCIZ @'WCF' (RMER1, BIT 5) @
30	100120	127	122	111	EMS34: .ASCIZ @WRITE CHECK ERROR-'WCE' (RMCS2, BIT 14) @
31	100171	042	115	104	EMS35: .ASCIZ @'MDPE' (RMCS2, BIT 8) @
32	100220	042	104	103	EMS36: .ASCIZ @'DCK' (RMER1, BIT 15) @
33	100247	042	105	103	EMS37: .ASCIZ @'ECH' (RMER1, BIT 6) @
34	100275	042	104	114	EMS40: .ASCIZ @'DLT' (RMCS2, BIT 15) @
35	100324	042	115	130	EMS41: .ASCIZ @'MXF' (RMCS2, BIT 9) @
36	100352	042	104	124	EMS42: .ASCIZ @'DTE' (RMER1, BIT 12) @
37	100401	042	110	103	EMS43: .ASCIZ @'HCR' (RMER1, BIT 8) @
38	100430	110	105	101	EMS44: .ASCIZ @HEADER COMPARE ERROR 'HCE' (RMER1, BIT 7) @
39	100503	106	117	122	EMS45: .ASCIZ @FORMAT ERROR 'FER' (RMER1, BIT 4) @
40	100546	042	111	101	EMS46: .ASCIZ @'IAE' (RMER1, BIT 10) @
41	100575	042	117	120	EMS47: .ASCIZ @'DPI' (RMER1, BIT 13) @
42	100624	042	123	113	EMS50: .ASCIZ @'SKI' (RMER2, BIT 14) @
43	100653	042	120	111	EMS51: .ASCIZ @'PIP' (RMDS, BIT 13) @
44	100701	124	110	105	EMS52: .ASCIZ @THE RM @
45	100711	104	105	124	EMS53: .ASCIZ @DETECTED @
46	100723	101	124	040	EMS54: .ASCIZ @AT AN UNEXPECTED @
47	100745	111	116	103	EMS55: .ASCIZ @INCORRECT DATA DURING @
48	100774	111	116	126	EMS56: .ASCIZ @INVALID COMMAND ERROR 'IVC' (RMER2, BIT 12) @
49	101051	104	125	122	EMS57: .ASCIZ @DURING DATA TRANSFER @
50	101077	104	101	124	EMS60: .ASCIZ @DATA READ @
51	101112	104	117	105	EMS61: .ASCIZ @DOES NOT COMPARE WITH @
52	101141	104	101	124	EMS62: .ASCIZ @DATA WRITTEN @
53	101157	042	120	101	EMS63: .ASCIZ @'PAR' (RMER1, BIT 3) @
54	101205	111	123	040	EMS64: .ASCIZ @IS INCORRECT @
55	101223	103	117	115	EMS65: .ASCIZ @COMPOSITE ERROR 'ERR' (RMDS, BIT 14) @
56	101271	104	101	124	EMS66: .ASCIZ @DATA PARITY ERROR 'DPE' (RMER2, BIT 3) @
57	101341	104	125	122	EMS67: .ASCIZ @DURING SEEK COMMAND @

58	101366	111	123	.040	EMS70:	.ASCIZ	@IS RESET @
59	101400	105	122	122	EMS71:	.ASCIZ	@ERRONEOUS @
60	101413	111	123	040	EMS72:	.ASCIZ	@IS SET @
61	101423	104	111	104	EMS73:	.ASCIZ	@DID NOT SET @
62	101440	104	111	104	EMS74:	.ASCIZ	@DID NOT RESET @
63	101457	114	117	123	EMS75:	.ASCIZ	@LOST @
64	101465	104	125	122	EMS76:	.ASCIZ	@DURING PACK ACK COMMAND @
65	101516	042	122	115	EMS77:	.ASCIZ	@'RMR' (RMER1, BIT 2) @
66	101544	042	111	114	EMS100:	.ASCIZ	@'ILR' (RMER1, BIT 1) @
67	101572	042	111	114	EMS101:	.ASCIZ	@'ILF' (RMER1, BIT 0) @
68	101620	104	125	122	EMS102:	.ASCIZ	@DURING SEARCH COMMAND @
69	101647	105	122	122	EMS103:	.ASCIZ	@ERROR @
70	101656	042	114	102	EMS104:	.ASCIZ	@'LBC' (RMER2, BIT 10) @
71	101705	042	114	123	EMS105:	.ASCIZ	@'LSC' (RMER2, BIT 11) @
72	101734	110	105	101	EMS106:	.ASCIZ	@HEADER ERRORS @
73	101753	102	125	123	EMS110:	.ASCIZ	@BUS TIMEOUT (04 TRAP) @
74	102002	101	104	104	EMS111:	.ASCIZ	@ADDRESS @
75	102013	127	110	105	EMS112:	.ASCII	@WHEN READING/WRTING RH REGISTERS @
76	102055	101	124	040		.ASCIZ	@AT THE FOLLOWING @
77	102077	124	110	105	EMS113:	.ASCIZ	@THE SELECTED DEVICE IS @
78	102127	116	117	116	EMS114:	.ASCIZ	@NONEXISTENT DEVICE @
79	102153	015	012	000	EMS115:	.ASCIZ	<CR><LF>
80	102156	124	110	105	EMS116:	.ASCIZ	@THE MASSBUS CONTROLLER @
81	102206	106	101	111	EMS117:	.ASCIZ	@FAILED TO DETECT @
82	102230	116	117	124	EMS120:	.ASCIZ	@NOT AN RM05/3/2 @
83	102251	103	117	116	EMS121:	.ASCIZ	@CONTROL STATUS REGISTER 1, RMCS1, @
84	102314	102	125	123	EMS122:	.ASCIZ	@BUS ADDRESS REGISTER, RMBA, @
85	102351	103	117	116	EMS123:	.ASCIZ	@CONTROL STATUS REGISTER 2, RMCS2, @
86	102414	105	122	122	EMS124:	.ASCIZ	@ERROR REGISTER 1, RMER1, @
87	102446	101	124	124	EMS125:	.ASCIZ	@ATTENTION SUMMARY REGISTER, RMAS, @
88	102511	115	101	111	EMS126:	.ASCIZ	@MAINTENANCE REGISTER #1, RMMR #1, @
89	102554	105	103	103	EMS127:	.ASCIZ	@ECC POSITION REGISTER, RMEC1, @
90	102613	105	103	103	EMS130:	.ASCIZ	@ECC PATTERN REGISTER, RMEC2, @
91	102651	115	101	111	EMS131:	.ASCIZ	@MAINTENANCE REGISTER 2, RMMR2, @
92	102711	116	117	124	EMS132:	.ASCIZ	@NOT INITIALIZED BY @
93	102735	125	116	111	EMS133:	.ASCIZ	@UNIBUS INITIALIZE @
94	102760	103	117	116	EMS134:	.ASCIZ	@CONTROLLER CLEAR, I.E. BIT 5 OF @
95	103022	122	110	057	EMS135:	.ASCIZ	@RH/RM ERROR CLEAR (RMCS1, BIT 14) @
96	103065	104	122	111	EMS136:	.ASCIZ	@DRIVE CLEAR COMMAND @
97	103112	104	122	111	EMS137:	.ASCIZ	@DRIVE STATUS REGISTER, RMDS @
98	103147	115	105	104	EMS140:	.ASCIZ	@MEDIUM OFF LINE @
99	103170	042	115	117	EMS141:	.ASCIZ	@'MOL' (RMDS, BIT 12) @
100	103216	104	122	111	EMS142:	.ASCIZ	@DRIVE FAULT @
101	103233	042	104	126	EMS143:	.ASCIZ	@'DVC' (RMER2, BIT 7) @
102	103261	125	116	123	EMS144:	.ASCIZ	@UNSAFE @
103	103271	042	125	116	EMS145:	.ASCIZ	@'UNS' (RMER1, BIT 14) @
104	103320	123	110	117	EMS146:	.ASCIZ	@SHOULD BE SET @
105	103337	117	106	106	EMS147:	.ASCIZ	@OFFSET MODE @
106	103354	126	117	114	EMS150:	.ASCIZ	@VOLUME VALID @
107	103372	042	117	115	EMS151:	.ASCIZ	@'OM' (RMDS, BIT 0) @
108	103416	042	126	126	EMS152:	.ASCIZ	@'VV' (RMDS, BIT 6) @
109	103442	120	101	103	EMS153:	.ASCIZ	@PACK ACK COMMAND @
110	103464	116	117	124	EMS154:	.ASCIZ	@NOT SET BY @
111	103500	117	106	106	EMS155:	.ASCIZ	@OFFSET COMMAND @
112	103520	116	117	124	EMS156:	.ASCIZ	@NOT RESET BY @
113	103536	122	124	103	EMS157:	.ASCIZ	@RTC COMMAND @
114	103553	122	111	120	EMS160:	.ASCIZ	@RIP COMMAND @

115	103570	117	106	106	EMS161:	.ASCIZ	@OFFSET REGISTER (RMOF) @
116	103620				EMS162:		@<UNUSED>
117	103620	104	125	122	EMS163:	.ASCIZ	@DURING RECALIBRATE @
118	103644	111	123	040	EMS164:	.ASCII	@IS INTERMITTENT OR DRIVE DIDNT DROP ON @
119	103713	103	131	114		.ASCIZ	@CYLINDER @
120	103725	125	116	105	EMS165:	.ASCIZ	@UNEXPECTED @
121	103741	122	105	103	EMS166:	.ASCIZ	@RECALIBRATE COMMAND @
122	103766	042	101	124	EMS167:	.ASCIZ	@'ATA' (RMDS, BIT15) @
123	104013	127	110	105	EMS170:	.ASCIZ	@WHEN READING REGISTER @
124	104042	105	122	122	EMS171:	.ASCIZ	@ERROR REGISTER #2, RMER2, @
125	104075	116	117	116	EMS172:	.ASCIZ	@NONRECOVERABLE @
126	104115	122	105	103	EMS173:	.ASCIZ	@RECOVERABLE @
127	104132	104	101	124	EMS174:	.ASCIZ	@DATA @
128	104140	104	125	122	EMS175:	.ASCIZ	@DURING WRITE COMMAND @
129	104166	042	117	120	EMS176:	.ASCIZ	@'DPE' (RMER2, BIT 13) @
130	104215	111	116	040	EMS177:	.ASCIZ	@IN WRITE PROTECT @
131	104237	103	101	116	EMS200:	.ASCIZ	@CAN NOT SET @
132	104254	104	111	101	EMS201:	.ASCIZ	@DIAGNOSTIC MODE 'DMD' (RMMR1, BIT 0) @
133	104322	104	125	122	EMS202:	.ASCIZ	@DURING DIAGNOSTIC MODE @
134	104352	111	116	103	EMS203:	.ASCIZ	@INCORRECT @
135	104365	042	127	122	EMS204:	.ASCIZ	@'WRL' (RMDS, BIT 11) @
136	104413	105	130	105	EMS205:	.ASCIZ	@EXECUTED @
137	104425	127	111	124	EMS206:	.ASCIZ	@WITH COMP ERROR SET @
138	104452	042	107	117	EMS207:	.ASCIZ	@'GO' (RMCS1, BIT 0) @
139	104477	127	122	111	EMS210:	.ASCIZ	@WRITING @
140	104510	127	101	123	EMS211:	.ASCIZ	@WAS RESET BY @
141	104526	120	122	117	EMS212:	.ASCIZ	@PROGRAM INTERRUPT @
142	104551	127	101	123	EMS213:	.ASCIZ	@WAS NOT GENERATED @
143	104574	123	105	105	EMS214:	.ASCIZ	@SEEK COMMAND @
144	104612	120	122	117	EMS215:	.ASCIZ	@PROGRAM TIMEOUT @
145	104633	104	125	122	EMS216:	.ASCIZ	@DURING LOOK AHEAD TEST @
146	104663	114	117	117	EMS217:	.ASCIZ	@LOOK AHEAD REGISTER, RMLA, @
147	104716	123	105	101	EMS220:	.ASCIZ	@SEARCH COMMAND @
148	104736	102	101	104	EMS221:	.ASCIZ	@BAD SECTOR ERROR 'BSE' (RMER2, BIT 15) @
149	105006	101	040	104	EMS222:	.ASCIZ	@A DATA TRANSFER COMMAND @
150	105037	110	105	101	EMS223:	.ASCIZ	@HEADER COMPARE INHIBIT 'HCI' (RMOF, BIT 10) @
151	105114	116	117	116	EMS224:	.ASCIZ	@NONEXISTENT MEMORY 'NEM' (RMCS2, BIT 11) @

1	105166	105	130	120	EH1:	.ASCIIZ	@EXPCTD	RECEVD@		
2	105205	102	125	123	EH110:	.ASCIIZ	@BUSADRA			
3	105214	040	122	115	EH111:	.ASCIIZ	@ RMCS2	RMCS1@		
4										
5	105233	122	105	103	EH114:	.ASCIIZ	@RECEVD	SNGPRT	DULPRT@	
6	105262	105	130	120	EH223:	.ASCIIZ	@EXPCTD	RECEVD	DATA@	
7	105310	105	130	120	EH256:	.ASCII	@EXPCTD	RECEVD	RGSTRA<CRLF>	
8	105336	123	124	101		.ASCIIZ	@STATUS	STATUS	INDEX@	
9										
10	105364	107	104	101	EH336:	.ASCIIZ	@GDADRS	GDDATA	BDADRS	BDDATA@
11	105423	122	115	103	EH337:	.ASCII	@RMCS2	STATUS	FAILING	DATA@<CRLF>
12	105462	137	137	137		.ASCII	@			@<CRLF>
13	105521	105	130	120		.ASCIIZ	@EXPCTD	RECEVD	--BIT--	ADDRESS@
14										
15	105560	122	115	105	EH344:	.ASCII	@RMER1	STATUS	HEADER	FAILING@<CRLF>
16	105620	137	137	137		.ASCII	@		WORD	BIT@<CRLF>
17	105656	105	130	120		.ASCIIZ	@EXPCTD	RECEVD	NUMBER	POSITION@
18	105716	105	130	120	EH353:	.ASCII	@EXPCTD	RECEVD@<CRLF>		
19	105735	074	103	122		.ASCIIZ	@<CRLF>	RMLA	RMLA	RMOF @
20										
21	105772	040	122	115	STSH1:	.ASCII	@ RMCS1	RMCS2	RMDS	RMER1 RMER2@
22	106037	040	040	040		.ASCIIZ	@ RMA@			
23	106047	040	122	115	STSH2:	.ASCII	@ RMWC	RMBA	RMDA	RMOF RMDCA
24	106114	040	040	040		.ASCIIZ	@ RMEC1	RMEC2@		
25	106136	040	122	115	STSH3:	.ASCIIZ	@ RMDA	RMDC	RMOF	RMLA@
26	106174	040	122	115	STSH4:	.ASCIIZ	@ RMMR1	RMMR2	RMDT	RMSNA
27						.EVEN				

1	106232	001140	001142	000000	ED1:	.WORD	\$GDDAT,\$BDDAT,0
2	106240	001276	000000		ED110:	.WORD	\$BASE,0
3	106244	001174	001176	000000	ED111:	.WORD	\$TMP0,\$TMP1,0
4							
5	106252	001364	001176	001200	ED114:	.WORD	RMDT1,\$TMP1,\$TMP2,0
6	106262	001140	001142	001174	ED223:	.WORD	\$GDDAT,\$BDDAT,\$TMP0,0
7							
8	106272	001134	001140	001136	ED336:	.WORD	\$GDADR,\$GDDAT,\$BDADR,\$BDDAT,0
9							
10	106304	001140	001142	001174	ED337:	.WORD	\$GDDAT,\$BDDAT,\$TMP0,\$TMP1,0
11	106316	001140	001142	001444	ED353:	.WORD	\$GDDAT,\$BDDAT,RMOF0,0
12							
13	106326	001336	001346	001350	STSD1:	.WORD	RMCS1I, RMCS2I, RMDSI, RMER1I, RMER2I, RMASI, 0
14	106344	001340	001342	001344	STSD2:	.WORD	RMWC1, RMBAI, RMDAI, RMOF1, RMDCI, RMEC1I
15	106360	001404	000000			.WORD	RMEC2I, 0
16	106364	001344	001372	001370	STSD3:	.WORD	RMDAI, RMDCI, RMOF1, RMLAI, 0
17	106376	001362	001376	001364	STSD4:	.WORD	RMMR1I, RMMR2I, RMDT1, RMSNI, 0

1	106410	000	.	EF110:	.BYTE	0
2	106411	000	000	EF111:	.BYTE	0,0
3	106413	000	000	000	EF114:	.BYTE 0,0,0
4	106416	000	000	000	EF336:	.BYTE 0,0,0,0
5	106422	000	000	000	EF337:	.BYTE 0,0,0,0,0
6						
7	106427	000	000	000	STSF:	.BYTE 0,0,0,0,0,0,0,0
8					.EVEN	

```
1      ;STORAGE FOR GENERAL DATA TRANSFERS
2 106436 BUFFER:
3 106436 BUFO1E: .BLKW 258.
4 107442 BUFTWO: .BLKW 258.
5
6      ;STORAGE FOR MANUFACTURES 16 BIT MODE BAD SECTOR FILE
7 110446 000000 000000 MFGFIL: .WORD 0,0 ;2 HEADER WORDS
8 110452 .BLKW 256. ;256. WORDS OF DATA
9 111452 177777 .WORD -1 ;TERMINATOR IF FILE IS FULL
10
11      ;STORAGE FOR USERS 16 BIT MODE BAD SECTOR FILE
12 111454 000000 000000 USRFIL: .WORD 0,0 ;2 HEADER WORDS
13 111460 .BLKW 256. ;256. WORDS OF DATA
14 112460 177777 .WORD -1 ;TERMINATOR IF FILE IS FULL
15
16      .=BUFFER
17
18 106436
19 106436 200
20 106437 200
21 106440 114 111 123
22 106456 055 055 055
23 106474 124 061 011
24 106526 124 062 011
25 106553 124 063 011
26 106607 124 064 011
27 106660 124 065 011
28 106704 124 066 011
29 106737 124 067 011
30 107007 124 061 060
31 107057 124 061 061
32 107117 124 061 062
33 107170 124 061 063
34 107244 124 061 064
35 107301 124 061 065
36 107330 124 061 066
37 107365 124 061 067
38 107416 124 062 060
39 107461 124 062 061
40 107543 124 062 062
41 107604 124 062 063
42 107645 124 062 064
43 107700 124 062 065
44 107766 200
45 107767 117 120 105
46 10023 055 055 055
47 110057 123 127 111
48 110074 055 055 055
49 110131 040 040 061
50 110155 040 040 061
51 110200 040 040 061
52 110235 040 040 061
53 110244 C 0 040 061
54 110275 040 040 061
55 110321 040 040 040
56 110345 040 040 040
57 110404 040 040 040

HELP:
.ASCII <CRLF>
.ASCII <CRLF>
.ASCII @LIST OF TESTS@<CRLF>
.ASCII @-----@<CRLF>
.ASCII @T1 CONTROLLER ACCESS TEST@<CRLF>
.ASCII @T2 WRITE, READ ZEROS@<CRLF>
.ASCII @T3 WRITE, WRITE CHECK ZEROS@<CRLF>
.ASCII @T4 WRITE, WRITE CHECK ZEROS W/ WCE ERROR@<CRLF>
.ASCII @T5 WRITE, READ ONES@<CRLF>
.ASCII @T6 WRITE, WRITE CHECK ONES@<CRLF>
.ASCII @T7 WRITE, WRITE CHECK ONES W/ WCE ERROR@<CRLF>
.ASCII @T10 WRITE, WRITE CHECK MULTIPLE SECTORS@<CRLF>
.ASCII @T11 WRITE, READ W/ IMPLIED SEEK@<CRLF>
.ASCII @T12 WRITE, WRITE CHECK W/ HEAD SWITCHING@<CRLF>
.ASCII @T13 WRITE, WRITE CHECK W/ MID-TRANSFER SEEK@<CRLF>
.ASCII @T14 WRITE, READ W/ HCE ERROR@<CRLF>
.ASCII @T15 WRITE, READ W/ HCl@<CRLF>
.ASCII @T16 WRITE, READ W/ IVC ERROR@<CRLF>
.ASCII @T17 WRITE, READ W/ ABORT@<CRLF>
.ASCII @T20 WRITE, READ EACH CURRENT LEVEL@<CRLF>
.ASCII @T21 WRITE, WRITE CHECK W/ CURRENT LEVEL SWITCHING@<CRLF>
.ASCII @T22 WRITE, READ EARLY PEAK SHIFT@<CRLF>
.ASCII @T23 WRITE, READ MIXED PEAK SHIFT@<CRLF>
.ASCII @T24 WRITE, READ EACH TRACK@<CRLF>
.ASCII @T25 READ, WRITE CHECK MULTIPLE SECTORS IN OFFSET MODE@<CRLF>
.ASCII <CRLF>
.ASCII @OPERATIONAL SWITCH SETTINGS@<CRLF>
.ASCII @-----@<CRLF>
.ASCII @SWITCH USE@<CRLF>
.ASCII @-----@<CRLF>
.ASCII @ 15 HALT ON ERROR@<CRLF>
.ASCII @ 14 LOOP ON TEST@<CRLF>
.ASCII @ 13 INHIBIT ERROR TYPEOUTS@<CRLF>
.ASCII @ 12 @<CRLF>
.ASCII @ 11 INHIBIT ITERATIONS@<CRLF>
.ASCII @ 10 BELL ON ERROR@<CRLF>
.ASCII @ 9 LOOP ON ERROR@<CRLF>
.ASCII @ 8 LOOP ON TEST IN SWR<7:0>@<CRLF>
.ASCII @ 7 TN128@<CRLF>
```

58	110420	040	040	040	.ASCII	@	6	TN64@<CRLF>
59	110433	040	040	040	.ASCII	@	5	TN32@<CRLF>
60	110446	040	040	040	.ASCII	@	4	TN16@<CRLF>
61	110461	040	040	040	.ASCII	@	3	TN8@<CRLF>
62	110473	040	040	040	.ASCII	@	2	TN4@<CRLF>
63	110505	040	040	040	.ASCII	@	1	TN2@<CRLF>
64								
65	000200				.END		200	

ABASE = 176700	ATNMSK 000377	BUFONE 106436	EBL - 020000	EH344 105560
ACDW1 = 000000	ATNTBL 071746	BUFTWO 107442	ECH = 000100	EH353 105716
ACDW2 = 000000	AUNIT = 000000	CC - 004000	ECI = 004000	EMS1 076772
ACKSTS 053432	AUSWR = 000000	CH - 002000	ECRC = 001000	EMS10 077326
ACPUOP= 000000	AVECT1= 120254	CHGADR 001330	EDT1 076614	EMS100 101544
ADDW0 = 000000	AVECT2= 000000	CHRCNT 066107	EDT110 076632	EMS101 101572
ADDW1 = 000000	A16 = 000400	CKSWR = 104410	EDT111 076634	EMS102 101620
ADDW10= 000000	A17 = 001000	CLKADR 001520	EDT114 076636	EMS103 101647
ADDW11= 000000	BACK = 000001	CLKVCT 001522	EDT2 076624	EMS104 101656
ADDW12= 000000	BADSCT 040242	CLR = 000040	EDT223 076640	EMS105 101705
ADDW13= 000000	BADTMO 005342	CLRSTS 052552	EDT336 076650	EMS106 101734
ADDW14= 000000	BAI = 000010	CMNSTA 007476	EDT337 076660	EMS11 077370
ADDW15= 000000	BB00 = 000001	CMPSUF 042432	EDT344 076670	EMS110 101753
ADDW2 = 000000	BB01 = 000002	CMPEER 050550	EDT353 076702	EMS111 102002
ADDW3 = 000000	BB02 = 000004	CNSL01 071105	ED1 106232	EMS112 102013
ADDW4 = 000000	BB03 = 000010	CNSL02 071115	ED110 106240	EMS113 102077
ADDW5 = 000000	BB04 = 000020	CNSL03 071157	ED111 106244	EMS114 102127
ADDW6 = 000000	BB05 = 000040	CNSL04 071166	ED114 106252	EMS115 102153
ADDW7 = 000000	BB06 = 000100	CNSL07 071222	ED223 106262	EMS116 102156
ACDW8 = 000000	BB07 = 000200	CNSL08 071364	ED336 106272	EMS117 102206
ADDW9 = 000000	BB08 = 000400	CNSL09 071365	ED337 106304	EMS12 077401
ADEVCT= 000000	BB09 = 001000	CNTCLR 052434	ED353 106316	EMS120 102230
ADEVN 000000	BIT0 = 000001	COMMA 070525	EECC = 000020	EMS121 102251
ADR - 000001	BIT00 = 000001	CONT = 000100	EFT1 076704	EMS122 102314
AENV = 000000	BIT01 = 000002	CPSAVE 065230	EFT110 076722	EMS123 102351
AENVN = 000000	BIT02 = 000004	CR = 000015	EFT111 076724	EMS124 102414
AFATAL= 000000	BIT03 = 000010	CRLF = 000200	EFT114 076726	EMS125 102446
ALL 070514	BIT04 = 000020	CTLFG 001326	EFT2 076714	EMS126 102511
AMADR1= 000000	BIT05 = 000040	CYLSK= 001777	EFT223 076730	EMS127 102554
AMADR2= 000000	BIT06 = 000100	DBCK = 100000	EFT336 076740	EMS13 077442
AMADR3= 000000	BIT07 = 000200	DBEN = 040000	EFT337 076750	EMS130 102613
AMADR4= 000000	BIT08 = 000400	DBL = 002000	EFT344 076760	EMS131 102651
AMAMS1= 000000	BIT09 = 001000	DCK = 100000	EFT353 076770	EMS132 102711
AMAMS2 000000	BIT1 = 000002	DDISP = 177570	EF110 106410	EMS133 102735
AMAMS3 000000	BIT10 = 002000	DEBL = 020000	EF111 106411	EMS134 102760
AMAMS4= 000000	BIT11 = 004000	DEVSEL 050762	EF114 106413	EMS135 103022
AMSGAD= 000000	BIT12 = 010000	DISPLA 001156	EF336 106416	EMS136 103065
AMSGLG= 000000	BIT13 = 020000	DISPRE 000174	EF337 106422	EMS137 103112
AMSGTY= 000000	BIT14 = 040000	DLT = 100000	EHT1 076470	EMS14 077453
AMTYP1= 000000	BIT15 = 100000	DMU = 000001	EHT110 076512	EMS140 103147
AMTYP2= 000000	BIT2 = 000004	DPE = 000010	EHT111 076516	EMS141 103170
AMTYP3= 000000	BIT3 = 000010	DPEHI = 040000	EHT114 076522	EMS142 103216
AMTYP4= 000000	BIT4 = 000020	DPELO = 020000	EHT2 076502	EMS143 103233
AOE - 001000	BIT5 = 000040	DPR = 000400	EHT223 076526	EMS144 103261
APASS = 000000	BIT6 = 000100	DRIVES 071600	EHT256 076540	EMS145 103271
APE - 100000	BIT7 = 000200	DRQ = 004000	EHT336 076552	EMS146 103320
APRIOR= 000000	BIT8 = 000400	DRVCLR= 000010	EHT337 076564	EMS147 103337
ATCSU= 000040	BIT9 = 001000	DRVSTS 055770	EHT344 076576	EMS15 077464
APTENV= 000001	BLNKS1 071643	DRY = 000200	EHT353 076610	EMS150 103354
APTSIZ= 000200	BLNKS2 071642	DSWR = 177570	EH1 105166	EMS151 103372
APTSPO= 000100	BLNKS3 071641	DTASTS 056572	EH110 105205	FMS152 103416
ARGS = 000004	BLNKS4 071640	DTE - 010000	EH111 105214	EMS153 103442
ASNDA 001516	BOTADR 066104	DTO - 010000	EH114 105233	EMS154 103464
ASNDC 001514	BOTFLG 066106	DULPRT= 024024	EH223 105262	EMS155 103500
ASWREG= 000000	BPTVEC= 000014	DVA - 004000	EH256 105310	EMS156 103520
ATA = 100000	BSE 100000	DVC 000200	EH336 105364	EMS157 103536
ATESTN= 000000	BUFFER 106436	EARLY 072272	EH337 105423	EMS16 077506

EMS160	103553	EMS40	100275	EMT121	073716	EMT203	074602	EMT266	075354
EMS161	103570	EMS41	100324	EMT122	073726	EMT204	074614	EMT267	075366
EMS162	103620	EMS42	100352	EMT123	073736	EMT205	074632	EMT27	072630
EMS163	103620	EMS43	100401	EMT124	073746	EMT206	074642	EMT270	075404
EMS164	103644	EMS44	100430	EMT125	073756	EMT207	074652	EMT271	075420
EMS165	103725	EMS45	100503	EMT126	073766	EMT21	072550	EMT272	075436
EMS166	103741	EMS46	100546	EMT127	074000	EMT210	074664	EMT273	075454
EMS167	103766	EMS47	100575	EMT13	072472	EMT211	074672	EMT274	075472
EMS17	077536	EMS5	077164	EMT130	074012	EMT212	074706	EMT275	075510
EMS170	104013	EMS50	100624	EMT131	074024	EMT213	074722	EMT276	075522
EMS171	104042	EMS51	100653	EMT132	074036	EMT214	074732	EMT277	075536
EMS172	104075	EMS52	100701	EMT133	074050	EMT215	074752	EMT3	072416
EMS173	104115	EMS53	100711	EMT134	074062	EMT216	074764	EMT30	072640
EMS174	104132	EMS54	100723	EMT135	074074	EMT217	074774	EMT300	075554
EMS175	104140	EMS55	100745	EMT136	074106	EMT22	072560	EMT301	075574
EMS176	104166	EMS56	100774	EMT137	074120	EMT220	075004	EMT302	075616
EMS177	104215	EMS57	101051	EMT14	072504	EMT221	075014	EMT303	075630
EMS2	077041	EMS6	077214	EMT140	074130	EMT222	075024	EMT304	075642
EMS20	077576	EMS60	101077	EMT141	074140	EMT223	075034	EMT305	075654
EMS200	104237	EMS61	101112	EMT142	074150	EMT224	075044	EMT306	075666
EMS201	104254	EMS62	101141	EMT143	074160	EMT225	075046	EMT307	075676
EMS202	104322	EMS63	101157	EMT144	074170	EMT226	075050	EMT31	072650
EMS203	104352	EMS64	101205	EMT145	074200	EMT227	075052	EMT310	075716
EMS204	104365	EMS65	101223	EMT146	074210	EMT23	072570	EMT311	075730
EMS205	104413	EMS66	101271	EMT147	074220	EMT230	075054	EMT312	075742
EMS206	104425	EMS67	101341	EMT15	072512	EMT231	075056	EMT313	075754
EMS207	104452	EMS7	077261	EMT150	074230	EMT232	075060	EMT314	075774
EMS21	077621	EMS70	101366	EMT151	074240	EMT233	075062	EMT315	076006
EMS210	104477	EMS71	101400	EMT152	074252	EMT234	075064	EMT316	076020
EMS211	104510	EMS72	101413	EMT153	074264	EMT235	075066	EMT317	076032
EMS212	104526	EMS73	101423	EMT154	074302	EMT236	075070	EMT32	072660
EMS213	104551	EMS74	101440	EMT155	074320	EMT237	075072	EMT320	076042
EMS214	104574	EMS75	101457	EMT156	074332	EMT24	072600	EMT321	076052
EMS215	104612	EMS76	101465	EMT157	074344	EMT240	075074	EMT322	076062
EMS216	104633	EMS77	101516	EMT16	072520	EMT241	075076	EMT323	076070
EMS217	104663	EMTVEC=	000030	EMT160	074356	EMT242	075100	EMT324	076100
EMS22	077644	EMT1	072404	EMT161	074366	EMT243	075102	EMT325	076112
EMS220	104716	EMT10	072454	EMT162	074400	EMT244	075104	EMT326	076124
EMS221	104736	EMT100	073452	EMT163	074412	EMT245	075106	EMT327	076142
EMS222	105006	EMT101	073470	EMT164	074422	EMT246	075110	EMT33	072670
EMS223	105037	EMT102	073506	EMT165	074430	EMT247	075120	EMT330	076154
EMS224	105114	EMT103	073516	EMT166	074436	EMT25	072610	EMT331	076164
EMS23	077660	EMT104	073530	EMT167	074444	EMT250	075132	EMT332	076176
EMS24	077706	EMT105	073546	EMT17	072530	EMT251	075146	EMT333	076210
EMS25	077735	EMT106	073556	EMT170	074452	EMT252	075154	EMT334	076226
EMS26	077752	EMT107	073566	EMT171	074464	EMT253	075162	EMT335	076244
EMS27	077773	EMT11	072460	EMT172	074474	EMT254	075200	EMT336	076264
EMS3	077056	EMT110	073606	EMT173	074504	EMT255	075210	EMT337	076274
EMS30	100004	EMT111	073620	EMT174	074514	EMT256	075220	EMT34	072700
EMS31	100014	EMT112	073626	EMT175	074526	EMT257	075230	EMT340	076304
EMS32	100043	EMT113	073634	EMT176	074534	EMT26	072620	EMT341	076316
EMS33	100072	EMT114	073650	EMT177	074542	EMT260	075242	EMT342	076324
EMS34	100120	EMT115	073656	EMT2	072410	EMT261	075254	EMT343	076336
EMS35	100171	EMT116	073666	EMT20	072540	EMT262	075274	EMT344	076350
EMS36	100220	EMT117	073676	EMT200	074544	EMT263	075304	EMT345	076362
EMS37	100247	EMT12	072464	EMT201	074556	EMT264	075314	EMT346	076372
FMS4	077121	EMT120	073706	EMT202	074570	EMT265	075334	EMT347	076406


```

ILRG76= 000076
IOTVEC= 000020
IPCK0 = 000001
IPCK1 = 000002
IPCK2 = 000004
IPCK3 = 000010
IR      = 000100
IVC     = 010000
LBC     = 002000
LBT     = 002000
LF      = 000012
LODEV   071506
LS      = 000004
LSC     = 004000
LST     = 000002
LSTRK   001334
MCLK    = 004000
MCPE    = 020000
MDF     = 000100
MDPE    = 000400
MEDENB  001512
MFGFIL  110446
MI      = 000004
MIXED   071756
MOC     = 000400
MOH     = 020000
MOL     = 010000
MRD     = 002000
MS      = 000040
MSC     = 000002
MSDRVS  071406
MSE     = 100000
MSER    = 000200
MSGDRV  071422
MSHELP  070530
MUR     = 001000
MWD     = 000010
MWP     = 000010
MXF     = 001000
N       071634
NDTMSK- 115760
NED     = 010000
NEM     = 004000
NONE    071627
NOP     = 000000
NOTAVL  071540
NOTPRS  071523
NOTRM   071465
NSA     = 100000
OCC     = 100000
OFD     = 000200
OFFSET= 000014
OM      = 000001
ONES    072016
OPE     = 020000
OPI     020000
OR      000200

```

```

PACACK= 000022
PAKACK= 000022
PAR      = 000010
PAT      = 000020
PDA      = 000400
PFECH    066152
PFECH1   066162
PFECH2   066250
PFECH3   066266
PFECH4   066274
PGE      = 002000
PGM      = 001000
PHA      = 000200
PIP      = 020000
PIRQ     = 177772
PIRQVE   = 000240
PLFS     = 002000
PRIERR   044056
PRO      = 000000
PR1      = 000040
PR2      = 000100
PR3      = 000140
PR4      = 000200
PR5      = 000240
PR6      = 000300
PR7      = 000340
PS       = 177776
PSEL     = 002000
PSW      = 177776
PUT      043330
PUTBUF   001412
PUTINX   001553
PWRVEC   = 000024
QUES     070521
RCLSTS   054226
RD        = 000070
RDCHR    = 104411
RDLIN    = 104412
RDOCT    = 104413
RDY      = 000200
READY    007670
RECAL    = 000006
RESREG   = 104415
RESVEC   = 000010
REX      = 010000
RG       = 040000
RGDTPT   071756
RH       = 000072
RIP      = 000020
RELEASE  = 000012
RMAS     = 000016
RMASI    001354
RMASO    001430
RMB      = 000004
RMBAE    = 000050
RMBAEI   001406
RMBAE0   001462

```

RMBAI	=	001342
RMBAO		001416
RMCS1	=	000000
RMCS1I		001336
RMCS10		001412
RMCS2	=	000010
RMCS2I		001346
RMCS20		001422
RMCS3	=	000052
RMCS3I		001410
RMCS30		001464
RMDA	=	000006
RMDAI		001344
RMDAO		001420
RMDB	=	000022
RMDBI		001360
RMDBO		001434
RMDC	=	000034
RMDCI		001372
RMDCO		001446
RMDS	=	000012
RMDSI		001350
RMDSO		001424
RMDT	=	000026
RMDTI		001364
RMDTO		001440
RMEC1		000044
RMEC1I		001402
RMEC10		001456
RMEC2	=	000046
RMEC2I		001404
RMEC20		001460
RMER1	=	000014
RMER1I		001352
RMER10		001426
RMER2	=	000042
RMER2I		001400
RMER20		001454
RMHR		000036
RMHRI		001374
RMHRO		001450
RMLA	=	000020
RMLAI		001356
RMLAO		001432
RMMR1	=	000024
RMMR1I		001362
RMMR10		001436
RMMR2	=	000040
RMMR2I		001376
RMMR20		001452
RMOF	=	000032
RMOFI		001370
RMOFO		001444
RMR	=	000004
RMSN	=	000030
RMSNI		001366
RMSNO		001442

RMWC = 000002	SW04 = 000020	TST25 = 035624	SCNTLG = 067520	\$MADR4 = 001270
RMWCI = 001340	SW05 = 000040	TST3 = 011274	SCNTLU = 067513	\$MAIL = 001222
RMWCO = 001414	SW06 = 000100	TST4 = 012252	\$CPUOP = 001250	\$MAMS1 = 001252
RQA = 100000	SW07 = 000200	TST5 = 013446	\$CRLF = 001217	\$MAMS2 = 001256
RQB = 040000	SW08 = 000400	TST6 = 014454	\$DBLK = 063262	\$MAMS3 = 001262
RTC = 000016	SW09 = 001000	TST7 = 015432	\$DDW0 = 001306	\$MAMS4 = 001266
R6 = 2000006	SW1 = 000002	TYPBN = 104406	\$DDW1 = 001310	\$MBADR = 001102
R7 = 2000007	SW10 = 002000	TYPDS = 104405	\$DDW2 = 001312	\$MFLG = 070422
SADMSK = 000377	SW11 = 004000	TYPE = 104401	\$DDW3 = 001314	\$MNEW = 067536
SAVREG = 104414	SW12 = 010000	TYPDC = 104402	\$DDW4 = 001316	\$MSGAD = 001236
SA1 = 000001	SW13 = 020000	TYPON = 104404	\$DDW5 = 001320	\$MSGLG = 001240
SA16 = 000020	SW14 = 040000	TYPDS = 104403	\$DDW6 = 001322	\$MSGTY = 001222
SA2 = 000002	SW15 = 100000	UNS = 040000	\$DDW7 = 001324	\$MSWR = 067525
SA4 = 000004	SW2 = 000004	UNTMASK = 000007	\$DEVCT = 001232	\$MTYP1 = 001253
SA8 = 000010	SW3 = 000010	UNTOFF = 071557	\$DEVN = 001300	\$MTYP2 = 001257
SC = 100000	SW4 = 000020	UNTON = 071570	\$DOAGN = 037252	\$MTYP3 = 001263
SCOPE = 000004	SW5 = 000040	UPE = 020000	\$DTBL = 063252	\$MTYP4 = 001267
SCTMSG = 070426	SW6 = 000100	USE = 040000	\$ENDAD = 037242	\$MXCNT = 064536
SCTMSK = 003700	SW7 = 000200	USRFIL = 111454	\$ENDCT = 037100	\$NULL = 001170
SCO = 000100	SW8 = 000400	U0 = 000001	\$ENULL = 037256	\$NWTST = 000001
SC1 = 000200	SW9 = 001000	U1 = 000002	\$ENV = 001242	\$OCNT = 063514
SC2 = 000400	SYSTAT = 071430	U2 = 000004	\$ENVM = 001243	\$OMODE = 063516
SC3 = 001000	TADMSK = 177400	VV = 000100	\$EOP = 037044	\$OVER = 064522
SC4 = 002000	TAG = 020000	WC = 000040	\$EOPCT = 037072	\$PASS = 001230
SEARCH = 000030	TAGADR = 001114	WCD = 000050	\$EOSP = 037006	\$PASTM = 001106
SEERR = 044710	TAP = 040000	WCE = 040000	\$ERFLG = 001117	\$POWER = 070150
SEEK = 000004	TA1 = 000400	WCEH1 = 010000	\$ERMAX = 001131	\$PWRDN = 070002
SEKSTS = 051174	TA16 = 010000	WCELO = 004000	\$ERROR = 064640	\$PWRMG = 070136
SHUT = 037262	TA2 = 001000	WCF = 000040	\$ERRPC = 001132	\$PWRUP = 070054
SHUT2 = 067550	TA4 = 002000	WCH = 000052	\$ERRTB = 001602	\$QUES = 001216
SIZCLK = 043550	TA8 = 004000	WD = 000060	\$ERTTL = 001126	\$RDCHR = 067152
SKI = 040000	TBITVE = 000014	WH = 000062	\$ESCAP = 001210	\$RDLIN = 067242
SNGPRT = 020024	TIMOUT = 043672	WLE = 004000	\$FTABL = 001242	\$RDOCT = 067572
STACK = 001100	TKVEC = 000060	WRL = 004000	\$ETEND = 001326	\$RDSZ = 000010
STANDA = 007020	TPVEC = 000064	XSIZ = 006464	\$FATAL = 001224	\$RESRE = 062734
START = 005432	TRAPVE = 000034	XXDP = 001332	\$FFLG = 070424	\$RM02 = 071446
START1 = 005422	TRE = 040000	Y = 071636	\$FILLC = 001172	\$RM03 = 071453
START2 = 005436	TRTVEC = 000014	ZEROS = 072060	\$FILLS = 001171	\$RM05 = 071460
STCDRV = 062172	TST = 010000	\$APTHD = 001100	\$GDADR = 001134	\$RTNAD = 037254
STKLMT = 177774	TSTNMB = 066100	\$ATYC = 070204	\$GDDAT = 001140	\$SAVRE = 062676
STOP = 064612	TSTPRP = 037316	\$ATY1 = 070160	\$GET42 = 037232	\$SAVR6 = 070146
STSD1 = 106326	TSTQUE = 001466	\$ATY3 = 070166	\$GTSWR = 066700	\$SCOPE = 064054
STSD2 = 106344	TST1 = 010066	\$ATY4 = 070176	\$GT42P = 037226	\$SETUP = 000137
STSD3 = 106364	TST10 = 016630	\$AUTOB = 001150	\$HD = 000000	\$STUP = 177777
STSD4 = 106376	TST11 = 017634	\$BASE = 001276	\$HIBTS = 001100	\$SVLAD = 064466
STSF = 106427	TST12 = 021060	\$BDADR = 001136	\$HIOCT = 067672	\$SVPC = 000210
STSH1 = 105772	TST13 = 022266	\$BDDAT = 001142	\$ICNT = 001120	\$SWR = 167400
STSH2 = 106047	TST14 = 023310	\$BELL = 001212	\$ILLUP = 070142	\$SWREG = 001244
STSH3 = 106136	TST15 = 025156	\$BIN = 063044	\$INTAG = 001151	\$SWRMK = 000000
STSH4 = 106174	TST16 = 026562	\$CDW1 = 001302	\$ITEMB = 001130	\$SW08T = 064540
SWR = 001154	TST17 = 027602	\$CDW2 = 001304	\$LF = 001220	\$TESTN = 001226
SWREG = 000176	TST2 = 010266	\$CHARC = 064050	\$LFLG = 070423	\$TIMES = 001206
SW0 = 000001	TST20 = 030520	\$CKSWR = 066610	\$LPADR = 001122	\$TKB = 001162
SW00 = 000001	TST21 = 031556	\$CMTAG = 001114	\$LPERR = 001124	\$TKCNT = 066300
SW01 = 000002	TST22 = 032562	\$CM3 = 000000	\$MADR1 = 001254	\$TKINT = 066310
SW02 = 000004	TST23 = 033570	\$CM4 = 000005	\$MADR2 = 001260	\$TKQEN = 066307
SW03 = 000010	TST24 = 034576	\$CNTLL = 067506	\$MADR3 = 001264	\$TKQIN = 066302

\$TKQOU	066304	\$TMP4	001204	\$TRPAD	067746	\$TYPEX	064052	\$VECT2	001274
\$TKQSR	066306	\$TN	= 000026	\$TSTM	001104	\$TYPDC	063316	\$XOFF	= 000023
\$TKS	001160	\$TPB	001166	\$TSTNM	001116	\$TYPON	063332	\$XON	= 000021
\$TKSRV	066360	\$TPFLG	001173	\$TTYIN	067476	\$TYPOS	063272	\$XTSTR	064076
\$TMP0	001174	\$TPS	001164	\$TYPBN	062772	\$UNIT	001234	\$GET4	= 000000
\$TMP1	001176	\$TRAP	067674	\$TYPDS	063046	\$UNITM	001110	\$SW08	= 000026
\$TMP2	001200	\$TRAP2	067734	\$TYPE	063520	\$USWR	001246	\$FILL	063515
\$TMP3	001202	\$TRP	= 000016	\$YPEC	063732	\$VECT1	001272	\$.X	= 001100

. ABS. 112462 000
000000 001

ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 62208 WORDS (243 PAGES)
DYNAMIC MEMORY AVAILABLE FOR 70 PAGES
CZRMOC.BIC,CZRMOC/C-CZRMOC.DOC,CZRMOC.SYSMAC/M

\$GET4	4-20	14-20#												
\$ESW08	42-1	42-1	42-1	42-1	42-1	42-1	42-1	42-1	42-1	42-1	42-1	42-1	42-1	42-1
	42-1	42-1	42-1	42-1	42-1	42-1	42-1	42-1	42-1	42-1	42-1	42-1	42-1	42-1
	42-1	42-1	42-1	42-1	42-1	42-1	42-1	42-1	42-1	42-1	42-1	42-1	42-1	42-1
	42-1#	42-1#	42-1#	42-1#	42-1#	42-1#	42-1#	42-1#	42-1#	42-1#	42-1#	42-1#	42-1#	42-1#
	42-1#	42-1#	42-1#	42-1#	42-1#	42-1#	42-1#	42-1#	42-1#	42-1#	42-1#	42-1#	42-1#	42-1#
\$OFILL	40-1	40-1#	40-1*	40-1*										
\$LOCAT	42-1	43-1												
\$APTHD	5-8	5-8#												
\$ASTAT	49-1	49-1												
\$ATY1	49-1#													
\$ATY3	41-1	49-1#												
\$ATY4	43-1	49-1#												
\$ATYC	49-1	49-1#												
\$AUTOB	6-0#	10-28*	45-1	45-1	45-1									
\$BASE	6-0#	10-75	11-33	11-42*	12-57	13-2	13-33	13-180	13-322	13-497	13-645	13-790	13-969	13-;20
	13-05	13->88	13-a38	13-C24	13-E49	13-F81	13-G99	13-J54	13-K05	13-L50	13-M95	13-046	20-36	21-30
	23-12	28-23	30-15	44-25	60-2									
\$BDADR	6-0#	13-477*	13-478*	13-479	13-949*	13-950*	13-951	18-84*	60-8					
\$BDDAT	6-0#	13-465*	13-473*	13-483	13-935*	13-945*	13-955	13-A56*	13-A73*	13-A84*	13-B49*	13-B66*	13-B77*	13-D30*
	13-D31*	13-D34*	13-D37*	13-D92*	13-D93*	13-D96*	13-D99*	13-F10*	13-F72*	13-G27*	13-G82*	18-82*	24-47*	24-48*
	24-51	24-69*	24-75*	24-91*	24-108*	24-124*	24-152*	25-40*	25-41*	25-53*	25-54*	25-64*	25-65*	25-67
	25-85*	25-86*	25-87	25-107*	25-108*	25-109	25-140*	25-141*	25-142	25-156*	25-157*	25-158	25-170*	25-171*
	25-183*	25-184*	25-200*	25-201*	25-226*	25-227*	25-241*	25-242*	25-256*	25-257*	25-271*	25-272*	25-284*	25-285*
	25-314*	25-315*	25-329*	25-330*	25-344*	25-345*	25-361*	25-362*	25-387*	25-388*	25-402*	25-403*	25-417*	25-418*
	25-432*	25-433*	25-455*	25-456*	25-470*	25-471*	25-492*	25-493*	26-10*	26-34*	26-92*	26-93*	26-94	26-120*
	26-121*	26-122	26-137*	26-138	26-150*	26-151	27-39*	27-54*	29-37*	29-66*	29-67*	29-68	29-88*	29-99*
	29-100*	29-106	29-117	29-130*	29-143*	29-168*	29-182*	29-183*	29-196*	29-212*	31-29*	31-30*	31-32	31-42*
	31-51*	31-59	31-69*	31-70*	31-79*	31-80*	31-82	31-92*	31-101*	31-102*	31-104	31-114*	31-115*	31-124*
	31-125*	31-127	32-27*	32-41*	32-57*	32-71*	32-85*	32-99*	32-113*	33-35*	33-49*	33-65*	33-82*	33-103*
	33-120*	33-134*	33-159*	33-174*	33-189*	33-205*	33-222*	33-240*	33-254*	33-270*	34-16*	34-17*	34-19	34-28*
	34-29*	34-31	34-41*	34-50*	34-57*	34-60	34-69*	34-70*	34-72	34-81*	34-82*	34-84	34-94*	34-103*
	35-31*	35-47*	35-67*	35-87*	35-116*	35-135*	35-149*	35-163*	35-189*	35-190*	35-191	35-203*	35-204*	35-240*
	35-256*	35-270*	35-288*	35-301*	35-313*	35-356*	35-369*	35-382*	35-395*	35-417*	35-442*	35-457*	35-470*	35-483*
	35-496*	35-517*	35-534*	35-550*	36-46*	36-62*	36-78*	36-98*	36-112*	60-1	60-6	60-8	60-10	60-11
\$BELL	6-0#	43-1	43-1	43-1	45-1	45-1	45-1							
\$BIN	38-1	38-1#	38-1*	38-1*										
\$CDW1	6-0#													
\$CDW2	6-0#													
\$CHARC	41-1	41-1#	41-1*	41-1*	41-1*									
\$CKSWR	45-1#	47-1	47-1											
\$CM3	6-0	6-0#												
\$CM4	6-0	6-0	6-0	6-0	6-0	6-0	6-0	6-0	6-0	6-0	6-0#	6-0#	6-0#	6-0#
	6-0#	6-0#												
\$CMTAG	6-0#	10-23	10-23	10-23	10-23	10-23	10-23	10-23						
\$CNTLC	45-1	45-1	45-1	45-1	45-1#									
\$CNTLG	45-1	45-1#												
\$CNTLU	45-1	45-1	45-1#											
\$CPUOP	6-0#													
\$CRLF	6-0#	10-48	10-80	10-140	11-20	11-28	11-62	11-100	12-25	12-68	14-20	41-1	41-1	41-1
	43-1	43-1	43-1	44-20	44-61	44-104	44-114	44-126	44-134	44-154	45-1	45-1	45-1	45-1
\$DBLK	39-1	39-1	39-1#											
\$DDW0	6-0#													
\$DDW1	6-0#													
\$DDW2	6-0#													
\$DDW3	6-0#													

\$DDW4	6-0#													
\$DDW5	6-0#													
\$DDW6	6-0#													
\$DDW7	6-0#													
\$DEVCT	6-0#													
\$DEVN	6-0#	10-67*	10-78	10-117*	10-121*	11-63*	11-70*	11-97*	11-98	12-4	12-38	13-29*	14-28*	
\$DOAGN	14-20	14-20	14-20#											
\$DTBL	39-1	39-1#												
\$ENDAD	5-5	10-28	14-20#	43-1										
\$ENDCT	10-23	14-20#												
\$ENULL	14-20#													
\$ENV	6-0#	10-28	41-1	43-1	49-1	49-1								
\$ENVN	6-0#	10-23	10-71	41-1	41-1	49-1								
\$EOP	13-30	14-20#	14-29											
\$EOPCT	10-23*	14-20	14-20#											
\$EOSP	14-9#													
\$ERFLG	6-0#	42-1	42-1	42-1	42-1	42-1	42-1*	43-1	43-1	43-1*				
\$ERMAX	6-0#	10-23*	42-1	42-1	42-1	42-1*								
\$ERROR	10-23	43-1#												
\$ERRPC	6-0#	43-1	43-1	43-1	43-1*	43-1*	44-54							
\$ERRTB	8-0#	44-72												
\$ERTTL	6-0#	14-20	14-20	14-20*	43-1	43-1	43-1*							
\$ESCAP	6-0#	10-23*	42-1*	43-1	43-1	43-1	43-1							
\$ETABL	6-0#													
\$ETEND	5-8	6-0#												
\$FATAL	6-0#	49-1*												
\$FFLG	49-1	49-1#	49-1*	49-1*	49-1*									
\$FILLC	6-0#	41-1	41-1	41-1										
\$FILLS	6-0#	41-1	41-1											
\$GDADR	6-0#	13-476*	13-479	13-948*	13-951	18-83*	60-8							
\$GGDAT	6-0#	13-463*	13-464*	13-474*	13-475*	13-483	13-933*	13-934*	13-946*	13-947*	13-955	13-A57*	13-A72*	13-A85*
	13-850*	13-865*	13-878*	13-D26*	13-D88*	13-F11*	13-F12*	13-F70*	13-F71*	13-G25*	13-G26*	13-G80*	13-G81*	18-81*
	24-49*	24-50*	24-51	24-67*	24-68*	24-74*	24-76*	24-88*	24-89*	24-90*	24-106*	24-107*	24-122*	24-123*
	24-150*	24-151*	25-42*	25-55*	25-66*	25-67	25-79*	25-84*	25-87	25-98*	25-106*	25-109	25-135*	25-138*
	25-139*	25-142	25-154*	25-155*	25-158	25-165*	26-16*	26-23*	26-24*	26-25*	26-26*	26-30*	26-32	26-86*
	26-91*	26-94	26-104*	26-117*	26-122	26-134*	26-135*	26-136*	26-138	26-148*	26-149*	26-151	27-40*	27-55*
	29-36*	29-47*	29-63*	29-68	29-87*	29-101*	29-102*	29-129*	29-142*	29-166*	29-167*	29-181*	29-194*	29-195*
	29-210*	29-211*	31-31*	31-32	31-41*	31-57*	31-59	31-68*	31-81*	31-82	31-91*	31-103*	31-104	31-113*
	31-126*	31-127	32-25*	32-26*	32-39*	32-40*	32-58*	32-59*	32-72*	32-73*	32-86*	32-87*	32-100*	32-101*
	32-114*	32-115*	33-33*	33-34*	33-47*	33-48*	33-63*	33-64*	33-80*	33-81*	33-101*	33-102*	33-118*	33-119*
	33-132*	33-133*	33-157*	33-158*	33-173*	33-175*	33-187*	33-188*	33-203*	33-204*	33-220*	33-221*	33-238*	33-239*
	33-252*	33-253*	33-268*	33-269*	34-18*	34-19	34-30*	34-31	34-40*	34-71*	34-72	34-83*	34-84	34-92*
	35-29*	35-30*	35-45*	35-46*	35-65*	35-66*	35-85*	35-86*	35-114*	35-115*	35-133*	35-134*	35-147*	35-148*
	35-161*	35-162*	35-172*	35-187*	35-191	35-205*	35-206*	35-238*	35-239*	35-254*	35-255*	35-268*	35-269*	35-286*
	35-287*	35-299*	35-300*	35-314*	35-315*	35-323*	35-354*	35-355*	35-367*	35-368*	35-380*	35-381*	35-393*	35-394*
	35-415*	35-416*	35-440*	35-441*	35-455*	35-456*	35-468*	35-469*	35-481*	35-482*	35-494*	35-495*	35-515*	35-516*
	35-532*	35-533*	35-548*	35-549*	36-44*	36-45*	36-60*	36-76*	36-77*	36-96*	36-97*	36-110*	36-111*	60-1
	60-6	60-8	60-10	60-11										
\$GET42	12-34	12-43	14-20#											
\$GT42P	14-20	14-20#												
\$GTSWR	45-1#	47-1	47-1											
\$HD	4-480	4-480	4-480											
\$HIBTS	5-8#													
\$HIOCT	46-1#	46-1*												
\$ICNT	6-0#	42-1	42-1	42-1	42-1*	42-1*								
\$ILLUP	48-1	48-1	48-1#											
\$INTAG	6-0#	45-1	45-1	45-1	45-1	45-1*								

ITEM#	6-0#	43-1	43-1	43-1	43-1	43-1	43-1*	43-1*	44-49							
SLF	6-0#	41-1	41-1	43-1	43-1	45-1	45-1	45-1								
SLFLG	49-1#	49-1*														
SLPADR	6-0#	10-23*	42-1	42-1	42-1	42-1*	42-1*	42-1*								
SLPERR	6-0#	10-23*	13-84*	13-130*	13-231*	13-277*	13-372*	13-418*	13-550*	13-597*	13-698*	13-745*	13-843*	13-889*		
	13-:22*	13-:75*	13-:74*	13-<36*	13--59*	13->27*	13-?41*	13-?93*	13-@99*	13-A95*	13-C87*	13-D50*	13-F20*	13-G42*		
	13-H49*	13-H99*	13-J04*	13-J57*	13-K56*	13-L02*	13-M01*	13-M47*	13-N49*	13-N95*	13-096*	13-P53*	42-1	42-1		
	42-1	42-1*	43-1													
\$MADR1	6-0#															
\$MADR2	6-0#															
\$MADR3	6-0#															
\$MADR4	6-0#															
\$MAIL	5-8	5-8	6-0#	10-23	10-28	13-2	13-33	13-180	13-322	13-497	13-645	13-790	13-969	13-:20		
	13--05	13->88	13-@38	13-G24	13-E49	13-F81	13-G99	13-I54	13-K05	13-L50	13-M95	13-046	41-1	42-1		
	43-1															
\$MAMS1	6-0#															
\$MAMS2	6-0#															
\$MAMS3	6-0#															
\$MAMS4	6-0#															
\$MBADR	5-8#															
\$MFLG	49-1	49-1#	49-1*	49-1*												
\$MNEW	45-1	45-1#														
\$MSGAD	6-0#	49-1	49-1*													
\$MSGLG	6-0#	49-1*														
\$MSGTY	6-0#	49-1	49-1	49-1*	49-1*											
\$MSWR	45-1	45-1#														
\$MTYP1	6-0#															
\$MTYP2	6-0#															
\$MTYP3	6-0#															
\$MTYP4	6-0#															
\$MXCNT	42-1	42-1	42-1	42-1#												
\$NULL	6-0#	41-1	41-1	41-1												
\$NWTST	13-2	13-2#	13-2#	13-33	13-33#	13-33#	13-180	13-180#	13-180#	13-322	13-322#	13-322#	13-497	13-497#		
	13-497#	13-645	13-645#	13-645#	13-790	13-790#	13-790#	13-969	13-969#	13-969#	13-:20	13-:20#	13-:20#	13-:20#	13-=05	
	13--05#	13--05#	13->88	13->88#	13->88#	13-@38	13-@38#	13-@38#	13-C24	13-C24#	13-C24#	13-E49	13-E49#	13-E49#	13-E49#	
	13-F81	13-F81#	13-F81#	13-G99	13-G99#	13-G99#	13-I54	13-I54#	13-I54#	13-K05	13-K05#	13-K05#	13-L50	13-L50#		
	13-L50#	13-M95	13-M95#	13-M95#	13-046	13-046#	13-046#									
\$SOCNT	40-1#	40-1*	40-1*													
\$SOMODE	40-1	40-1#	40-1*	40-1*	40-1*	40-1*										
\$OVER	42-1	42-1	42-1	42-1	42-1#											
\$PASS	6-0#	10-23*	14-20	14-20	14-20	14-20*	14-20*	42-1	42-1	42-1						
\$PASTM	5-8#															
\$POWER	48-1	48-1#														
\$PWDRN	10-23	48-1	48-1#													
\$PWRMG	48-1#															
\$PWRUP	48-1	48-1#														
\$QUES	6-0#	41-1	41-1	43-1	43-1	45-1	45-1	45-1	45-1							
\$R2A	47-1															
\$RDCHR	45-1#	47-1	47-1													
\$RDDEC	47-1															
\$RDLIN	45-1#	47-1	47-1													
\$RDOCT	46-1#	47-1	47-1													
\$RDSZ	45-1	45-1#														
\$RESRE	37-1#	47-1														
\$RM02	10-92	44-32	50-24#													
\$RM03	10-97	44-28	50-25#													
\$RM05	10-102	44-36	50-26#													

SEQ 0310

	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1
	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1
	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1
	47-1#	47-1#	47-1#	47-1#	47-1#	47-1#	47-1#	47-1#	47-1#	47-1#	47-1#	47-1#	47-1#	47-1#
\$TRPAD	47-1	47-1#												
\$YSTM	5-8#													
\$YSTNM	6-0#	12-45*	14-20*	42-1	42-1	42-1	42-1	42-1	42-1*	42-1*	43-1	43-1	43-1	45-4
\$ITYIN	45-1	45-1	45-1	45-1	45-1	45-1#								
\$YPSN	38-1#	47-1	47-1											
\$YPDS	39-1#	47-1	47-1											
\$YPE	41-1#	47-1	47-1	49-1										
\$YPEC	41-1	41-1	41-1	41-1#	45-1									
\$YPEX	41-1	41-1	41-1#											
\$YPOC	40-1#	47-1	47-1											
\$YPON	40-1	40-1#	47-1											
\$YPOS	40-1#	47-1												
\$UNIT	6-0#	12-50*	12-70	24-49	44-22									
\$UNITM	5-8#													
\$USWR	6-0#													
\$VECT1	6-0#	11-46	11-56*											
\$VECT2	6-0#													
\$XOFF	41-1	41-1												
\$XON	41-1	41-1	45-1											
\$XTSTR	42-1#													
.\$ASTA	49-1	49-1												
.\$X	5-8	5-8#												
A16	4-721#													
A17	4-720#													
ABASE	4-772#	6-0	6-0											
ACDW1	6-0	6-0												
ACDW2	6-0	6-0												
ACKSTS	15-152	16-137	32-15#											
ACPUOP	6-0	6-0												
ADDW0	6-0	6-0												
ADDW1	6-0	6-0												
ADDW10	6-0													
ADDW11	6-0													
ADDW12	6-0													
ADDW13	6-0													
ADDW14	6-0													
ADDW15	6-0													
ADDW2	6-0	6-0												
ADDW3	6-0	6-0												
ADDW4	6-0	6-0												
ADDW5	6-0	6-0												
ADDW6	6-0	6-0												
ADDW7	6-0	6-0												
ADDW8	6-0													
ADDW9	6-0													
ADEVCT	6-0	6-0												
ADEVVM	6-0	6-0												
ADR	13-2	13-2#	13-2#	13-33	13-33#	13-33#	13-180	13-180#	13-180#	13-322	13-322#	13-322#	13-497	13-497#
	13-497#	13-645	13-645#	13-645#	13-790	13-790#	13-790#	13-969	13-969#	13-969#	13-:20	13-:20#	13-:20#	13--05
	13--05#	13--05#	13->88	13->88#	13->88#	13-@38	13-@38#	13-@38#	13-C24	13-C24#	13-C24#	13-E49	13-E49#	13-E49#
	13-F81	13-F81#	13-F81#	13-G99	13-G99#	13-G99#	13-154	13-154#	13-154#	13-K05	13-K05#	13-K05#	13-L50	13-L50#
	13-L50#	13-M95	13-M95#	13-M95#	13-046	13-046#	13-046#							
AENV	6-0	6-0												

AENVN	6-0	6-0												
AFATAL	6-0	6-0												
ALL	11-69	50-5#												
AMADR1	6-0	6-0												
AMADR2	6-0	6-0												
AMADR3	6-0	6-0												
AMADR4	6-0	6-0												
AMAMS1	6-0	6-0												
AMAMS2	6-0	6-0												
AMAMS3	6-0	6-0												
AMAMS4	6-0	6-0												
AMSGAD	6-0	6-0												
AMSLG	6-0	6-0												
AMSGTY	6-0	6-0												
AMTYP1	6-0	6-0												
AMTYP2	6-0	6-0												
AMTYP3	6-0	6-0												
AMTYP4	6-0	6-0												
AOE	4-572#	4-583	25-311	25-326	25-341	25-356	25-360	26-117	26-121	51-77	51-78	51-81	51-82	51-85
	51-86													
APASS	6-0	6-0												
APE	4-750#													
APRIOR	6-0													
APTC SU	41-1	49-1#												
APTENV	41-1	43-1	49-1	49-1#										
APTSIZ	10-23	49-1#												
APTSPO	41-1	49-1	49-1#											
ARGS	13-41	13-41#	13-52#	13-68#	13-71#	13-73#	13-76#	13-80#	13-93	13-93#	13-110#	13-113#	13-115#	13-118#
	13-122#	13-124#	13-126#	13-139	13-139#	13-155#	13-158#	13-160#	13-163#	13-167#	13-169#	13-171#	13-175#	13-188
	13-188#	13-199#	13-215#	13-218#	13-220#	13-223#	13-227#	13-240	13-240#	13-257#	13-260#	13-262#	13-265#	13-269#
	13-271#	13-273#	13-286	13-286#	13-301#	13-304#	13-306#	13-309#	13-313#	13-315#	13-317#	13-330	13-330#	13-341#
	13-356#	13-359#	13-361#	13-364#	13-368#	13-381	13-381#	13-398#	13-401#	13-403#	13-406#	13-410#	13-412#	13-414#
	13-429	13-429#	13-444#	13-447#	13-449#	13-452#	13-456#	13-461#	13-489#	13-505	13-505#	13-516#	13-534#	13-537#
	13-539#	13-542#	13-546#	13-560	13-560#	13-577#	13-580#	13-582#	13-585#	13-589#	13-591#	13-593#	13-606	13-606#
	13-622#	13-625#	13-627#	13-630#	13-634#	13-636#	13-638#	13-640#	13-653	13-653#	13-664#	13-682#	13-685#	13-687#
	13-690#	13-694#	13-708	13-708#	13-725#	13-728#	13-730#	13-733#	13-737#	13-739#	13-741#	13-754	13-754#	13-769#
	13-772#	13-774#	13-777#	13-781#	13-783#	13-785#	13-798	13-798#	13-809#	13-827#	13-830#	13-832#	13-835#	13-839#
	13-852	13-852#	13-869#	13-872#	13-874#	13-877#	13-881#	13-883#	13-885#	13-900	13-900#	13-915#	13-918#	13-920#
	13-923#	13-927#	13-931#	13-961#	13-977	13-977#	13-988#	13-:06#	13-:09#	13-:11#	13-:14#	13-:18#	13-:40	13-:40#
	13-:55#	13-:58#	13-:60#	13-:63#	13-:67#	13-:69#	13-:71#	13-:84	13-:84#	13-:99#	13-:02#	13-:04#	13-:07#	13-:11#
	13-:13#	13-:15#	13-:28	13-:28#	13-:40#	13-:58#	13-:61#	13-:63#	13-:66#	13-:70#	13-:83	13-:83#	13-:88#	13-:91#
	13-:93#	13-:96#	13-<00#	13-<18#	13-<21#	13-<24#	13-<28#	13-<30#	13-<32#	13-<45	13-<45#	13-<50#	13-<53#	13-<55#
	13-<58#	13-<62#	13-<79#	13-<82#	13-<85#	13-<89#	13-<91#	13-<93#	13-<95#	13-=20#	13-=29	13-=29#	13-=43#	13-=46#
	13-=48#	13-=51#	13-=55#	13-=76	13-=76#	13-=80#	13-=83#	13-=85#	13-=88#	13-=92#	13->07#	13->10#	13->12#	13->15#
	13->19#	13->21#	13->23#	13->36	13->36#	13->40#	13->43#	13->45#	13->48#	13->52#	13->67#	13->70#	13->72#	13->75#
	13->79#	13->81#	13->83#	13-?03#	13-?12	13-?12#	13-?25#	13-?28#	13-?30#	13-?33#	13-?37#	13-?58	13-?58#	13-?73#
	13-?76#	13-?78#	13-?81#	13-?85#	13-?87#	13-?89#	13-a02	13-a02#	13-a17#	13-a20#	13-a22#	13-a25#	13-a29#	13-a31#
	13-a33#	13-a48	13-a48#	13-a59#	13-a83#	13-a86#	13-a88#	13-a91#	13-a95#	13-A08	13-A08#	13-A25#	13-A28#	13-A30#
	13-A33#	13-A37#	13-A54#	13-A70#	13-A82#	13-A91#	13-B03	13-B03#	13-B20#	13-B23#	13-B25#	13-B28#	13-B32#	13-B47#
	13-B63#	13-B75#	13-B84#	13-C07	13-C07#	13-C18#	13-C34	13-C34#	13-C45#	13-C71#	13-C74#	13-C76#	13-C79#	13-C83#
	13-C96	13-C96#	13-D13#	13-D16#	13-D18#	13-D21#	13-D25#	13-D44#	13-D46#	13-D59	13-D59#	13-D75#	13-D78#	13-D80#
	13-D83#	13-D87#	13-E06#	13-E08#	13-E10#	13-E32	13-E32#	13-E43#	13-E63#	13-E70	13-E70#	13-E91#	13-E94#	13-E96#
	13-E99#	13-F03#	13-F08#	13-F16#	13-F29	13-F29#	13-F51#	13-F54#	13-F56#	13-F59#	13-F63#	13-F68#	13-F76#	13-F96
	13-F96#	13-G15#	13-G18#	13-G21#	13-G31#	13-G34#	13-G36#	13-G38#	13-G51	13-G51#	13-G70#	13-G73#	13-G76#	13-G87#
	13-G90#	13-G92#	13-G94#	13-H06	13-H06#	13-H17#	13-H33#	13-H36#	13-H38#	13-H41#	13-H45#	13-H59	13-H59#	13-H77#
	13-H80#	13-H82#	13-H85#	13-H89#	13-H91#	13-H95#	13-I08	13-I08#	13-I26#	13-I29#	13-I31#	13-I34#	13-I38#	13-I40#
	13-I44#	13-I46#	13-I60	13-I60#	13-I72#	13-I88#	13-I91#	13-I93#	13-I96#	13-J00#	13-J14	13-J14#	13-J35#	13-J38#

	13-J40#	13-J43#	13-J47#	13-J49#	13-J53#	13-J66	13-J66#	13-J82#	13-J85#	13-J87#	13-J90#	13-J94#	13-J96#	13-K00#
	13-K13	13-K13#	13-K24#	13-K40#	13-K43#	13-K45#	13-K48#	13-K52#	13-K65	13-K65#	13-K82#	13-K85#	13-K87#	13-K90#
	13-K94#	13-K96#	13-K98#	13-L11	13-L11#	13-L27#	13-L30#	13-L32#	13-L35#	13-L39#	13-L41#	13-L43#	13-L45#	13-L58
	13-L58#	13-L69#	13-L85#	13-L88#	13-L90#	13-L93#	13-L97#	13-M10	13-M10#	13-M27#	13-M30#	13-M32#	13-M35#	13-M39#
	13-M41#	13-M43#	13-M56	13-M56#	13-M72#	13-M75#	13-M77#	13-M80#	13-M84#	13-M86#	13-M88#	13-M90#	13-N03	13-N03#
	13-N15#	13-N33#	13-N36#	13-N38#	13-N41#	13-N45#	13-N58	13-N58#	13-N75#	13-N78#	13-N80#	13-N83#	13-N87#	13-N89#
	13-N91#	13-004	13-004#	13-020#	13-023#	13-025#	13-028#	13-032#	13-034#	13-036#	13-038#	13-053	13-053#	13-064#
	13-020#	13-083#	13-085#	13-088#	13-092#	13-P05	13-P05#	13-P17#	13-P23#	13-P26#	13-P33#	13-P36#	13-P38#	13-P41#
	13-P45#	13-P47#	13-P49#	13-P61	13-P61#	13-P74#	13-P83#	13-P86#	13-P94#	13-P97#	13-P99#	13-Q02#	13-Q06#	13-Q08#
	13-Q10#													
ASND	7-0#	16-299*	16-331	16-358*	16-359	16-361*	16-362*	16-363	16-365*	16-397				
ASND	7-0#	16-298*	16-330	16-366*	16-367	16-369*	16-396							
ASWREG	6-0	6-0												
ATA	4-552#	25-138	25-139	25-141	29-156	29-157	29-192	29-195	33-145	33-146	33-185	33-188	51-58	51-59
	51-60	51-63	51-64	51-67	51-68	51-69	51-70	51-71	51-72	51-73	51-74	51-75	51-76	51-79
	51-80	51-83	51-84	51-87	51-88									
ATESTN	6-0	6-0												
ATNMSK	4-589#	34-55												
ATNTBL	10-78	10-117	10-121	11-97	12-17	52-3#								
AUNIT	6-0	6-0												
AUSWR	6-0	6-0												
AVECT1	4-773#	6-0	6-0											
AVECT2	6-0	6-0												
BACK	13-41	13-41#	13-52	13-52#	13-68	13-68#	13-71#	13-73#	13-76	13-76#	13-80	13-80#	13-80#	13-93
	13-93#	13-110	13-110#	13-113#	13-115#	13-118	13-118#	13-122	13-122#	13-122#	13-124	13-124#	13-124#	13-126
	13-126#	13-126#	13-139	13-139#	13-155	13-155#	13-158#	13-160#	13-163	13-163#	13-167	13-167#	13-167#	13-169
	13-169#	13-169#	13-171	13-171#	13-171#	13-175	13-175#	13-188	13-188#	13-199	13-199#	13-215	13-215#	13-218#
	13-220#	13-223	13-223#	13-227	13-227#	13-227#	13-240	13-240#	13-257	13-257#	13-260#	13-262#	13-265	13-265#
	13-269	13-269#	13-269#	13-271	13-271#	13-271#	13-273	13-273#	13-273#	13-286	13-286#	13-301	13-301#	13-304#
	13-306#	13-309	13-309#	13-313	13-313#	13-313#	13-315	13-315#	13-315#	13-317	13-317#	13-317#	13-330	13-330#
	13-341	13-341#	13-356	13-356#	13-359#	13-361#	13-364	13-364#	13-368	13-368#	13-368#	13-381	13-381#	13-398
	13-398#	13-401#	13-403#	13-406	13-406#	13-410	13-410#	13-410#	13-412	13-412#	13-412#	13-414	13-414#	13-414#
	13-429	13-429#	13-444	13-444#	13-447#	13-449#	13-452	13-452#	13-456	13-456#	13-456#	13-461	13-461#	13-461#
	13-489	13-489#	13-489#	13-505	13-505#	13-516	13-516#	13-534	13-534#	13-537#	13-539#	13-542	13-542#	13-546
	13-546#	13-546#	13-560	13-560#	13-577	13-577#	13-580#	13-582#	13-585	13-585#	13-589	13-589#	13-589#	13-591
	13-591#	13-591#	13-593	13-593#	13-593#	13-606	13-606#	13-622	13-622#	13-625#	13-627#	13-630	13-630#	13-634
	13-634#	13-634#	13-636	13-636#	13-636#	13-638	13-638#	13-638#	13-640	13-640#	13-653	13-653#	13-664	13-664#
	3-682	13-682#	13-685#	13-687#	13-690	13-690#	13-694	13-694#	13-694#	13-708	13-708#	13-725	13-725#	13-728#
	13-730#	13-733	13-733#	13-737	13-737#	13-737#	13-739	13-739#	13-739#	13-741	13-741#	13-741#	13-754	13-754#
	13-769	13-769#	13-772#	13-774#	13-777	13-777#	13-781	13-781#	13-781#	13-783	13-783#	13-783#	13-785	13-785#
	13-785#	13-798	13-798#	13-809	13-809#	13-827	13-827#	13-830#	13-832#	13-835	13-835#	13-839	13-839#	13-839#
	13-852	13-852#	13-869	13-869#	13-872#	13-874#	13-877	13-877#	13-881	13-881#	13-881#	13-883	13-883#	13-883#
	13-885	13-885#	13-885#	13-900	13-900#	13-915	13-915#	13-918#	13-920#	13-923	13-923#	13-927	13-927#	13-927#
	13-931	13-931#	13-931#	13-961	13-961#	13-961#	13-977	13-977#	13-988	13-988#	13-:06	13-:06#	13-:09#	13-:11#
	13-:14	13-:14#	13-:18	13-:18#	13-:18#	13-:40	13-:40#	13-:55	13-:55#	13-:58#	13-:60#	13-:63	13-:63#	13-:67
	13-:67#	13-:67#	13-:69	13-:69#	13-:69#	13-:71	13-:71#	13-:71#	13-:84	13-:84#	13-:99	13-:99#	13-:02#	13-:04#
	13-:07	13-:07#	13-:11	13-:11#	13-:11#	13-:13	13-:13#	13-:13#	13-:15	13-:15#	13-:15#	13-:28	13-:28#	13-:40
	13-:40#	13-:58	13-:58#	13-:61#	13-:63#	13-:66	13-:66#	13-:70	13-:70#	13-:70#	13-:83	13-:83#	13-:88	13-:88#
	13-:91#	13-:93#	13-:96	13-:96#	13-<00	13-<00#	13-<00#	13-<18	13-<18#	13-<21#	13-<24	13-<24#	13-<28	13-<28#
	13-<28#	13-<30	13-<30#	13-<30#	13-<32	13-<32#	13-<32#	13-<45	13-<45#	13-<50	13-<50#	13-<53#	13-<55#	13-<58
	13-<58#	13-<62	13-<62#	13-<62#	13-<79	13-<79#	13-<82#	13-<85	13-<85#	13-<89	13-<89#	13-<89#	13-<91	13-<91#
	13-<91#	13-<93	13-<93#	13-<93#	13-<95	13-<95#	13-20	13-20#	13-29	13-29#	13-43	13-43#	13-46#	13-48#
	13-51	13-51#	13-55	13-55#	13-55#	13-76	13-76#	13-80	13-80#	13-83#	13-85#	13-88	13-88#	13-92
	13-92#	13-92#	13->07	13->07#	13->10#	13->12#	13->15	13->15#	13->19	13->19#	13->19#	13->21	13->21#	13->21#
	13->23	13->23#	13->23#	13->36	13->36#	13->40	13->40#	13->43#	13->45#	13->48	13->48#	13->52	13->52#	13->52#
	13->7	13->67#	13->70#	13->72#	13->75	13->75#	13->79	13->79#	13->79#	13->81	13->81#	13->81#	13->83	13->83#
	13->83#	13-?03	13-?03#	13-?12	13-?12#	13-?25	13-?25#	13-?28#	13-?30#	13-?33	13-?33#	13-?37	13-?37#	13-?37#

	13-258	13-253#	13-273	13-273#	13-276#	13-278#	13-281	13-281#	13-285	13-285#	13-285#	13-287	13-287#	13-287#
	13-289	13-289#	13-289#	13-202	13-202#	13-217	13-217#	13-220#	13-222#	13-225	13-225#	13-229	13-229#	13-229#
	13-231	13-231#	13-231#	13-233	13-233#	13-233#	13-248	13-248#	13-259	13-259#	13-283	13-283#	13-286#	13-288#
	13-291	13-291#	13-295	13-295#	13-295#	13-A08	13-A08#	13-A25	13-A25#	13-A28#	13-A30#	13-A33	13-A33#	13-A37
	13-A37#	13-A37#	13-A54	13-A54#	13-A54#	13-A70	13-A70#	13-A70#	13-A82	13-A82#	13-A82#	13-A91	13-A91#	13-A91#
	13-B03	13-B03#	13-B20	13-B20#	13-B23#	13-B25#	13-B28	13-B28#	13-B32	13-B32#	13-B32#	13-B47	13-B47#	13-B47#
	13-B63	13-B63#	13-B63#	13-B75	13-B75#	13-B75#	13-B84	13-B84#	13-B84#	13-C07	13-C07#	13-C18	13-C18#	13-C34
	13-C34#	13-C45	13-C45#	13-C71	13-C71#	13-C74#	13-C76#	13-C79	13-C79#	13-C83	13-C83#	13-C83#	13-C96	13-C96#
	13-D13	13-D13#	13-D16#	13-D18#	13-D21	13-D21#	13-D25	13-D25#	13-D25#	13-D44	13-D44#	13-D44#	13-D46	13-D46#
	13-D46#	13-D59	13-D59#	13-D75	13-D75#	13-D78#	13-D80#	13-D83	13-D83#	13-D87	13-D87#	13-D87#	13-E06	13-E06#
	13-E06#	13-E08	13-E08#	13-E08#	13-E10	13-E10#	13-E32	13-E32#	13-E43	13-E43#	13-E63	13-E63#	13-E70	13-E70#
	13-E91	13-E91#	13-E94#	13-E96#	13-E99	13-E99#	13-F03	13-F03#	13-F03#	13-F08	13-F08#	13-F08#	13-F16	13-F16#
	13-F16#	13-F29	13-F29#	13-F51	13-F51#	13-F54#	13-F56#	13-F59	13-F59#	13-F63	13-F63#	13-F63#	13-F68	13-F68#
	13-F68#	13-F76	13-F76#	13-F76#	13-F96	13-F96#	13-G15	13-G15#	13-G18#	13-G21	13-G21#	13-G31#	13-G34	13-G34#
	13-G36	13-G36#	13-G36#	13-G38	13-G38#	13-G38#	13-G51	13-G51#	13-G70	13-G70#	13-G73#	13-G76	13-G76#	13-G87#
	13-G90	13-G90#	13-G92	13-G92#	13-G92#	13-G94	13-G94#	13-G94#	13-H06	13-H06#	13-H17	13-H17#	13-H33	13-H33#
	13-H36#	13-H38#	13-H41	13-H41#	13-H45	13-H45#	13-H45#	13-H59	13-H59#	13-H77	13-H77#	13-H80#	13-H82#	13-H85
	13-H85#	13-H89	13-H89#	13-H89#	13-H91	13-H91#	13-H91#	13-H95	13-H95#	13-H95#	13-I08	13-I08#	13-I26	13-I26#
	13-I29#	13-I31#	13-I34	13-I34#	13-I38	13-I38#	13-I38#	13-I40	13-I40#	13-I40#	13-I44	13-I44#	13-I44#	13-I46
	13-I46#	13-I60	13-I60#	13-I72	13-I72#	13-I88	13-I88#	13-I91#	13-I93#	13-I96	13-I96#	13-J00	13-J00#	13-J00#
	13-J14	13-J14#	13-J35	13-J35#	13-J38#	13-J40#	13-J43	13-J43#	13-J47	13-J47#	13-J47#	13-J49	13-J49#	13-J49#
	13-J53	13-J53#	13-J53#	13-J66	13-J66#	13-J82	13-J82#	13-J85#	13-J87#	13-J90	13-J90#	13-J94	13-J94#	13-J94#
	13-J96	13-J96#	13-J96#	13-K00	13-K00#	13-K00#	13-K13	13-K13#	13-K24	13-K24#	13-K40	13-K40#	13-K43#	13-K45#
	13-K48	13-K48#	13-K52	13-K52#	13-K52#	13-K65	13-K65#	13-K82	13-K82#	13-K85#	13-K87#	13-K90	13-K90#	13-K94
	13-K94#	13-K94#	13-K96	13-K96#	13-K96#	13-K98	13-K98#	13-K98#	13-L11	13-L11#	13-L27	13-L27#	13-L30#	13-L32#
	13-L35	13-L35#	13-L39	13-L39#	13-L39#	13-L41	13-L41#	13-L41#	13-L43	13-L43#	13-L43#	13-L45	13-L45#	13-L58
	13-L58#	13-L69	13-L69#	13-L85	13-L85#	13-L88#	13-L90#	13-L93	13-L93#	13-L97	13-L97#	13-L97#	13-M10	13-M10#
	13-M27	13-M27#	13-M30#	13-M32#	13-M35	13-M35#	13-M39	13-M39#	13-M39#	13-M41	13-M41#	13-M41#	13-M43	13-M43#
	13-M43#	13-M56	13-M56#	13-M72	13-M72#	13-M75#	13-M77#	13-M80	13-M80#	13-M84	13-M84#	13-M84#	13-M86	13-M86#
	13-M86#	13-M88	13-M88#	13-M88#	13-M90	13-M90#	13-N03	13-N03#	13-N15	13-N15#	13-N33	13-N33#	13-N36#	13-N38#
	13-N41	13-N41#	13-N45	13-N45#	13-N45#	13-N58	13-N58#	13-N75	13-N75#	13-N78#	13-N80#	13-N83	13-N83#	13-N87
	13-N87#	13-N87#	13-N89	13-N89#	13-N89#	13-N91	13-N91#	13-N91#	13-004	13-004#	13-020	13-020#	13-023#	13-025#
	13-028	13-028#	13-032	13-032#	13-032#	13-034	13-034#	13-034#	13-036	13-036#	13-036#	13-038	13-038#	13-053
	13-053#	13-064	13-064#	13-080	13-080#	13-083#	13-085#	13-088	13-088#	13-092	13-092#	13-092#	13-P05	13-P05#
	13-P17	13-P17#	13-P23	13-P23#	13-P26#	13-P33	13-P33#	13-P36#	13-P38#	13-P41	13-P41#	13-P45	13-P45#	13-P45#
	13-P47	13-P47#	13-P47#	13-P49	13-P49#	13-P49#	13-P61	13-P61#	13-P74	13-P74#	13-P83	13-P83#	13-P86#	13-P94
	13-P94#	13-P97#	13-P99#	13-Q02	13-Q02#	13-Q06	13-Q06#	13-Q06#	13-Q08	13-Q08#	13-Q08#	13-Q10	13-Q10#	13-Q10#
BADSCT	13-52	13-199	13-341	13-516	13-664	13-809	13-988	13-:40	13-=20	13-?03	13-a59	13-c45	13-E63	13-H17
	13-172	13-K24	13-L69	13-N15	13-064	16-33#								
BADTMO	10-3#	10-25												
BAI	4-739#	26-28												
B800	4-670#													
B801	4-670#													
B802	4-670#													
B803	4-670#													
B804	4-670#													
B805	4-670#													
B806	4-670#													
B807	4-670#													
B808	4-670#													
B809	4-670#													
BIT0	4-484#	23-18												
BIT00	4-484	4-484#	4-496	4-544	4-562	4-581	4-618	4-636	4-670	4-742	4-760	42-1	42-1	43-1
	43-1													
BIT01	4-484	4-484#	4-495	4-543	4-580	4-617	4-635	4-670	4-742	4-759				
BIT02	4-484	4-484#	4-494	4-542	4-579	4-616	4-634	4-670	4-742	4-758				
BIT03	4-484	4-484#	4-493	4-541	4-578	4-615	4-633	4-670	4-681	4-739	4-757			

BIT04	4-484	4-484#	4-492	4-540	4-577	4-632	4-670	4-738						
BIT05	4-484	4-484#	4-491	4-576	4-614	4-631	4-670	4-737						
BIT06	4-484	4-484#	4-561	4-575	4-597	4-613	4-630	4-670	4-723	4-736	4-756			
BIT07	4-484	4-484#	4-560	4-574	4-596	4-612	4-629	4-653	4-670	4-680	4-722	4-735		
BIT08	4-484	4-484#	4-537	4-559	4-573	4-595	4-611	4-628	4-670	4-721	4-734	42-1		
BIT09	4-484	4-484#	4-536	4-558	4-572	4-594	4-610	4-627	4-670	4-720	4-733	42-1	42-1	43-1
	43-1													
BIT11	4-484#	16-304	17-26	26-48										
BIT10	4-484#	4-535	4-557	4-571	4-593	4-609	4-626	4-652	4-667	4-679	4-719	4-732	4-755	43-1
BIT11	4-484#	4-490	4-534	4-556	4-570	4-608	4-625	4-643	4-651	4-666	4-678	4-731	4-754	15-165
	42-1													
BIT12	4-484#	4-533	4-555	4-569	4-607	4-624	4-650	4-665	4-677	4-730	4-753	15-111		
BIT13	4-484#	4-554	4-568	4-606	4-623	4-642	4-664	4-676	4-718	4-729	4-752	43-1		
BIT14	4-484#	4-553	4-567	4-605	4-622	4-641	4-663	4-675	4-686	4-717	4-728	4-751	15-73	42-1
BIT15	4-484#	4-552	4-566	4-604	4-621	4-640	4-662	4-674	4-685	4-716	4-727	4-750	18-33	18-90
BIT2	4-484#	23-18												
BIT3	4-484#	15-196	35-320	35-407										
BIT4	4-484#	15-141	35-426											
BIT5	4-484#													
BIT6	4-484#	15-87												
BIT7	4-484#	10-71	23-24											
BIT8	4-484#													
BIT9	4-484#													
BLNKS1	50-40#													
BLNKS2	10-132	11-34	11-48	50-39#										
BLNKS3	50-38#													
BLNKS4	10-82	50-37#												
BOTADR	44-81*	44-99*	44-102	44-117	44-161#									
BCTFLG	44-62*	44-109*	44-112	44-115*	44-162#									
BPTVEC	4-484#													
BSE	4-674#	13-A79	13-A85	13-B72	13-B78	13-D35	13-D37	13-D97	13-D99	16-225	25-431			
BUFFER	16-49*	16-284	62-2#	62-16										
BUFONE	13-47	13-98	13-175	13-194	13-245	13-336	13-387	13-511	13-565	13-640	13-659	13-713	13-804	13-858
	13-983	13-:26	13-:35	13-<07	13-<95	13-=16	13-=63	13->99	13-?46	13-254	13-266	13-268*	13-270*	13-A13
	13-C03	13-C40	13-C52	13-C54*	13-C56*	13-D03	13-E10	13-E28	13-E58	13-F89	13-H12	13-H66	13-I46	13-167
	13-J21	13-K19	13-K71	13-L45	13-L64	13-M16	13-M90	13-N10	13-N63	13-O38	13-O59	13-P00	62-3#	
BUF TWO	13-144	13-175	13-426*	13-467	13-474	13-476	13-491*	13-611	13-640	13-897*	13-937	13-946	13-948	13-963*
	13-<69	13-<95	13-B08	13-D65	13-E10	13-F39	13-I14	13-I46	13-L17	13-L45	13-M61	13-M90	13-009	13-038
	62-4#													
CC	4-666#													
CH	4-667#													
CHGADR	7-0#	10-14*	10-17*	11-25	11-27*									
CHRCNT	44-63*	44-86*	44-92*	44-93	44-96*	44-100	44-105*	44-116*	44-163#					
CKSWR	42-1	43-1	43-1	47-1#										
CLKADR	7-0#	22-8*	22-10	22-14*	22-16	23-13								
CLKVCT	7-0#	22-9*	22-15*											
CLR	4-737#	10-84	12-58	30-16										
CLRSTS	15-98	31-22#												
CMNSTA	10-141	12-2#												
CMPEUF	13-175	13-640	13-<95	13-E10	13-I46	13-L45	13-M90	13-O38	18-17#					
CMPEER	27-21#													
CNSLO1	11-32	50-13#												
CNSLO2	11-40	50-14#												
CNSLO3	11-44	50-15#												
CNSLO4	11-54	50-16#												
CNSLO7	11-61	50-17#												
CNSLO8	11-92	50-19#												

CNSL09	50-20#													
CNTCLR	15-76	30-12#												
COMMA	11-86	12-6	12-14	50-7#										
CONT	4-630#													
CPSAVE	42-1	42-1	42-1	42-1*	42-1*	43-1	43-1	43-1	43-1	43-1	43-1#	43-1*	43-1*	44-180
CR	4-484#	11-73	11-84	41-1	41-1	44-84	58-79							
CRLF	4-484#	10-6	10-28	10-28	10-42	10-53	10-59	10-60	12-30	41-1	41-1	50-3	50-5	50-8
	50-9	50-10	50-11	50-11	50-12	50-13	50-14	50-16	50-17	50-18	50-19	50-20	50-21	50-23
	50-33	59-7	59-11	59-12	59-15	59-16	59-18	62-19	62-20	62-21	62-22	62-23	62-24	62-25
	62-26	62-27	62-28	62-29	62-30	62-31	62-32	62-33	62-34	62-35	62-36	62-37	62-38	62-39
	62-40	62-41	62-42	62-43	62-44	62-45	62-46	62-47	62-48	62-49	62-50	62-51	62-52	62-53
	62-54	62-55	62-56	62-57	62-58	62-59	62-60	62-61	62-62	62-63				
CTLFG	7-0#	12-47*	14-22	45-3*										
CYLSK	4-657#													
DBCK	4-604#													
DBEN	4-605#													
DBL	4-755#													
DCK	4-566#	4-583	16-222	18-23	25-469	25-489	35-413	35-416						
DDISP	4-484#	6-0	10-23											
DEBL	4-606#													
DEVSEL	15-62	28-12#												
DISPLA	6-0#	10-23*	10-23*	42-1*	43-1*									
DISPRE	5-1#	10-23												
DLT	4-727#	25-313	35-492	35-495										
DMD	4-618#	4-636#	13-E74	13-F33										
DPE	4-681#	24-137	25-255	29-32	33-31	35-43	35-466	35-469						
DPEHI	4-751#													
DPELO	4-752#													
DPR	4-559#	29-195	31-125	31-126	34-30									
DRIVES	12-3	50-33#												
DRQ	4-643#													
DRVCLR	4-504#	16-87	34-18											
DRVSTS	16-105	34-9#												
DRY	4-560#	24-104	25-38	25-41	25-42	29-195	31-125	31-126	34-30					
DSWR	4-484#	6-0	10-23											
DTASTS	13-80	13-124	13-169	13-227	13-271	13-315	13-368	13-412	13-461	13-546	13-591	13-636	13-694	13-739
	13-783	13-839	13-883	13-931	13-:18	13-:69	13-:13	13-:70	13-<30	13-<91	13-=55	13->21	13->81	13-?37
	13-?87	13-a31	13-a95	13-A54	13-A70	13-A82	13-B47	13-B63	13-B75	13-C83	13-D44	13-E06	13-F08	13-F68
	13-H45	13-H91	13-I40	13-J00	13-J49	13-J96	13-K52	13-K96	13-L41	13-L97	13-M41	13-M86	13-N45	13-N89
	13-034	13-092	13-P47	13-Q08	16-200	35-17#								
DTE	4-569#	4-583	35-279	35-297	35-300									
DTO	4-607#													
DULPRT	4-646#													
DVA	4-490#	10-89	20-43	21-37	24-65	24-68	28-37	31-31	34-17	34-18				
DVC	4-680#	29-127	31-115	33-92	33-130	33-133	33-266	35-232	35-236	35-239	35-284	36-89	36-94	36-97
EARLY	13-K26	53-107#												
EBL	4-623#													
ECH	4-575#	4-583	16-222	18-25	25-452	25-467	25-485	25-491	35-423	51-77	51-78	51-85	51-86	
ECI	4-651#	18-21	25-487	35-420										
ECRC	4-627#													
ED1	56-1	60-1#												
ED110	56-4	60-2#												
ED111	56-5	60-3#												
ED114	56-7	60-5#												
ED223	56-8	60-6#												
ED336	56-10	60-8#												
ED337	56-11	56-12	60-10#											

ED353	56-14	60-11#												
EDT1	8-3	8-6	8-9	8-12	8-15	8-18	8-24	8-27	8-30	8-33	8-36	8-39	8-42	8-45
	8-48	8-51	8-54	8-57	8-60	8-63	8-66	8-69	8-72	8-75	8-78	8-81	8-84	8-87
	8-90	8-93	8-96	8-99	8-102	8-105	8-108	8-111	8-114	8-117	8-120	8-123	8-126	8-130
	8-133	8-136	8-139	8-142	8-146	8-150	8-154	8-160	8-163	8-166	8-169	8-172	8-175	8-178
	8-182	8-185	8-188	8-191	8-194	8-197	8-200	8-203	8-206	8-209	8-212	8-215	8-218	8-236
	8-239	8-242	8-245	8-248	8-251	8-254	8-257	8-260	8-263	8-266	8-269	8-272	8-275	8-278
	8-281	8-284	8-287	8-290	8-293	8-296	8-299	8-302	8-305	8-308	8-311	8-314	8-317	8-320
	8-323	8-326	8-329	8-332	8-335	8-338	8-341	8-344	8-347	8-354	8-357	8-360	8-363	8-366
	8-369	8-372	8-375	8-378	8-381	8-384	8-390	8-393	8-396	8-399	8-402	8-405	8-408	8-411
	8-414	8-417	8-420	8-423	8-432	8-435	8-438	8-441	8-444	8-504	8-507	8-510	8-519	8-522
	8-525	8-531	8-534	8-537	8-540	8-543	8-546	8-549	8-552	8-556	8-559	8-567	8-566	8-569
	8-573	8-577	8-581	8-586	8-590	8-594	8-597	8-600	8-603	8-606	8-609	8-612	8-615	8-618
	8-621	8-624	8-627	8-630	8-633	8-636	8-639	8-642	8-645	8-648	8-651	8-654	8-657	8-660
	8-663	8-666	8-669	8-672	8-675	8-678	8-693	8-696	8-702	8-705	8-708	8-711	8-714	8-723
	56-1#													
EDT110	8-221	56-4#												
EDT111	8-224	8-227	56-5#											
EDT114	8-233	56-7#												
EDT2	8-426	8-429	8-513	8-516	56-2#									
EDT223	8-447	8-528	56-8#											
EDT336	8-351	8-681	8-687	8-690	56-10#									
EDT337	8-684	56-11#												
EDT344	8-699	56-12#												
EDT353	8-720	56-14#												
EECC	4-632#													
EF110	57-4	61-1#												
EF111	57-1	57-5	61-2#											
EF114	57-7	57-8	57-14	61-3#										
EF336	57-10	57-11	57-12	61-4#										
EF337	61-5#													
EFT1	8-3	8-6	8-9	8-12	8-15	8-18	8-24	8-27	8-30	8-33	8-36	8-39	8-42	8-45
	8-48	8-51	8-54	8-57	8-60	8-63	8-66	8-69	8-72	8-75	8-78	8-81	8-84	8-87
	8-90	8-93	8-96	8-99	8-102	8-105	8-108	8-111	8-114	8-117	8-120	8-123	8-126	8-130
	8-133	8-136	8-139	8-142	8-146	8-150	8-154	8-160	8-163	8-166	8-169	8-172	8-175	8-178
	8-182	8-185	8-188	8-191	8-194	8-197	8-200	8-203	8-206	8-209	8-212	8-215	8-218	8-236
	8-239	8-242	8-245	8-248	8-251	8-254	8-257	8-260	8-263	8-266	8-269	8-272	8-275	8-278
	8-281	8-284	8-287	8-290	8-293	8-296	8-299	8-302	8-305	8-308	8-311	8-314	8-317	8-320
	8-323	8-326	8-329	8-332	8-335	8-338	8-341	8-344	8-347	8-354	8-357	8-360	8-363	8-366
	8-369	8-372	8-375	8-378	8-381	8-384	8-390	8-393	8-396	8-399	8-402	8-405	8-408	8-411
	8-414	8-417	8-420	8-423	8-432	8-435	8-438	8-441	8-444	8-504	8-507	8-510	8-519	8-522
	8-525	8-531	8-534	8-537	8-540	8-543	8-546	8-549	8-552	8-556	8-559	8-563	8-566	8-569
	8-573	8-577	8-581	8-586	8-590	8-594	8-597	8-600	8-603	8-606	8-609	8-612	8-615	8-618
	8-621	8-624	8-627	8-630	8-633	8-636	8-639	8-642	8-645	8-648	8-651	8-654	8-657	8-660
	8-663	8-666	8-669	8-672	8-675	8-678	8-693	8-696	8-702	8-705	8-708	8-711	8-714	8-723
	57-1#													
EFT110	8-221	57-4#												
EFT111	8-224	8-227	57-5#											
EFT114	8-233	57-7#												
EFT2	8-426	8-429	8-513	8-516	57-2#									
EFT223	8-447	8-528	57-8#											
EFT336	8-351	8-681	8-687	8-690	57-10#									
EFT337	8-684	57-11#												
EFT344	8-699	57-12#												
EFT353	8-720	57-14#												
EM1	55-1	59-1#												
EM110	55-4	59-2#												

EH111	55-5	59-3#												
EH114	55-7	59-5#												
EH223	55-8	59-6#												
EH256	55-9	59-7#												
EH336	55-11	59-10#												
EH337	55-12	59-11#												
EH344	55-13	59-15#												
EH353	55-15	59-18#												
EHT1	8-3	8-6	8-9	8-12	8-15	8-18	8-24	8-27	8-30	8-33	8-36	8-39	8-42	8-45
	8-48	8-51	8-54	8-57	8-60	8-63	8-66	8-69	8-72	8-75	8-78	8-81	8-84	8-87
	8-90	8-93	8-96	8-99	8-102	8-105	8-108	8-111	8-114	8-117	8-120	8-123	8-126	8-130
	8-133	8-136	8-139	8-142	8-146	8-150	8-154	8-160	8-163	8-166	8-169	8-172	8-175	8-178
	8-182	8-185	8-188	8-191	8-194	8-197	8-200	8-203	8-206	8-209	8-212	8-215	8-218	8-236
	8-239	8-242	8-245	8-248	8-251	8-254	8-257	8-260	8-263	8-266	8-269	8-272	8-275	8-278
	8-281	8-284	8-287	8-290	8-293	8-296	8-299	8-302	8-305	8-308	8-311	8-314	8-317	8-320
	8-323	8-326	8-329	8-332	8-335	8-338	8-341	8-344	8-347	8-354	8-357	8-360	8-363	8-366
	8-369	8-372	8-375	8-378	8-381	8-384	8-390	8-393	8-396	8-399	8-402	8-405	8-408	8-411
	8-414	8-417	8-420	8-423	8-432	8-435	8-438	8-441	8-444	8-504	8-507	8-510	8-519	8-522
	8-525	8-531	8-534	8-537	8-540	8-543	8-546	8-549	8-552	8-556	8-559	8-563	8-566	8-569
	8-573	8-577	8-581	8-586	8-590	8-594	8-597	8-600	8-603	8-606	8-609	8-612	8-615	8-618
	8-621	8-624	8-627	8-630	8-633	8-636	8-639	8-642	8-645	8-648	8-651	8-654	8-657	8-660
	8-663	8-666	8-669	8-672	8-675	8-678	8-693	8-696	8-702	8-705	8-708	8-711	8-714	8-723
	55-1#													
EHT110	8-221	55-4#												
EHT111	8-224	8-227	55-5#											
EHT114	8-233	55-7#												
EHT2	8-426	8-429	8-513	8-516	55-2#									
EHT223	8-447	55-8#												
EHT256	8-528	55-9#												
EHT336	8-351	8-681	8-687	8-690	55-11#									
EHT337	8-684	55-12#												
EHT344	8-699	55-13#												
EHT353	8-720	55-15#												
EMS1	54-3	58-3#												
EMS10	54-7	54-11	58-10#											
EMS100	54-39	54-71	54-176	54-196	58-66#									
EMS101	54-40	54-59	54-174	54-197	58-67#									
EMS102	54-177	54-178	54-181	54-182	54-183	54-184	54-186	54-187	54-188	54-189	54-234	54-235	58-68#	
EMS103	54-56	54-57	54-135	54-141	54-196	54-197	54-198	54-203	54-204	54-205	54-206	54-207	54-208	54-209
	54-210	54-211	54-213	54-214	54-215	54-219	54-220	58-69#						
EMS104	54-136	54-180	54-203	58-70#										
EMS105	54-204	58-71#												
EMS106	54-212	54-232	58-72#											
EMS11	54-12	54-15	54-16	54-17	54-18	54-19	54-20	54-21	54-22	54-23	54-24	54-25	54-26	54-27
	54-28	54-29	54-30	54-31	54-32	54-33	54-43	54-177	58-11#					
EMS110	54-9	54-74	54-77	58-73#										
EMS111	54-74	54-226	58-74#											
EMS112	54-74	58-75#												
EMS113	54-75	54-76	54-78	54-127	58-77#									
EMS114	54-77	58-78#												
EMS115	54-44	54-45	54-48	54-50	54-51	54-52	54-54	54-55	54-60	54-61	54-63	54-66	54-67	54-70
	54-73	54-77	54-107	54-108	54-109	54-110	54-134	54-137	54-139	54-140	54-142	54-179	54-182	54-183
	54-185	54-186	54-187	54-188	54-189	54-190	54-192	54-193	54-194	54-195	54-201	54-205	54-221	54-222
	54-223	54-232	54-234	58-79#										
EMS116	54-74	54-77	54-191	54-225	58-80#									
EMS117	54-77	54-143	54-148	54-149	54-178	54-185	54-201	54-225	54-228	54-229	54-230	54-231	58-81#	
EMS12	54-1	58-12#												

[illegible]

EMS2	54-4	54-5	58-4#														
EMS20	54-34	54-58	54-110	54-118	54-119	54-120	54-121	54-212	54-232	54-238	58-18#						
EMS200	54-128	58-131#															
EMS201	54-128	58-132#															
EMS202	54-122	54-130	54-131	54-132	54-133	54-134	54-136	54-180	58-133#								
EMS203	54-122	54-130	54-131	54-132	54-133	54-174	54-175	54-176	54-237	58-134#							
EMS204	54-131	58-135#															
EMS205	54-55	54-142	54-179	54-233	58-136#												
EMS206	54-55	54-142	54-170	54-179	54-233	58-137#											
EMS207	54-168	54-170	58-138#														
EMS21	54-15	58-19#															
EMS210	54-169	54-170	54-173	58-139#													
EMS211	54-170	58-140#															
EMS212	54-171	54-172	58-141#														
EMS213	54-171	58-142#															
EMS214	54-55	58-143#															
EMS215	54-236	58-144#															
EMS216	54-237	58-145#															
EMS217	54-237	58-146#															
EMS22	54-16	58-20#															
EMS220	54-179	58-147#															
EMS221	54-231	54-238	58-148#														
EMS222	54-233	58-149#															
EMS223	54-232	58-150#															
EMS224	54-121	58-151#															
EMS23	54-17	58-21#															
EMS24	54-18	58-22#															
EMS25	54-19	54-115	58-23#														
EMS26	54-20	54-116	54-173	58-24#													
EMS27	54-16	54-19	54-20	54-115	54-116	54-173	58-25#										
EMS3	54-4	54-76	58-5#														
EMS30	54-17	54-18	54-21	54-22	54-23	54-24	54-25	54-26	54-27	54-28	54-29	54-30	54-31	54-32			
	54-33	54-43	54-122	54-130	54-131	54-132	54-133	54-144	54-145	54-146	54-147	54-174	54-175	54-176			
	54-177	54-199	58-26#														
EMS31	54-21	54-146	54-207	54-208	58-27#												
EMS32	54-22	58-28#															
EMS33	54-23	54-219	58-29#														
EMS34	54-24	54-225	54-226	54-227	58-30#												
EMS35	54-25	54-216	58-31#														
EMS36	54-26	54-213	54-214	54-215	58-32#												
EMS37	54-27	58-33#															
EMS4	54-5	54-75	58-6#														
EMS40	54-28	54-220	58-34#														
EMS41	54-29	54-190	54-191	58-35#													
EMS42	54-30	54-206	58-36#														
EMS43	54-31	54-209	58-37#														
EMS44	54-32	54-211	54-230	58-38#													
EMS45	54-33	54-210	54-229	58-39#													
EMS46	54-43	54-58	54-120	54-145	54-177	54-185	54-199	54-201	58-40#								
EMS47	54-37	54-44	54-45	54-52	54-60	54-61	54-66	54-117	54-118	54-147	54-186	54-187	54-188	54-192			
	54-193	54-222	58-41#														
EMS5	54-6	54-7	58-7#														
EMS50	54-46	54-48	54-64	54-70	54-132	54-135	54-139	54-141	54-181	54-185	54-189	54-2	54-201	54-221			
	58-42#																
EMS51	54-48	54-55	54-70	54-130	54-139	54-142	54-179	54-189	54-221	58-43#							
EMS52	54-55	54-142	54-143	54-148	54-149	54-178	54-179	54-185	54-201	54-228	54-229	54-230	54-231	54-233			
	58-44#																

EMS53	54-37	54-38	54-39	54-40	54-44	54-45	54-46	54-47	54-62	54-64	54-65	54-73	54-140	54-141
	54-181	54-182	54-184	54-186	54-187	54-190	54-191	54-192	54-193	54-194	54-195	54-200	54-202	54-203
	54-204	54-206	54-207	54-216	54-218	54-219	54-220	54-226	58-45#					
EMS54	54-226	58-46#												
EMS55	54-227	58-47#												
EMS56	54-50	54-51	54-54	54-62	54-63	54-67	54-119	54-143	54-144	54-178	54-182	54-183	54-194	54-195
	54-223	54-228	54-234	58-48#										
EMS57	54-190	54-191	54-192	54-193	54-194	54-195	54-196	54-197	54-198	54-199	54-200	54-201	54-202	54-203
	54-204	54-205	54-206	54-207	54-208	54-209	54-210	54-211	54-213	54-214	54-215	54-220	54-221	54-222
	54-223	54-228	54-229	54-230	58-49#									
EMS6	54-6	58-8#												
EMS60	54-224	58-50#												
EMS61	54-224	58-51#												
EMS62	54-224	58-52#												
EMS63	54-42	54-149	54-190	58-53#										
EMS64	54-8	54-14	54-41	54-144	54-145	54-146	54-147	54-199	58-54#					
EMS65	54-41	58-55#												
EMS66	54-34	54-218	58-56#											
EMS67	54-43	54-44	54-45	54-46	54-47	54-48	54-49	54-50	54-51	54-52	54-54	58-57#		
EMS7	54-10	58-9#												
EMS70	54-48	54-50	54-52	54-54	54-66	54-70	54-73	54-107	54-110	54-139	54-182	54-189	54-194	54-205
	54-221	58-58#												
EMS71	54-51	54-59	54-63	54-183	54-195	54-208	58-59#							
EMS72	54-36	54-48	54-51	54-55	54-56	54-57	54-60	54-61	54-61	54-63	54-108	54-109	54-134	54-135
	54-139	54-140	54-142	54-179	54-183	54-187	54-193	54-195	54-232	58-60#				
EMS73	54-49	54-54	54-67	54-68	54-136	54-180	54-188	54-222	54-223	54-234	54-235	58-61#		
EMS74	54-69	54-70	54-134	54-189	54-217	54-221	58-62#							
EMS75	54-35	54-66	54-67	54-137	54-138	54-188	54-222	54-223	54-234	58-63#				
EMS76	54-35	54-36	54-37	54-38	54-39	54-40	58-64#							
EMS77	54-38	54-72	54-148	54-198	58-65#									
EMT1	8-3	54-3#												
EMT10	8-24	54-10#												
EMT100	8-197	54-66#												
EMT101	8-200	54-67#												
EMT102	8-203	54-68#												
EMT103	8-206	54-69#												
EMT104	8-209	54-70#												

EMT127	8-266	54-89#
EMT13	8-33	54-13#
EMT130	8-269	54-90#
EMT131	8-272	54-91#
EMT132	8-275	54-92#
EMT133	8-278	54-93#
EMT134	8-281	54-94#
EMT135	8-284	54-95#
EMT136	8-287	54-96#
EMT137	8-290	54-97#
EMT14	8-36	54-14#
EMT140	8-293	54-98#
EMT141	8-296	54-99#
EMT142	8-299	54-100#
EMT143	8-302	54-101#
EMT144	8-305	54-102#
EMT145	8-308	54-103#
EMT146	8-311	54-104#
EMT147	8-314	54-105#
EMT15	8-39	54-15#
EMT150	8-317	54-106#
EMT151	8-320	54-107#
EMT152	8-323	54-108#
EMT153	8-326	54-109#
EMT154	8-329	54-110#
EMT155	8-332	54-111#
EMT156	8-335	54-112#
EMT157	8-338	54-113#
EMT16	8-42	54-16#
EMT160	8-341	54-114#
EMT161	8-344	54-115#
EMT162	8-347	54-116#
EMT163	54-117#	
EMT164	8-354	54-118#
EMT165	8-357	54-119#
EMT166	8-360	54-120#
EMT167	8-363	54-121#
EMT17	8-45	54-17#
EMT170	8-366	54-122#
EMT171	8-369	54-123#
EMT172	8-372	54-124#
EMT173	8-375	54-125#
EMT174	8-378	54-126#
EMT175	8-381	54-127#
EMT176	8-384	54-128#
EMT177	54-129#	
EMT2	8-6	54-4#
EMT20	8-48	54-18#
EMT200	8-390	54-130#
EMT201	8-393	54-131#
EMT202	8-396	54-132#
EMT203	8-399	54-133#
EMT204	8-402	54-134#
EMT205	8-405	54-135#
EMT206	8-408	54-136#
EMT207	8-411	54-137#
EMT21	8-51	54-19#

EMT210	8-414	54-138#
EMT211	8-417	54-139#
EMT212	8-420	54-140#
EMT213	8-423	54-141#
EMT214	8-426	54-142#
EMT215	8-429	54-143#
EMT216	8-432	54-144#
EMT217	8-435	54-145#
EMT22	8-54	54-20#
EMT220	8-438	54-146#
EMT221	8-441	54-147#
EMT222	8-444	54-148#
EMT223	8-447	54-149#
EMT224	54-150#	
EMT225	54-151#	
EMT226	54-152#	
EMT227	54-153#	
EMT23	8-57	54-21#
EMT230	54-154#	
EMT231	54-155#	
EMT232	54-156#	
EMT233	54-157#	
EMT234	54-158#	
EMT235	54-159#	
EMT236	54-160#	
EMT237	54-161#	
EMT24	8-60	54-22#
EMT240	54-162#	
EMT241	54-163#	
EMT242	54-164#	
EMT243	54-165#	
EMT244	54-166#	
EMT245	54-167#	
EMT246	8-504	54-168#
EMT247	8-507	54-169#
EMT25	8-63	54-23#
EMT250	8-510	54-170#
EMT251	8-513	54-171#
EMT252	8-516	54-172#
EMT253	8-519	54-173#
EMT254	8-522	54-174#
EMT255	8-525	54-175#
EMT256	8-528	54-176#
EMT257	8-531	54-177#
EMT26	8-66	54-24#
EMT260	8-534	54-178#
EMT261	8-537	54-179#
EMT262	8-540	54-180#
EMT263	8-543	54-181#
EMT264	8-546	54-182#
EMT265	8-549	54-183#
EMT266	8-552	54-184#
EMT267	8-555	54-185#
EMT27	8-69	54-25#
EMT270	8-559	54-186#
EMT271	8-563	54-187#
EMT272	8-566	54-188#

EMT273	8-569	54-189#
EMT274	8-573	54-190#
EMT275	8-577	54-191#
EMT276	8-581	54-192#
EMT277	8-586	54-193#
EMT3	8-9	54-5#
EMT30	8-72	54-26#
EMT300	8-590	54-194#
EMT301	8-594	54-195#
EMT302	8-597	54-196#
EMT303	8-600	54-197#
EMT304	8-603	54-198#
EMT305	8-606	54-199#
EMT306	8-609	54-200#
EMT307	8-612	54-201#
EMT31	8-75	54-27#
EMT310	8-615	54-202#
EMT311	8-618	54-203#
EMT312	8-621	54-204#
EMT313	8-624	54-205#
EMT314	8-627	54-206#
EMT315	8-630	54-207#
EMT316	8-633	54-208#
EMT317	8-636	54-209#
EMT32	8-78	54-28#
EMT320	8-639	54-210#
EMT321	8-642	54-211#
EMT322	8-645	54-212#
EMT323	8-648	54-213#
EMT324	8-651	54-214#
EMT325	8-654	54-215#
EMT326	8-657	54-216#
EMT327	8-660	54-217#
EMT33	8-81	54-29#
EMT330	8-663	54-218#
EMT331	8-666	54-219#
EMT332	8-669	54-220#
EMT333	8-672	54-221#
EMT334	8-675	54-222#
EMT335	8-678	54-223#
EMT336	8-351	8-681 54-224#
EMT337	8-684	54-225#
EMT34	8-84	54-30#
EMT340	8-687	54-226#
EMT341	8-690	54-227#
EMT342	8-693	54-228#
EMT343	8-696	54-229#
EMT344	8-699	54-230#
EMT345	8-702	54-231#
EMT346	8-705	54-232#
EMT347	8-708	54-233#
EMT35	8-87	54-31#
EMT350	8-711	54-234#
EMT351	8-714	54-235#
EMT352	8-717	54-236#
EMT353	8-720	54-237#
EMT354	8-723	54-238#

[illegible]

FO	4-495#	25-29												
F1	4-494#	25-29												
F2	4-493#	25-29	26-113											
F3	4-492#	25-29												
F4	4-491#	25-29												
FER	4-577#	4-583	13-A67	13-A72	13-B60	13-B65	13-D28	13-D31	13-D90	13-D93	16-222	25-416	35-348	35-365
FIND	35-368	35-391	35-394											
	13-41	13-41#	13-41#	13-52	13-52#	13-52#	13-68	13-68#	13-68#	13-71	13-71#	13-73	13-73#	13-76
	13-76#	13-76#	13-80	13-80#	13-80#	13-93	13-93#	13-93#	13-110	13-110#	13-110#	13-113	13-113#	13-115
	13-115#	13-118	13-118#	13-118#	13-122	13-122#	13-122#	13-124	13-124#	13-124#	13-126	13-126#	13-126#	13-139
	13-139#	13-139#	13-155	13-155#	13-155#	13-158	13-158#	13-160	13-160#	13-163	13-163#	13-163#	13-167	13-167#
	13-167#	13-169	13-169#	13-169#	13-171	13-171#	13-171#	13-175	13-175#	13-175#	13-188	13-188#	13-188#	13-199
	13-199#	13-199#	13-215	13-215#	13-215#	13-218	13-218#	13-220	13-220#	13-223	13-223#	13-223#	13-227	13-227#
	13-227#	13-240	13-240#	13-240#	13-257	13-257#	13-257#	13-260	13-260#	13-262	13-262#	13-265	13-265#	13-265#
	13-269	13-269#	13-269#	13-271	13-271#	13-271#	13-273	13-273#	13-273#	13-286	13-286#	13-286#	13-301	13-301#
	13-301#	13-304	13-304#	13-306	13-306#	13-309	13-309#	13-309#	13-313	13-313#	13-313#	13-315	13-315#	13-315#
	13-317	13-317#	13-317#	13-330	13-330#	13-330#	13-341	13-341#	13-341#	13-356	13-356#	13-356#	13-359	13-359#
	13-361	13-361#	13-364	13-364#	13-364#	13-368	13-368#	13-368#	13-381	13-381#	13-381#	13-398	13-398#	13-398#
	13-401	13-401#	13-403	13-403#	13-406	13-406#	13-406#	13-410	13-410#	13-410#	13-412	13-412#	13-412#	13-414
	13-414#	13-414#	13-429	13-429#	13-429#	13-444	13-444#	13-444#	13-447	13-447#	13-449	13-449#	13-452	13-452#
	13-452#	13-456	13-456#	13-456#	13-461	13-461#	13-461#	13-489	13-489#	13-489#	13-505	13-505#	13-505#	13-516
	13-516#	13-516#	13-534	13-534#	13-534#	13-537	13-537#	13-539	13-539#	13-542	13-542#	13-542#	13-546	13-546#
	13-546#	13-560	13-560#	13-560#	13-577	13-577#	13-577#	13-580	13-580#	13-582	13-582#	13-585	13-585#	13-585#
	13-589	13-589#	13-589#	13-591	13-591#	13-591#	13-593	13-593#	13-593#	13-606	13-606#	13-606#	13-622	13-622#
	13-622#	13-625	13-625#	13-627	13-627#	13-630	13-630#	13-630#	13-634	13-634#	13-634#	13-636	13-636#	13-636#
	13-638	13-638#	13-638#	13-640	13-640#	13-640#	13-653	13-653#	13-653#	13-664	13-664#	13-664#	13-682	13-682#
	13-682#	13-685	13-685#	13-687	13-687#	13-690	13-690#	13-690#	13-694	13-694#	13-694#	13-708	13-708#	13-708#
	13-725	13-725#	13-725#	13-728	13-728#	13-730	13-730#	13-733	13-733#	13-733#	13-737	13-737#	13-737#	13-739
	13-739#	13-739#	13-741	13-741#	13-741#	13-754	13-754#	13-754#	13-769	13-769#	13-769#	13-772	13-772#	13-774
	13-774#	13-777	13-777#	13-777#	13-781	13-781#	13-781#	13-783	13-783#	13-783#	13-785	13-785#	13-785#	13-798
	13-798#	13-798#	13-809	13-809#	13-809#	13-827	13-827#	13-827#	13-830	13-830#	13-832	13-832#	13-835	13-835#
	13-835#	13-839	13-839#	13-839#	13-852	13-852#	13-852#	13-869	13-869#	13-869#	13-872	13-872#	13-874	13-874#
	13-877	13-877#	13-877#	13-881	13-881#	13-881#	13-883	13-883#	13-883#	13-885	13-885#	13-885#	13-900	13-900#
	13-900#	13-915	13-915#	13-915#	13-918	13-918#	13-920	13-920#	13-923	13-923#	13-923#	13-927	13-927#	13-927#
	13-931	13-931#	13-931#	13-961	13-961#	13-961#	13-977	13-977#	13-977#	13-988	13-988#	13-988#	13-:06	13-:06#
	13-:06#	13-:09	13-:09#	13-:11	13-:11#	13-:14	13-:14#	13-:14#	13-:18	13-:18#	13-:18#	13-:40	13-:40#	13-:40#
	13-:55	13-:55#	13-:55#	13-:58	13-:58#	13-:60	13-:60#	13-:63	13-:63#	13-:63#	13-:67	13-:67#	13-:67#	13-:69
	13-:69#	13-:69#	13-:71	13-:71#	13-:71#	13-:84	13-:84#	13-:84#	13-:99	13-:99#	13-:99#	13-:02	13-:02#	13-:04
	13-:04#	13-:07	13-:07#	13-:07#	13-:11	13-:11#	13-:11#	13-:13	13-:13#	13-:13#	13-:15	13-:15#	13-:15#	13-:28
	13-:28#	13-:28#	13-:40	13-:40#	13-:40#	13-:58	13-:58#	13-:58#	13-:61	13-:61#	13-:63	13-:63#	13-:66	13-:66#
	13-:66#	13-:70	13-:70#	13-:70#	13-:83	13-:83#	13-:83#	13-:88	13-:88#	13-:88#	13-:91	13-:91#	13-:93	13-:93#
	13-:96	13-:96#	13-:96#	13-:00	13-:00#	13-:00#	13-:18	13-:18#	13-:18#	13-:21	13-:21#	13-:24	13-:24#	13-:24#
	13-:28	13-:28#	13-:28#	13-:30	13-:30#	13-:30#	13-:32	13-:32#	13-:32#	13-:45	13-:45#	13-:45#	13-:50	13-:50#
	13-:50#	13-:53	13-:53#	13-:55	13-:55#	13-:58	13-:58#	13-:58#	13-:62	13-:62#	13-:62#	13-:79	13-:79#	13-:79#
	13-:82	13-:82#	13-:85	13-:85#	13-:85#	13-:89	13-:89#	13-:89#	13-:91	13-:91#	13-:91#	13-:93	13-:93#	13-:93#
	13-:95	13-:95#	13-:95#	13-:20	13-:20#	13-:20#	13-:29	13-:29#	13-:29#	13-:43	13-:43#	13-:43#	13-:46	13-:46#
	13-:48	13-:48#	13-:51	13-:51#	13-:51#	13-:55	13-:55#	13-:55#	13-:76	13-:76#	13-:76#	13-:80	13-:80#	13-:80#
	13-:83	13-:83#	13-:85	13-:85#	13-:88	13-:88#	13-:88#	13-:92	13-:92#	13-:92#	13-:07	13-:07#	13-:07#	13-:10
	13->10#	13->12	13->12#	13->15	13->15#	13->15#	13->19	13->19#	13->19#	13->21	13->21#	13->21#	13->23	13->23#
	13->23#	13->36	13->36#	13->36#	13->40	13->40#	13->40#	13->43	13->43#	13->45	13->45#	13->48	13->48#	13->48#
	13->52	13->52#	13->52#	13->67	13->67#	13->67#	13->70	13->70#	13->72	13->72#	13->75	13->75#	13->75#	13->79
	13->79#	13->79#	13->81	13->81#	13->81#	13->83	13->83#	13->83#	13-?03	13-?03#	13-?03#	13-?12	13-?12#	13-?12#
	13-?25	13-?25#	13-?25#	13-?28	13-?28#	13-?30	13-?30#	13-?33	13-?33#	13-?33#	13-?37	13-?37#	13-?37#	13-?58
	13-?58#	13-?58#	13-?73	13-?73#	13-?73#	13-?76	13-?76#	13-?78	13-?78#	13-?81	13-?81#	13-?81#	13-?85	13-?85#
	13-?85#	13-?87	13-?87#	13-?87#	13-?89	13-?89#	13-?89#	13-?02	13-?02#	13-?02#	13-?17	13-?17#	13-?17#	13-?20
	13-?20#	13-?22	13-?22#	13-?25	13-?25#	13-?25#	13-?29	13-?29#	13-?29#	13-?31	13-?31#	13-?31#	13-?33	13-?33#
	13-?33#	13-?48	13-?48#	13-?48#	13-?59	13-?59#	13-?59#	13-?83	13-?83#	13-?83#	13-?86	13-?86#	13-?88	13-?88#

	13-291	13-291#	13-291#	13-295	13-295#	13-295#	13-A08	13-A08#	13-A08#	13-A25	13-A25#	13-A25#	13-A28	13-A28#
	13-A30	13-A30#	13-A33	13-A33#	13-A33#	13-A37	13-A37#	13-A37#	13-A54	13-A54#	13-A54#	13-A70	13-A70#	13-A70#
	13-A82	13-A82#	13-A82#	13-A91	13-A91#	13-A91#	13-B03	13-B03#	13-B03#	13-B20	13-B20#	13-B20#	13-B23	13-B23#
	13-B25	13-B25#	13-B28	13-B28#	13-B28#	13-B32	13-B32#	13-B32#	13-B47	13-B47#	13-B47#	13-B63	13-B63#	13-B63#
	13-B75	13-B75#	13-B75#	13-B84	13-B84#	13-B84#	13-C07	13-C07#	13-C07#	13-C18	13-C18#	13-C18#	13-C34	13-C34#
	13-C34#	13-C45	13-C45#	13-C45#	13-C71	13-C71#	13-C71#	13-C74	13-C74#	13-C76	13-C76#	13-C79	13-C79#	13-C79#
	13-C83	13-C83#	13-C83#	13-C96	13-C96#	13-C96#	13-D13	13-D13#	13-D13#	13-D16	13-D16#	13-D18	13-D18#	13-D21
	13-D21#	13-D21#	13-D25	13-D25#	13-D25#	13-D44	13-D44#	13-D44#	13-D46	13-D46#	13-D46#	13-D59	13-D59#	13-D59#
	13-D75	13-D75#	13-D75#	13-D78	13-D78#	13-D80	13-D80#	13-D83	13-D83#	13-D83#	13-D87	13-D87#	13-D87#	13-E06
	13-E06#	13-E06#	13-E08	13-E08#	13-E08#	13-E10	13-E10#	13-E10#	13-E32	13-E32#	13-E32#	13-E43	13-E43#	13-E43#
	13-E63	13-E63#	13-E63#	13-E70	13-E70#	13-E70#	13-E91	13-E91#	13-E91#	13-E94	13-E94#	13-E96	13-E96#	13-E99
	13-E99#	13-E99#	13-F03	13-F03#	13-F03#	13-F08	13-F08#	13-F08#	13-F16	13-F16#	13-F16#	13-F29	13-F29#	13-F29#
	13-F51	13-F51#	13-F51#	13-F54	13-F54#	13-F56	13-F56#	13-F59	13-F59#	13-F59#	13-F63	13-F63#	13-F63#	13-F68
	13-F68#	13-F68#	13-F76	13-F76#	13-F76#	13-F96	13-F96#	13-F96#	13-G15	13-G15#	13-G15#	13-G18	13-G18#	13-G21
	13-G21#	13-G21#	13-G31	13-G31#	13-G34	13-G34#	13-G34#	13-G36	13-G36#	13-G36#	13-G38	13-G38#	13-G38#	13-G51
	13-G51#	13-G51#	13-G70	13-G70#	13-G70#	13-G73	13-G73#	13-G76	13-G76#	13-G76#	13-G87	13-G87#	13-G90	13-G90#
	13-G90#	13-G92	13-G92#	13-G92#	13-G94	13-G94#	13-G94#	13-H06	13-H06#	13-H06#	13-H17	13-H17#	13-H17#	13-H33
	13-H33#	13-H33#	13-H36	13-H36#	13-H38	13-H38#	13-H41	13-H41#	13-H41#	13-H45	13-H45#	13-H45#	13-H59	13-H59#
	13-H59#	13-H77	13-H77#	13-H77#	13-H80	13-H80#	13-H82	13-H82#	13-H85	13-H85#	13-H85#	13-H89	13-H89#	13-H89#
	13-H91	13-H91#	13-H91#	13-H95	13-H95#	13-H95#	13-I08	13-I08#	13-I08#	13-I26	13-I26#	13-I26#	13-I29	13-I29#
	13-I31	13-I31#	13-I34	13-I34#	13-I34#	13-I38	13-I38#	13-I38#	13-I40	13-I40#	13-I40#	13-I44	13-I44#	13-I44#
	13-I46	13-I46#	13-I46#	13-I60	13-I60#	13-I60#	13-I72	13-I72#	13-I72#	13-I88	13-I88#	13-I88#	13-I91	13-I91#
	13-I93	13-I93#	13-I96	13-I96#	13-I96#	13-J00	13-J00#	13-J00#	13-J14	13-J14#	13-J14#	13-J35	13-J35#	13-J35#
	13-J38	13-J38#	13-J40	13-J40#	13-J43	13-J43#	13-J43#	13-J47	13-J47#	13-J47#	13-J49	13-J49#	13-J49#	13-J53
	13-J53#	13-J53#	13-J66	13-J66#	13-J66#	13-J82	13-J82#	13-J82#	13-J85	13-J85#	13-J87	13-J87#	13-J90	13-J90#
	13-J90#	13-J94	13-J94#	13-J94#	13-J96	13-J96#	13-K00	13-K00#	13-K00#	13-K13	13-K13#	13-K13#	13-K24	13-K24#
	13-K24#	13-K24#	13-K40	13-K40#	13-K40#	13-K43	13-K43#	13-K45	13-K45#	13-K48	13-K48#	13-K48#	13-K52	13-K52#
	13-K52#	13-K65	13-K65#	13-K65#	13-K82	13-K82#	13-K85	13-K85#	13-K85#	13-K87	13-K87#	13-K90	13-K90#	13-K90#
	13-K94	13-K94#	13-K94#	13-K96	13-K96#	13-K96#	13-K98	13-K98#	13-K98#	13-L11	13-L11#	13-L11#	13-L27	13-L27#
	13-L27#	13-L30	13-L30#	13-L32	13-L32#	13-L35	13-L35#	13-L35#	13-L39	13-L39#	13-L39#	13-L41	13-L41#	13-L41#
	13-L43	13-L43#	13-L43#	13-L45	13-L45#	13-L45#	13-L58	13-L58#	13-L58#	13-L69	13-L69#	13-L69#	13-L85	13-L85#
	13-L85#	13-L88	13-L88#	13-L90	13-L90#	13-L93	13-L93#	13-L93#	13-L97	13-L97#	13-L97#	13-M10	13-M10#	13-M10#
	13-M27	13-M27#	13-M27#	13-M30	13-M30#	13-M32	13-M32#	13-M35	13-M35#	13-M35#	13-M39	13-M39#	13-M39#	13-M41
	13-M41#	13-M41#	13-M43	13-M43#	13-M43#	13-M56	13-M56#	13-M56#	13-M72	13-M72#	13-M72#	13-M75	13-M75#	13-M77
	13-M77#	13-M80	13-M80#	13-M80#	13-M84	13-M84#	13-M84#	13-M86	13-M86#	13-M86#	13-M88	13-M88#	13-M88#	13-M90
	13-M90#	13-M90#	13-N03	13-N03#	13-N03#	13-N15	13-N15#	13-N15#	13-N33	13-N33#	13-N33#	13-N36	13-N36#	13-N38
	13-N38#	13-N41	13-N41#	13-N41#	13-N45	13-N45#	13-N45#	13-N58	13-N58#	13-N58#	13-N75	13-N75#	13-N75#	13-N78
	13-N78#	13-N80	13-N80#	13-N83	13-N83#	13-N83#	13-N87	13-N87#	13-N87#	13-N89	13-N89#	13-N89#	13-N91	13-N91#
	13-N91#	13-004	13-004#	13-004#	13-020	13-020#	13-020#	13-023	13-023#	13-025	13-025#	13-028	13-028#	13-028#
	13-032	13-032#	13-032#	13-034	13-034#	13-034#	13-036	13-036#	13-036#	13-038	13-038#	13-038#	13-053	13-053#
	13-053#	13-064	13-064#	13-064#	13-080	13-080#	13-080#	13-083	13-083#	13-085	13-085#	13-088	13-088#	13-088#
	13-092	13-092#	13-092#	13-P05	13-P05#	13-P05#	13-P17	13-P17#	13-P17#	13-P23	13-P23#	13-P23#	13-P26	13-P26#
	13-P33	13-P33#	13-P33#	13-P36	13-P36#	13-P38	13-P38#	13-P41	13-P41#	13-P41#	13-P45	13-P45#	13-P45#	13-P47
	13-P47#	13-P47#	13-P49	13-P49#	13-P49#	13-P61	13-P61#	13-P61#	13-P74	13-P74#	13-P74#	13-P83	13-P83#	13-P83#
	13-P86	13-P86#	13-P94	13-P94#	13-P94#	13-P97	13-P97#	13-P99	13-P99#	13-Q02	13-Q02#	13-Q02#	13-Q06	13-Q06#
	13-Q06#	13-Q08	13-Q08#	13-Q08#	13-Q10	13-Q10#	13-Q10#							
FMT16	4-650#	13-49	13-196	13-338	13-513	13-661	13-806	13-985	13-:37	13--15	13->98	13-a56	13-A44	13-B37
	13-C02	13-C42	13-D01	13-E27	13-E60	13-F91	13-H14	13-I69	13-K21	13-L66	13-N12	13-061	16-60	17-32
	17-34	18-27	29-58	35-183										
FNCDTB	25-130	51-55#												
FNCMSK	4-497#	34-17	35-341											
GENRUF	13-56	13-203	13-345	13-520	13-668	13-813	13-992	13-:30	13-:44	13-=26	13--67	13-?09	13-?49	13-a63
	13-C05	13-C49	13-E30	13-E67	13-H21	13-I76	13-J25	13-K28	13-L73	13-N19	13-068	17-20#		
GET	13-76	13-118	13-163	13-223	13-265	13-309	13-364	13-406	13-452	13-471	13-542	13-585	13-630	13-690
	13-733	13-777	13-835	13-877	13-923	13-942	13-:14	13-:63	13-:07	13-:66	13-:96	13-<24	13-<58	13-<85
	13-=51	13-=88	13->15	13->48	13->75	13-?33	13-?81	13-a25	13-a91	13-A33	13-B28	13-C79	13-D21	13-D83
	13-E99	13-F59	13-G21	13-G34	13-G76	13-G90	13-H41	13-H85	13-I34	13-I96	13-J43	13-J90	13-K48	13-K90

	13-L35	13-L93	13-M35	13-M80	13-N41	13-N83	13-O28	13-O88	13-P41	13-Q02	15-47	15-90	15-114	15-144
GETBUF	15-168	15-199	16-76	16-97	16-129	16-163	16-192	20-31#						
GETINX	7-0#	20-37												
GETSTS	7-0#	13-471*	13-471*	13-942*	13-942*	19-13	20-38							
	13-71	13-113	13-158	13-218	13-260	13-304	13-359	13-401	13-447	13-537	13-580	13-625	13-685	13-728
	13-772	13-830	13-872	13-918	13-:09	13-:58	13-:02	13-:61	13-:91	13-<53	13-=46	13-=83	13->10	13->43
	13->70	13-?28	13-?76	13-a20	13-a86	13-A28	13-B23	13-C74	13-D16	13-D78	13-E94	13-F54	13-G18	13-G73
	13-H36	13-H80	13-I29	13-I91	13-J38	13-J85	13-K43	13-K85	13-L30	3-L88	13-M30	13-M75	13-N36	13-N78
	13-O23	13-O83	13-P36	13-P97	15-46	16-75	19-10#							
GNS	5-1	5-1	10-6	10-28	10-42	10-53	10-59	10-60	12-30	14-20	14-20	47-1	47-1	47-1
	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1
	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1					
GO	4-496#	13-50	13-100	13-145	13-197	13-247	13-291	13-339	13-388	13-434	13-514	13-567	13-612	13-662
	13-715	13-759	13-807	13-859	13-905	13-986	13-:45	13-:89	13-:38	13-:86	13-<08	13-<48	13-r68	13-=18
	13-=78	13-=97	13->38	13->57	13-?01	13-?63	13-a07	13-a57	13-A15	13-B10	13-C09	13-C43	13-D02	13-D64
	13-E34	13-E79	13-F38	13-F92	13-G05	13-G60	13-H15	13-H67	13-I16	13-I70	13-J22	13-J72	13-K22	13-K72
	13-L16	13-L67	13-M17	13-M62	13-N13	13-N65	13-O10	13-O62	13-P18	13-P28	13-P78	13-P89	15-126	15-181
	16-87	16-119	16-153	16-182	23-21	24-90	24-102	24-107	24-123	25-51	25-54	35-342		
G*SWR	10-28	47-1#												
HCE	4-574#	4-583	13-A51	13-A57	13-B44	13-B50	13-D28	13-D31	13-D90	13-D93	16-222	25-379	25-384	25-399
	25-401	25-414	25-429	35-348	35-378	35-381	35-391	35-394	51-77	51-78	51-81	51-85	51-86	
HCP	4-652#	13-D01	25-377	35-344										
HCRC	4-573#	4-583	16-222	25-386	35-348	35-352	35-355	35-391	35-394					
HELP	11-17	62-18#												
HT	4-484#	41-1	41-1	44-90										
IAE	4-571#	25-283	26-89	26-105	26-132	29-47	29-67	33-24	33-78	33-81	35-172	35-190	51-59	51-69
	51-77	51-78	51-81	51-82	51-85	51-86								
IBSAVE	43-1	43-1	43-1	43-1	43-1	43-1#	43-1*	43-1*	43-1*					
IDXMSK	4-710#	20-51	20-54	21-45	21-48									
IE	4-723#	4-756#												
ILF	4-581#	25-155	25-157	32-51	32-111	32-115	33-24	33-45	33-48	35-128	35-145	35-148	51-58	51-67
	51-68	51-70	51-71	51-72										

ILRG66	4-708#														
ILRG70	4-708#														
ILRG72	4-708#														
ILRG74	4-708#														
ILRG76	4-708#														
IOTVEC	4-484#	10-23*	10-23*												
IPCK0	4-760#														
IPCK1	4-759#														
IPCK2	4-758#														
IPCK3	4-757#														
IR	4-736#	31-56													
IJC	4-677#	13-F05	13-F12	13-F65	13-F71	25-198	25-199	29-85	29-208	33-92	33-99	33-102	33-171	35-111	
	35-115	35-546	36-58	51-58	51-59	51-60	51-62	51-63	51-64	51-67	51-68	51-69	51-70	51-71	
	51-72	51-73	51-74	51-75	51-76	51-77	51-78	51-79	51-80	51-81	51-82	51-83	51-84	51-85	
	51-86	51-87	51-88												
LBC	4-679#	35-232	35-250	35-255											
LBT	4-557#	25-358	26-91	26-93											
LF	4-484#	41-1	41-1	44-88	50-16	58-79									
LODEV	10-128	50-28#													
LS	4-534#	31-80	34-70												
LSC	4-678#	35-232	35-266	35-269											
LST	4-635#	31-80	34-70												
LSTRK	7-0#	12-56*	12-64*	13->96	13-?05	13-165	13-041	16-58	16-363	16-384	17-50	26-62	29-53	35-178	
MCLK	4-608#														
MCPE	4-718#	24-90	24-107	24-120	24-123	35-27	35-30								
MDF	4-613#														
MDPE	4-734#	25-454	35-438	35-441											
MEDENB	7-0#	12-51*	15-125*	16-40	16-283*										
MFGFIL	16-61	16-265*	16-273*	16-322	62-7#										
MI	4-616#														
MIXED	13-L71	13-N17	53-4#												
MOC	4-611#														
MOH	4-642#														
MOL	4-555#	10-109	29-146	29-156	29-157	29-162	29-167	29-195	32-37	32-40	33-68	33-145	33-146	33-153	
	33-158	34-30	35-88	35-252	35-505	35-506	35-528	35-533	36-35	36-36	36-40	36-45			
MRD	4-609#														
MS	4-614#														
MSC	4-617#														
MSDRVS	11-64	50-21#													
MSE	4-685#	13-A46	13-B39	17-29											
MSER	4-612#														
MSGDRV	12-69	50-22#													
MSHELP	11-11	50-8#													
MUR	4-610#														
MWD	4-633#	31-81	34-71												
MWP	4-615#														
MXF	4-733#	25-343	35-50	35-63	35-66										
N	11-19	50-35#													
NDTMSK	4-583#														
NED	4-730#	10-87	24-72	24-76	28-31										
NEM	4-731#	25-328													
NONE	12-7	50-34#													
NOP	4-500#														
NOTAVL	10-116	50-30#													
NOTPRS	10-113	50-29#													
NOTRPM	10-107	50-27#													
NSA	4-640#														

OCC	4-62#													
OFD	4-653#	13-012	13-014											
OFFSET	4-506#	13-P18	13-P78											
OM	4-562#	33-145	33-201	33-204	34-29	35-453	35-456							
ONES	13-518	13-666	13-811	13-:28	13-:42	13-:24	13-:47	13-C47	13-174	13-J23	53-21#			
OPE	4-676#													
OPI	4-568#	25-182	29-140	29-164	32-51	32-69	32-73	33-24	33-61	33-64	33-155	35-83	35-86	35-530
	36-42	51-57	51-58	51-59	51-60	51-61	51-62	51-63	51-64	51-65	51-66	51-67	51-68	51-69
	51-70	51-71	51-72	51-73	51-74	51-75	51-76	51-77	51-78	51-79	51-80	51-81	51-82	51-83
	51-84	51-85	51-86	51-87	51-88									
OR	4-735#													
PACACK	4-510#	16-119												
PAKACK	4-509#	4-510	15-126											
PAR	4-578#	24-135	24-139	24-151	29-30	33-24	33-29	33-34	35-41	35-46				
PAT	4-738#													
PDA	4-628#													
PFECH	44-67	44-172#												
PFECH1	44-172	44-173#												
PFECH2	44-172	44-176#												
PFECH3	44-172	44-179#												
PFECH4	44-172	44-181#												
PGE	4-732#													
PGM	4-558#	34-29												
PHA	4-629#													
PIP	4-554#	13-G23	13-G26	13-G78	13-G81	15-178	16-150	29-156	29-177	29-182	33-145	33-216	33-221	34-29
	35-505	35-511	35-516	36-35	36-72	36-77								
PIRO	4-484#													
PIROVE	4-484#													
PLFS	4-626#													
PRO	4-484#													
PR1	4-484#													
PR2	4-484#													
PR3	4-484#	42-5												
PR4	4-484#													
PR5	4-484#	12-49	42-9											
PR6	4-484#	10-26	13-7	20-40	21-34	22-7	23-11	28-22	30-14					
PR7	4-484#													
PRIERR	13-122	13-167	13-269	13-313	13-410	13-456	13-589	13-634	13-737	13-781	13-881	13-927	13-:67	13-:11
	13-<28	13-<89	13->19	13->79	13-?85	13-@29	13-A37	13-B32	13-D25	13-D87	13-F03	13-F63	13-G36	13-G92
	13-H89	13-138	13-J47	13-J94	13-K94	13-L39	13-M39	13-M84	13-N87	13-032	13-P45	13-Q06	24-39#	
PS	4-484	4-484#												
PSEL	4-719#													
PSW	4-484#													
PUT	13-68	13-110	13-155	13-215	13-257	13-301	13-356	13-398	13-444	13-534	13-577	13-622	13-682	13-725
	13-769	13-827	13-869	13-915	13-:06	13-:55	13-:99	13-:58	13-:88	13-<18	13-<50	13-<79	13-43	13-80
	13->07	13->40	13->67	13-?25	13-?73	13-@17	13-@83	13-A25	13-B20	13-C18	13-C71	13-D13	13-D75	13-E43
	13-E74	13-E91	13-F33	13-F51	13-G00	13-G15	13-G55	13-G70	13-H33	13-H77	13-I26	13-I88	13-J35	13-J82
	13-K40	13-K82	13-L27	13-L85	13-M27	13-M72	13-N33	13-N75	13-O20	13-O80	13-P17	13-P23	13-P33	13-F74
	13-P83	13-P94	15-129	15-184	16-88	16-120	16-154	16-183	21-28#					
PUTBLF	7-0#	16-49	16-284*	21-31										
PUTINX	7-0#	13-59	13-101	13-146	13-206	13-248	13-292	13-347	13-389	13-435	13-523	13-568	13-613	13-671
	13-716	13-760	13-816	13-860	13-906	13-995	13-:46	13-:90	13-:47	13-<09	13-<70	13-33	13-98	13->58
	13-?16	13-?64	13-@08	13-@74	13-A16	13-B11	13-C10	13-C60	13-D04	13-D66	13-E35	13-E74*	13-E74*	13-E81
	13-F33*	13-F33*	13-F41	13-G00*	13-G00*	13-G06	13-G55*	13-G55*	13-G61	13-H24	13-H68	13-I17	13-I79	13-J26
	13-J73	13-K31	13-K73	13-L18	13-L76	13-M18	13-M63	13-N22	13-N66	13-O11	13-O71	13-P09	13-P19	13-P29
	13-P66	13-P79	13-P90	15-127*	15-128*	15-182*	15-183*	16-63	21-32	24-142				
PWRVEC	4-484#	10-23*	10-23*	48-1*	48-1*	48-1*	48-1*	48-1*	48-1*					

RMCS3	4-770#														
RMCS31	7-0#														
RMCS30	7-0#														
RMDA	4-691#	13-61	13-102	13-147	13-208	13-249	13-293	13-349	13-390	13-436	13-525	13-569	13-614	13-673	
	13-717	13-761	13-818	13-861	13-907	13-997	13-:47	13-:91	13-:49	13-:10	13-:71	13-:35	13-:99	13-:59	
	13-?18	13-?65	13-?09	13-?76	13-A17	13-B12	13-C11	13-C62	13-D05	13-D67	13-E36	13-E83	13-F43	13-G07	
	13-G62	13-H26	13-H69	13-I18	13-I81	13-J27	13-J74	13-K33	13-K74	13-L19	13-L78	13-M19	13-M64	13-N24	
	13-N67	13-012	13-073	13-P10	13-P67	16-64									
RMDAI	7-0#	26-137	60-14	60-16											
RMDAO	7-0#	13-46*	13-193*	13-335*	13-510*	13-653*	13-803*	13-982*	13-:33*	13-:13*	13-:14*	13-:22	13-:96*	13-:97*	
	13-?05	13-?53*	13-C39*	13-E57*	13-E65	13-F88*	13-H11*	13-I65*	13-I66*	13-K18*	13-L63*	13-M08*	13-O40*	13-O41	
	13-058*	16-58*	16-232*	16-233	16-235	16-255*	16-299	16-397*	17-24	26-60	26-61	29-53	29-56	29-60	
	35-178	35-181	35-185												
RMDB	4-768#	13-18*	13-19	13-471	13-942	19-20									
RMDBI	7-0#	13-473	13-942*	13-945											
RMDBO	7-0#														
RMDC	4-700#	13-60	13-103	13-149	13-207	13-250	13-295	13-348	13-391	13-438	13-524	13-570	13-616	13-672	
	13-718	13-763	13-817	13-862	13-909	13-996	13-:48	13-:93	13-:48	13-:11	13-:73	13-:34	13-:00	13-:61	
	13-?17	13-?66	13-?11	13-?75	13-A18	13-B14	13-C13	13-C61	13-D06	13-D69	13-E38	13-E84	13-F45	13-G08	
	13-G64	13-H25	13-H70	13-I20	13-I80	13-J28	13-J76	13-K32	13-K75	13-L21	13-L77	13-M20	13-M66	13-N23	
	13-N68	13-014	13-072	13-P11	13-P69	16-65									
RMDCI	7-0#	26-150	60-14	60-16											
RMDCO	7-0#	13-45*	13-192*	13-334*	13-509*	13-657*	13-802*	13-981*	13-:32*	13-:34	13-:85*	13-:05*	13-:47*	13-:67*	
	13-12*	13-:95*	13-?52*	13-C38*	13-E56*	13-F87*	13-H10*	13-I48*	13-I49	13-I64*	13-K17*	13-L62*	13-N07*	13-O57*	
	16-57*	16-298	16-396*	17-23	26-59	29-49	35-174								
RMDS	4-692#	10-86	10-109												
RMDSI	7-0#	13-G23	13-G25	13-G27	13-G78	13-G80	13-G82	15-122	15-178	16-116	16-150	24-104	25-38	25-40	
	25-85	25-102	25-136	25-140	25-196	25-223	25-358	26-92	29-91	29-146	29-155	29-162	29-166	29-168	
	29-177	29-181	29-183	29-192	29-194	29-196	29-206	29-210	29-212	31-124	32-23	32-25	32-27	32-37	
	32-39	32-41	33-68	33-106	33-144	33-153	33-157	33-159	33-169	33-173	33-174	33-185	33-187	33-189	
	33-201	33-203	33-205	33-216	33-220	33-222	34-28	35-88	35-90	35-119	35-252	35-318	35-453	35-455	
	35-457	35-504	35-511	35-515	35-517	35-528	35-532	35-534	35-544	35-548	35-550	36-34	36-40	36-44	
	36-46	36-56	36-60	36-61*	36-62	36-72	36-76	36-78	60-13						
RMDSO	7-0#														
RMDT	4-697#	10-91	12-60	44-26											
RMDTI	7-0#	60-5	60-17												
RMDTO	7-0#														
RMEC1	4-704#														
RMEC1I	7-0#	18-32	60-14												
RMEC1O	7-0#														
RMEC2	4-705#	19-15													
RMEC2I	7-0#	18-53	19-14	31-92	34-94	60-15									
RMEC2O	7-0#														
RMER1	4-693#	13-G00	13-G55	24-143											
RMER1I	7-0#	13-A51	13-A56	13-A67	13-A73	13-B44	13-B49	13-B60	13-B66	13-D28	13-D30	13-D90	13-D92	16-222	
	18-23	18-25	24-135	24-150	24-152	25-80	25-156	25-183	25-226	25-241	25-284	25-361	25-387	25-402	
	25-417	25-470	25-489	25-492	26-89	26-105	26-120	26-132	27-24	29-30	29-37	29-66	29-140	29-143	
	29-164	31-69	32-51	32-55	32-57	32-58	32-69	32-71	32-72	32-83	32-85	32-86	32-97	32-99	
	32-100	32-111	32-113	32-114	33-24	33-29	33-33	33-35	33-45	33-47	33-49	33-61	33-63	33-65	
	33-78	33-80	33-82	33-155	33-232	33-236	33-238	33-240	33-250	33-252	33-254	33-264	33-268	33-270	
	34-41	35-41	35-45	35-47	35-83	35-85	35-87	35-128	35-131	35-133	35-135	35-145	35-147	35-149	
	35-159	35-161	35-163	35-189	35-279	35-282	35-286	35-288	35-297	35-299	35-301	35-311	35-313	35-314	
	35-348	35-352	35-354	35-356	35-365	35-367	35-369	35-378	35-380	35-382	35-391	35-393	35-395	35-413	
	35-415	35-417	35-423	35-479	35-481	35-483	35-530	36-42	60-13						
RMER1O	7-0#	13-G00*	13-G55*	24-139											
RMER2	4-703#														
RMER2I	7-0#	13-A79	13-A84	13-B72	13-B77	13-D34	13-D35	13-D96	13-D97	13-F05	13-F10	13-F11	13-F65	13-F75	

SCOPE	4-484#	13-2	13-33	13-180	13-322	13-497	13-645	13-790	13-969	13-;20	13-=05	13->88	13-a38	13-C24
	13-E49	13-F81	13-G99	13-I54	13-K05	13-L50	13-M95	13-046	14-9					
SCTMSG	13-52	13-199	13-341	13-516	13-664	13-809	13-988	13-;40	13-=20	13-?03	13-a59	13-C45	13-E63	13-H17
	13-172	13-K24	13-L69	13-N15	13-064	50-3#								
SCTMSK	4-599#													
SEARCH	4-513#	25-104	26-5											
SELCERR	13-126	13-171	13-273	13-317	13-414	13-489	13-593	13-638	13-741	13-785	13-885	13-961	13-;71	13-;15
	13-<32	13-<93	13->23	13->83	13-?89	13-a33	13-A91	13-B84	13-D46	13-E08	13-F16	13-F76	13-G38	13-G94
	13-H95	13-I44	13-J53	13-K00	13-K98	13-L43	13-M43	13-M88	13-N91	13-036	13-P49	13-Q10	25-24#	
SEEK	4-502#	13-;86	13-<48	13-=78	13->38									
SEKSTS	13-<00	13-<62	13- 92	13->52	29-19#									
SHUT	14-16	14-20	14-22#	45-6										
SHUT2	45-1	45-1	45-3#											
SIZLLK	12-28	22-3#												
SKI	4-675#	15-176	16-148	29-48	29-51	29-70	29-100	29-102	29-106	29-117	29-179	31-115	33-92	33-116
	33- 19	33-218	35-173	35-176	35-193	35-204	35-206	35-207	35-209	35-221	35-513	36-74	36-89	36-108
	36-111													
SNGPRT	4-645#													
STACK	4-484#	10-23	13-2	13-33	13-180	13-322	13-497	13-645	13-790	13-969	13-;20	13-=05	13->88	13-a38
	13-C24	13-E49	13-F81	13-G99	13-I54	13-K05	13-L50	13-M95	13-046					
STANDA	10-66	11-3#												
START	5-1	10-17#	12-33	12-42	14-27									
START1	5-3	10-14#	13-28											
START2	10-15	10-18#												
STCDRV	36-27#													
STKLMT	4-484#													
STOP	42-1	42-5#												
STSD1	56-1	56-2	56-8	56-10	56-11	56-12	60-13#							
STSD2	56-1	56-2	56-8	56-10	56-11	56-12	60-14#							
STSD3	60-16#													
STSD4	56-1	56-2	56-8	56-10	56-11	56-12	60-17#							
STSF	57-1	57-1	57-1	57-2	57-2	57-2	57-8	57-8	57-8	57-10	57-10	57-10	57-11	57-11
	57-11	57-12	57-12	57-12	61-7#									
STSH1	55-1	55-2	55-8	55-9	55-11	55-12	55-13	59-21#						
STSH2	55-1	55-2	55-8	55-9	55-11	55-12	55-13	59-23#						
STSH3	59-25#													
STSH4	55-1	55-2	55-8	55-9	55-11	55-12	55-13	59-26#						
SW0	4-484#		</											

SW4	4-484#													
SW5	4-484#													
SW6	4-484#													
SW7	4-484#													
SW8	4-484#													
SW9	4-484#													
SWR	6-0#	10-23	10-23	10-23*	10-23*	10-23*	10-28	16-245	42-1	42-1	42-1	42-1	42-1	42-1
	42-1*	42-1*	42-1*	42-1*	43-1	43-1	43-1	43-1	43-1	44-14	45-1	45-1	45-1*	48-1
	48-1*													
SWREG	5-1#	10-23	10-28	45-1	45-1	45-1								
SYSTAT	10-76	50-23#												
TA1	4-537#													
TA16	4-533#	12-64												
TA2	4-536#	12-64												
TA4	4-535#	12-56												
TA8	4-534#													
TADMSK	4-547#	26-66												
TAG	4-664#													
TAGADR	5-9#	6-0												
TAP	4-641#													
TBITVE	4-484#													
TIMOUT	13-73	13-115	13-160	13-220	13-262	13-306	13-361	13-403	13-449	13-539	13-582	13-627	13-687	13-730
	13-774	13-832	13-874	13-920	13-:11	13-:60	13-:04	13-:63	13-:93	13-<21	13-<55	13-<82	13--48	13-=85
	13->12	13->45	13->72	13-?30	13-?78	13-a22	13-a88	13-A30	13-B25	13-C20	13-C76	13-D18	13-D80	13-E45
	13-E96	13-F56	13-G31	13-G87	13-H38	13-H82	13-I31	13-I93	13-J40	13-J87	13-K45	13-K87	13-L32	13-L90
	13-M32	13-M77	13-N38	13-N80	13-025	13-085	13-P26	13-P38	13-P86	13-P99	15-137	15-192	16-96	16-128
	16-162	16-191	23-9#											
TKVEC	4-484#	45-1*	45-1*											
TPVEC	4-484#													
TRAPVE	4-484#	10-23*	10-23*											
TRE	4-717#	16-209	24-90	24-107	24-123	25-106	25-108	26-12						
TRTVEC	4-484#													
TST	4-665#	31-103	34-83											
TST1	13-2#	42-1												
TST10	13-969#	42-1												
TST11	13-:20#	42-1												
TST12	13-=05#	42-1												
TST13	13->88#	42-1												
TST14	13-a38#	42-1												
TST15	13-C24#	42-1												
TST16	13-E49#	42-1												
TST17	13-F81#	42-1												
TST2	13-33#	42-1												
TST20	13-G99#	42-1												
TST21	13-I54#	42-1												
TST22	13-K05#	42-1												
TST23	13-L50#	42-1												
TST24	13-M95#	42-1												
TST25	13-O46#	42-1												
TST3	13-180#	42-1												
TST4	13-322#	42-1												
TST5	13-497#	42-1												
TST6	13-645#	42-1												
TST7	13-790#	42-1												
TSTNMB	44-44*	44-45*	44-47	44-159#										
TSTPRP	13-41	13-93	13-139	13-188	13-240	13-286	13-330	13-381	13-429	13-505	13-560	13-606	13-653	13-708
	13-754	13-798	13-852	13-900	13-977	13-:40	13-:84	13-:28	13-:83	13-<45	13-=29	13-=76	13->36	13-?12

	13-258	13-202	13-248	13-A08	13-B03	13-C07	13-C34	13-C96	13-D59	13-E32	13-E70	13-F29	13-F96	13-G51
	13-H06	13-H59	13-I08	13-I60	13-J14	13-J66	13-K13	13-K65	13-L11	13-L58	13-M10	13-M56	13-N03	13-N58
	13-004	13-053	13-P05	13-P61	15-38#									
YSTQUE	7-0#	12-8	12-9*	12-50	12-59	13-2	13-33	13-180	13-322	13-497	13-645	13-790	13-969	13-;20
	13-05	13->88	13-238	13-C24	13-E49	13-F81	13-G99	13-I54	13-K05	13-L50	13-M95	13-046	14-11	14-13*
	14-17	14-17*	28-24	30-17	31-54	34-53								
TYPBN	44-149	47-1#												
TYPDS	14-20	14-20	44-143	47-1#										
TYPE	10-6	10-28	10-42	10-48	10-53	10-59	10-60	10-76	10-80	10-82	10-107	10-113	10-116	10-120
	10-128	10-131	10-132	10-133	10-140	11-11	11-16	11-17	11-19	11-20	11-28	11-32	11-34	11-40
	11-44	11-48	11-54	11-61	11-62	11-64	11-69	11-75	11-86	11-87	11-92	11-100	12-3	12-6
	12-7	12-14	12-25	12-30	12-68	12-69	13-52	13-199	13-341	13-516	13-664	13-809	13-988	13-;40
	13-20	13-203	13-259	13-C45	13-E63	13-H17	13-I72	13-K24	13-L69	13-N15	13-064	14-20	14-20	14-20
	38-1	39-1	40-1	41-1	43-1	43-1	44-20	44-21	44-39	44-40	44-46	44-51	44-53	44-61
	44-104	44-110	44-114	44-126	44-132	44-134	44-152	44-154	45-1	45-1	45-1	45-1	45-1	45-1
	45-1	45-1	45-1	45-1	45-1	45-1	45-1	45-1	45-1	45-1	45-1	45-1	47-1#	48-1
TYPOC	10-8	11-33	11-47	44-146	45-1	47-1#								
TYPON	47-1#													
TYPOS	10-45	10-56	10-81	12-16	12-70	44-22	44-47	44-52	44-54	47-1#				
U0	4-742#													
U1	4-742#													
U2	4-742#													
UNS	4-567#	13-G00	13-G55	31-70	32-51	32-55	32-59	33-232	33-264	33-269	35-279	35-282	35-287	
UNMSK	4-746#	24-48	24-50											
UNOFF	10-120	50-31#												
UNTON	10-131	50-32#												
UPE	4-729#	25-270												
USE	4-686#	13-A46	13-B39	17-29										
USRFIL	16-253	16-256	16-266*	16-274*	16-325	62-12#								
VV	4-561#	15-122	16-116	25-196	29-91	29-156	29-157	29-206	29-211	32-23	32-26	33-106	33-145	33-146
	33-169	33-175	34-29	35-90	35-119	35-505	35-506	35-544	35-549	36-35	36-36	36-56	36-61	
WC	4-631#	31-80	34-70											
WCD	4-517#	13-291	13-434	13-759	13-905	13-;89	13->57	13-207	13-J72	13-P89				
WCE	4-728#	13-458	13-464	13-929	13-934	25-169	51-77	51-78						
WCEHI	4-753#													
WCELO	4-754#													
WCF	4-576#	4-583	25-240	35-479	35-482									
WCH	4-518#													
WD	4-521#	13-100	13-247	13-388	13-567	13-715	13-859	13-;27	13-;45	13-<08	13-;66	13-;97	13-244	13-263
	13-A15	13-D02	13-E61	13-E79	13-F92	13-G05	13-H67	13-J22	13-K72	13-M17	13-N65			
WH	4-522#	13-50	13-197	13-339	13-514	13-662	13-807	13-986	13-;38	13-;18	13-201	13-257	13-C04	13-C09
	13-C43	13-E29	13-E34	13-H15	13-170	13-K22	13-L67	13-N13	13-062	35-342				
WLE	4-570#	4-583	25-221	25-225	25-238	25-253	25-268	35-279	35-311	35-315	35-323	51-81	51-82	
WRL	4-556#	25-223	35-318											
XSIZ	10-70#	11-71	11-101											
XADP	7-0#	10-33*	10-36*	10-37	10-39*	10-44	- 5	10-124	10-126					
Y	50-36#													
ZEROS	13-54	13-201	13-343	13-990	13-;64	13-207	13-261	13-H19	13-066	53-38#				

SSCMRE	5-10#													
SSCMTM	5-10#	6-0	6-0	6-0	6-0	6-0								
SSESCA	4-484#													
SSNEWI	4-484#	13-2	13-33	13-180	13-322	13-497	13-645	13-790	13-969	13-;20	13-=05	13->88	13-a38	13-C24
	13-E49	13-F81	13-G99	13-I54	13-K05	13-L50	13-M95	13-046						
SSSET	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1#
SSSETM	10-23	10-23#												
SSSKIP	4-484#													
.SACT1	4-476#	5-5												
.SAPTB	4-476#	6-0	6-0#											
.SAPTH	4-476#	5-8												
.SAPTY	4-476#	49-1												
.SCATC	4-472#	5-1												
.SCMTA	4-473#	5-10												
.SEOP	4-473#	14-20												
.SERRO	4-473#	43-1												
.SPOWE	4-475#	48-1												
.SRDDE	4-474#													
.SRDOC	4-474#	46-1												
.SREAD	4-474#	45-1												
.SSAVE	4-475#	37-1												
.SSCOP	4-473#	42-1												
.SSIZE	4-475#													
.STRAP	4-475#	47-1												
.STYPB	4-474#	38-1												
.STYPD	4-474#	39-1												
.STYPE	4-473#	41-1												
.STYPO	4-474#	40-1												
.EQUAT	4-472#	4-484												
.HEADE	4-472#	4-480												
.SETUP	4-472#	4-775												
.SWRHI	4-472#	4-481												
.SWRLO	4-472#	4-481#	4-482											
CALCLR	4-187#													
CALSUB	4-201#	13-41	13-52	13-68	13-71	13-73	13-76	13-80	13-93	13-110	13-113	13-115	13-118	13-122
	13-124	13-126	13-139	13-155	13-158	13-160	13-163	13-167	13-169	13-171	13-175	13-188	13-199	13-215
	13-218	13-220	13-223	13-227	13-240	13-257	13-260	13-262	13-265	13-269	13-271	13-273	13-286	13-301
	13-304	13-306	13-309	13-313	13-315	13-317	13-330	13-341	13-356	13-359	13-361	13-364	13-368	13-381
	13-398	13-401	13-403	13-406	13-410	13-412	13-414	13-429	13-444	13-447	13-449	13-452	13-456	13-461
	13-489	13-505	13-516	13-534	13-537	13-539	13-542	13-546	13-560	13-577	13-580	13-582	13-585	13-589
	13-591	13-593	13-606	13-622	13-625	13-627	13-630	13-634	13-636	13-638	13-640	13-653	13-664	13-682
	13-685	13-687	13-690	13-694	13-708	13-725	13-728	13-730	13-733	13-737	13-739	13-741	13-754	13-769
	13-772	13-774	13-777	13-781	13-783	13-785	13-798	13-809	13-827	13-830	13-832	13-835	13-839	13-852
	13-869	13-872	13-874	13-877	13-881	13-883	13-885	13-900	13-915	13-918	13-920	13-923	13-927	13-931
	13-961	13-977	13-988	13-:06	13-:09	13-:11	13-:14	13-:18	13-:40	13-:55	13-:58	13-:60	13-:63	13-:67
	13-:69	13-:71	13-:84	13-:99	13-:02	13-:04	13-:07	13-:11	13-:13	13-:15	13-:28	13-:40	13-:58	13-:61
	13-:63	13-:66	13-:70	13-:83	13-:88	13-:91	13-:93	13-:96	13-<00	13-<18	13-<21	13-<24	13-<28	13-<30
	13-<32	13-<45	13-<50	13-<53	13-<55	13-<58	13-<62	13-<79	13-<82	13-<85	13-<89	13-<91	13-<93	13-<95
	13-=20	13-=29	13--43	13--46	13-=48	13-=51	13-=55	13-=76	13-=80	13-=83	13-=85	13-=88	13-=92	13->07
	13->10	13->12	13->15	13->19	13->21	13->23	13->36	13->40	13->43	13->45	13->48	13->52	13->67	13->70
	13->72	13->75	13->79	13->81	13->83	13-?03	13-?12	13-?25	13-?28	13-?30	13-?33	13-?37	13-?58	13-?73
	13-?76	13-?78	13-?81	13-?85	13-?87	13-?89	13-a02	13-a17	13-a20	13-a22	13-a25	13-a29	13-a31	13-a33
	13-a48	13-a59	13-a83	13-a86	13-a88	13-a91	13-a95	13-A08	13-A25	13-A28	13-A30	13-A33	13-A37	13-A54
	13-A70	13-A82	13-A91	13-B03	13-B20	13-B23	13-B25	13-B28	13-B32	13-B47	13-B63	13-B75	13-B84	13-C07
	13-C18	13-C34	13-C45	13-C71	13-C74	13-C76	13-C79	13-C83	13-C96	13-D13	13-D16	13-D18	13-D21	13-D25
	13-D44	13-D46	13-D59	13-D75	13-D78	13-D80	13-D83	13-D87	13-E06	13-E08	13-E10	13-E32	13-E43	13-E63

	13-E70	13-E91	13-E94	13-E96	13-E99	13-F03	13-F08	13-F16	13-F29	13-F51	13-F54	13-F56	13-F59	13-F63
	13-F68	13-F76	13-F96	13-G15	13-G18	13-G21	13-G31	13-G34	13-G36	13-G38	13-G51	13-G70	13-G73	13-G76
	13-G87	13-G90	13-G92	13-G94	13-H06	13-H17	13-H33	13-H36	13-H38	13-H41	13-H45	13-H59	13-H77	13-H80
	13-H82	13-H85	13-H89	13-H91	13-H95	13-I08	13-I26	13-I29	13-I31	13-I34	13-I38	13-I40	13-I44	13-I46
	13-I60	13-I72	13-I88	13-I91	13-I93	13-I96	13-J00	13-J14	13-J35	13-J38	13-J40	13-J43	13-J47	13-J49
	13-J53	13-J66	13-J82	13-J85	13-J87	13-J90	13-J94	13-J96	13-K00	13-K13	13-K24	13-K40	13-K43	13-K45
	13-K48	13-K52	13-K65	13-K82	13-K85	13-K87	13-K90	13-K94	13-K96	13-K98	13-L11	13-L27	13-L30	13-L32
	13-L35	13-L39	13-L41	13-L43	13-L45	13-L58	13-L69	13-L85	13-L89	13-L90	13-L93	13-L97	13-M10	13-M27
	13-M30	13-M32	13-M35	13-M39	13-M41	13-M43	13-M56	13-M72	13-M75	13-M77	13-M80	13-M84	13-M86	13-M88
	13-M90	13-N03	13-N15	13-N33	13-N36	13-N38	13-N41	13-N45	13-N58	13-N75	13-N78	13-N80	13-N83	13-N87
	13-N89	13-N91	13-004	13-020	13-023	13-025	13-028	13-032	13-034	13-036	13-038	13-053	13-064	13-080
	13-083	13-085	13-088	13-092	13-P05	13-P17	13-P23	13-P26	13-P33	13-P36	13-P38	13-P41	13-P45	13-P47
	13-P49	13-P61	13-P74	13-P83	13-P86	13-P94	13-P97	13-P99	13-Q02	13-Q06	13-Q08	13-Q10		
COMMEN	4-484#													
ENDCOM	4-484#													
ERROR	4-484#	13-25	13-41	13-52	13-68	13-76	13-80	13-93	13-110	13-118	13-122	13-124	13-126	13-139
	13-155	13-163	13-167	13-169	13-171	13-175	13-188	13-199	13-215	13-223	13-227	13-240	13-257	13-265
	13-269	13-271	13-273	13-286	13-301	13-309	13-313	13-315	13-317	13-330	13-341	13-356	13-364	13-368
	13-381	13-398	13-406	13-410	13-412	13-414	13-429	13-444	13-452	13-456	13-461	13-468	13-471	13-481
	13-485	13-489	13-505	13-516	13-534	13-542	13-546	13-560	13-577	13-585	13-589	13-591	13-593	13-606
	13-622	13-630	13-634	13-636	13-638	13-640	13-653	13-664	13-682	13-690	13-694	13-708	13-725	13-733
	13-737	13-739	13-741	13-754	13-769	13-777	13-781	13-783	13-785	13-798	13-809	13-827	13-835	13-839
	13-852	13-869	13-877	13-881	13-883	13-885	13-900	13-915	13-923	13-927	13-931	13-938	13-942	13-953
	13-957	13-961	13-977	13-988	13-:06	13-:14	13-:18	13-:40	13-:55	13-:63	13-:67	13-:69	13-:71	13-:84
	13-:99	13-:07	13-:11	13-:13	13-:15	13-:28	13-:40	13-:58	13-:66	13-:70	13-:83	13-:88	13-:96	13-<00
	13-<18	13-<24	13-<28	13-<30	13-<32	13-<45	13-<50	13-<58	13-<62	13-<79	13-<85	13-<89	13-<91	13-<93
	13-<95	13-=20	13-=29	13-=43	13-=51	13-=55	13-=76	13-=80	13-=88	13-=92	13->07	13->15	13->19	13->21
	13->23	13->36	13->40	13->48	13->52	13->67	13->75	13->79	13->81	13->83	13-?03	13-?12	13-?25	13-?33
	13-?37	13-?58	13-?73	13-?81	13-?85	13-?87	13-?89	13-a02	13-a17	13-a25	13-a29	13-a31	13-a33	13-a48
	13-a59	13-a83	13-a91	13-a95	13-A08	13-A25	13-A33	13-A37	13-A54	13-A62	13-A70	13-A74	13-A82	13-A86
	13-A91	13-B03	13-B20	13-B28	13-B32	13-B47	13-B55	13-B63	13-B67	13-B75	13-B79	13-B84	13-C07	13-C18
	13-C34	13-C45	13-C71	13-C79	13-C83	13-C96	13-D13	13-D21	13-D25	13-D39	13-D44	13-D46	13-D59	13-D75
	13-D83	13-D87	13-E01	13-E06	13-E08	13-E10	13-E32	13-E43	13-E63	13-E70	13-E74	13-E91	13-E99	13-F03
	13-F08	13-F13	13-F16	13-F29	13-F33	13-F51	13-F59	13-F63	13-F68	13-F73	13-F76	13-F96	13-G00	13-G15
	13-G21	13-G28	13-G34	13-G36	13-G38	13-G51	13-G55	13-G70	13-G76	13-G83	13-G90	13-G92	13-G94	13-H06
	13-H17	13-H33	13-H41	13-H45	13-H59	13-H77	13-H85	13-H89	13-I.91	13-H95	13-I08	13-I26	13-I34	13-I38
	13-I40	13-I44	13-I46	13-I60	13-I72	13-I88	13-I96	13-J00	13-J14	13-J35	13-J43	13-J47	13-J49	13-J53
	13-J66	13-J82	13-J90	13-J94	13-J96	13-K00	13-K13	13-K24	13-K40	13-K48	13-K52	13-K65	13-K82	13-K90
	13-K94	13-K96	13-K98	13-L11	13-L27	13-L35	13-L39	13-L41	13-L43	13-L45	13-L58	13-L69	13-L85	13-L93
	13-L97	13-M10	13-M27	13-M35	13-M39	13-M41	13-M43	13-M56	13-M72	13-M80	13-M84	13-M86	13-M88	13-M90
	13-N03	13-N15	13-N33	13-N41	13-N45	13-N58	13-N75	13-N83	13-N87	13-N89	13-N91	13-004	13-020	13-028
	13-032	13-034	13-036	13-038	13-053	13-064	13-080	13-088	13-092	13-P05	13-P17	13-P23	13-P33	13-P41
	13-P45	13-P47	13-P49	13-P61	13-P74	13-P83	13-P94	13-Q02	13-Q06	13-Q08	13-Q10	42-1		
ESCAPE	4-484#													
GETPRI	4-484#													
GETREG	4-164#	13-471	13-942											
GETSWR	4-484#	10-28	10-28#											
MSG	4-8#													
MULT	4-484#													
NEWST	4-484#	13-2	13-33	13-180	13-322	13-497	13-645	13-790	13-969	13-:20	13-=05	13->88	13-a38	13-C24
	13-E49	13-F81	13-G99	13-I54	13-K05	13-L50	13-M95	13-046						
NWTST	4-36#	13-2	13-33	13-180	13-322	13-497	13-645	13-790	13-969	13-:20	13-=05	13->88	13-a38	13-C24
	13-E49	13-F81	13-G99	13-I54	13-K05	13-L50	13-M95	13-046						
POP	4-484#	13-17	13-20	13-24	16-71	16-278	16-288	16-398	17-65	18-95	19-25	19-26	19-27	20-70
	21-59	22-20	22-21	23-35	24-145	24-149	25-131	28-51	30-25	31-58	34-58	34-59	37-1	39-1
	46-1	48-1	48-1	49-1	49-1									
PUSH	4-484#	13-5	13-14	16-47	16-262	16-281	16-297	17-20	18-17	19-10	19-11	19-12	20-35	21-29

	22-4	22-5	23-9	24-141	25-128	28-20	30-12	31-52	34-51	34-52	37-1	39-1	46-1	48-1
PUTREG	48-1	49-1	49-1	49-1	13-G55									
REPORT	4-442#	13-E74	13-F33	13-G00										
RGBFMC	4-484#													
SETPRI	4-69#	7-0	7-0											
SETTRA	4-484#	12-49	42-5	42-9	45-1									
SETUP	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1#
SKIP	4-484#	10-23												
SLASH	4-484#													
STARS	4-484#													
	13-133	13-180	13-180	13-182	13-234	13-280	13-322	13-322	13-324	13-375	13-420	13-497	13-497	13-499
	13-553	13-600	13-645	13-645	13-647	13-701	13-748	13-790	13-790	13-792	13-846	13-891	13-969	13-969
	13-971	13-:33	13-:78	13-:20	13-:20	13-:22	13-:77	13-<39	13-=05	13-=05	13-=07	13-=70	13->30	13->88
	13->88	13->90	13-?52	13-?96	13-a38	13-a38	13-a40	13-A02	13-A97	13-B97	13-B99	13-C24	13-C24	13-C26
	13-C90	13-D53	13-E22	13-E24	13-E49	13-E49	13-E51	13-F23	13-F81	13-F81	13-F83	13-G45	13-G99	13-G99
	13-H02	13-H52	13-I02	13-I54	13-I54	13-I57	13-J07	13-J60	13-K05	13-K05	13-K07	13-K59	13-L05	13-L50
	13-L50	13-L52	13-M04	13-M50	13-M95	13-M95	13-M97	13-N52	13-N98	13-046	13-046	13-048	13-P01	13-P56
	14-20	15-57	15-70	15-84	15-107	15-139	15-161	15-194	16-213	16-290	16-294	25-26	25-34	25-119
	26-1	26-3	37-1	38-1	39-1	40-1	41-1	42-1	43-1	45-1	45-1	45-1	45-1	45-1
	46-1	47-1	48-1	48-1	49-1									
SWRSU	4-484#	10-23	10-23#											
TAGS	4-103#	6-0												
TRMTRP	47-1#													
TYPBIN	4-484#													
TYPDEC	4-484#	14-20	14-20											
TYPNAM	4-472#	4-484#	10-28											
TYPNUM	4-484#													
TYPOCS	4-484#	10-81	12-16	12-70	44-22	44-47	44-52	44-54						
TYPOCT	4-484#	11-33	45-1											
TYPTXT	4-484#	10-6	10-42	10-53	10-59	10-60	12-30	14-20	14-20					
XPER	4-12#	8-3	8-6	8-9	8-12	8-15	8-18	8-21	8-24	8-27	8-30	8-33	8-36	8-39
	8-42	8-45	8-48	8-51	8-54	8-57	8-60	8-63	8-66	8-69	8-72	8-75	8-78	8-81
	8-84	8-87	8-90	8-93	8-96	8-99	8-102	8-105	8-108	8-111	8-114	8-117	8-120	8-123
	8-126	8-130	8-133	8-136	8-139	8-142	8-146	8-150	8-154	8-157	8-160	8-163	8-166	8-169
	8-172	8-175	8-178	8-182	8-185	8-188	8-191	8-194	8-197	8-200	8-203	8-206	8-209	8-212
	8-215	8-218	8-221	8-224	8-227	8-230	8-233	8-236	8-239	8-242	8-245	8-248	8-251	8-254
	8-257	8-260	8-263	8-266	8-269	8-272	8-275	8-278	8-281	8-284	8-287	8-290	8-293	8-296
	8-299	8-302	8-305	8-308	8-311	8-314	8-317	8-320	8-323	8-326	8-329	8-332	8-335	8-338
	8-341	8-344	8-347	8-351	8-354	8-357	8-360	8-363	8-366	8-369	8-372	8-375	8-378	8-381
	8-384	8-387	8-390	8-393	8-396	8-399	8-402	8-405	8-408	8-411	8-414	8-417	8-420	8-423
	8-426	8-429	8-432	8-435	8-438	8-441	8-444	8-447	8-450	8-453	8-456	8-459	8-462	8-465
	8-468	8-471	8-474	8-477	8-480	8-483	8-486	8-489	8-492	8-495	8-498	8-501	8-504	8-507
	8-510	8-513	8-516	8-519	8-522	8-525	8-528	8-531	8-534	8-537	8-540	8-543	8-546	8-549
	8-552	8-556	8-559	8-563	8-566	8-569	8-573	8-577	8-581	8-586	8-590	8-594	8-597	8-600
	8-603	8-606	8-609	8-612	8-615	8-618	8-621	8-624	8-627	8-630	8-633	8-636	8-639	8-642
	8-645	8-648	8-651	8-654	8-657	8-660	8-663	8-666	8-669	8-672	8-675	8-678	8-681	8-684
	8-687	8-690	8-693	8-696	8-699	8-702	8-705	8-708	8-711	8-714	8-717	8-720	8-723	