

RP04/5/6

RP04/5/6 DSKLS 1
CZRJGDO

AH 9210D MC
COPYRIGHT 76-79
FICHE 1 OF 2

NOV 1979
digital
MADE IN USA

The main body of the document consists of a 10x10 grid of small, illegible data tables. Each table appears to be a structured data set, possibly a ledger or a series of reports, but the text is too faint to read. The tables are arranged in 10 rows and 10 columns, filling most of the page area below the header.

RP04/5/6

RP04/5/6 DSKLS 1
CZRJGD0

AH 9210D MC

COPYRIGHT 76 79
FICHE 2 OF 2

NOV 1979

digital

MADE IN USA

Table with multiple columns and rows of data, including headers like 'NAME', 'ADDRESS', 'CITY', 'STATE', 'ZIP', and 'PHONE'. The text is very faint and difficult to read.

.REM @

IDENTIFICATION

PRODUCT CODE: AC-9208D-MC
PRODUCT NAME: CZRJGDO RP04/5/6 DISKLESS CONTROLLER TEST-PART I
DATE CREATED: MAY, 1979
MAINTAINER: DIAGNOSTIC ENGINEERING
AUTHOR: PETE BLACKSTONE

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

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

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

COPYRIGHT (C) 1976,1979 DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION

| | | | |
|---------|-------|---------|----------|
| DIGITAL | PDP | UNIBUS | MASSBUSS |
| DEC | DECUS | DECTAPE | |

CONTENTS

- 1. ABSTRACT
- 2. REQUIREMENTS
 - 2.1 EQUIPMENT
 - 2.2 STORAGE
 - 2.3 PRELIMINARY PROGRAMS
- 3. LOADING PROCEDURE
 - 3.1 METHOD
- 4. STARTING PROCEDURE
 - 4.1 CONTROL SWITCH SETTINGS
 - 4.2 STARTING ADDRESS OR ADDRESSES
 - 4.3 PROGRAM AND/OR OPERATOR ACTION
- 5. OPERATING PROCEDURE
 - 5.1 OPERATIONAL SWITCH SETTINGS
 - 5.2 SUB-ROUTINE ABSTRACTS
- 6. ERRORS
 - 6.1 'FATAL' ERRORS
- 7. RESTRICTIONS
- 8. MISCELLANEOUS
 - 8.1 EXECUTION TIME
 - 8.2 STACK POINTER
 - 8.3 OPERATOR SELECTABLE SCOPE LOOPS
 - 8.4 PROGRAM REVISION HISTORY
- 9.0 PROGRAM DESCRIPTION

1.0 ABSTRACT

THE DIAGNOSTIC IS USED TO TEST RP04/5/6 DEVICE CONTROL LOGIC CONNECTED TO EITHER AN RH11 OR RH70 DISK DRIVE CONTROLLER

THIS DIAGNOSTIC TESTS THE RH11 AND DCL OF AN RJP04/5/6 SUBSYSTEM. IT DOES NOT USE THE DISK SURFACE OR ANY SIGNALS FROM THE MDLI. IT REQUIRES THAT THE DCL CABLE BE PLUGGED INTO THE MDLI OR BE APPROPRIATELY TERMINATED. IF THE DISK IS POWERED UP, IT IS REQUIRED TO GET THE DISK TO THE "HEADS UNLOADED" POSITION. AFTER A SUCCESSFUL RUN (WITH NO ERRORS) OF THIS DIAGNOSTIC IT CAN BE ASSERTED THAT, "THAT PART OF THE DCL THAT HANDLES DATA OR DATA ASSOCIATED LOGIC IS WORKING PROPERLY". THIS IMPLIES THAT, THE PART OF THE LOGIC WHICH HANDLES MECHANICAL COMMANDS OR ITS ASSOCIATED LOGIC IS NOT TESTED IN THIS DIAGNOSTIC. ALL DATA COMMANDS USE THE MAINTENANCE REGISTER IN THE WRAPAROUND MODE.

THE DIAGNOSTIC DOES NOT DO ANY TESTING OF THE RH70 CONTROLLER WHEN IT IS USED ON AN RWP04/5/6 SYSTEM TO TEST RP04/5/6 DISK DRIVES CONNECTED TO THAT TYPE OF CONTROLLER. IT IS ASSUMED THAT THE RH70 SPECIFIC CONTROLLER DIAGNOSTIC HAVE BEEN SUCCESSFULLY RUN TO COMPLETION BEFORE THIS DIAGNOSTIC IS RUN.

2.0 REQUIREMENTS

2.1 EQUIPMENT

PDP-11 COMPUTER WITH CONSOLE TELETYPE, AND A RP04/5/6 DISK SYSTEM. THE RP04/5/6 DISK SYSTEM WILL CONSIST OF AN RH11/RH70 CONTROLLER, AND DISK CONTROL LOGIC (DCL), THE CABLE FROM THE DCL CAN BE CONNECTED TO THE MDLI, BUT IF NOT THAT CABLE MUST BE PROPERLY TERMINATED.

2.2 STORAGE

THIS PROGRAM REQUIRES 16K WORDS OF MEMORY.

2.3 PRELIMINARY PROGRAMS

THIS CAN BE THE FIRST PROGRAM RUN ON AN RJP04/5/6 SYSTEM BUT THE CONTROLLER DIAGNOSTICS MUST BE RUN FIRST IN THE CASE OF AN RWP04/5/6 SYSTEM.

3.0 LOADING PROCEDURE

USE STANDARD PROCEDURE FOR LOADING .ABS TAPES

4.0 STARTING PROCEDURE

SWITCH 12 MUST BE SET WHEN THIS PROGRAM IS TO BE RUN USING AN RH70 CONTROLLER. IT CAN BE SET AT THE FRON PAEL, OR IN THE SOFTWARE SWITCH REGISTER IF THE OPERATOR SO DESIRES. SE PARAGRAPH 5.1 FOR A DESCRIPTION OF SOFTWARE SWITCH REGISTER OPERATION.

4.1 CONTROL SWITCH SETTINGS

SEE SECTION 5.1

4.2 STARTING ADDRESS

START AT ADDRESS 200---FOR NORMAL RUN
START AT ADDRESS 204---TO SELECT NON-DEFAULT ADDRESSES
START AT ADDRESS 210---FOR UNIT SELECTION

200 START

ALL SWITCHES MUST BE DOWN FOR WORST CASE RUN. WITH THIS STARTING ADDRESS ALL THE RP04/5/6S ON THE SYSTEM WILL BE TESTED ONE AT A TIME BEFORE "END PASS" IS PRINTED OUT. TESTING WILL START WITH THE LOWEST UNIT NUMBER DRIVE THAT IS POWERED UP (THAT IS THE LOWEST UNIT NUMBER RHAS REGISTER THAT RESPONDS) THEN GO ON TO THE NEXT HIGHER UNIT NUMBER THAT IS POWERED UP.

204 RESTART

SAME AS START 200 WITH THE FOLLOWING EXCEPTION: THE PROGRAM WILL INTERROGATE THE OPERATOR FOR A NON-STANDARD C.S.R. AND VECTOR ADDRESS BEFORE STARTING. ONCE THE QUESTIONS HAVE BEEN CORRECTLY ANSWERED, AND IT IS ALSO NECESSARY TO SELECT A PARTICULAR UNIT FOR TEST (TYPICAL PROGRAM EXECUTION FROM ADDRESS 210), THE PROCESSOR MAY BE HALTED AND RESTARTED FROM ADDRESS 210. THE NEW PARAMETERS WILL NOT BE CHANGED UNLESS THE PROGRAM IS AGAIN RESTARTED FROM ADDRESS 204. IF ALL UNITS ARE TO BE CHECKED, THE PROCESSOR NEED NOT BE TOUCHED. THE PROGRAM WILL AUTOMATICALLY RESTART AT 200 AFTER RECEIVING THE NEW DEVICE PARAMETERS.

210 START

ALL SWITCHES MUST BE DOWN FOR WORST CASE RUN. WITH THIS STARTING ADDRESS THE CONSOLE TELETYPE WILL ASK FOR THE UNIT NUMBER TO BE TESTED. THEN ONLY THAT UNIT WILL BE TESTED FOR EACH PASS OF THE PROGRAM.

4.3 PROGRAM AND/OR OPERATOR ACTION

1. LOAD THE PROGRAM INTO MEMORY.
2. SET STARTING ADDRESS ON THE SWITCH REGISTER
3. PRESS "LOAD ADDRESS".
4. SET "OPERATIONAL SWITCH SETTINGS" (SEE SECTION 5.1)
WORST CASE IS ALL SWITCHES DOWN.
5. PRESS "START".
6. FOR THE FIRST PASS EACH TEST WILL BE EXECUTED ONCE ON THE DRIVES PRESENT OR DRIVE SELECTED BEFORE "END PASS" IS PRINTED. THE FIRST PASS WILL REQUIRE OPERATOR

INTERVENTION IF THE PROGRAM IS NOT RUN UNDER AN "ACT-11" MONITOR. THE SECOND AND SUBSEQUENT PASSES WILL EXECUTE EACH TEST FOUR TIMES ON EACH DRIVES PRESENT OR DRIVE SELECTED BEFORE "END PASS" IS PRINTED. THE SECOND AND SUBSEQUENT PASSES DO NOT NEED ANY OPERATOR INTERVENTION.

5.0 OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS

IF THE PROGRAM IS BEING RUN ON A SWITCHLESS PROCESSOR (I. E. AN 11/34) IT WILL DETERMINE THAT A HARDWARE SWITCH REGISTER IS NOT PRESENT, AND WILL USE A "SOFTWARE" SWITCH REGISTER. THE SETTINGS OF THE "SOFTWARE" SWITCHES ARE CONTROLLED THROUGH A KEYBOARD ROUTINE WHICH IS CALLED BY TYPING A 'CONTROL G'. THE PROGRAM WILL RECOGNIZE A 'CONTROL G' AT ANY TIME EXCEPT WHEN IT IS AT A HIGHER PRIORITY PROCESSING A RP04/5/6 INTERRUPT. THE "SOFTWARE" SWITCH VALUEA ARE ENTERED AS AN OCTAL NUMBER IN RESPONSE TO A PROMPT FROM THE SWITCH ENTRY ROUTINE:

'SWR = NNNNNN NEW ='

EACH TIME SWITCH SETTINGS ARE ENTERED, THE ENTIRE SWITCH REGISTER IMAGE MUST BE ENTERED. LEADING ZEROS ARE NOT REQUIRED. 'RUBOUT' AND 'CONTROL U' FUNCTIONS MAY BE USED TO CORRECT TYPING ERRORS DURING SWITCH ENTRY.

ON PROCESSORS WITH HARDWARE SWITCH REGISTERS, THE "SOFTWARE" SWITCH REGISTER MAY ALSO BE USED. IF THE PROGRAM FINDS ALL 16 SWITCHES IN THE 'UP' POSITION WHEN IT IS STARTED, ALL SWITCH REGISTER REFERENCES WILL BE TO THE "SOFTWARE" REGISTER AND THE PROCEDURES DESCRIBED ABOVE MUST BE FOLLOWED.

SWITCH DEFINITIONS ARE GIVEN IN SECTION 9 "OPERATIONAL SWITCH SETTINGS" HOWEVER THE DETAIL DESCRIPTION ARE GIVEN HERE.

SWITCH 15 - HALT ON ERROR
WHEN THIS SWITCH IS SET, IF THE PROGRAM FINDS AN ERROR THEN THE APPROPRIATE INFORMATION WILL BE PRINTED OUT AND THEN THE PROGRAM WILL HALT. AFTER THIS HALT, PRESSING "CONTINUE" WILL CONTINUE WITH THE PROGRAM TILL THE NEXT ERROR IS FOUND WHEN THE SAME THING WILL HAPPEN.

SWITCH 14 - LOOP ON TEST
WHEN THIS SWITCH IS SET THE PROGRAM WILL BEGIN TO LOOP ON THE CURRENT TEST BEING EXECUTED. FOR EXAMPLE IF THIS SWITCH IS SET WHEN THE PROGRAM IS IN TEST 10 THEN THE PROGRAM WILL KEEP EXECUTING ALL OF TEST 10 REPEATEDLY. ONE WAY TO BE SURE THAT THE PROGRAM IS IN THE EXPECTED TEST IS TO SET THIS SWITCH DURING AN ERROR PRINTOUT OR DURING A PROGRAM HALT.

SWITCH 13 - INHIBIT ERROR TYPEOUTS
WHEN THIS SWITCH IS SET FURTHER ERROR PRINTOUTS WILL
CEASE, HOWEVER OPERATOR INSTRUCTIONS SUCH AS "STOP DRIVE X"
WILL CONTINUE. AT THE END OF PASS "TOTAL NUMBER OF ERRORS
ON THIS PASS ON DRIVE X" WILL BE TRUE, THAT IS, ALTHOUGH
PRINTOUTS WERE INHIBITED IF THAT PASS FOUND 6 ERRORS,
IT WILL SAY SO.

SWITCH 12 - RH70 CONTROFLLER SELECT
THIS SWICH MUST BE SET AT THE START OF THE PROGRAM WHEN THE
DISK DRIVES TO BE TESTED ARE CONNECTED TO AN RH70
CONTROLLER. IT MUST NOT BE SET WHEN DISK DRIVES TO BE TESTED
ARE CONNECTED TO AN RH11 CONTROLLER.

SWITCH 11 - INHIBIT ITERATIONS
WHEN THIS SWITCH IS SET THE PROGRAM ON SECOND PASS WILL
NOT REPEAT EACH TEST FOUR TIMES BUT WILL DO EACH TEST
ONCE ONLY.

SWITCH 10 - BELL ON ERROR
WHEN THIS SWITCH IS SET, IF THE PROGRAM FINDS AN ERROR
THE "BELL" OR "ALARM" WILL BE SOUNDED. THIS SWITCH IS USEFUL
WHEN SWITCH 11 IS SET YET INFORMATION IS NEEDED WHEN ANY ERROR
IS DETECTED. TAKE THE EXAMPLE OF A PROGRAM LOOPING ON A TEST WITH
SWITCH 11 SET TO HELP SCOPING. THEN IF THIS SWITCH IS
SET AND THE BELL OR ALARM SOUNDS IT MEANS THAT THE ERROR
IS PRESENT BUT IF THE BELL OR ALARM STOPS IT MEANS THAT
THE ERROR IS NOT PRESENT.

SWITCH 9 - LOOP ON ERROR
WHEN THIS SWITCH IS SET, IF THE PROGRAM FINDS AN ERROR
THEN GENERALLY THE PROGRAM WILL LOOP BACK TO THE LAST
EXECUTED "SCOPE" STATEMENT. IF ON THE SECOND TIME
THROUGH AN ERROR IS FOUND IT WILL AGAIN LOOP BACK TO
THAT "SCOPE" STATEMENT. THIS LOOPING WILL CONTINUE AS LONG
AS THE ERROR IS PRESENT AND THIS SWITCH IS SET. HOWEVER
IF THE ERROR IS NOT PRESENT AT ANY TIME THEN IT WILL
CONTINUE NORMALLY WITH THE PROGRAM. EACH TIME THE ERROR
IS ENCOUNTERED PRINTOUT WILL TAKE PLACE UNLESS SWITCH 11
IS ALSO SET. DURING BEGUG, USING A SCOPE, IT IS RECOMMENDED
THAT SWITCH 11 IS ALSO SET.

NOTE: ALSO SEE SECTION 8.3

SWITCH 8 - LOOP ON TEST IN SWR <7:0>
THIS IS A SPECIAL SWITCH. WHEN SET SWITCHES 0 THRU 7
HAVE ONE MEANING AND WHEN RESET SWITCHES 0 THRU 7 HAVE
ANOTHER MEANING. THIS MEANS THAT ANY SETTING OF SWITCH
0 THRU 7 MUST BE DONE WITH SWITCH 8 IN THE APPROPRIATE
POSITION. WHEN THIS SWITCH IS SET THEN SWITCHES 0 THRU
7 GIVE THE TEST NUMBER TO BE LOOPED ON. FOR EXAMPLE
WITH SWITCH 8 SET AND SWITCH 3 SET THE PROGRAM WILL LOOP

ON TEST 10. HOWEVER THIS SETTING MUST BE DONE AT THE BEGINNING OF THE PROGRAM THEN ALL THE TESTS FROM 1 TO 10 WILL BE EXECUTED AND THEN TEST 10 WILL BE REPEATED OVER AND OVER AGAIN. WHEN THIS SWITCH IS NOT SET THEN SWITCHES 0 THRU 7 HAVE THE MEAING ITS NAME INDICATES. FOR EXAMPLE SWITCH 7 IS "STOP FURTHER COMPARES: THAT IS IF SWITCH 8 IS NOT SET AND SWITCH 7 IS SET THEN WHEN A DATA ERROR IS DETECTED NO FURTHER COMPARES WILL BE DONE. FOR EXAMPLE IN A 256 WORD BUFFER IF ALL THE WORDS ARE IN ERROR THEN AFTER SEEING THE PRINTOUT FOR THE FIRST FEW WORDS SETTING SWITCH 7 ONLY WILL STOP FURTHER PRINTOUTS OF THIS ERROR AND GO ON WITH THE TEST RATHER THAN PRINT ALL THE 256 WORDS. HOWEVER IF THIS WAS DONE WITH SWITCH 11 THEN THE NEXT ERROR THAT THE PROGRAM DETECTS IN A SUBSEQUENT TEST WILL ALSO BE LOST. BUT WITH SWITCH 7, ONLY THIS GROUP OF DATA ERRORS ARE NOT PRINTED OUT. ANOTHER EXAMPLE OF SWITCH 8 BEING LOW IS WITH SWITCH 6, WHICH IS "ECC TEST-COMPARE END RESULT ONLY". THAT IS IF SWITCH 8 IS NOT SET AND SWITCH 6 IS SET THEN ON ECC TESTS (TEST 120 THRU TEST 134) INSTEAD OF COMPARING CONTENTS OF THE POSITION REGISTER AND PATTERN REGISTER AFTER EVERY CLOCK, COMPARES WILL ONLY BE DONE AT THE END OF ALL THE CLOCKS.

NOTE: ALSO SEE SECTION 8.3

SWITCH 7 - STOP FURTHER COMPARES IF SW08 IS LOW. IF SWITCH 8 IS SET AND THIS SWITCH IS ALSO SET THEN THIS SWITCH GIVES THE TEST NUMBER TO BE LOOPED ON AS INDICATED IN THE DESCRIPTION OF SWITCH 8. IF SWITCH 8 IS NOT SET AND THIS SWITCH IS SET THEN THE PROGRAM WILL DO AS THE NAME INDICATES. FOR EXAMPLE IN A 256 WORD BUFFER IF ALL THE WORDS ARE IN ERROR THEN AFTER SEEING THE ERROR PRINTOUTS FOR THE FIRST FEW WORDS THEN SETTING SWITCH 7 WITH SWITCH 8 NOT SET WILL STOP THE PRINTOUT OF ALL 256 WORDS BUT WILL NOT STOP THE PRINTOUT OF ANOTHER ERROR IN ANY SUBSEQUENT TEST. IT IS EXPECTED THAT SWITCH 7 AFTER BEING SET FOR A WHILE TO STOP PRINTING ALL THE 256 WORDS WILL BE RESET AGAIN TO ENABLE THE PRINTING OF OTHER DATA ERRORS.

SWITCH 6 - ECC TEST-COMPARE END RESULTS ONLY IF SW08 IS LOW IF SWITCH 8 IS SET AND THIS SWITCH IS ALSO SET THEN THIS SWITCH GIVES THE TEST NUMBER TO BE LOOPED ON AS INDICATED IN THE DESCRIPTION OF SWITCH 8. IF SWITCH 8 IS NOT SET AND THIS SWITCH IS SET THEN ON ECC TESTS (TEST 120 THRU TEST 134) INSTEAD OF COMPARING CONTENTS OF THE POSITION AND PATTERN REGISTERS AFTER EVERY CLOCK, COMPARES WILL BE DONE ONLY AT THE END OF ALL THE CLOCKS.

5.2 SUB-ROUTINE ABSTRACTS

SEE SECTION 9 "SUBROUTINES"

6.0 ERRORS

ERROR PRINTOUTS CONTAIN THE ERROR ADDRESS AND OTHER PERTINENT INFORMATION CONCERNING THE PARTICULAR FAILURE. THIS INFORMATION MAY BE THE CONTENTS OF RELEVANT RP04/5/6 REGISTERS OR GOOD/RECEIVED DATA. IF THE ERROR OCCURRED IN A SUBROUTINE, THE ADDRESS OF THE SUBROUTINE CALL IS ALSO GIVEN. REFER TO THE PROGRAM LISTING AT THE STATED ADDRESS TO DETERMINE THE CAUSE OF THE ERROR.

6.1 'FATAL' ERRORS

IN THE EVENT THAT THE DISK DRIVE BECOMES UNAVAILABLE TO THE CONTROLLER, POWERS DOWN, OR CERTAIN CRITICAL STATUS BITS CANNOT BE CLEARED PRIOR TO THE START OF A TEST SEQUENCE - THIS INFORMATION WILL BE COMMUNICATED TO THE OPERATOR. IN ADDITION THE TTY BELL WILL RING AND THE PROGRAM WILL HALT.

IT IS SUGGESTED THAT IF THIS HAPPENS THE OPERATOR LOAD ADDRESS 200 (210) AND RESTART THE PROGRAM AS A FIRST ATTEMPT TO SOLVE THE PROBLEM. IF THE FAILURE CONTINUES TO OCCUR, LOOK IN THE TEST LISTING FOR THE 'HALT' INSTRUCTION AND REPLACE IT PLUS THE TWO WORDS ('TYPE ,CPHALT') ABOVE WITH 'NOP'S. WITH TTY ERROR PRINTOUTS INHIBITED A SCOPE LOOP CAN BE INITIATED FOR THE TEST IN QUESTION.

IT IS ALSO POSSIBLE TO CONTINUE FROM THE HALT POINT, BUT IT IS NOT RECOMMENDED AS ALL FOLLOWING TESTS WILL EXHIBIT THE SAME SYMPTOMS AND GIVE MISLEADING ERROR PRINTOUTS.

7.0 RESTRICTIONS

IF THERE IS A DRIVE CONNECTED THEN THE OPERATOR MUST HAVE THE DRIVE PORT SWITCH LOCKED EITHER ON PORT A OR PORT B BUT NEVER LEAVE IT IN THE PROGRAMMABLE STATE. IF THERE IS NO DRIVE CONNECTED THEN THE CABLE NORMALLY GOING FROM THE DCL TO THE MDLI MUST BE PROPERLY TERMINATED.

SWITCH 12 MUST BE SET WHEN RUNNING ON AN RH70 CONTROLLER AND IT MUST NOT BE SET WHEN RUNNING ON AN RH11 CONTROLLER. BECAUSE OF THE REQUIREMENT FOR IT TO BE SET WHEN USING AN RH70, THE PROGRAM CANNOT BE RUN IN CHAIN MODE WHEN USING THE SOFTWARE SWITCH REGISTER FEATURE WHILE RUNNING ON AN RH70. THIS IS BECAUSE THE ROUTINE WHICH GETS "SOFTWARE" SWITCH SETTINGS IS NOT OPERABLE WHEN IN CHAIN MODE.

8.0 MISCELLANEOUS

8.1 EXECUTION TIME

THE FIRST PASS OF THE PROGRAM WILL TAKE 30 SECONDS PER DRIVE. SUBSEQUENT PASSES WILL TAKE 1 MINUTE.

8.2 STACK POINTER

THE STACK IS INITIALLY SET TO 1000

496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523

```
;*INTERNAL PROGRAM MACROS BEGIN HERE  
:*****  
:*****  
*NOTE: MACROS BEGINNING WITH ".S" ARE SUPPLIED BY AN  
* EXTERNAL SYSMAC.SML SYSTEM MACRO PACKAGE WHICH  
* MUST BE MADE AVAILABLE TO THE SOURCE PROGRAM  
* AT ASSEMBLY TIME.  
:*****  
:*****
```

```
.SBTTL OPERATIONAL SWITCH SETTINGS  
:*  
:* SWITCH USE  
:* -----  
:* 15 HALT ON ERROR  
:* 14 LOOP ON TEST  
:* 13 INHIBIT ERROR TYPEOUTS  
:* 12 RH70 CONTROLLER SELECT  
:* 11 INHIBIT ITERATIONS  
:* 10 BELL ON ERROR  
:* 9 LOOP ON ERROR  
:* 8 LOOP ON TEST IN SWR<7:0>  
:* 7 STOP FURTHER COMPARES IF SW08 IS LOW  
:* 6 ECC TEST-COMPARE END RESULTS ONLY IF SW08 IS LOW
```

.SBTTL BASIC DEFINITIONS

;*INITIAL ADDRESS OF THE STACK POINTER *** 1000 ***

STACK= 1000

.EQUIV EMT,ERROR ;;BASIC DEFINITION OF ERROR CALL

.EQUIV IOT,SCOPE ;;BASIC DEFINITION OF SCOPE CALL

;*MISCELLANEOUS DEFINITIONS

HT= 11 ;;CODE FOR HORIZONTAL TAB

LF= 12 ;;CODE FOR LINE FEED

CR= 15 ;;CODE FOR CARRIAGE RETURN

CRLF= 200 ;;CODE FOR CARRIAGE RETURN-LINE FEED

PS= 177776 ;;PROCESSOR STATUS WORD

.EQUIV PS,PSW

STKLMT= 177774 ;;STACK LIMIT REGISTER

PIRQ= 177772 ;;PROGRAM INTERRUPT REQUEST REGISTER

DSWR= 177570 ;;HARDWARE SWITCH REGISTER

DDISP= 177570 ;;HARDWARE DISPLAY REGISTER

;*GENERAL PURPOSE REGISTER DEFINITIONS

R0= %0 ;;GENERAL REGISTER

R1= %1 ;;GENERAL REGISTER

R2= %2 ;;GENERAL REGISTER

R3= %3 ;;GENERAL REGISTER

R4= %4 ;;GENERAL REGISTER

R5= %5 ;;GENERAL REGISTER

R6= %6 ;;GENERAL REGISTER

R7= %7 ;;GENERAL REGISTER

SP= %6 ;;STACK POINTER

PC= %7 ;;PROGRAM COUNTER

;*PRIORITY LEVEL DEFINITIONS

PR0= 0 ;;PRIORITY LEVEL 0

PR1= 40 ;;PRIORITY LEVEL 1

PR2= 100 ;;PRIORITY LEVEL 2

PR3= 140 ;;PRIORITY LEVEL 3

PR4= 200 ;;PRIORITY LEVEL 4

PR5= 240 ;;PRIORITY LEVEL 5

PR6= 300 ;;PRIORITY LEVEL 6

PR7= 340 ;;PRIORITY LEVEL 7

;*"SWITCH REGISTER" SWITCH DEFINITIONS

SW15= 100000

SW14= 40000

SW13= 20000

SW12= 10000

SW11= 4000

SW10= 2000

SW09= 1000

SW08= 400

SW07= 200

SW06= 100

SW05= 40

SW04= 20

SW03= 10

SW02= 4

524
525
526
527 001000
528
529
530
531
532 000011
533 000012
534 000015
535 000200
536 177776
537
538 177774
539 177772
540 177570
541 177570
542
543
544 000000
545 000001
546 000002
547 000003
548 000004
549 000005
550 000006
551 000007
552 000006
553 000007
554
555
556 000000
557 000040
558 000100
559 000140
560 000200
561 000240
562 000300
563 000340
564
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      SW01= 2
581      000001      SW00= 1
582      .EQUIV SW09,SW9
583      .EQUIV SW08,SW8
584      .EQUIV SW07,SW7
585      .EQUIV SW06,SW6
586      .EQUIV SW05,SW5
587      .EQUIV SW04,SW4
588      .EQUIV SW03,SW3
589      .EQUIV SW02,SW2
590      .EQUIV SW01,SW1
591      .EQUIV SW00,SW0
592
593      ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
594      100000      BIT15= 100000
595      040000      BIT14= 40000
596      020000      BIT13= 20000
597      010000      BIT12= 10000
598      004000      BIT11= 4000
599      002000      BIT10= 2000
600      001000      BIT09= 1000
601      000400      BIT08= 400
602      000200      BIT07= 200
603      000100      BIT06= 100
604      000040      BIT05= 40
605      000020      BIT04= 20
606      000010      BIT03= 10
607      000004      BIT02= 4
608      000002      BIT01= 2
609      000001      BIT00= 1
610      .EQUIV BIT09,BIT9
611      .EQUIV BIT08,BIT8
612      .EQUIV BIT07,BIT7
613      .EQUIV BIT06,BIT6
614      .EQUIV BIT05,BIT5
615      .EQUIV BIT04,BIT4
616      .EQUIV BIT03,BIT3
617      .EQUIV BIT02,BIT2
618      .EQUIV BIT01,BIT1
619      .EQUIV BIT00,BIT0
620
621      ;*BASIC "CPU" TRAP VECTOR ADDRESSES
622      000004      ERRVEC= 4      ;;TIME OUT AND OTHER ERRORS
623      000010      RESVEC= 10     ;;RESERVED AND ILLEGAL INSTRUCTIONS
624      000014      TBITVEC=14    ;; "T" BIT
625      000014      TRTVEC= 14    ;;TRACE TRAP
626      000014      BPTVEC= 14    ;;BREAKPOINT TRAP (BPT)
627      000020      IOTVEC= 20    ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
628      000024      PWRVEC= 24    ;;POWER FAIL
629      000030      EMTVEC= 30    ;;EMULATOR TRAP (EMT) **ERROR**
630      000034      TRAPVEC=34    ;; "TRAP" TRAP
631      000060      TKVEC= 60     ;;TTY KEYBOARD VECTOR
632      000064      TPVEC= 64     ;;TTY PRINTER VECTOR
633      000240      PIRQVEC=240   ;;PROGRAM INTERRUPT REQUEST VECTOR
634
```

```
635 .SBTTL TRAP CATCHER
636
637 000000 . = 0
638 ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
639 ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
640 ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
641 000174 . = 174
642 000174 000000 DISPREG: .WORD 0 ;:SOFTWARE DISPLAY REGISTER
643 000176 000000 SWREG: .WORD 0 ;:SOFTWARE SWITCH REGISTER
644
645 .SBTTL ACT11 HOOKS
646
647 ;:*****
648 ;:HOOKS REQUIRED BY ACT11
649 000200 $SVPC= . ;SAVE PC
650 000046 . = 46
651 000046 041520 $ENDAD ;:1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
652 000052 . = 52
653 000052 020000 .WORD 20000 ;:2)SET LOC.52 TO 20000
654 000200 . = $SVPC ;: RESTORE PC
655
656 .SBTTL STARTING ADDRESS
657
658 . = 200
659 000200 000137 004244 RA: JMP @#BEGIN ;NORMAL START
660 000204 000137 046356 ADDMOD: JMP @#BASECH ;MODIFY ADDRESSES
661 000210 000137 004234 JMP @#BEGIN2 ;SELECT DRIVE START
662
663
664 ;**STARTING ADDRESS 200 FOR NORMAL STARTS
665 ;**THIS WILL TEST ALL DRIVES ON THE SYSTEM A SINGLE DRIVE AT A TIME
666
667 ;**STARTING ADDRESS 204 FOR PARAMETER MODIFICATION,
668 ;**RESTART AUTOMATICALLY FROM 200 AFTER PARAMETER MODIFICATION.
669
670 ;**STARTING ADDRESS 210 WILL TEST ONLY ONE SPECIFIED DRIVE
671
```

672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709

000250

177572
177574
177576
172516

172300
172302
172304
172306
172310
172312
172314
172316

172340
172342
172344
172346
172350
172352
172354
172356

001110

.SBTTL MEMORY MANAGEMENT DEFINITIONS

;*KT11 VECTOR ADDRESS

MMVEC= 250

;*KT11 STATUS REGISTER ADDRESSES

SR0= 177572
SR1= 177574
SR2= 177576
SR3= 172516

;*KERNEL "I" PAGE DESCRIPTOR REGISTERS

KIPDR0= 172300
KIPDR1= 172302
KIPDR2= 172304
KIPDR3= 172306
KIPDR4= 172310
KIPDR5= 172312
KIPDR6= 172314
KIPDR7= 172316

;*KERNEL "I" PAGE ADDRESS REGISTERS

KIPAR0= 172340
KIPAR1= 172342
KIPAR2= 172344
KIPAR3= 172346
KIPAR4= 172350
KIPAR5= 172352
KIPAR6= 172354
KIPAR7= 172356

::*****
. =1110 ; ?

| Line | Address | Value | Label | Description |
|------|---------|---------------|---|--|
| 710 | | | .SBTTL COMMON TAGS | |
| 711 | | | | |
| 712 | | | ::***** | |
| 713 | | | ::*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS | |
| 714 | | | ::*USED IN THE PROGRAM. | |
| 715 | | | | |
| 716 | 001100 | | .=1100 | |
| 717 | 001100 | | \$CMTAG: | :: START OF COMMON TAGS |
| 718 | 001100 | 000000 | \$PASS: .WORD 0 | :: CONTAINS PASS COUNT |
| 719 | 001102 | 000 | \$TSTNM: .BYTE 0 | :: CONTAINS THE TEST NUMBER |
| 720 | 001103 | 000 | \$ERFLG: .BYTE 0 | :: CONTAINS ERROR FLAG |
| 721 | 001104 | 000000 | \$ICNT: .WORD 0 | :: CONTAINS SUBTEST ITERATION COUNT |
| 722 | 001106 | 000000 | \$LPADR: .WORD 0 | :: CONTAINS SCOPE LOOP ADDRESS |
| 723 | 001110 | 000000 | \$LPERR: .WORD 0 | :: CONTAINS SCOPE RETURN FOR ERRORS |
| 724 | 001112 | 000000 | \$ERTTL: .WORD 0 | :: CONTAINS TOTAL ERRORS DETECTED |
| 725 | 001114 | 000 | \$ITEMB: .BYTE 0 | :: CONTAINS ITEM CONTROL BYTE |
| 726 | 001115 | 001 | \$ERMAX: .BYTE 1 | :: CONTAINS MAX. ERRORS PER TEST |
| 727 | 001116 | 000000 | \$ERRPC: .WORD 0 | :: CONTAINS PC OF LAST ERROR INSTRUCTION |
| 728 | 001120 | 000000 | \$GDADR: .WORD 0 | :: CONTAINS ADDRESS OF 'GOOD' DATA |
| 729 | 001122 | 000000 | \$BDADR: .WORD 0 | :: CONTAINS ADDRESS OF 'BAD' DATA |
| 730 | 001124 | 000000 | \$GDDAT: .WORD 0 | :: CONTAINS 'GOOD' DATA |
| 731 | 001126 | 000000 | \$BDDAT: .WORD 0 | :: CONTAINS 'BAD' DATA |
| 732 | 001130 | 000000 | .WORD 0 | :: RESERVED--NOT TO BE USED |
| 733 | 001132 | 000000 | .WORD 0 | |
| 734 | 001134 | 000 | \$AUTOB: .BYTE 0 | :: AUTOMATIC MODE INDICATOR |
| 735 | 001135 | 000 | \$INTAG: .BYTE 0 | :: INTERRUPT MODE INDICATOR |
| 736 | 001136 | 000000 | .WORD 0 | |
| 737 | 001140 | 177570 | \$SWR: .WORD DSWR | :: ADDRESS OF SWITCH REGISTER |
| 738 | 001142 | 177570 | \$DISPLAY: .WORD DDISP | :: ADDRESS OF DISPLAY REGISTER |
| 739 | 001144 | 177560 | \$TKS: 177560 | :: TTY KBD STATUS |
| 740 | 001146 | 177562 | \$TKB: 177562 | :: TTY KBD BUFFER |
| 741 | 001150 | 177564 | \$TPS: 177564 | :: TTY PRINTER STATUS REG. ADDRESS |
| 742 | 001152 | 177566 | \$TPB: 177566 | :: TTY PRINTER BUFFER REG. ADDRESS |
| 743 | 001154 | 000 | \$NULL: .BYTE 0 | :: CONTAINS NULL CHARACTER FOR FILLS |
| 744 | 001155 | 002 | \$FILLS: .BYTE 2 | :: CONTAINS # OF FILLER CHARACTERS REQUIRED |
| 745 | 001156 | 012 | \$FILLC: .BYTE 12 | :: INSERT FILL CHARS. AFTER A 'LINE FEED' |
| 746 | 001157 | 000 | \$TPFLG: .BYTE 0 | :: "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES) |
| 747 | 001160 | 000000 | \$REGAD: .WORD 0 | :: CONTAINS THE ADDRESS FROM WHICH (\$REG0) WAS OBTAINED |
| 748 | | | | |
| 749 | 001162 | 000000 | \$REG0: .WORD 0 | :: CONTAINS ((\$REGAD)+0) |
| 750 | 001164 | 000000 | \$REG1: .WORD 0 | :: CONTAINS ((\$REGAD)+2) |
| 751 | 001166 | 000000 | \$REG2: .WORD 0 | :: CONTAINS ((\$REGAD)+4) |
| 752 | 001170 | 000000 | \$REG3: .WORD 0 | :: CONTAINS ((\$REGAD)+6) |
| 753 | 001172 | 000000 | \$REG4: .WORD 0 | :: CONTAINS ((\$REGAD)+10) |
| 754 | 001174 | 000000 | \$REG5: .WORD 0 | :: CONTAINS ((\$REGAD)+12) |
| 755 | 001176 | 000000 | \$TMP0: .WORD 0 | :: USER DEFINED |
| 756 | 001200 | 000000 | \$TMP1: .WORD 0 | :: USER DEFINED |
| 757 | 001202 | 000000 | \$TMP2: .WORD 0 | :: USER DEFINED |
| 758 | 001204 | 000000 | \$TMP3: .WORD 0 | :: USER DEFINED |
| 759 | 001206 | 000000 | \$TMP4: .WORD 0 | :: USER DEFINED |
| 760 | 001210 | 000000 | \$TMP5: .WORD 0 | :: USER DEFINED |
| 761 | 001212 | 000000 | \$TIMES: 0 | :: MAX. NUMBER OF ITERATIONS |
| 762 | 001214 | 000000 | \$ESCAPE: 0 | :: ESCAPE ON ERROR ADDRESS |
| 763 | 001216 | 177607 000377 | \$BELL: .ASCIZ <207><377><377> | :: CODE FOR BELL |
| 764 | 001222 | 077 | \$QUES: .ASCII /?/ | :: QUESTION MARK |
| 765 | 001223 | 015 | \$CRLF: .ASCII <15> | :: CARRIAGE RETURN |

CZRJGDO,RP04/5/6 DSKLS CTRLR1
CZRJGD.P11 10-SEP-79 11:00

MACY11 30A(1052) 10-SEP-79 11:11 PAGE 17
COMMON TAGS

D 2

SEQ 0016

766 001224 000012
767

\$LF: .ASCIZ <12> ;:LINE FEED
:*****

```
768 .SBTTL ERROR POINTER TABLE
769
770 ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
771 ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
772 ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
773 ;*NOTE1: IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
774 ;*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
775
776 ;* EM ;:POINTS TO THE ERROR MESSAGE
777 ;* DH ;:POINTS TO THE DATA HEADER
778 ;* DT ;:POINTS TO THE DATA
779 ;* DF ;:POINTS TO THE DATA FORMAT
780
781
782 001226 $ERRTB:
783
784
785
786 ;*ITEM1
787 001226 057572 EM1 ;:WRONG DATA IN READING OR WRITING HARDWARE REGISTER
788 001230 063045 DH1 ;:PC
789 ;:REG. ADDR.
790 ;:GOOD DATA
791 ;:RECEIVED DATA
792 001232 067424 DT1 ;:$ERRPC,REGADR,$GDDAT,$BDDAT
793 001234 070142 DF1 ;:0,0,0,0,0
794
795
796
797
798
799 ;*ITEM2
800 001236 057655 EM2 ;:ERROR ON DATA COMMAND
801
802 001240 066202 DH33 ;:PC
803 ;:PC OF JSR
804 ;:TEST NO
805 ;:WORD NO.
806 ;:GOOD DATA
807 ;:CONTENTS OF RHCS1
808 ;:CONTENTS OF RHDS1
809 ;:CONTENTS OF RHER1
810 001242 070004 DT33 ;:$ERRPC,PCJSR,$ISTNM,ERWORD,$GDDAT,CS1,DS1,ER1
811 001244 070312 DF33 ;:0,0,0,1,0,0,0,0
812
813
814 ;*ITEM3
815 001246 057655 EM2 ;:ERROR ON DATA COMMAND
816
817 001250 065757 DH32 ;:PC
818 ;:PC OF JSR
819 ;:TEST NO
820 ;:WORD NO.
821 ;:GOOD DATA
822 ;:BAD DATA
823 ;:CONTENTS OF RHCS1
```

| | | | | | |
|-----|--------|--------|------|---------|--|
| 824 | | | | | :CONTENTS OF RHDS1 |
| 825 | | | | | :CONTENTS OF RHER1 |
| 826 | | | | | |
| 827 | 001252 | 067760 | DT32 | | :\$ERRPC,PCJSR,\$STSTNM,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1 |
| 828 | 001254 | 070301 | DF32 | | :0,0,0,1,0,0,0,0,0, |
| 829 | | | | | |
| 830 | | | | | |
| 831 | | | | :*ITEM4 | |
| 832 | 001256 | 057655 | EM2 | | :ERROR ON DATA COMMAND |
| 833 | | | | | |
| 834 | 001260 | 065554 | DH31 | | :PC |
| 835 | | | | | :TEST NO |
| 836 | | | | | :WORD NO. |
| 837 | | | | | :GOOD DATA |
| 838 | | | | | :BAD DATA |
| 839 | | | | | :CONTENTS OF RHCS1 |
| 840 | | | | | :CONTENTS OF RHDS1 |
| 841 | | | | | :CONTENTS OF RHER1 |
| 842 | | | | | |
| 843 | 001262 | 067736 | DT31 | | :\$ERRPC,\$STSTNM,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1 |
| 844 | 001264 | 070271 | DF31 | | :0,0,1,0,0,0,0,0, |
| 845 | | | | | |
| 846 | | | | | |
| 847 | | | | | |
| 848 | | | | :*ITEM5 | |
| 849 | 001266 | 000000 | 0 | | : |
| 850 | 001270 | 000000 | 0 | | : |
| 851 | 001272 | 067736 | DT31 | | :\$ERRPC,\$STSTNM,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1 |
| 852 | 001274 | 070271 | DF31 | | :0,0,1,0,0,0,0,0, |
| 853 | | | | | |
| 854 | | | | | |
| 855 | | | | :*ITEM6 | |
| 856 | 001276 | 057704 | EM6 | | :ERROR ON WRITE HEADER AND DATA |
| 857 | | | | | |
| 858 | 001300 | 065757 | DH32 | | :PC |
| 859 | | | | | :PC OF JSR |
| 860 | | | | | :TEST NO |
| 861 | | | | | :WORD NO. |
| 862 | | | | | :GOOD DATA |
| 863 | | | | | :BAD DATA |
| 864 | | | | | :CONTENTS OF RHCS1 |
| 865 | | | | | :CONTENTS OF RHDS1 |
| 866 | | | | | :CONTENTS OF RHER1 |
| 867 | | | | | |
| 868 | 001302 | 067760 | DT32 | | :\$ERRPC,PCJSR,\$STSTNM,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1 |
| 869 | 001304 | 070301 | DF32 | | :0,0,0,1,0,0,0,0,0, |
| 870 | | | | | |
| 871 | | | | | |
| 872 | | | | | |
| 873 | | | | :*ITEM7 | |
| 874 | 001306 | 057704 | EM6 | | :ERROR ON WRITE HEADER AND DATA |
| 875 | 001310 | 063170 | DH2 | | :PC |
| 876 | | | | | :TEST NO |
| 877 | | | | | :WORD NO. |
| 878 | | | | | :GOOD DATA |
| 879 | | | | | :BAD DATA |

| | | | | |
|-----|--------|--------|----------|--|
| 880 | 001312 | 067452 | DT3 | ;\$ERRPC,\$TSTNM,ERWORD,\$GDDAT,\$BDDAT |
| 881 | 001314 | 070153 | DF3 | :0,0,1,0,0, |
| 882 | | | | |
| 883 | | | | |
| 884 | | | ;*ITEM10 | |
| 885 | 001316 | 000000 | 0 | : |
| 886 | 001320 | 000000 | 0 | : |
| 887 | 001322 | 067452 | DT3 | ;\$ERRPC,\$TSTNM,ERWORD,\$GDDAT,\$BDDAT |
| 888 | 001324 | 070153 | DF3 | :0,0,1,0,0, |
| 889 | | | | |
| 890 | | | | |
| 891 | | | ;*ITEM11 | |
| 892 | 001326 | 057743 | EM11 | :CONTROLLER OR DRIVE STATUS |
| 893 | 001330 | 063313 | DH11 | :PC |
| 894 | | | | :TEST NO |
| 895 | | | | :FAILING REG. ADDR |
| 896 | | | | :CONTENTS OF RHCS1 |
| 897 | | | | :CONTENTS OF RHCS2 |
| 898 | | | | :CONTENTS OF RHDS1 |
| 899 | | | | :CONTENTS OF RHER1 |
| 900 | 001332 | 067466 | DT11 | ;\$ERRPC,\$TSTNM,\$BDADR,CS1,CS2,DS1,ER1 |
| 901 | 001334 | 070160 | DF11 | :0,0,0,0,0,0 |
| 902 | | | | |
| 903 | | | | |
| 904 | | | ;*ITEM12 | |
| 905 | 001336 | 057743 | EM11 | :WRONG DATA FROM SILO |
| 906 | | | | |
| 907 | 001340 | 063045 | DH1 | :PC |
| 908 | | | | :REG. ADDR |
| 909 | | | | :GOOD DATA |
| 910 | | | | :RECEIVED DATA |
| 911 | 001342 | 067424 | DT1 | ;\$ERRPC,REGADR,\$GDDAT,\$BDDAT |
| 912 | 001344 | 070142 | DF1 | :0,0,0,0 |
| 913 | | | | |
| 914 | | | | |
| 915 | | | ;*ITEM13 | |
| 916 | 001346 | 000000 | 0 | |
| 917 | 001350 | 000000 | 0 | |
| 918 | 001352 | 067424 | DT1 | ;\$ERRPC,TSTNM,REGADR,\$GDDAT,\$BDDAT |
| 919 | 001354 | 070142 | DF1 | :0,0,0,0,0 |
| 920 | | | | |
| 921 | | | | |
| 922 | | | ;*ITEM14 | |
| 923 | 001356 | 057776 | EM14 | :REGISTER FAILED |
| 924 | 001360 | 063473 | DH14 | :PC |
| 925 | | | | :FAILING REG. ADDR |
| 926 | | | | :CONTENTS OF FAILING REG. |
| 927 | | | | :CONTENTS OF RHCS1 |
| 928 | | | | :CONTENTS OF RHCS2 |
| 929 | | | | :CONTENTS OF RHDS1 |
| 930 | | | | :CONTENTS OF RHER1 |
| 931 | 001362 | 067506 | DT14 | ;\$ERRPC,\$BDADR,\$BDDAT,CS1,CS2,DS1,ER1 |
| 932 | 001364 | 070167 | DF14 | :0,0,0,0,0,0,0 |
| 933 | | | | |
| 934 | | | | |
| 935 | | | ;*ITEM15 | |

| | | | | |
|-----|--------|--------|------|---------------------------------------|
| 936 | 001366 | 060016 | EM15 | :SPECIFIED REG. NON EXISTANT SO ABORT |
| 937 | | | | :PROGRAM |
| 938 | 001370 | 063673 | DH15 | :PC |
| 939 | | | | :ADDR. OF REG |
| 940 | 001372 | 067530 | DT15 | :\$ERRPC,TEMP1 |
| 941 | 001374 | 070177 | DF15 | :0,0 |
| 942 | | | | |
| 943 | | | | |
| 944 | | | | :*ITEM16 |
| 945 | 001376 | 060067 | EM16 | :WAIT LOOP FAILED |
| 946 | 001400 | 063724 | DH16 | :PC |
| 947 | | | | :WAT PC |
| 948 | | | | :BIT WANTED |
| 949 | | | | :REG. ADR. |
| 950 | | | | :REG. CONT. |
| 951 | 001402 | 067540 | DT16 | :\$ERRPC,\$TMP3,\$TMP1,\$TMPO,\$BDDAT |
| 952 | 001404 | 070202 | DF16 | :0,0,0,0 |
| 953 | | | | |
| 954 | | | | |
| 955 | | | | :*ITEM17 |
| 956 | 001406 | 060110 | EM17 | :WRITE CHECK FAILING |
| 957 | 001410 | 064067 | DH17 | :PC |
| 958 | | | | :TEST NO |
| 959 | | | | :CONTENTS OF RHBA |
| 960 | | | | :CONTENTS OF RHDB |
| 961 | | | | :CONTENTS OF RHWC |
| 962 | | | | :CONTENTS OF RHCS1 |
| 963 | | | | :CONTENTS OF RHCS2 |
| 964 | 001412 | 067556 | DT17 | :\$ERRPC,\$TSTNM,\$BA,DB,WC,CS1,CS2 |
| 965 | 001414 | 070207 | DF17 | :0,0,0,0,0,0,0 |
| 966 | | | | |
| 967 | | | | |
| 968 | | | | :*ITEM20 |
| 969 | 001416 | 060134 | EM20 | :REGISTER FAILING |
| 970 | 001420 | 064252 | DH20 | :PC |
| 971 | | | | :TST NO |
| 972 | | | | :CONTENTS OF RHER1 |
| 973 | | | | :CONTENTS OF RHER2 |
| 974 | | | | :CONTENTS OF RHER3 |
| 975 | | | | :CONTENTS OF RHAS |
| 976 | | | | :CONTENTS OF RHDS1 |
| 977 | 001422 | 067576 | DT20 | :\$ERRPC,TSTNM ER1,ER2,ER3,AS,DS1 |
| 978 | 001424 | 070216 | DF20 | :0,0,0,0,0,0,0 |
| 979 | | | | |
| 980 | | | | :*ITEM21 |
| 981 | | | | |
| 982 | 001426 | 060155 | EM21 | :INTERRUPT FAILING |
| 983 | 001430 | 064435 | DH21 | :PC |
| 984 | | | | :TEST NO |
| 985 | | | | :CONTENTS OF RHCS1 |
| 986 | | | | :CONTENTS OF RHAS |
| 987 | | | | :CONTENTS OF RHDS1 |
| 988 | 001432 | 067616 | DT21 | :\$ERRPC,TSTNM,CS1,AS,DS1 |
| 989 | 001434 | 070225 | DF21 | :0,0,0,0,0 |
| 990 | | | | |
| 991 | | | | |

```
992          ;*ITEM22
993 001436 060177          EM22          ;ERROR IN DRIVE PRESENT -
994          ;LOOKING AT RHAS AND RHCS2-NED(BIT#12)
995          ;DRIVES PRESENT DO NOT AGREE
996          ;NOTE: ON DUAL PORT SYSTEM
997          ;DRIVE ON OTHER PORT WILL NOT GIVE NED
998          ;HENCE THERE WILL BE A MISSMATCH
999 001440 064555          DH22          ;PC
1000         ;TEST NO
1001         ;RHAS UNIT (RHER1 BITS SET)
1002         ;RHCS2 UNIT ('NED' BIT TEST)
1003 001442 067632          DT22          ;$ERRPC,TSTNM
1004 001444 070232          DF22          ;0,0
1005
1006
1007         ;*ITEM23
1008 001446 000000          0          ;NO LONGER USED DUE TO SPECIAL 'NED'
1009 001450 000000          0          ;TEST TABLE TYPE OUT ROUTINE
1010 001452 000000          0
1011 001454 000000          0
1012
1013
1014         ;*ITEM 24
1015 001456 060657          EM24          ;LOOK AHEAD REGISTER AT THE
1016         ;BEGINNING OF A SECTOR IS IN
1017         ;ERROR
1018 001460 064660          DH24          ;PC
1019         ;RHDST
1020         ;BAD RHLA
1021         ;GOOD RHLA
1022         ;SECTOR NO
1023         ;SECTOR CLOCK
1024 001462 067640          DT24          ;$ERRPC,DST,$BDDAT,$TMP1,$TMP2,$TMP3
1025 001464 070236          DF24          ;0,0,0,0,0
1026
1027         ;*ITEM 25
1028 001466 060752          EM25          ;LOOK AHEAD REGISTER IS
1029         ;IN ERROR
1030
1031 001470 064660          DH24          ;PC
1032         ;RHDST
1033         ;BAD RHLA
1034         ;GOOD RHLA
1035         ;SECTOR NO
1036         ;SECTOR CLOCK
1037 001472 067640          DT24          ;$ERRPC,DST,$BDDAT,$TMP1,$TMP2,$TMP3
1038 001474 070236          DF24          ;0,0,0,0,0
1039         ;*ITEM26
1040 001476 057743          EM11          ;CONTROLLER OR DRIVE STATUS
1041
1042 001500 065043          DH26          ;PC
1043         ;PC OF JSR
1044         ;FAILING REGISTER ADDRESS
1045         ;CONTENTS OF RHCS1
1046         ;CONTENTS OF RHCS2
1047         ;CONTENTS OF RHDS1
```

| | | | | |
|------|--------|--------|----------|--|
| 1048 | | | | :CONTENTS OF RHER1 |
| 1049 | | | | |
| 1050 | 001502 | 067660 | DT26 | :\$ERRPC,PCJSR,\$BDADR,CS1,CS2,DS1,ER1 |
| 1051 | 001504 | 070245 | DF26 | :0,0,0,0,0,0, |
| 1052 | | | | |
| 1053 | | | | |
| 1054 | | | | |
| 1055 | | | ;*ITEM27 | |
| 1056 | 001506 | 057572 | EM1 | :ERROR IN READING OR WRITING HARDWARE REGISTER |
| 1057 | | | | |
| 1058 | 001510 | 065246 | DH27 | :PC |
| 1059 | | | | :PC OF JSR |
| 1060 | | | | :TEST NUMBER |
| 1061 | | | | :FAILING REGISTER |
| 1062 | | | | :GOOD DATA |
| 1063 | | | | :RECEIVED DATA |
| 1064 | | | | |
| 1065 | 001512 | 067702 | DT27 | :\$ERRPC,PCJSR,TSTNM,REGADR,\$GDDAT,\$BDDAT |
| 1066 | 001514 | 070255 | DF27 | :0,0,0,0,0,0 |
| 1067 | | | | |
| 1068 | | | | |
| 1069 | | | | |
| 1070 | | | ;*ITEM30 | |
| 1071 | 001516 | 061012 | EM30 | :CURRENT CYLINDER DOES NOT REFLECT DESIRED CYLINDER REG. |
| 1072 | 001520 | 065411 | DH30 | :PC |
| 1073 | | | | :PC OF JSR |
| 1074 | | | | :REGISTER ADDRESS |
| 1075 | | | | :GOOD DATA |
| 1076 | | | | :BAD DATA |
| 1077 | | | | |
| 1078 | 001522 | 067720 | DT30 | :\$ERRPC,PCJSR,REGADR,\$GDDAT,\$BDDAT |
| 1079 | 001524 | 070263 | DF30 | :0,0,0,0,0 |
| 1080 | | | | |
| 1081 | | | | |
| 1082 | | | | |
| 1083 | | | ;*ITEM31 | |
| 1084 | 001526 | 061133 | EM31 | :ECC GENERATED IS INCORRECT |
| 1085 | | | | :EVERY WORD IN THIS SECTOR IS GIVEN IN "DATA USED" |
| 1086 | | | | |
| 1087 | 001530 | 066405 | DH34 | :PC |
| 1088 | | | | :TEST NUMBER |
| 1089 | | | | :GOOD ECC1 |
| 1090 | | | | :GOOD EC2C |
| 1091 | | | | :WRITTEN ECC1 |
| 1092 | | | | :WRITTEN ECC2 |
| 1093 | | | | :DATA USED |
| 1094 | | | | |
| 1095 | 001532 | 070026 | DT34 | :\$ERRPC,TSTNM,GECC1,GECC2,WECC1,WECC2,DISK |
| 1096 | | | | |
| 1097 | 001534 | 070322 | DF34 | :0,0,0,0,0,0,0 |
| 1098 | | | | |
| 1099 | | | | |
| 1100 | | | ;*ITEM32 | |
| 1101 | 001536 | 061256 | EM32 | :ON READ COMMAND AFTER DATA AND ECC HAVE BEEN READ |
| 1102 | | | | :ECC REGISTER OR RHER1 IS IN ERROR |
| 1103 | | | | :ONLY LOWER 11 BITS OF PATTERN REGISTER |

| | | | | |
|------|--------|--------|------|--|
| 1104 | | | | :CAN BE READ |
| 1105 | | | | :THIS SHUOLD MATCH LOWER 11 BITS OF ECC1 |
| 1106 | | | | |
| 1107 | 001540 | 066570 | DH35 | :PC |
| 1108 | | | | :TEST NUMBER |
| 1109 | | | | :GOOD ECC1 |
| 1110 | | | | :GOOD FCC2 |
| 1111 | | | | :PATTERN REGISTER |
| 1112 | | | | :RHER1 |
| 1113 | | | | |
| 1114 | 001542 | 070046 | DT35 | :\$ERRPC,TSTNM,GECC1,GECC2,EC2,ER1 |
| 1115 | | | | |
| 1116 | 001544 | 070331 | DF35 | :0,0,0,0,0,0 |
| 1117 | | | | |
| 1118 | | | | |
| 1119 | | | | |
| 1120 | | | | |
| 1121 | 001546 | 061545 | EM33 | :HIGH COUNT BIT NOT HIGH AFTER 38859 CLOCKS |
| 1122 | 001550 | 066773 | DH36 | :PC |
| 1123 | | | | :PC OF JSR |
| 1124 | | | | :TEST NUMBER |
| 1125 | | | | :RHRM |
| 1126 | | | | :POSITION REG. |
| 1127 | | | | :PATTERN REGISTER |
| 1128 | | | | |
| 1129 | 001552 | 070070 | DT36 | :\$ERRPC,PCJSR,TSTNM,MR,EC1,EC2 |
| 1130 | | | | |
| 1131 | 001554 | 070341 | DF36 | :0,0,0,0,0,0 |
| 1132 | | | | |
| 1133 | | | | |
| 1134 | 001556 | 061617 | EM34 | :ZERO DETECT BIT NOT HIGH WHEN THE |
| 1135 | | | | :32 BIT ECC REGISTER HAS ITS 21 BITS |
| 1136 | | | | :OF ZEROS |
| 1137 | | | | :ERROR PRINTOUT WILL CONTINUE TILL |
| 1138 | | | | :ZERO DETECT BIT IS HIGH |
| 1139 | 001560 | 066773 | DH36 | :PC |
| 1140 | | | | :PC OF JSR |
| 1141 | | | | :TEST NUMBER |
| 1142 | | | | :RHRM |
| 1143 | | | | :POSITION REG. |
| 1144 | | | | :PATTERN REGISTER |
| 1145 | | | | |
| 1146 | 001562 | 070070 | DT36 | :\$ERRPC,PCJSR,TSTNM,MR,EC1,EC2 |
| 1147 | | | | |
| 1148 | 001564 | 070341 | DF36 | :0,0,0,0,0,0 |
| 1149 | | | | |
| 1150 | | | | |
| 1151 | | | | |
| 1152 | | | | |
| 1153 | 001566 | 061712 | EM35 | :POSITION REGISTER OR 11 BITS OF |
| 1154 | | | | :PATTERN REGISTER INCORRECT |
| 1155 | | | | :LOWER 11 BITS OF PATTERN REGISTER |
| 1156 | | | | :SHOULD MATCH LOWER 11 BITS OF GOOD ECC1 |
| 1157 | | | | :DATA ENVELOPE AND N-CODE ZEROS ARE IN DECIMAL |
| 1158 | | | | |
| 1159 | 001570 | 067132 | DH37 | :PC |

| Line | Code | Address | Code | Field | Description |
|------|--------|---------|------|----------|--|
| 1160 | | | | | :TEST NUMBER |
| 1161 | | | | | :ECC POSITION |
| 1162 | | | | | :GOOD POSITION |
| 1163 | | | | | :GOOD ECC1 |
| 1164 | | | | | :GOOD ECC2 |
| 1165 | | | | | :ECC PATTERN |
| 1166 | | | | | :DATA ENVELOPE |
| 1167 | | | | | :N-CODE ZEROS |
| 1168 | | | | | |
| 1169 | 001572 | 070106 | DT37 | | :\$ERRPC,TSTNM,EC1,POSITI,GECC1,GECC2,EC2,DATENV,ZCODE |
| 1170 | | | | | |
| 1171 | 001574 | 070347 | DF37 | | :0,0,0,0,0,0,0,0,0 |
| 1172 | | | | | |
| 1173 | | | | | |
| 1174 | | | | | |
| 1175 | | | | | |
| 1176 | 001576 | 062211 | EM36 | ;*ITEM36 | :ON A READ COMMAND WITH NON CORRECTABLE |
| 1177 | | | | | :ERROR INSERTED DCK AND ECH SHOULD BE SET |
| 1178 | 001600 | 066570 | DH35 | | :PC |
| 1179 | | | | | :TEST NUMBER |
| 1180 | | | | | :GOOD ECC1 |
| 1181 | | | | | :GOOD ECC2 |
| 1182 | | | | | :PATTERN REGISTER |
| 1183 | | | | | :POSITION REGISTER |
| 1184 | | | | | :RHER1 |
| 1185 | | | | | |
| 1186 | 001602 | 070046 | DT35 | | :\$ERRPC,TSTNM,GECC1,GECC2,EC2,EC1,ER1 |
| 1187 | | | | | |
| 1188 | 001604 | 070331 | DF35 | | :0,0,0,0,0,0,0 |
| 1189 | | | | | |
| 1190 | | | | | |
| 1191 | 001606 | 062322 | EM37 | ;*ITEM37 | :PGE ERROR |
| 1192 | 001610 | 063313 | DH11 | | :PC |
| 1193 | | | | | :TEST NO |
| 1194 | | | | | :FAILING REG. ADDR |
| 1195 | | | | | :CONTENTS OF RHCS1 |
| 1196 | | | | | :CONTENTS OF RHCS2 |
| 1197 | | | | | :CONTENTS OF RHDS1 |
| 1198 | | | | | :CONTENTS OF RHER1 |
| 1199 | 001612 | 067466 | DT11 | | :\$ERRPC,\$TSTNM,\$BDADR,CS1,CS2,DS1,ER1 |
| 1200 | 001614 | 070160 | DF11 | | :0,0,0,0,0,0 |
| 1201 | | | | | |
| 1202 | | | | | |
| 1203 | 001616 | 062457 | EM40 | ;*ITEM40 | :RHC DID NOT = 0 AFTER A READ OR |
| 1204 | | | | | :WRITE HEADER AND DATA |
| 1205 | 001620 | 067350 | DH40 | | :PC |
| 1206 | | | | | :TEST NO |
| 1207 | | | | | :CONTENTS OF RHC |
| 1208 | 001622 | 070132 | DT40 | | :\$ERRPC,TSTNM,\$BDDAT |
| 1209 | 001624 | 070360 | DF40 | | :0,0,0 |
| 1210 | | | | | |
| 1211 | | | | | ***** |
| 1212 | | | | | ;*RH11/RH70 REGISTERS |
| 1213 | | | | | ***** |
| 1214 | | | | | |
| 1215 | | | | | |

```
1216
1217      ;*WORD COUNT REGISTER (RHWC)
1218      ;*EACH BIT IS CALLED BY BIT NUMBER
1219
1220
1221
1222      ;*BUS ADDRESS REGISTER (RHBA)
1223      ;*EACH BIT IS CALLED BY BIT NUMBER
1224
1225
1226
1227      ;*CONTROL AND STATUS REGISTER 2 (RHCS2)
1228
1229      000001      US1=      1      ;UNIT SELECT (BIT #0)
1230      000002      US2=      2      ;UNIT SELECT (BIT #1)
1231      000004      US4=      4      ;UNIT SELECT (BIT #2)
1232      000010      BA1=     10      ;BUS ADDRESS INCREMENT INHIBIT (BIT #3)
1233      000020      PAT=     20      ;INVERT PARITY ON MASS BUS TO EVEN (BIT #4)
1234      000040      CLR=     40      ;CLEAR (BIT #5)
1235      000100      IR=     100     ;INPUT READY (BIT #6)
1236      000200      OR=     200     ;OUTPUT READY (BIT #7)
1237      000400      MPE=    400     ;MASS BUS PARITY ERROR (BIT #8)
1238      001000      MXF=   1000     ;MISSED TRANSFER ERROR (BIT #9)
1239      002000      PGE=   2000     ;PROGRAM ERROR (BIT #10)
1240      004000      NEM=   4000     ;NON EXISTANT MEMORY (BIT #11)
1241      010000      NED=  10000     ;NON EXISTANT DRIVE (BIT #12)
1242      020000      UPE=  20000     ;UNIBUS PARITY ERROR (BIT #13)
1243      040000      WCE=  40000     ;WRITE CHECK ERROR (BIT #14)
1244      100000      DLT= 100000     ;DATA LATE (BIT #15)
1245
1246      ;*DATA BUFFER REGISTER (RHDB)
1247      ;*EACH BIT IS CALLED BY BIT NUMBER
1248
1249
1250
1251      ;*****
1252      ;*RP04 REGISTERS
1253      ;*****
1254
1255
1256
1257      ;*CONTROL AND STATUS 1 REGISTER. (#00)
1258
1259      000001      GO=      1      ;GO (BIT #0)
1260      000100      IE=     100     ;INTERRUPT ENABLE (BIT #6)
1261      000200      RDY=    200     ;READY (BIT #7)
1262      000400      A16=   400     ;HIGH ORDER UNIBUS BITS (BIT #8)
1263      001000      A17=  1000     ;HIGH ORDER UNIBUS BITS (BIT #9)
1264      002000      PSEL=  2000     ;PORT SELECT (BIT #10)
1265      004000      DVA=   4000     ;DEVICE AVAILABLE (BIT #11)
1266      020000      MCPE= 20000     ;MASSBUSS PARITY ERROR (BIT #13)
1267      040000      TRE=   40000     ;TRANSFER ERROR (BIT #14)
1268      100000      SC=  100000     ;SPECIAL CONDITION (BIT #15)
1269
1270      ;*STATUS REGISTER (RHDS1) (#01)
1271
```

| | | | |
|------|--------|--|--|
| 1272 | 000001 | DF5= 1 | :DRIVE FORWARD 5''/SEC. (BIT #0) |
| 1273 | 000002 | DFF20= 2 | :DRIVE FORWARD 20''/SEC. (BIT #1) |
| 1274 | 000004 | DIGB= 4 | :DRIVE TO INNER GAVRD BAND (BIT #2) |
| 1275 | 000010 | GRV= 10 | :GO REVERSE (BIT #3) |
| 1276 | 000020 | DL64= 20 | :DIFFERENCE LESS THAN 64 (BIT #4) |
| 1277 | 000040 | DE1= 40 | :DIFFERENCE EQUALS 1 (BIT #5) |
| 1278 | 000100 | VV= 100 | :VOLUME VALID (BIT #6) |
| 1279 | 000200 | DRY= 200 | :DRIVE READY (BIT #7) |
| 1280 | 000400 | DPR= 400 | :DRIVE PRESENT (BIT #8) |
| 1281 | 001000 | PROG= 1000 | :PROGRAMABLE (BIT #9) |
| 1282 | 002000 | LST= 2000 | :LAST SECTOR TRANSFERRED (BIT #10) |
| 1283 | 004000 | WRL= 4000 | :WRITE LOCK (BIT #11) |
| 1284 | 010000 | MOL= 10000 | :MEDIUM ON-LINE (BIT #12) |
| 1285 | 020000 | PIP= 20000 | :POSITIONING OPERATION IN PROGRESS (BIT #13) |
| 1286 | 040000 | ERR= 40000 | :COMPOSIT ERROR. (BIT #14) |
| 1287 | 100000 | ATA= 100000 | :ATTENTION ACTIVE (BIT #15) |
| 1288 | | | |
| 1289 | | :*ERROR REGISTER #01 (RHER1) (#02) | |
| 1290 | 000001 | ILF= 1 | :ILLEGAL FUNCTION (BIT #0) |
| 1291 | 000002 | ILR= 2 | :ILLEGAL REGISTER (BIT #1) |
| 1292 | 000004 | RMR= 4 | :REGISTER MODIFICATION REFUSED (BIT #2) |
| 1293 | 000010 | PAR= 10 | :PARITY ERROR (BIT #3) |
| 1294 | 000020 | FER= 20 | :FORMAT ERROR (BIT #4) |
| 1295 | 000040 | WCF= 40 | :WRITE CLOCK FAIL (BIT #5) |
| 1296 | 000100 | ECH= 100 | :ECC HARD ERROR (BIT #6) |
| 1297 | 000200 | HCE= 200 | :HEADER COMPARE ERROR (BIT #7) |
| 1298 | 000400 | HCRC= 400 | :HEADER CRC ERROR (BIT #8) |
| 1299 | 001000 | AOE= 1000 | :ADDRESS OVERFLOW ERROR (BIT #9) |
| 1300 | 002000 | IAE= 2000 | :INVALID ADDRESS ERROR (BIT #10) |
| 1301 | 004000 | WLE= 4000 | :WRITE LOCK ERROR (BIT #11) |
| 1302 | 010000 | DTE= 10000 | :DRIVE TIMING ERROR (BIT #12) |
| 1303 | 020000 | OPI= 20000 | :OPERATION INCOMPLETE (BIT #13) |
| 1304 | 040000 | UNS= 40000 | :DRIVE UNSAFE (BIT #14) |
| 1305 | 100000 | DCK= 100000 | :DATA CHECK ERROR (BIT 15) |
| 1306 | | | |
| 1307 | | :*MAINTAINABILITY REGISTER (RHMR) (#03) | |
| 1308 | | | |
| 1309 | 000001 | DMD= 1 | :DIAGINOSTIC MODE (BIT #0) |
| 1310 | 000002 | MCLK= 2 | :MAINTAINABILITY CLOCK (BIT #1) |
| 1311 | 000004 | MINX= 4 | :MAINTAINABILITY INDEX (BIT #2) |
| 1312 | 000010 | MSTCK= 10 | :MAINTAINABILITY SECTOR CLOCK (BIT #3) |
| 1313 | 000020 | MRD= 20 | :MAINTAINABILITY READ (BIT #4) |
| 1314 | 000040 | MWR= 40 | :MAINTAINABILITY WRITE (BIT #5) |
| 1315 | 000200 | DENVL= 200 | :DATA ENVELOPE (BIT #7) |
| 1316 | 000400 | ZER= 400 | :ZERO DETECT (BIT #8) |
| 1317 | 001000 | DTSY= 1000 | :MAINTAINABILITY SYNC DETECTED (BIT #9) |
| 1318 | | | |
| 1319 | | :*ATTENTION SUMMARY PSEUDO-REGISTER (RHAS) (#04) | |
| 1320 | | | |
| 1321 | 000001 | AT0= 1 | :DEVICE 0 (BIT #0) |
| 1322 | 000002 | AT1= 2 | :DEVICE 1 (BIT #1) |
| 1323 | 000004 | AT2= 4 | :DEVICE 2 (BIT #2) |
| 1324 | 000010 | AT3= 10 | :DEVICE 3 (BIT #3) |
| 1325 | 000020 | AT4= 20 | :DEVICE 4 (BIT #4) |
| 1326 | 000040 | AT5= 40 | :DEVICE 5 (BIT #5) |
| 1327 | 000100 | AT6= 100 | :DEVICE 6 (BIT #6) |

| | | | |
|------|--------|---------------|--|
| 1328 | 000200 | AT7= 200 | :DEVICE 7 (BIT #7) |
| 1329 | | | |
| 1330 | | | |
| 1331 | | | |
| 1332 | | | |
| 1333 | | | |
| 1334 | | | :*DESIRED SECTOR/TRACK ADDRESS REGISTER (RHDST) (#1) |
| 1335 | | | :*EACH BIT IS CALLED BY BIT NUMBER |
| 1336 | | | |
| 1337 | | | |
| 1338 | | | |
| 1339 | | | |
| 1340 | | | |
| 1341 | | | :*DRIVE TYPE REGISTER (RHDT) (#06) |
| 1342 | | | :*EACH BIT IS CALLED BY BIT NUMBER |
| 1343 | | | |
| 1344 | | | |
| 1345 | | | |
| 1346 | | | |
| 1347 | | | |
| 1348 | | | :*LOOK-AHEAD REGISTER (RHLA) (#07) |
| 1349 | | | |
| 1350 | 000001 | EXT1= 1 | :EXTENSION 1 (BIT #0) |
| 1351 | 000002 | EXT2= 2 | :EXTENSION 2 (BIT #1) |
| 1352 | 000004 | EXT4= 4 | :EXTENSION 3 (BIT #2) |
| 1353 | 000010 | EXT10= 10 | :EXTENSION 4 (BIT #3) |
| 1354 | 000020 | EXT20= 20 | :EXTENSION 5 (BIT #4) |
| 1355 | 000040 | EXT40= 40 | :EXTENSION 6 (BIT #5) |
| 1356 | 000100 | SC1= 100 | :SECTOR COUNT FIELD 0 (BIT #6) |
| 1357 | 000200 | SC2= 200 | :SECTOR COUNT FIELD 1 (BIT #7) |
| 1358 | 000400 | SC4= 400 | :SECTOR COUNT FIELD 2 (BIT #8) |
| 1359 | 001000 | SC10= 1000 | :SECTOR COUNT FIELD 3 (BIT #9) |
| 1360 | 002000 | SC20= 2000 | :SECTOR COUNT FIELD 4 (BIT #10) |
| 1361 | 004000 | TRK1= 4000 | :TRACK FIELD 1 (BIT #11) |
| 1362 | 010000 | TRK2= 10000 | :TRACK FIELD 2 (BIT #12) |
| 1363 | 020000 | TRK4= 20000 | :TRACK FIELD 3 (BIT #13) |
| 1364 | 040000 | TRK10= 40000 | :TRACK FIELD 4 (BIT #14) |
| 1365 | 100000 | TRK20= 100000 | :TRACK FIELD 5 (BIT #15) |
| 1366 | | | |
| 1367 | | | :*RP04 ERROR REGISTER #2 (RHER2) (#10) |
| 1368 | | | |
| 1369 | 000001 | WCU= 1 | :WRITE CURRENT UNSAFE (BIT #0) |
| 1370 | 000002 | CSF= 2 | :CURRENT SINK FAILURE (BIT #1) |
| 1371 | 000004 | WSU= 4 | :WRITE SELECT UNSAFE (BIT #2) |
| 1372 | 000010 | CSU= 10 | :CURRENT SWITCH UNSAFE (BIT #3) |
| 1373 | 000020 | MSE= 20 | :MOTOR SEQUENCE ERROR (BIT #4) |
| 1374 | 000040 | TDF= 40 | :TRANSITIONS DETECTOR FAILURE (BIT #5) |
| 1375 | 000100 | TUF= 100 | :TRANSITIONS UNSAFE (BIT #6) |
| 1376 | 000200 | FEN= 200 | :FAILSAFE ENABLED (BIT #7) |
| 1377 | 000400 | WRU= 400 | :WRITE READY UNSAFE (BIT #8) |
| 1378 | 001000 | MHS= 1000 | :MULTIPLE HEAD SELECT (BIT #9) |
| 1379 | 002000 | NHS= 2000 | :NO HEAD SELECTION (BIT #10) |
| 1380 | 004000 | IXE= 4000 | :INDEX ERROR (BIT #11) |
| 1381 | 010000 | VU30= 10000 | :30VOLT UNSAFE (BIT #12) |
| 1382 | 020000 | PLU= 20000 | :PLO UNSAFE (BIT #13) |
| 1383 | 100000 | ACU= 100000 | :ACUNSAFE (BIT #15) |

```
1384
1385
1386
1387      000001      WCU=      1      ;WRITE CURENT UNSAFE
1388      000002      CSF=      2      ;CURRENT SINK FAILURE
1389      000004      WSU=      4      ;CURRENT SELECT UNSAFE
1390      000010      CSU=     10      ;CURRENT SWITCH UNSAFE
1391      000020      RAW=     20      ;READ AND WRITE
1392      000040      TDF=     40      ;TRANSITIONS DETECTOR FAILURE
1393      000100      TUF=    100      ;TRANSITIONS UNSAFE
1394      000200      ABS=    200      ;ABNORMAL STOP
1395      000400      WRU=    400      ;WRITE READY UNSAFE
1396      001000      MHS=   1000      ;MULTIPLE HEAD SELECT
1397      002000      NHS=   2000      ;NO HEAD SELECTION
1398      004000      IXE=   4000      ;INDEX ERROR
1399      020000      PLU=  20000      ;PLO UNSAFE
1400
1401      ;*OFFSET REGISTER (RHOF) (#11)
1402
1403      000001      OF25=     1      ;OFFSET 25 MICRO INCHES (BIT #0)
1404      000002      OF50=     2      ;OFFSET 50 MICRO INCHES (BIT #1)
1405      000004      OF100=    4      ;OFFSET 100 MICRO INCHES (BIT #2)
1406      000010      OF200=   10      ;OFFSET 200 MICRO INCHES (BIT #3)
1407      000020      OF400=   20      ;OFFSET 400 MICRO INCHES (BIT #4)
1408      000040      OF800=   40      ;OFFSET 800 MICRO INCHES (BIT #5)
1409
1410      000200      OFREV=   200      ;OFFSET NEGATIVE (REVERSE) (BIT #7)
1411      002000      HCI=   2000      ;HEADER COMPARE INHIBIT (BIT #10)
1412      004000      ECI=   4000      ;ERROR CORRECTION CODE INHIBIT (BIT #11)
1413      010000      FMT22= 10000      ;FORMAT BIT (BIT #12)
1414
1415
1416
1417
1418
1419
1420      ;*DESIRED CYLINDER ADDRESS (RHCA) (#12)
1421      ;*EACH BIT IS CALLED BY BIT NUMBER.
1422
1423
1424
1425
1426
1427      ;*CURRENT CYLINDER ADDRESS (RHCC) (#13)
1428      ;*EACH BIT IS CALLED BY BIT NUMBER
1429
1430
1431
1432
1433
1434      ;*SERIAL NUMBER REGISTER (RHSN) (#14)
1435      ;*EACH IS CALLED BY BIT NUMBER
1436
1437
1438
1439
```

| | | | |
|------|--------|--------------|--|
| 1440 | | | |
| 1441 | | | |
| 1442 | | | |
| 1443 | 000001 | PSU= 1 | :PACK SPEED UNSAFE (BIT #0) |
| 1444 | 000002 | VUF= 2 | :VELOCITY UNSAFE (BIT #1) |
| 1445 | 000010 | UWR= 10 | :ANY UNSAFE EXCEPT READ/WRITE (BIT #3) |
| 1446 | 000020 | PRE= 20 | :DISK PACK ROTATION ERROR (BIT #4) |
| 1447 | 000040 | ACL= 40 | :AC LOW (BIT #5) |
| 1448 | 000100 | DCL= 100 | :DC LOW (BIT #6) |
| 1449 | 040000 | SKI= 40000 | :SEEK INCOMPLETE (BIT #14) |
| 1450 | 100000 | OCYL= 100000 | :OFF CYLINDER (BIT #15) |
| 1451 | | | |
| 1452 | | | |
| 1453 | | | |
| 1454 | | | |
| 1455 | | | |
| 1456 | | | |
| 1457 | | | |
| 1458 | | | |
| 1459 | | | |
| 1460 | | | |
| 1461 | | | |
| 1462 | | | |
| 1463 | | | |
| 1464 | | | |
| 1465 | | | |
| 1466 | | | |

:*ECC POSITION REGISTER (RHEC1) (#16)
:*EACH BIT IS CALLED BY BIT NUMBER

:*ECC PATTERN REGISTER (RHEC2) (#17)
:*EACH BIT IS CALLED BY BIT NUMBER

```
1467  
1468  
1469  
1470  
1471  
1472  
1473  
1474  
1475 001626 000254  
1476  
1477  
1478  
1479  
1480  
1481  
1482  
1483  
1484  
1485 001630 176722  
1486 001632 176702  
1487 001634 176704  
1488 001636 176710  
1489  
1490  
1491  
1492  
1493  
1494  
1495  
1496  
1497 001640 176700  
1498 001642 176714  
1499 001644 176706  
1500 001646 176740  
1501 001650 176732  
1502 001652 176734  
1503 001654 176742  
1504 001656 176716  
1505 001660 176724  
1506 001662 176712  
1507 001664 176726  
1508 001666 176730  
1509 001670 176744  
1510 001672 176746  
1511 001674 176720  
1512 001676 176736  
1513  
1514  
1515  
1516 001700 176750  
1517 001702 176752
```

.SBTTL REGISTER ADDRESSES

;*RP04 VECTOR ADDRESS

RPVEC: 254 ;RP04 VECTOR ADDRESS

;*RP04/5/6 DISK I/O REGISTERS LOCATED IN THE RH11 CONTROLLER
;*NOTE: THE CONTENTS OF THESE LOCATIONS WILL BE DIFFERENT
;* IF THE "CHANGE BASE ADDRESS" ROUTINE IS USED.
;* THIS ROUTINE STARTS AT LOCATION TAGED "BASECH"

RHDB: 176722 ;DATA BUFFER SEE NOTE ABOVE
RHWC: 176702 ;WORD COUNT SEE NOTE ABOVE
RHBA: 176704 ;BUS ADDRESS SEE NOTE ABOVE
RHCS2: 176710 ;CONTROL AND STATUS 2 SEE NOTE ABOVE

;*RP04/5/6 DISK I/O REGISTERS LOCATED IN THE DEVICE CONTROL LOGIC (DCL)
;*NOTE: THE CONTENTS OF THESE LOCATIONS WILL BE DIFFERENT
;* IF THE "CHANGE BASE ADDRESS ROUTINE IS USED.
;* THIS ROUTINE STARTS AT LOCATION TAGED "BASECH"

RHCS1: 176700 ;CONTROL AND STATUS 1 SEE NOTE ABOVE
RHER1: 176714 ;ERROR #1 SEE NOTE ABOVE
RHDST: 176706 ;DESIRED SECTOR/TRACK ADDRESS SEE NOTE ABOVE
RHER2: 176740 ;ERROR #2 SEE NOTE ABOVE
RHOF: 176732 ;OFFSET SEE NOTE ABOVE
RHCA: 176734 ;DESIRED CYLINDER ADDRESS SEE NOTE ABOVE
RHER3: 176742 ;ERROR #3 SEE NOTE ABOVE
RHAS: 176716 ;ATTENTION SUMMARY SEE NOTE ABOVE
RHMR: 176724 ;MAINTAINABILITY SEE NOTE ABOVE
RHDS1: 176712 ;DRIVE STATUS SEE NOTE ABOVE
RHDT: 176726 ;DRIVE TYPE SEE NOTE ABOVE
RHSN: 176730 ;SERIAL NUMBER SEE NOTE ABOVE
RHEC1: 176744 ;ECC POSITION SEE NOTE ABOVE
RHEC2: 176746 ;ECC PATTERN SEE NOTE ABOVE
RHLA: 176720 ;LOOK-AHEAD SEE NOTE ABOVE
RHCC: 176736 ;CURRENT CYLINDER ADDRESS SEE NOTE ABOVE

;*ADDITIONAL REGISTERS LOCATED IN THE RH70 CONTROLLER LOGIC

RHBAE: 176750 ;BUS ADDRESS EXTENSION REGISTER
RHCS3: 176752 ;CONTROL AND STATUS REGISTER #3

1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546

001704 000000
001706 000000
001710 000000
001712 000000

001714 000000
001716 000000
001720 000000
001722 000000
001724 000000
001726 000000
001730 000000
001732 000000
001734 000000
001736 000000
001740 000000
001742 000000
001744 000000
001746 000000
001750 000000
001752 000000

; *THE FOLLOWING LOCATIONS ARE RESERVED FOR REGISTER SAVES
; *ANY TIME THERE IS AN ERROR ALL THESE WILL BE FILLED
; *ONLY SOME MAY BE PRINTED BUT ALL WILL BE FILLED TRUE
; *FOR THE TIME JUST AFTER THE "ERROR" ERROR COMMAND

DB: 0 ;DATA BUFFER
WC: 0 ;WORD COUNT
BA: 0 ;BUS ADDRESS
CS2: 0 ;CONTROL AND STATUS 2

CS1: 0 ;CONTROL AND STATUS 1
ER1: 0 ;ERROR #1
DST: 0 ;DESIRED SECTOR/TRACK ADDRESS
ER2: 0 ;ERROR #2
OF: 0 ;OFFSET
CA: 0 ;DESIRED CYLINDER ADDRESS
ER3: 0 ;ERROR #3
AS: 0 ;ATTENTION SUMMARY
MR: 0 ;MAINTAINABILITY
DS1: 0 ;DRIVE STATUS
DT: 0 ;DRIVE TYPE
SN: 0 ;SERIAL NUMBER
EC1: 0 ;ECC POSITION
EC2: 0 ;ECC PATTERN
LA: 0 ;LOOK-AHEAD
CC: 0 ;CURRENT CYLINDER ADDRESS

| | | | | | |
|------|--------|--------|---------------|----|---|
| 1547 | | | | | |
| 1548 | | | | | |
| 1549 | | | | | |
| 1550 | 001754 | 000010 | UNITS: .BLKW | 8. | ;TABLE OF DRIVES PRESENT TO TEST |
| 1551 | 001774 | 000000 | UNIT: .WORD | 0 | ;UNIT UNDER TEST |
| 1552 | 001776 | 000000 | NOUNIT: .WORD | 0 | ;NUMBER OF UNITS PRESENT |
| 1553 | | | | | ;USED TO KEEP TRACK OF UNIT UNDER TEST |
| 1554 | 002000 | 000000 | NUNIT: .WORD | 0 | ;USED TO DETERMINE IF THERE IS MORE |
| 1555 | | | | | ;THAN ONE UNIT |
| 1556 | 002002 | 000000 | SELECT: .WORD | 0 | ;ALL ONES INDICATE UNIT TO BE SELECTED |
| 1557 | 002004 | 000000 | UNITSL: .WORD | 0 | ;UNIT NO. SELECTED |
| 1558 | | | | | |
| 1559 | 002006 | 000000 | ERFLG\$: | 0 | ;ERROR FLAG |
| 1560 | | | | | |
| 1561 | 002010 | 000000 | SAVDT: | 0 | ;SAVE DRIVE TYPE REGISTER |
| 1562 | | | | | ;FOR COMPARISON IN DRIVE CLEAR TEST |
| 1563 | | | | | ;AND RH INIT TEST |
| 1564 | 002012 | 000000 | SAVSN: | 0 | ;SAVE SERIAL NUMBER REGISTER |
| 1565 | | | | | ;FOR COMPARISON IN DRIVE CLEAR TEST |
| 1566 | | | | | ;AND RH INIT TEST |
| 1567 | | | | | |
| 1568 | 002014 | 000000 | PCJSR: | 0 | ;SAVE PC OF JSR WHICH GAVE THE ERROR |
| 1569 | | | | | |
| 1570 | 002016 | 000000 | ATTENT: | 0 | ;ATTENTION BIT FOR PRESENT UNIT |
| 1571 | 002020 | 000000 | TOTALAT: | 0 | ;TATAL ATTENTION BITS |
| 1572 | | | | | |
| 1573 | 002022 | 000000 | TMPILL: | 0 | ;TEMPORARY ILLEGAL FUNCTION |
| 1574 | | | | | |
| 1575 | 002024 | 000000 | TSECC: | 0 | ;FLAG TO SAY IF ECC TEST OR NOT |
| 1576 | | | | | ;WHEN =177777 IT IS AN ECC TEST |
| 1577 | | | | | ;WHEN =0IT IS NOT AN ECC TEST |
| 1578 | | | | | |
| 1579 | 002026 | 000000 | TESDTE: | 0 | ;FLAG TO SAY IF DRIVE TIMING ERROR OR NOT |
| 1580 | | | | | ;WHEN = 177777 IT IS A DTE TEST |
| 1581 | | | | | ;WHEN = 0 IT IS NOT A DTE TEST |
| 1582 | | | | | |
| 1583 | 002030 | 000000 | TAGDTE: | 0 | ;TEMPORARY TAG USED IN DRIVE TIMING |
| 1584 | | | | | ;ERROR TEST |
| 1585 | | | | | |
| 1586 | 002032 | 000000 | TSTNM: | 0 | ;TEST NUMBER |
| 1587 | | | | | |
| 1588 | 002034 | 000000 | FIRST: | 0 | ;IF ZERO WILL TYPE HEADER |
| 1589 | | | | | |
| 1590 | 002036 | 000000 | RP06: | 0 | ;IF 0 PROGRAM WILL TREAT DRIVE AS RP04 |
| 1591 | | | | | |
| 1592 | 002040 | 000000 | RH70: | 0 | ;IF 1 PROGRAM IS RUNNING ON RH70 |
| 1593 | | | | | ;IF 0 PROGRAM IS ON AN RH11 |
| 1594 | 002042 | 000000 | SILOSZ: .WORD | 0 | ;RH SILO SIZE |

```
1595
1596
1597
1598
1599
1600
1601 002044
1602 002044 000000
1603 002046 000002
1604 002050 000006
1605 002052 000010
1606 002054 000012
1607 002056 000030
1608 002060 000050
1609 002062 000052
1610 002064 000060
1611 002066 000062
1612 002070 000070
1613 002072 000072
1614 002074 000004
1615 002076 000014
1616 002100 000016
1617 002102 000022
1618 002104 000020
1619 002106 000000
1620
1621
1622
1623
1624
1625 002114 000422
1626 003160 000422
1627
1628
1629
1630
1631
1632
1633 004224 001 002 004
1634 004227 010 020 040
1635 004232 100 200
1636
```

;*FUNCTION EQUATES

;*TABLE OF FUNCTIONS FOR RHCS1, THEN "GO" BIT HAS TO BE SET

| | | | | |
|---------|-------|--|--|---------------------------------|
| FUTABL: | | | | |
| NOPEA: | 0 | | | :NO OPERATION |
| UNLOAD: | 2 | | | :UNLOAD (STAND BY) |
| RECALI: | 6 | | | :RECALIBRATE |
| DCLEAR: | 10 | | | :DRIVE CLEAR |
| RELEAS: | 12 | | | :RELEASE (DUAL-PORT OPERATION) |
| SERCH: | 30 | | | :SEARCH COMMAND |
| WRCHK: | 50 | | | :WRITE CHECK DATA |
| WRCHDT: | 52 | | | :WRITE CHECK HEADER AND DATA |
| WRIDAT: | 60 | | | :WRITE DATA |
| WRIFOR: | 62 | | | :WRITE HEADER AND DATA (FORMAT) |
| READAT: | 70 | | | :READ DATA |
| REFOR: | 72 | | | :READ HEADER AND DATA |
| SEECOM: | 4 | | | :SEEK COMMAND |
| OFSETC: | 14 | | | :OFFSET COMMAND |
| RETCL: | 16 | | | :RETURN TO CENTERLINE |
| PKACK: | 22 | | | :PACK ACKNOWLEDGE |
| READIN: | 20 | | | :READ IN |
| ILLEGL: | .WORD | | | :COMPUTED ILLEGAL FUNCTION |

;*DATA BUFFERS FOR READ WRITE

| | | | | |
|---------|-------|------|--|-------------------------|
| WRFROM: | .BLKW | 274. | | :WRITE FROM THIS BUFFER |
| REINTO: | .BLKW | 274. | | :READ INTO THIS BUFFER |

;*TABLE FOR ATTENTION BITS

;*ATTENTION TABLE

| | | | | |
|---------|-------|------------------------|--|--|
| ATABLE: | .BYTE | 1,2,4,10,20,40,100,200 | | |
|---------|-------|------------------------|--|--|

```
1637
1638
1639 .SBTTL ***DIAGNOSTIC CODE***
1640 .SBTTL
1641
1642 .SBTTL SETUP TESTS
1643
1644 004234 012737 177777 002002 BEGIN2: MOV #-1,@#SELECT ;SELECT UNIT
1645 004242 000402 BR START
1646 004244 005037 002002 BEGIN: CLR @#SELECT ;DO NOT SELECT UNIT
1647 ;NORMAL RUN
1648
1649 004250 START:
1650 004250 000005 RESET
1651 .SBTTL INITIALIZE THE COMMON TAGS
1652 ;;CLEAR THE COMMON TAGS (%CMTAG) AREA
1653 004252 012706 001100 MOV #CMTAG,R6 ;;FIRST LOCATION TO BE CLEARED
1654 004256 005026 CLR (R6)+ ;;CLEAR MEMORY LOCATION
1655 004260 022706 001140 CMP #SWR,R6 ;;DONE?
1656 004264 001374 BNE -6 ;;LOOP BACK IF NO
1657 004266 012706 001000 MOV #STACK,SP ;;SETUP THE STACK POINTER
1658 ;;INITIALIZE A FEW VECTORS
1659 004272 012737 054332 000020 MOV #SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
1660 004300 012737 000340 000022 MOV #340,@#IOTVEC+2 ;;LEVEL 7
1661 004306 012737 056550 000030 MOV #ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
1662 004314 012737 000340 000032 MOV #340,@#EMTVEC+2 ;;LEVEL 7
1663 004322 012737 057320 000034 MOV #STRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
1664 004330 012737 000340 000036 MOV #340,@#TRAPVEC+2;LEVEL 7
1665 004336 012737 057410 000024 MOV #SPWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
1666 004344 012737 000340 000026 MOV #340,@#PWRVEC+2 ;;LEVEL 7
1667 004352 005037 001212 CLR $TIMES ;;INITIALIZE NUMBER OF ITERATIONS
1668 004356 005037 001214 CLR $ESCAPE ;;CLEAR THE ESCAPE ON ERROR ADDRESS
1669 004362 112737 000001 001115 MOVB #1,$ERMAX ;;ALLOW ONE ERROR PER TEST
1670 004370 012737 004370 001106 MOV #,$LPADR ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
1671 004376 012737 004376 001110 MOV #,$LPERR ;;SETUP THE ERROR LOOP ADDRESS
1672 ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
1673 ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
1674 004404 013746 000004 MOV @#ERRVEC,-(SP) ;;SAVE ERROR VECTOR
1675 004410 012737 004444 000004 MOV #64$,@#ERRVEC ;;SET UP ERROR VECTOR
1676 004416 012737 177570 001140 MOV #DSWR,SWR ;;SETUP FOR A HARDWARE SWICH REGISTER
1677 004424 012737 177570 001142 MOV #DDISP,DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
1678 004432 022777 177777 174500 CMP #-1,@SWR ;;TRY TO REFERENCE HARDWARE SWR
1679 004440 001012 BNE 66$ ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
1680 ;;AND THE HARDWARE SWR IS NOT = -1
1681 004442 000403 BR 65$ ;;BRANCH IF NO TIMEOUT
1682 004444 012716 004452 64$: MOV #65$,(SP) ;;SET UP FOR TRAP RETURN
1683 004450 000002 RTI
1684 004452 012737 000176 001140 65$: MOV #SWREG,SWR ;;POINT TO SOFTWARE SWR
1685 004460 012737 000174 001142 MOV #DISPREG,DISPLAY
1686 004466 012637 000004 66$: MOV (SP)+,@#ERRVEC ;;RESTORE ERROR VECTOR
1687
1688
1689
1690
1691 004472 012737 000000 177776 STARTA: MOV #0,PS ;SET PROCESSOR STATUS TO 0
1692 004500 012777 054250 175120 MOV #RPVECT,@RPVEC ;THIS IS FOR UNTIMELY DRIVE INTERRUPTS
```

```

1693 004506 004737 055310 JSR PC,@#STKINT ;INITIALIZE THE TTY KEYBOARD
1694 004512 005737 002034 TST @#FIRST ;IS THIS FIRST TIME ROUND ?
1695 004516 001001 BNE 1$ ;SKIP HEADER IF NOT
1696 004520 000402 BR 2$ ;DO HEADER IF SO
1697
1698 004522 000137 005332 1$: JMP @#SND1 ;SKIP OVERALL PROGRAM HEADER
1699 004526 000402 2$:
1700 004526 104401 004534 TYPE ,65$ ;;TYPE ASCIZ STRING
1701 004532 000434 BR 64$ ;;GET OVER THE ASCIZ
1702 ;;65$: .ASCIZ <15><12>?RP04/5/6 DISKLESS CONTROLLER TEST - PART I - CZRJG-D?
1703 004624 64$:
1704
1705 004624 104401 004632 TYPE ,67$ ;;TYPE ASCIZ STRING
1706 004630 000417 BR 66$ ;;GET OVER THE ASCIZ
1707 ;;67$: .ASCIZ <15><12>/REVISION DATE: 01-MAY-79/<15><12>
1708 004670 66$:
1709
1710 004670 104401 004676 TYPE ,69$ ;;TYPE ASCIZ STRING
1711 004674 000433 BR 68$ ;;GET OVER THE ASCIZ
1712 ;;69$: .ASCIZ <15><12>/ALL DCL'S UNDER TEST MUST BE LOCKED ON CORRECT PORT/
1713 004764 68$:
1714 004764 104401 004772 TYPE ,71$ ;;TYPE ASCIZ STRING
1715 004770 000427 BR 70$ ;;GET OVER THE ASCIZ
1716 ;;71$: .ASCIZ <15><12>/IF CHANGES ARE REQUIRED ON PORT SWITCH THEN/
1717 005050 70$:
1718 005050 104401 005056 TYPE ,73$ ;;TYPE ASCIZ STRING
1719 005054 000430 BR 72$ ;;GET OVER THE ASCIZ
1720 ;;73$: .ASCIZ <15><12>/A CYCLE UP SEQUENCE IS REQUIRED FOR STROBING/
1721 005136 72$:
1722 005136 104401 005144 TYPE ,75$ ;;TYPE ASCIZ STRING
1723 005142 000415 BR 74$ ;;GET OVER THE ASCIZ
1724 ;;75$: .ASCIZ <15><12>/THE PORT SELECT FLOP/<15><12>
1725 005176 74$:
1726 005176 104401 005204 TYPE ,77$ ;;TYPE ASCIZ STRING
1727 005202 000430 BR 76$ ;;GET OVER THE ASCIZ
1728 ;;77$: .ASCIZ <15><12>/ALL DCL'S NOT UNDER TEST MUST BE SWITCHED OFF/
1729 005264 76$:
1730 005264 104401 005272 TYPE ,79$ ;;TYPE ASCIZ STRING
1731 005270 000420 BR 78$ ;;GET OVER THE ASCIZ
1732 ;;79$: .ASCIZ <15><12>/OR LOCKED ON THE OTHER PORT/<15><12>
1733 005332 78$:
1734
1735 005332 012737 177777 002034 SND1: MOV #-1,@#FIRST ;NEXT TIME DO NOT GIVE HEADER
1736
1737 .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
1738 005340 005737 000042 TST @#42 ;;ARE WE RUNNING UNDER XXDP/ACT?
1739 005344 001006 BNE 64$ ;;BRANCH IF YES
1740 005346 023727 001140 000176 CMP SWR,#SWREG ;;SOFTWARE SWITCH REG SELECTED?
1741 005354 001005 BNE 65$ ;;BRANCH IF NO
1742 005356 104406 GTSWR ;;GET SOFT-SWR SETTINGS
1743 005360 000403 BR 65$
1744 005362 112737 000001 001134 64$: MOVB #1,$AUTOB ;;SET AUTO-MODE INDICATOR
1745 005370 65$:
1746
1747 005370 032777 010000 173542 RH70CK: BIT #SW12,@SWR ;LOOK TO SEE IF USING RH70
1748 005376 001403 BEQ 3$ ;IF SW12 = 0, SKIP NEXT

```

```
1749 005400 012737 000001 002040      MOV      #1,@#RH70      ;IF SW12 = 1, CU IS AN RH70
1750                                     3$:      TST      @#SELECT      ;200 START?
1751 005406 005737 002002                BEQ      TST1          ;SKIP NEXT IF STARTING FROM 200 -----)
1752 005412 001434                        TYPE     ,65$          ;;TYPE ASCIZ STRING
1753 005414 104401 005422                BR       64$          ;;GET OVER THE ASCIZ
1754 005420 000422                        ;;65$: .ASCIZ <15><12>/SELECT UNIT NUMBER TO BE TESTED ?/
1755                                     64$:
1756 005466                                RDOCT
1757 005466 104412                        BIC     #177770,(SP)  ;ONLY KEEP LAST 3 BITS
1758 005470 042716 177770                MOV     (SP),@#UNIT  ;SAVE UNIT TO BE TESTED
1759 005474 011637 001774                MOV     (SP)+,@#UNITSL ;SAVE UNIT TO BE TESTED
1760 005500 012637 002004
1761
1762
```

```
1763
1764
1765
1766
1767
1768
1769
1770
1771 005504 000004
1772 005506 012737 000001 001212
1773 005514 012706 001000
1774 005520 012737 000001 002032
1775
1776 005526 012737 056560 000030
1777
1778 005534 012737 005562 000004
1779 005542 012700 000024
1780 005546 012701 001630
1781 005552 013102 1$:
1782 005554 005300
1783 005556 001375
1784 005560 000454
1785 005562 012737 000006 000004 2$:
1786 005570 022626
1787 005572 016137 177776 001200
1788 005600 104015
1789 005602 032777 020000 173330
1790 005610 001036
1791
1792 005612 104401 005620
1793 005616 000427
1794
1795 005676
1796
1797 005676 012746 000204
1798
1799 005702 104402
1800 005704 000000
1801
1802 005706 000137 041432 4$:
1803
1804 005712 012737 005772 000004 3$:
1805 005720 005037 002040
1806 005724 005777 173750
1807 005730 005237 002040
1808 005734 104401 005742
1809 005740 000413
1810
1811 005770
1812 005770 000417
1813 005772 022626 TRP:
1814 005774 104401 006002
1815 006000 000413
1816
1817 006030
1818 006030 012737 056550 000030 RTN:

```

: *TEST 1 REFERENCE EACH REGISTER
: ** REFERENCE EACH REGISTER BY A MOVE INSTRUCTION

TST1: SCOPE
MOV #1,\$TIMES ;;DO 1 ITERATION
MOV #STACK, SP ;;SET UP STACK POINTER
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
MOV #REGSA1,@#EMTVEC;ERROR VECTOR SO THAT
;NO REGISTERS ARE SAVED
MOV #2\$,@#ERRVEC ;SET UP FOR BUS TIMEOUT
MOV #24,R0 ;THERE ARE 24 REG TO TEST
MOV #RHDB,R1 ;R1 NOW HAS ADDR OF ADDR OF FIRST REG.
1\$: MOV @(R1)+,R2 ;READ HARDWARE REG.
DEC R0 ;COUNT DOWN
BNE 1\$;BRANCH IF 24 NOT DONE
BR 3\$;BRANCH IF 24 DONE
2\$: MOV #ERRVEC+2,@#ERRVEC ;RESTORE TRAP CATCHER
CMP (SP)+,(SP)+ ;CLEAN STACK
MOV -2(R1), \$TMP1 ;STORE FAILING REG ADDR
ERROR 15 ;REGISTER NON EXISTANT
BIT #SW13,@SWR ;INHIBIT ERROR PRINTOUT ?
BNE 4\$;BRANCH IF YES
TYPE ,65\$;;TYPE ASCIZ STRING
BR 64\$;;GET OVER THE ASCIZ
64\$: .ASCIZ <15><12>/TO CHANGE BASE ADDRESS, RESTART AT ADDRESS /
MOV #ADDMOD,-(SP) ;GET READY TO TYPE STARTING ADDRESS
;OF "CHANGE OF BASE ADDRESS" ROUTINE
TYPOC
HALT ;FORCE THE RESTART!
4\$: JMP @#\$EOP ;GO TO END OF PROGRAM ----->
3\$: MOV #TRP,@#4 ;INITIALIZE VECTOR
CLR RH70 ;INIT RH INDICATOR ++ C.W
TST @RHBAE ;ADDRESS RPBAE (RH11/RH70?)
INC RH70 ;FOUND AN RH70-SET MASK
TYPE ,67\$;;TYPE ASCIZ STRING
BR 66\$;;GET OVER THE ASCIZ
66\$: .ASCIZ <15><12>/RH70 CONTROLLER /
TRP: BR RTN
CMP (SP)+,(SP)+ ;ADJUST THE STACK
TYPE ,65\$;;TYPE ASCIZ STRING
BR 64\$;;GET OVER THE ASCIZ
64\$: .ASCIZ <15><12>/RH11 CONTROLLER /
RTN: MOV # \$ERROR,@#EMTVEC;RESTORE ERROR VECTOR

```
1819                                     ;SO THAT REGISTERS ARE SAVED
1820 006036 012737 000006 000004      MOV  #ERRVEC+2,@#ERRVEC ;RESTORE TRAP CATCHER
1821
1822                                     ;FIND THE SILO SIZE
1823
1824 006044 004737 042732                JSR  PC, @#CLDISK      ;CONTROLLER CLEAR
1825 006050 005037 002042                CLR  SILOSZ           ;CLEAR SILO COUNTER
1826 006054 013777 002042 173546 13$:  MOV  SILOSZ,@RHDB      ;LOAD SILO
1827 006062 005237 002042                INC  SILOSZ           ;KEEP COUNT
1828 006066 032777 000100 173542        BIT  #IR, @RHCS2      ;IS THE SILO FULL?
1829 006074 001367                        BNE  13$              ;BRANCH IF NO
1830 006076 012737 000412 037416        MOV  #266., VAR1+2    ;VAR1 IN TEST 116
1831 006104 163737 002042 037416        SUB  SILOSZ, VAR1+2
1832 006112 005437 037416                NEG  VAR1+2
1833 006116 012737 002114 037424        MOV  #WRFROM, VAR2+2 ;VAR2 IN TEST 116
1834 006124 063737 002042 037424        ADD  SILOSZ, VAR2+2
1835 006132 063737 002042 037424        ADD  SILOSZ, VAR2+2
1836 006140 062737 001000 037424        ADD  #512., VAR2+2
1837 006146 004737 042732                JSR  PC, @#CLDISK      ;CONTROLLER CLEAR
```


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

006152 000004
006154 012737 000001 001212
006162 012706 001000
006166 012737 000002 002032

006174 005737 002040
006200 001402
006202 000137 006236
006206

006206 013737 011054 006226
006214 013737 001636 006230
006222 004537 042404
006226 000000
006230 000000
006232 104001
006234 000207

006236 000004
006240 012737 000001 001212
006246 012737 000003 002032
006254 013701 001656
006260 012711 177777
006264 011137 001126
006270 105737 001126
006274 001405
006276 005037 001124
006302 010137 042402
006306 104001

```
*****  
:*TEST 2          RHCS2-CONTROL AND STATUS 2  
*****  
:**          THIS PARTIALLY TESTS RHCS2 TO ENABLE DETERMINATION  
:**          OF THE NUMBER OF DRIVES PRESENT  
*****  
TST2:  SCOPE  
        MOV      #1,$TIMES          ;;DO 1 ITERATION  
        MOV      #STACK,SP          ;RESET STACK  
        MOV      #TTNO,@#TSTNM      ;THIS SAVES TEST NUMBER  
  
        ;*CHECK TO SEE IF THE PROGRAM IS RUNNING WITH AN RH70  
  
        TST      @#RH70              ;TEST FLAG FOR RH70 CONTROLLER  
        BEQ      30$                 ;IF FLAG = 1, THIS TEST IS SKIPPED  
        JMP      TST3                 ;JUMP TO NEXT TEST -----)  
30$:    ;IF FLAG = 0, DO THIS TEST  
  
        MOV      @#PRCS2+12,@#UN     ;TEST BITS IN REGISTER  
        MOV      @#RHCS2,@#UN+2     ;ONLY THESE BITS TESTED FOR READ/WRITE  
UN:     .WORD    0                   ;ADDRESS OF REG. BEING TESTED  
        .WORD    0                   ;IN CORRECT DATA RECEIVED  
        ERROR   1                   ;RETURN TO BLT3 ROUTINE  
        RTS     PC  
  
*****  
:*TEST 3          PARTIAL TEST OF RHAS FOR UNIT NUMBERS PRESENT  
*****  
TST3:  SCOPE  
        MOV      #1,$TIMES          ;;DO 1 ITERATION  
  
        MOV      #TTNO,@#TSTNM      ;THIS SAVES TEST NUMBER  
  
        MOV      @#RHAS,R1          ;R1 HAS ADDRESS OF RHAS  
        MOV      #-1,@R1            ;THIS CLEARS RHAS (SURPRISED!)  
        MOV      @R1,@#SBDDAT       ;TEST DATA  
        TSTB    @#SBDDAT  
        BEQ     TST4                 ;BRANCH IF GOOD  
        CLR     @#SGDDAT             ;GOOD DATA  
        MOV     R1,@#REGADR          ;FAILING REG. RHAS  
        ERROR   1                   ;RHAS DOES NOT CLEAR  
        ;BY WRITING ONES IN
```

```
1891
1892
1893
1894
1895
1896 006310 000004
1897 006312 012737 000001 001212
1898
1899 006320 000005
1900 006322 012737 000004 002032
1901 006330 004737 055310
1902 006334 032777 020000 172576
1903 006342 001147
1904
1905 006344 104401 006352
1906 006350 000430
1907
1908 006432
1909 006432 104401 006440
1910 006436 000427
1911
1912 006516
1913 006516 104401 006524
1914 006522 000425
1915
1916 006576
1917 006576 104401 006604
1918 006602 000427
1919
1920 006662
1921
1922 006662 013701 001656
1923 006666 013702 001636
1924 006672 005012
1925 006674 012700 000010
1926 006700 013704 001642
1927 006704 012714 177777
1928 006710 005212
1929 006712 005300
1930 006714 001373
1931
1932 006716 111137 002020
1933
1934 006722 105037 002021
1935 006726 105711
1936 006730 001402
1937 006732 000137 007304
1938
1939 006736 032777 020000 172174 2$:
1940 006744 001402
1941 006746 000137 007642
1942
1943
1944 006752 3$:
1945 006752 104401 006760
1946 006756 000421
```

```

*****
:TEST 4 TEST FOR DRIVES PRESENT USING RHAS AND RHCS2
*****
TST4: SCOPE
MOV #1,$TIMES ;;DO 1 ITERATION
RESET ;;START WITH AN INIT
MOV #TTNO,@#TSTNM ;;THIS SAVES TEST NUMBER
JSR PC,@#STKINT ;;INITILIZE TTY KEYBOARD
BIT #SW13,@SWR ;;INHIBIT ERROR TYPEOUT?
BNE 4$ ;;BRANCH IF YES
TYPE ,65$ ;;TYPE ASCIZ STRING
BR 64$ ;;GET OVER THE ASCIZ
::65$: .ASCIZ <15><12><15><12>/LOOKING AT RHAS - DRIVES ASSUMED PRESENT/<15><12>
64$:
TYPE ,67$ ;;TYPE ASCIZ STRING
BR 66$ ;;GET OVER THE ASCIZ
::67$: .ASCIZ <15><12>/*****/
66$:
TYPE ,69$ ;;TYPE ASCIZ STRING
BR 68$ ;;GET OVER THE ASCIZ
::69$: .ASCIZ <15><12>/ THIS MUST BE VERIFIED BY OPERATOR !!/
68$:
TYPE ,71$ ;;TYPE ASCIZ STRING
BR 70$ ;;GET OVER THE ASCIZ
::71$: .ASCIZ <15><12>/*****/
70$:
4$: MOV @#RHAS,R1 ;LOAD R1 WITH ADDR. OF RHAS
MOV @#RHCS2,R2 ;LOAD R2 WITH ADDR. OF RHCS2
CLR @R2 ;CLEAR RHCS2 (ADDRESS UNIT #0)
MOV #8.,R0 ;INITIALIZE DRIVE COUNTER
MOV @#RHER1,R4 ;LOAD R4 WITH ADDR. OF RHER1
1$: MOV #-1,@R4 ;MOVE ERRORS INTO RHER1 OF UNIT ADDRESSED
INC @R2 ;INCREMENT UNIT NO. (RHCS2)
DEC R0 ;COUNT DOWN DRIVE COUNTER
BNE 1$ ;TEST AND DO NEXT UNIT IF 8 NOT DONE
MOV @R1,@#TOTALAT ;SAVE ALL RESULTING ATTENTION BITS
; (USED IN DRIVE CLEAR TEST)
CLRB @#TOTALAT+1 ;CLEAR UPPER BYTE
TSTB @R1 ;TEST RHAS FOR ANY DRIVES PRESENT
BEQ 2$ ;NONE RESPONDING - TYPE THE MESSAGE
JMP XE2 ;SOME THERE - GO FILL "UNITS" TABLE
2$: BIT #SW13,@SWR ;INHIBIT ERROR TYPE OUT?
BEQ 3$ ;TYPE "NO DRIVES" MESSAGE IF NO
JMP SELTST ;CHECK FOR SELECTED UNIT START AND LOAD
;"UNITS" TABLE WITH DESIRED DRIVE IF SO
3$:
TYPE ,73$ ;;TYPE ASCIZ STRING
BR 72$ ;;GET OVER THE ASCIZ
```

```
1947      ::73$: .ASCIZ <15><12>/NO DRIVES PRESENT - RHAS = 0/<15><12>
1948 007022 72$:
1949 007022 104401 007030      TYPE      ,75$      ::TYPE ASCIZ STRING
1950 007026 000430      BR      74$      ::GET OVER THE ASCIZ
1951      ::75$: .ASCIZ <15><12>/WRITING ONES INTO RHER1 FOR ALL UNIT NUMBERS/
1952 007110 74$:
1953 007110 104401 007116      TYPE      ,77$      ::TYPE ASCIZ STRING
1954 007114 000430      BR      76$      ::GET OVER THE ASCIZ
1955      ::77$: .ASCIZ <15><12>/DOES NOT SET ANY BIT IN RHAS SO ABORT PROGRAM/
1956 007176 76$:
1957 007176 104401 007204      TYPE      ,79$      ::TYPE ASCIZ STRING
1958 007202 000436      BR      78$      ::GET OVER THE ASCIZ
1959      ::79$: .ASCIZ <15><12>/TO LOOP ON THIS TEST WO PRINTOUT, SET SWITCHES 13, 8 & 2/
1960 007300 78$:
1961
1962 007300 000137 041432      JMP      @#$EOP      ;GO OUT----->
1963
1964
1965      ;*SET UP DRIVES PRESENT TABLE
1966 007304      XE2:
1967
1968 007304 012700 000010      2$:      MOV      #8.,R0      ;LOAD 'UNITS' TABLE COUNTER
1969 007310 012703 001754      MOV      #UNITS,R3      ;LOAD 'UNITS' TABLE POINTER
1970 007314 012723 177777      3$:      MOV      #-1,(R3)+      ;PRESET 1ST TABLE BLOCK TO ALL ONES
1971 007320 005300      DEC      R0      ;COUNT DOWN
1972 007322 001374      BNE      3$      ;PRESET NEXT BLOCK IF 8 NOT DONE
1973
1974 007324 012703 001754      10$:     MOV      #UNITS,R3      ;RELOAD THE TABLE POINTER
1975 007330 005005      CLR      R5      ;INITIALIZE UNIT NO. TO 0
1976 007332 005037 001776      CLR      @#NOUNIT      ;NO. OF UNITS PRESENT
1977 007336 012700 000010      MOV      #8.,R0      ;RELOAD THE TABLE COUNTER
1978 007342 011137 001176      MOV      @R1,@#$TMPO      ;ADDR OF RHAS INTO TEMPORARY STORAGE
1979 007346 006037 001176      4$:      ROR      @#$TMPO      ;SET CARRY IF 0 BIT = 1 (UNIT ATTEN.)
1980 007352 103120      BCC      5$      ;CHECK NEXT UNIT IF ONE NOT IN BIT 0
1981
1982 007354 010577 172256      11$:     MOV      R5,@RHCS2      ;INSERT UNIT NO. INTO RHCS2 UNIT ADDR.
1983 007360 022777 024020 172276      CMP      #24020,@RHDT      ;READ RHDT - IS IT A DUAL PORT RP04 ?
1984 007366 001503      BEQ      6$      ;YES...TYPE THE UNIT NO.
1985 007370 022777 020020 172266      CMP      #20020,@RHDT      ;READ RHDT - IS IT A SINGLE PORT RP04 ?
1986 007376 001477      BEQ      6$      ;YES...TYPE THE UNIT NO.
1987
1988
1989
1990      ;:*****
1991 007400 022777 024021 172256      CMP      #24021,@RHDT      ;DUAL PORT RP05 ?
1992 007406 001473      BEQ      6$      ;TYPE UNIT NO. IF SO
1993 007410 022777 020021 172246      CMP      #20021,@RHDT      ;SINGLE PORT RP05 ?
1994 007416 001467      BEQ      6$      ;TYPE NO. IF SO
1995
1996 007420 022777 024022 172236      CMP      #24022,@RHDT      ;READ RHDT - IS IT A DUAL PORT RP06 ?
1997 007426 001463      BEQ      6$      ;YES...TYPE THE UNIT NO.
1998 007430 022777 020022 172226      CMP      #20022,@RHDT      ;READ RHDT - IS IT A SINGLE PORT RP06 ?
1999 007436 001457      BEQ      6$      ;YES...TYPE THE UNIT NO.
2000      ;:*****
2001
2002
```

```
2003
2004
2005      ;*NO...IT'S NOT AN RP04/RP05/RP06 DEVICE SO TYPE
2006      ;*OUT THE DEVICE TYPE
2007 007440 104401 007446      TYPE      ,65$      ;;TYPE ASCIZ STRING
2008 007444 000410      BR      64$      ;;GET OVER THE ASCIZ
2009      ;;65$: .ASCIZ <15><12>/UNIT NUMBER /
2010 007466      64$:
2011
2012 007466 010546      MOV      R5,-(SP)      ;PUT THE UNIT NUMBER ON STACK
2013 007470 104405      TYPDS      ;TYPE IT
2014 007472 104401 007500      TYPE      ,67$      ;;TYPE ASCIZ STRING
2015 007476 000406      BR      66$      ;;GET OVER THE ASCIZ
2016      ;;67$: .ASCIZ /, RHDT = /
2017 007514      66$:
2018 007514 017746 172144      MOV      @RHDT,-(SP)      ;PUT RHDT ON THE STACK
2019 007520 104402      TYPOC      ;TYPE IT
2020 007522 104401 007530      TYPE      ,69$      ;;TYPE ASCIZ STRING
2021 007526 000422      BR      68$      ;;GET OVER THE ASCIZ
2022      ;;69$: .ASCIZ ? - NOT AN RP04/RP05/RP06 DEVICE !!?
2023 007574      68$:
2024 007574 000407      BR      5$      ;UNIT NOT AN RP04/RP05/RP06 SO TEST NEXT ONE
2025
2026 007576 010523      6$: MOV      R5,(R3)+      ;LOAD TABLE POSITION AND INCR IT
2027 007600 104401 001223      TYPE      ,SCRLF      ;CRLF
2028 007604 010546      MOV      R5,-(SP)      ;PUT UNIT NO. ON THE STACK
2029 007606 104405      TYPDS      ;TYPE THE UNIT NO.
2030 007610 005237 001776      INC      @#NOUNIT      ;INCR THE TOTAL NO. OF UNITS
2031
2032 007614 005205      5$: INC      R5      ;'RHCS2' UNIT ADDRESS
2033 007616 005300      DEC      R0      ;DRIVE COUNTER DOWN ONE
2034 007620 001252      BNE      4$      ;TEST AND DO NEXT UNIT IF 8 NOT DONE
2035
2036 007622 013737 001754 001774 12$: MOV      @#UNITS,@#UNIT      ;SET UNIT NO. TO FIRST ONE FOUND/OR 0
2037 007630 013737 001776 002000      MOV      @#NOUNIT,@#NUNIT      ;SAVE NO. OF UNITS
2038 007636 005337 002000      DEC      @#NUNIT      ;IF NUNIT = 0 THEN ONLY ONE UNIT
2039      ;IF NUNIT > 0 THEN MORE THAN ONE UNIT
2040
2041 007642 005737 002002      SELTST: TST      @#SELECT      ;STARTING ADDRESS 200 ?
2042 007646 001403      BEQ      TST5      ;BRANCH IF STARTING FROM 200
2043 007650 013737 002004 001774      MOV      @#UNITSL,@#UNIT      ;CHANGE UNIT NUMBER TO SELECTED ONE
2044
```

2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100

007656 000004
007660 012737 000001 001212
007666 012737 010404 001106
007674 012737 000005 002032
007702 004737 042732
007706 005037 002016
007712 005737 001774
007716 001022
007720 012700 000041
007724 122710 000011
007730 001015
007732 005737 002002
007736 001012
007740 012700 001754
007744 005720
007746 022710 177777
007752 001404
007754 011037 001774
007760 005337 001776
007764 013700 001774
010077 171642
007774 005037 002036
010000 022777 024022 171656
010006 001405
010010 022777 020022 171646
010016 001401
010020 000403

```
*****  
:*TEST 5 TYPE SERIAL NUMBER AND DRIVE TYPE  
  
:** SET APPROPRIATE ATTENTION BIT OF UNIT UNDER TEST IN  
:** 'ATTENT' AND TYPE THE UNIT UNDER TEST  
  
:** READ SERIAL NUMBER REGISTER AND DRIVE TYPE REGISTER,  
:** TYPE THEM OUT AND PROCEED  
  
:** TO LOOP HERE SET SWITCH 8 AND THIS TEST NO. AND RESTART  
  
*****  
TST5: SCOPE  
MOV #1,$TIMES ;:DO 1 ITERATION  
MOV #1,$SLPADR ;:SET SCOPE LOOP ADDRESS  
  
MOV #TTNO,@#TSTNM ;:THIS SAVES TEST NUMBER  
JSR PC,@#CLDISK ;:FILL UNIT NO.  
CLR @#ATTENT ;:CLEAR  
  
:*TEST FOR UNIT #0  
  
TST @#UNIT ;:IS UNIT #0 NEXT IN THE UNITS TABLE ?  
BNE 10$ ;:IF NOT, TEST THIS UNIT  
MOV #41,R0 ;:IF SO, CHECK THE LOAD MEDIA LOCATION  
CMPB #11,(R0) ;:WAS IT AN RP04/5/6 ?  
BNE 10$ ;:NO...GO AHEAD WITH TESTING UNIT #0  
TST @#SELECT ;:WAS UNIT #0 SELECTED ?  
 ;:(IE. WAS IT A 210 START ?)  
BNE 10$ ;:IF SO...TEST IT  
  
:*INCREMENT THE UNITS TABLE TO NEXT DRIVE (IF ANY)  
:*& DECREMENT THE "NOUNITS" PRESENT (TO BE TESTED)  
  
MOV #UNITS,R0 ;:LOAD THE UNITS TABLE POINTER  
TST (R0)+ ;:SELECT THE NEXT UNIT IN THE TABLE  
 ;:(DOUBLE INCREMENT THE POINTER, R0)  
CMP #-1,(R0) ;:IS THERE ANOTHER TABLE ENTRY PRESENT ?  
BEQ 10$ ;:IF NOT (LOC = -1)...MUST USE UNIT #0  
MOV (R0),@#UNIT ;:SET UP TO BE THE UNIT UNDER TEST  
DEC @#NOUNITS ;:DECREMENT BECAUSE UNIT # 0 WONT'T BE TESTED  
10$: MOV @#UNIT,R0 ;:R0 CONTAINS UNIT NO.  
  
:*SET UP THE PROPER DEVICE TYPE FLAG  
  
*****  
MOV R0,@RHCS2 ;:SET UP UNIT ADDRESS  
CLR @#RP06 ;:CLEAR RP06 DEVICE TYPE FLAG  
CMP #24022,@RHDT ;:DUAL PORT RP06 ?  
BEQ 2$ ;:YES...SET THE FLAG  
CMP #20022,@RHDT ;:SINGLE PORT RP06 ?  
BEQ 2$ ;:YES..SET FLAG  
BR 3$ ;:NO...DON'T SET RP06 FLAG
```

```

2101 010022 012737 177777 002036 2$: MOV #-1,@#RP06 ;SET IT
2102
2103 010030 3$: ;ASSUME THE NEXT UNIT IS AN RP04
2104 ;:*****
2105
2106 010030 116037 004224 002016 MOVB ATABLE(R0),@#ATTENT ;SET APPROPRIATE ATTENTION BIT
2107
2108 010036 104401 010044 TYPE .65$ ;:TYPE ASCIZ STRING
2109 010042 000414 BR 64$ ;:GET OVER THE ASCIZ
2110 ;:65$: .ASCIZ <15><12>/TESTING DRIVE NUMBER/
2111 010074 64$:
2112 010074 013746 001774 MOV @#UNIT,-(SP) ;UNIT NO. TO STACK
2113 010100 104405 TYPDS ;:TYPE DRIVE NO.
2114 010102 104401 010110 TYPE .67$ ;:TYPE ASCIZ STRING
2115 010106 000410 BR 66$ ;:GET OVER THE ASCIZ
2116 ;:67$: .ASCIZ <15><12>/SERIAL NO. = /
2117 010130 66$:
2118 010130 017746 171532 MOV @#RHSN,-(SP) ;:SAVE @#RHSN FOR TYPEOUT
2119 010134 104402 TYPOC ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
2120 010136 104401 010144 TYPE .69$ ;:TYPE ASCIZ STRING
2121 010142 000410 BR 68$ ;:GET OVER THE ASCIZ
2122 ;:69$: .ASCIZ <15><12>/DRIVE TYPE = /
2123 010164 68$:
2124 010164 017746 171474 MOV @#RHDT,-(SP) ;:SAVE @#RHDT FOR TYPEOUT
2125 010170 104402 TYPOC ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
2126
2127 ;:*****
2128 010172 022777 024020 171464 CMP #24020,@#RHDT ;:DUAL PORT RP04 ?
2129 010200 001425 BEQ 4$ ;:TYPE ASCII MSG OUT
2130 010202 022777 020020 171454 CMP #20020,@#RHDT ;:SINGLE PORT RP04 ?
2131 010210 001421 BEQ 4$ ;:TYPE THE MESSAGE
2132
2133 010212 022777 024021 171444 CMP #24021,@#RHDT ;:DUAL PORT RP05 ?
2134 010220 001453 BEQ 6$ ;:TYPE MSG
2135 010222 022777 020021 171434 CMP #20021,@#RHDT ;:SINGLE PORT RP05 ?
2136 010230 001447 BEQ 6$ ;:TYPE MSG
2137
2138 010232 022777 024022 171424 CMP #24022,@#RHDT ;:DUAL PORT RP06 ?
2139 010240 001424 BEQ 5$ ;:TYPE MSG
2140 010242 022777 020022 171414 CMP #20022,@#RHDT ;:SINGLE PORT RP06 ?
2141 010250 001420 BEQ 5$ ;:TYPE MSG
2142 010252 000454 BR 1$ ;:DRIVE IS NOT AN RP04/RP05/RP06 - SO
2143 ;:DON'T TYPE THE ASCII MESSAGE
2144
2145 ;:-SHOULD NEVER HAPPEN AT THIS POINT
2146 ;:UNLESS DRIVE GOT SICK WHILE TESTING
2147 ;:WAS IN PROGRESS
2148 010254 4$:
2149 010254 104401 010262 TYPE .71$ ;:TYPE ASCIZ STRING
2150 010260 000413 BR 70$ ;:GET OVER THE ASCIZ
2151 ;:71$: .ASCIZ <15><12>/DRIVE IS AN RP04/<15><12>
2152 010310 70$:
2153 010310 000435 BR 1$ ;:SKIP NEXT MESSAGE
2154 010312 5$:
2155 010312 104401 010320 TYPE .73$ ;:TYPE ASCIZ STRING
2156 010316 000413 BR 72$ ;:GET OVER THE ASCIZ
    
```

2157
2158 010346
2159 010346 000416
2160 010350
2161 010350 104401 010356
2162 010354 000413
2163
2164 010404
2165
2166
2167 010404 005777 171256
2168 010410 005777 171250
2169 010414 017737 171246 002012
2170 010422 017737 171236 002010
2171
2172

```
:::73$: .ASCIZ <15><12>/DRIVE IS AN RP06/<15><12>  
72$:  
BR 1$ ;SKIP NEXT  
6$:  
TYPE ,75$ ;:TYPE ASCIZ STRING  
BR 74$ ;:GET OVER THE ASCIZ  
:::75$: .ASCIZ <15><12>/DRIVE IS AN RP05/<15><12>  
74$:  
:*****  
1$: TST @RHSN ;READ SERIAL NO. AND DRIVE TYPE  
TST @RHDT ;THESE TWO ARE TO HELP SCOPE LOOPS  
MOV @RHSN,@#SAVSN ;SAVE TO CHECK IF CLR RHCS2 BIT 5 CLEARS ANY BITS  
MOV @RHDT,@#SAVDT ;SAVE TO CHECK IF CLR RHCS2 BIT 5 CLEARS ANY BITS
```

2173
2174
2175
2176
2177
2178
2179
2180
2181
2182 010430 000004
2183 010432 012737 000006 002032
2184 010440 004737 042732
2185 010444 032713 010000
2186 010450 001551
2187 010452 104401 010460
2188 010456 000421
2189
2190 010522
2191 010522 104401 010530
2192 010526 000424
2193
2194 010600
2195 010600 104401 010606
2196 010604 000430
2197
2198 010666
2199 010666 032713 010000
2200 010672 001375
2201 010674 104401 010702
2202 010700 000435
2203
2204 010774

```
*****  
: *TEST 6 CHECK MOL TO BE LOW  
*****  
: ** MAKE SURE THAT DRIVE IS OFF LINE BEFORE STARTING PROGRAM  
: ** IF DRIVE IS ON LINE THEN AFTER TYPE OUT THE PROGRAM WILL  
: ** HANG FOR EVER WAITING FOR DRIVE TO GO OFF LINE  
*****  
TST6: SCOPE  
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER  
JSR PC,@#CLDISK ;GIVE INITILIZE  
BIT #MOL,@R3 ;CHECK MOL IN RHDS1  
BEQ TST7 ;BRANCH IF MOL LOW  
TYPE ,65$ ;:TYPE ASCIZ STRING  
BR 64$ ;:GET OVER THE ASCIZ  
::65$: .ASCIZ <15><12>/DRIVE IS ON LINE - MOL IS HIGH/  
64$:  
TYPE ,67$ ;:TYPE ASCIZ STRING  
BR 66$ ;:GET OVER THE ASCIZ  
::67$: .ASCIZ <15><12>/HIT STOP ON DRIVE TO GET IT OFF LINE/  
66$:  
TYPE ,69$ ;:TYPE ASCIZ STRING  
BR 68$ ;:GET OVER THE ASCIZ  
::69$: .ASCIZ <15><12>/PROGRAM WILL HANG TESTING MOL TILL MOL IS LOW/  
68$:  
1$: BIT #MOL,@R3 ;CHECK MOL IN RHDS1  
BNE 1$ ;BRANCH IF MOL IS HIGH  
TYPE ,71$ ;:TYPE ASCIZ STRING  
BR 70$ ;:GET OVER THE ASCIZ  
::71$: .ASCIZ <15><12>/GOOD - MOL IS NOW LOW . PROGRAM WILL NOW BE EXECUTED/<15><12>  
70$:
```


.SBTTL REGISTER TESTS

: *TEST 7 RHCS2 - CONTROL AND STATUS 2
: ** TEST LOADING AND READING OF ALL POSSIBLE BITS IN THE HARDWARE
: ** REGISTERS. USE A PATTERN OF WALKING 0'S (-2,-3,-5 ETC.) AND
: ** WALKING 1'S (1,2,4,10 ETC)

```
TST7: SCOPE
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER

; *CHECK TO SEE IF THE PROGRAM IS RUNNING WITH AN RH70

TST @#RH70 ;TEST FLAG FOR RH70 CONTROLLER
BEQ 30$ ;IF FLAG = 1, THIS TEST IS SKIPPED
JMP TST10 ;JUMP TO NEXT TEST -----)
30$: ;IF FLAG = 0, DO THIS TEST

JSR PC,@#CLDISK ;GIVE INITIALIZE

MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER

PRCS2: MOV @#RHCS2,@#PRCS2+14 ;GET REGISTER ADDRESS
MOV @#UNIT,@#RHCS2 ;MOVE UNIT NO. UNDER TEST
JSR R5,BITST ;TEST BITS IN REGISTER
.WORD 20017 ;ONLY THESE BITS ARE TESTED FOR READ/WRITE
.WORD 176710 ;ADDRESS OF REG. BEING TESTED
ERROR 1 ;INCORRECT DATA RECEIVED
RTS PC ;RETURN TO BLT3 ROUTINE
```

: *TEST 10 RHCS1 - CONTROL AND STATUS 1 REGISTER
: ** TEST LOADING AND READING OF ALL POSSIBLE BITS IN THE HARDWARE
: ** REGISTERS. USE A PATTERN OF WALKING 0'S (-2,-3,-5 ETC.) AND
: ** WALKING 1'S (1,2,4,10 ETC)

```
TST10: SCOPE

MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER

PRCS1: MOV @#RHCS1,@#PRCS1+14 ;GET REGISTER ADDRESS
MOV @#UNIT,@#RHCS2 ;MOVE UNIT NO. UNDER TEST
JSR R5,BITST ;TEST BITS IN REGISTER
.WORD 1476 ;ONLY THESE BITS ARE TESTED FOR READ/WRITE
.WORD 176700 ;ADDRESS OF REG. BEING TESTED
ERROR 1 ;INCORRECT DATA RECEIVED
```

2205
2206
2207
2208
2209
2210
2211
2212
2213
2214 010774 000004
2215 010776 012737 000007 002032
2216
2217
2218
2219 011004 005737 002040
2220 011010 001402
2221 011012 000137 011064
2222 011016
2223
2224 011016 004737 042732
2225
2226
2227 011022 012706 001000
2228 011026 012737 000007 002032
2229
2230 011034 013737 001636 011056
2231 011042 013777 001774 170566
2232 011050 004537 042404
2233 011054 020017
2234 011056 176710
2235 011060 104001
2236 011062 000207
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250 011064 000004
2251
2252 011066 012706 001000
2253 011072 012737 000010 002032
2254
2255 011100 013737 001640 011122
2256 011106 013777 001774 170522
2257 011114 004537 042404
2258 011120 001476
2259 011122 176700
2260 011124 104001

CZRJGDO,RP04/5/6 DSKLS CTRLR1
CZRJGD.P11 10-SEP-79 11:00

MACY11 30A(1052)
T10

10-SEP-79 11:11 PAGE 49
RHCS1 - CONTROL AND STATUS 1 REGISTER

J 4

SEQ 0048

2261 011126 000207
2262
2263

RTS PC ;RETURN TO BLT3 ROUTINE

```
2264  
2265  
2266  
2267  
2268  
2269  
2270  
2271  
2272 011130 000004  
2273  
2274 011132 012706 001000  
2275 011136 012737 000011 002032  
2276 011144 004737 042732  
2277  
2278  
2279  
2280 011150 052777 000020 170460  
2281  
2282 011156 005077 170460  
2283  
2284  
2285  
2286  
2287 011162 011137 001126  
2288 011166 022737 104200 001126  
2289  
2290 011174 001406  
2291 011176 012737 104200 001124  
2292 011204 010137 042402  
2293 011210 104001  
2294  
2295  
2296  
2297 011212 013746 001774 1$:  
2298 011216 052716 000120  
2299 011222 012637 001124  
2300 011226 011237 001126  
2301 011232 023737 001124 001126  
2302 011240 001403  
2303 011242 010237 042402  
2304 011246 104001  
2305  
2306  
2307 011250 011437 001126 2$:  
2308 011254 022737 000010 001126  
2309  
2310 011262 001406  
2311 011264 012737 000010 001124  
2312 011272 010437 042402  
2313 011276 104001  
2314  
2315  
2316 011300 3$:
```

```
*****  
: *TEST 11 RHCS1 - BIT # 13 - MCPE  
: ** THIS FORCES A MASS BUS CONTROL PARITY ERROR  
: ** BY SETTING 'PAT', LOOKING FOR 'PAR', WRITING RHCS1  
: ** AND READING RHER1  
: *****  
TST11: SCOPE  
  
MOV #STACK,SP ;RESET STACK  
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER  
JSR PC,@#CLDISK ;INIT AND SET UNIT NUMBER AND DEVICE -  
;CPU REG. CORRESPONDENCE (R1-R4)  
  
; *SET FORCED PARITY ERROR 'PAT'  
BIS #PAT,@RHCS2 ;SET 'PAT' TO INVERT PARITY  
;GENERATED  
CLR @RHER1 ;WRITE DCL REGISTER USING BAD CONTROL PARITY  
  
; *WITH THIS PARITY ERROR NOTHING WILL BE READ TILL IT IS  
; *CLEARED  
  
MOV @R1,@#SBDDAT ;RHCS1 ---> SBDDAT  
CMP #SC!DVA!RDY,@#SBDDAT ;COMPARE RHCS1 AFTER PARITY  
;ERROR  
BEQ 1$ ;BRANCH IF SC!DVA!RDY=1  
MOV #SC!DVA!RDY,@#SGDDAT ;GOOD DATA  
MOV R1,@#REGADR ;REGISTER ADDRESS RHCS1  
ERROR 1 ;SETTING PAT AND  
;WRITING DCL REGISTER RHCS1  
;DID NOT SET SC!DVA!RDY  
;WITH 'PAT' BIT HIGH  
1$: MOV @#UNIT,-(SP) ;GET UNIT NUMBER  
BIS #PAT!IR,(SP) ;INCLUDE PAT AND IR  
MOV (SP)+,@#SGDDAT ;PUT ON STACK  
MOV @R2,@#SBDDAT ;RHCS2 ---> SBDDAT  
CMP @#SGDDAT,@#SBDDAT ;COMPARE RHCS2  
BEQ 2$ ;OK - SC!DVA!RDY ARE HIGH  
MOV R2,@#REGADR ;REGISTER ADDRESS  
ERROR 1 ;READING DCL REGISTER RHCS2 DID NOT  
;SHOW UNIT#!PAT!IR BITS HIGH  
  
2$: MOV @R4,@#SBDDAT ;RHER1 ---> SBDDAT  
CMP #PAR,@#SBDDAT ;ERROR REGISTER RHER1 SHOULD  
;HAVE 'PAR' SET  
BEQ 3$ ;A - OK, IT DOES  
MOV #PAR,@#SGDDAT ;GOOD DATA  
MOV R4,@#REGADR ;FAILING REGISTER RHER1  
ERROR 1 ;PARITY ERROR DID NOT  
;SET 'PAR'  
  
3$:
```

2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364

```
*****  
: *TEST 12      RHWC - WORD COUNT REGISTER  
: **          TEST LOADING AND READING OF ALL POSSIBLE BITS IN THE HARDWARE  
: **          REGISTERS. USE A PATTERN OF WALKING 0'S (-2,-3,-5 ETC.) AND  
: **          WALKING 1'S (1,2,4,10 ETC)  
: *****
```

TST12: SCOPE

```
MOV #STACK,SP ;RESET STACK  
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER  
PRWC: MOV @#RHWC,@#PRWC+14 ;GET REGISTER ADDRESS  
MOV @#UNIT,@#RHCS2 ;MOVE UNIT NO. UNDER TEST  
JSR R5,BITST ;TEST BITS IN REGISTER  
.WORD 177777 ;ONLY THESE BITS ARE TESTED FOR READ/WRITE  
.WORD 176702 ;ADDRESS OF REG. BEING TESTED  
ERROR 1 ;INCORRECT DATA RECEIVED  
RTS PC ;RETURN TO BLT3 ROUTINE
```

```
*****  
: *TEST 13      RHBA - UNIBUS ADDRESS REGISTER  
: **          TEST LOADING AND READING OF ALL POSSIBLE BITS IN THE HARDWARE  
: **          REGISTERS. USE A PATTERN OF WALKING 0'S (-2,-3,-5 ETC.) AND  
: **          WALKING 1'S (1,2,4,10 ETC)  
: *****
```

TST13: SCOPE

```
MOV #STACK,SP ;RESET STACK  
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER  
PRBA: MOV @#RHBA,@#PRBA+14 ;GET REGISTER ADDRESS  
MOV @#UNIT,@#RHCS2 ;MOVE UNIT NO. UNDER TEST  
JSR R5,BITST ;TEST BITS IN REGISTER  
.WORD 177776 ;ONLY THESE BITS ARE TESTED FOR READ/WRITE  
.WORD 176704 ;ADDRESS OF REG. BEING TESTED  
ERROR 1 ;INCORRECT DATA RECEIVED  
RTS PC ;RETURN TO BLT3 ROUTINE
```

2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387

```
*****  
: *TEST 14      RHER1 - ERROR REGISTER #1  
: **          TEST LOADING AND READING OF ALL POSSIBLE BITS IN THE HARDWARE  
: **          REGISTERS. USE A PATTERN OF WALKING 0'S (-2,-3,-5 ETC.) AND  
: **          WALKING 1'S (1,2,4,10 ETC)  
: *****
```

```
TST14: SCOPE  
      MOV      #STACK,SP      ;RESET STACK  
      MOV      #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER  
      MOV      @#RHER1,@#PRER1+14 ;GET REGISTER ADDRESS  
PRER1: MOV      @#UNIT,@RHCS2 ;MOVE UNIT NO. UNDER TEST  
      JSR      R5,BITST      ;TEST BITS IN REGISTER  
      .WORD   177777        ;ONLY THESE BITS ARE TESTED FOR READ/WRITE  
      .WORD   176714        ;ADDRESS OF REG. BEING TESTED  
      ERROR   1             ;INCORRECT DATA RECEIVED  
      RTS     PC            ;RETURN TO BLT3 ROUTINE
```

```
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397 011454 000004
2398
2399 011456 012737 000015 002032
2400 011464 004737 042732
2401 011470 013700 001660
2402 011474 012701 000001
2403 011500 012702 000005
2404 011504 012710 000001
2405 011510 050110
2406 011512 010146
2407 011514 052716 000401
2408 011520 011637 001124
2409 011524 022610
2410 011526 001405
2411 011530 011037 001126
2412 011534 010037 042402
2413 011540 104001
2414
2415
2416 011542 000241
2417 011544 006101
2418 011546 052701 000400
2419 011552 042701 001000
2420 011556 005302
2421 011560 001351
2422
2423
2424
2425
2426 011562 012701 000435
2427 011566 012702 000005
2428 011572 012710 000001
2429 011576 050110
2430 011600 020110
2431 011602 001407
2432 011604 010137 001124
2433 011610 011037 001126
2434 011614 010037 042402
2435 011620 104001
2436
2437
2438 011622 000261
2439 011624 006101
2440 011626 042701 001340
2441 011632 052701 000400
2442 011636 005302
2443 011640 001354

;*****
;*TEST 15 RHMR - MAINTENANCE REGISTER
;** BIT 0 (DMD) MUST BE SET BEFORE THE OTHER BITS
;** ARE READ WRITE
;** ONLY 5 LOW ORDER BITS ARE TESTED (R2 HAS 5)
;*****
TST15: SCOPE
          MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
          JSR PC,@#CLDISK ;SET UNIT NUMBER AND INIT
          MOV @#RHMR,R0 ;R0 HAS MAINTENANCE REG. ADR.
          MOV #1,R1 ;R1 HAS DATA
          MOV #5,R2 ;R2 HAS COUNT OF NUMBER OF BITS
1$:      MOV #DMD,@R0 ;SET DIAGNOSTIC MODE BIT
          BIS R1,@R0 ;SET DATA IN RHMR
          MOV R1,-(SP) ;SAVE DATA FOR COMPARES
          BIS #DMD!400,(SP) ;INCLUDE BIT 0
          MOV (SP),@#$GDDAT ;SAVE FOR ERROR PRINTOUT
          CMP (SP)+,@R0 ;COMPARE DATA
          BEQ 2$ ;BRANCH IF GOOD
          MOV @R0,$BDDAT ;BAD DATA
          MOV R0,@#REGADR ;FAILING REG. ADR.
          ERROR 1 ;MAINTENANCE REGISTER
          ;FAILED TO SET INDICATED
          ;BITS
2$:      CLC ;CLEAR CARRY
          ROL R1 ;GET NEXT DATA
          BIS #400,R1 ;SET UNUSED BITS
          BIC #BIT09,R1 ;CLEAR READ ONLY BIT
          DEC R2 ;COUNT
          BNE 1$ ;BRANCH IF 5 BITS NOT DONE

;*NOW FLOAT A 0
          MOV #435,R1 ;R1 HAS DATA
          MOV #5,R2 ;R2 HAS COUNT BITS
3$:      MOV #DMD,@R0 ;SET DIAGNOSTIC MODE BITS
          BIS R1,@R0 ;SET DATA IN RHMR
          CMP R1,@R0 ;COMPARE DATA
          BEQ 4$ ;BRANCH IF GOOD
          MOV R1,@#$GDDAT ;GOOD DATA
          MOV @R0,@#$BDDAT ;BAD DATA
          MOV R0,@#REGADR ;FAILING REG. ADR. RHMR
          ERROR 1 ;MAINTENANCE REGISTER
          ;DOES NOT ALLOW WRITING
          ;ZEROS
4$:      SEC ;SET CARRY
          ROL R1 ;GET NEXT DATA
          BIC #BIT05!BIT06!BIT07!BIT09,R1 ;CLEAR READ ONLY BIT
          BIS #BIT08,R1 ;SET BIT ZEROED BY ROL
          DEC R2 ;COUNT IF 5 BITS DONE
          BNE 3$ ;BRANCH IF INCOMPLETE
```

2444
2445

2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497

```
*****  
: *TEST 16      RHDST - DESIRED SECTOR/TRACK ADDRESS  
: **          TEST LOADING AND READING OF ALL POSSIBLE BITS IN THE HARDWARE  
: **          REGISTERS. USE A PATTERN OF WALKING 0'S (-2,-3,-5 ETC.) AND  
: **          WALKING 1'S (1,2,4,10 ETC)  
: *****
```

TST16: SCOPE

011642 000004

```
MOV      #STACK,SP      ;RESET STACK  
MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER  
  
PRDST:  MOV      @#RHDST,@#PRDST+14      ;GET REGISTER ADDRESS  
        MOV      @#UNIT,@#RHCS2        ;MOVE UNIT NO. UNDER TEST  
        JSR      R5,BITST              ;TEST BITS IN REGISTER  
        .WORD    17437                  ;ONLY THESE BITS ARE TESTED FOR READ/WRITE  
        .WORD    176706                 ;ADDRESS OF REG. BEING TESTED  
        ERROR    1                      ;INCORRECT DATA RECEIVED  
        RTS      PC                    ;RETURN TO BLT3 ROUTINE
```

```
*****  
: *TEST 17      RHER2 - ERROR REGISTER #2  
: **          TEST LOADING AND READING OF ALL POSSIBLE BITS IN THE HARDWARE  
: **          REGISTERS. USE A PATTERN OF WALKING 0'S (-2,-3,-5 ETC.) AND  
: **          WALKING 1'S (1,2,4,10 ETC)  
: *****
```

TST17: SCOPE

011706 000004

```
MOV      #STACK,SP      ;RESET STACK  
MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER  
  
PRER2:  MOV      @#RHER2,@#PRER2+14      ;GET REGISTER ADDRESS  
        MOV      @#UNIT,@#RHCS2        ;MOVE UNIT NO. UNDER TEST  
        JSR      R5,BITST              ;TEST BITS IN REGISTER  
        .WORD    177777                 ;ONLY THESE BITS ARE TESTED FOR READ/WRITE  
        .WORD    176740                 ;ADDRESS OF REG. BEING TESTED  
        ERROR    1                      ;INCORRECT DATA RECEIVED  
        RTS      PC                    ;RETURN TO BLT3 ROUTINE
```


2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544

011752 000004

011754 012706 001000

011760 012737 000020 002032

011766 013737 001650 012010

011774 013777 001774 167634

012002 004537 042404

012006 016277

012010 176732

012012 104001

012014 000207

```
*****
*TEST 20      RHOF - MARGIN/OFFSET REGISTER
**          TEST LOADING AND READING OF ALL POSSIBLE BITS IN THE HARDWARE
**          REGISTERS. USE A PATTERN OF WALKING 0'S (-2,-3,-5 ETC.) AND
**          WALKING 1'S (1,2,4,10 ETC)
*****
```

TST20: SCOPE

```
MOV      #STACK,SP      ;RESET STACK
MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
MOV      @#RHOF,@#PROF+14 ;GET REGISTER ADDRESS
MOV      @#UNIT,@#RHCS2 ;MOVE UNIT NO. UNDER TEST
JSR      R5,BITST       ;TEST BITS IN REGISTER
.WORD    16277          ;ONLY THESE BITS ARE TESTED FOR READ/WRITE
.WORD    176732        ;ADDRESS OF REG. BEING TESTED
ERROR    1              ;INCORRECT DATA RECEIVED
RTS      PC             ;RETURN TO BLT3 ROUTINE
```

```
*****
*TEST 21      RHCA - DESIRED CYLINDER REGISTER
**          TEST LOADING AND READING OF ALL POSSIBLE BITS IN THE HARDWARE
**          REGISTERS. USE A PATTERN OF WALKING 0'S (-2,-3,-5 ETC.) AND
**          WALKING 1'S (1,2,4,10 ETC)
*****
```

TST21: SCOPE

```
MOV      #STACK,SP      ;RESET STACK
MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
MOV      @#RHCA,@#PRCA+14 ;GET REGISTER ADDRESS
MOV      @#UNIT,@#RHCS2 ;MOVE UNIT NO. UNDER TEST
JSR      R5,BITST       ;TEST BITS IN REGISTER
.WORD    1777          ;ONLY THESE BITS ARE TESTED FOR READ/WRITE
.WORD    176734        ;ADDRESS OF REG. BEING TESTED
ERROR    1              ;INCORRECT DATA RECEIVED
RTS      PC             ;RETURN TO BLT3 ROUTINE
```

2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578

```
*****  
:TEST 22          RHER3 - ERROR REGISTER #3  
:**              TEST LOADING AND READING OF ALL POSSIBLE BITS IN THE HARDWARE  
:**              REGISTERS. USE A PATTERN OF WALKING 0'S (-2,-3,-5 ETC.) AND  
:**              WALKING 1'S (1,2,4,10 ETC)  
*****  
TST22: SCOPE  
012062 000004  
012064 012706 001000      MOV    #STACK,SP      ;RESET STACK  
012070 012737 000022 002032  MOV    #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER  
012076 013737 001654 012120  PRER3: MOV    @#RHER3,@#PRER3+14 ;GET REGISTER ADDRESS  
012104 013777 001774 167524  MOV    @#UNIT,@RHCS2 ;MOVE UNIT NO. UNDER TEST  
012112 004537 042404      JSR    R5,BITST      ;TEST BITS IN REGISTER  
012116 177777      .WORD 177777      ;ONLY THESE BITS ARE TESTED FOR READ/WRITE  
012120 176742      .WORD 176742      ;ADDRESS OF REG. BEING TESTED  
012122 104001      ERROR 1          ;INCORRECT DATA RECEIVED  
012124 000207      RTS    PC          ;RETURN TO BLT3 ROUTINE
```

```
*****  
:**OF THE TWENTY REGISTERS (4 IN RH11, 16 IN RP04) ONLY 12 ARE  
:**CHECKED IN THE ABOVE TESTS  
:**TWO ARE ALREADY TESTED (SERIAL NO. AND DRIVE TYPE)  
:**THE OTHER 7 WHICH ARE RHDS1, RHLA, RHCC, RHEC1, RHEC1, RHEC2  
:**ARE READ ONLY REGISTERS. ONE OR ZERO CANNOT BE WRITTEN  
*****
```

```
2579
2580
2581
2582          ::*****
2583          :*TEST 23      CONTROL AND STATUS 2 (RHCS 2) - 'NED'
2584          :**      THIS TESTS THE UNIT SELECT BITS #0-2 (US1-4)
2585          :**      AND NON-EXISTENT DRIVE BIT #12 (NED)
2586
2587          :**      THE OTHER RHCS2 BITS ARE NOT TESTED HERE
2588          ::*****
2589 012126 000004          1ST23: SCOPE
2590 012130 012706 001000      MOV      #STACK,SP          ;RESET STACK
2591 012134 012737 000023 002032  MOV      #TTNO,@#TSTNM      ;THIS SAVES TEST NUMBER
2592
2593 012142 004737 042732      JSR      PC,@#CLDISK        ;HERE IT IS USED TO SETUP HARDWARE/
2594                                ;CPU REGISTER CORRESPONDENCE
2595                                ;R1=RHCS1
2596                                ;R2=RHCS2
2597                                ;R3=RHDS1
2598                                ;R4=RHER1
2599 012146 005037 002006      CLR      @#ERFLG$          ;CLEAR ERROR FLAG
2600
2601          :*SIMULATED DISK AREA WILL BE USED AS A TEMPORARY
2602          :*STORAGE TABLE FOR DRIVES PRESENT DETERMINED FROM 'NED' = 0 IN RHCS2
2603
2604 012152 012701 051534      MOV      #DISK,R1          ;LOAD TABLE POINTER
2605 012156 012700 000010      MOV      #8.,R0           ;LOAD TABLE LOCATION COUNTER
2606 012162 012721 177777      1$:    MOV      #-1,(R1)+    ;FILL 8 LOCATIONS WITH -1
2607 012166 005300              DEC      R0                ;COUNT DOWN ONE LOCATION
2608 012170 001374              BNE      1$                ;BRANCH IF 8 NOT DONE
2609
2610 012172 005012              CLR      @R2                ;SELECT UNIT NO.0 (U2!U1!U0=0)
2611 012174 012700 000010      MOV      #8.,R0           ;RELOAD TABLE LOCATION COUNTER
2612 012200 012701 051534      MOV      #DISK,R1          ;RELOAD THE TABLE POINTER
2613 012204 005714              2$:    TST      @R4            ;READ A DRIVE REGISTER (RHER1)
2614 012206 032712 010000      BIT      #NED,@R2         ;NON EXISTENT DRIVE BIT = 0 ?
2615 012212 001415              BEQ      3$                ;YES...DRIVE PRESENT, CHECK THE TYPE
2616 012214 005300              7$:    DEC      R0            ;NO...DECREMENT DRIVE COUNT
2617 012216 001454              BEQ      4$                ;CHECK RESULTS IF 8 DRIVES DONE
2618
2619 012220 011246              10$:   MOV      @R2,-(SP)         ;PUT RHCS2 ON THE STACK
2620 012222 042716 177770      BIC      #^C7,(SP)        ;MASK ALL BUT THE UNIT NUMBER
2621 012226 005216              INC      (SP)              ;INCREMENT THE UNIT NUMBER
2622 012230 013703 001640      MOV      @#RHCS1,R3       ;GET RHCS1 ADDRESS
2623 012234 005203              INC      R3                ;ADDRESS UPPER BYTE OF RHCS1
2624 012236 112713 000100      MOVB     #100,@R3         ;SET 'TRE' IN RHCS1
2625                                ;WITHOUT ADDRESSING DRIVE
2626 012242 012612              MOV      (SP)+,@R2        ;RHCS2 HAS THE INCREMENTED UNIT
2627                                ;WITH 'NED' CLEARED
2628 012244 000757              BR       2$                ;TEST FOR NEXT DRIVE
2629
2630
2631
2632          :*CHECK THE UNIT TYPE AND BUILD 'NED' DERIVED UNITS TABLE
2633
2634 012246 022777 024020 167410 3$:    CMP      #24020,@RHDT      ;IS THIS A DUAL PORT RP04 ?
```

```

2635 012254 001425          BEQ      8$          ;ENTER IN TABLE IF SO
2636 012256 022777 020020 167400  CMP      #20020,@RHDT ;IS THIS A SINGLE PORT RP04 ?
2637 012264 001421          BEQ      8$          ;ENTER IN TABLE IF SO
2638
2639
2640 012266 022777 024022 167370  CMP      #24022,@RHDT ;IS THIS A DUAL PORT RP06 ?
2641 012274 001415          BEQ      8$          ;ENTER IN TABLE IF SO
2642 012276 022777 020022 167360  CMP      #20022,@RHDT ;IS THIS A SINGLE PORT RP06 ?
2643 012304 001411          BEQ      8$          ;ENTER IN TABLE IF SO
2644
2645 012306 022777 024021 167350  CMP      #24021,@RHDT ;IS THIS A DUAL PORT RP05 ?
2646 012314 001405          BEQ      8$          ;ENTER IN TABLE IF SO
2647 012316 022777 020021 167340  CMP      #20021,@RHDT ;IS THIS A SINGLE PORT RP05 ?
2648 012324 001401          BEQ      8$          ;ENTER IN TABLE IF SO
2649
2650
2651 012326 000732          BR       7$          ;NO RP04 FOUND SO CHECK NEXT UNIT
2652
2653 012330 012746 000010 8$:     MOV      #8,-(SP)    ;LOAD MAX NO. OF DRIVES
2654 012334 160016          SUB      RO,(SP)     ;(SP) NOW HAS THE PRESENT DRIVE NO.
2655 012336 012621          MOV      (SP)+,(R1)+ ;LOAD TABLE, INCR TABLE LOCATION &
2656                                     ;RESTORE THE STACK TO WHERE IT WAS
2657 012340 005300          DEC      RO         ;DECREMENT THE DRIVE COUNT
2658 012342 001402          BEQ      4$          ;CHECK RESULTS IF 8 UNITS CHECKED
2659 012344 005212          INC      @R2        ;SELECT NEXT UNIT
2660 012346 000716          BR       2$          ;GO TEST IT
2661
2662
2663                                     ;*COMPARE 'NED' DERIVED UNITS TABLE WITH THAT DERIVED USING RHAS IN T4
2664
2665 012350 004037 043626 4$:     JSR      RO,@#COMPAR ;COMPARE RESULTS
2666 012354 001754          UNITS    ;RHER1/RHAS DERIVED DATA
2667 012356 051534          DISK    ;'NED' TEST DATA
2668 012360 000010          8.     ;NO.OF WORDS TO COMPARE
2669 012362 012370          5$     ;RETURN FOR ERROR HEADER
2670 012364 012416          6$     ;RETURN FOR ERROR DATA
2671 012366 012534          13$    ;RETURN FOR GOOD COMPARISON (NEXT TEST)
2672
2673
2674
2675                                     ;*SPECIAL 'NED'/'RHAS' TABLE TYPE OUT ROUTINE (BYPASSES .SERRTYP AND
2676                                     ;*HENCE IGNORES INHIBIT ERROR TIMEOUT SWITCH)
2677
2678 012370 104022          5$:     ERROR   22
2679 012372 012703 000010  MOV      #8,R3      ;LENGTH OF BOTH UNIT TABLES
2680 012376 012701 001754  MOV      #UNITS,R1   ;ADDRESS OF RHAS/RHER1 UNITS TABLE
2681 012402 012702 051534  MOV      #DISK,R2    ;ADDRESS OF 'NED' RHCS2 UNITS TABLE
2682 012406 012137 001124 14$:    MOV      (R1)+,@#$GDDAT ;LOAD RHAS UNIT NO.INTO '$GDDAT' AND
2683                                     ;INCREMENT THE TABLE LOCATION
2684 012412 012237 001126  MOV      (R2)+,@#$BDDAT ;LOAD 'NED' UNIT NO. INTO '$BDDAT'
2685                                     ;& INCR TABLE LOCATION
2686
2687 012416 032777 020000 166514 6$:     BIT      #SW13,@SWR  ;INHIBIT ERROR TYPE OUTS ?
2688 012424 001043          BNE     13$         ;YES...EXIT
2689 012426 022737 177777 001124  CMP      #-1,@#$GDDAT ;DOES RHAS UNIT TABLE LOCATION = -1 ?
2690 012434 001413          BEQ     11$         ;YES...DON'T TYPE IT - CHECK 'NED' TABLE
    
```


CZRJGDO,RP04/5/6 DSKLS CTRLR1 MACY11 30A(1052) 10-SEP-79 11:11 PAGE 61
CZRJGD.P11 10-SEP-79 11:00 123 CONTROL AND STATUS 2 (RHCS 2) - 'NED'

SEQ 0060

:/*/:*/:*/:*/:*/:*/:*/:*/:*/:*/:*/:*/:*/:*/:*/:*/*
:/*/:*/:*/:*/:*/:*/:*/:*/:*/:*/:*/:*/:*/:*/:*/:*/*

2750
2751
2752
2753
2754
2755
2756
2757
2758
2759
2760
2761
2762
2763
2764
2765
2766
2767
2768
2769
2770
2771
2772
2773
2774
2775
2776
2777
2778
2779
2780
2781
2782
2783
2784
2785
2786
2787
2788
2789
2790
2791
2792
2793
2794
2795
2796
2797
2798
2799
2800
2801
2802
2803
2804
2805

```
*****  
:*TEST 24 CONTROL AND STATUS 2 (RHCS2) - 'CLR'  
:*THIS TESTS THE UNIT SELECT BITS (US1-4) AND CLEAR BIT #5 (CLR)  
  
:*ALL REGISTERS ARE LOADED WITH ALL ONES EXCEPT BIT #0 AND #6  
:*WHICH ARE 'GO' AND 'INTERRUPT ENABLE', THEN 'CLR' IS GIVEN  
:*(RHDB IS READ FIRST AS THIS WILL SET DTL IN RHCS2 AND  
:*SC AND TRE IN RHCS1.)  
  
:*ANOTHER CLR IS GIVEN THEN ALL OTHER REGISTERS ARE READ  
:*****  
TST24: SCOPE
```

```
JSR PC,@#CLDISK ;SET REGISTERS AND CLEAR  
CLR @#ERFLG$ ;CLEAR ANY ERRORS  
  
:*FILL ALL POSSIBLE BITS WITH ONES  
  
MOV #177777,@RHDB ;BUS ADDRESS REGISTER GETS 177777  
MOV #177777,@RHWC ;WORD COUNT REGISTER GETS 177777  
MOV #177777,@RHBA ;BUS ADDRESS REGISTER GETS 177777  
BIS #157010,@RHCS2 ;CONTROL AND STATUS 2 GETS 157010  
MOV #1476,@RHCS1 ;CONTROL AND STATUS REGISTER GETS 1476  
MOV #177777,@RHER1 ;ERROR REGISTER1 GETS 177777  
MOV #17437,@RHDST ;DESIRED SECTOR TRACK  
MOV #177777,@RHER2 ;ERROR REGISTER 2  
MOV #16277,@RHOF ;OFFSET REGISTER  
MOV #177777,@RHCA ;DESIRED CYLINDER  
MOV #177777,@RHER3 ;ERROR REGISTER 3  
MOV #DMD,@RHMR ;MAINTENANCE REGISTER  
MOV #177777,@RHMR ;MAINTENANCE REGISTER  
  
BIS #CLR,@R2 ;CLEAR ALL POSSIBLE BITS  
MOV @#UNIT,@R2 ;REINSTATE UNIT NO.  
MOV #RHDB,R0 ;R0 CONTAINS ADDR. OF ADDR. OF REG.
```

;*DATA BUFFER REGISTER

```
MOV #177777,@#$GDDAT ;GOOD DATA FOR ERROR  
MOV @R0,@#REGADR ;REGISTER ADDRESS  
MOV @R0+,@#$BDDAT ;TEST DATA  
CMP @#$GDDAT,@#$BDDAT ;COMPARE GOOD WITH TEST DATA  
BEQ 2$ ;BRANCH IF GOOD  
ERROR 1 ;RHDB DID NOT HAVE ALL ONES  
;AFTER A CLR IN RHCS2
```

```
2$: BIS #CLR,@R2 ;SET CLEAR AGAIN BECAUSE  
;READING RHDB AFTER CLEARING WILL  
;SET DLT SC AND TRE  
MOV @#UNIT,@R2 ;REINSTATE UNIT NO.
```

;*WORD COUNT REGISTER

2806
 2807
 2808 012740 012737 177777 001124 3\$: MOV #177777,@#\$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
 2809 012746 013037 001126 MOV @ (R0)+,@#\$BDDAT ;TEST DATA
 2810 012752 022737 177777 001126 CMP #177777,@#\$BDDAT ;COMPARE DATA
 2811 012760 001402 BEQ 4\$;BRANCH IF GOOD
 2812 012762 004737 013546 JSR PC,@#ERCS2C ;JUMP TO ERROR FOR CLR (BIT 5)
 2813 ;IN RHCS2

;*BUS ADDRESS REGISTER

2814
 2815
 2816
 2817
 2818 012766 012737 000000 001124 4\$: MOV #0,@#\$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
 2819 012774 013037 001126 MOV @ (R0)+,@#\$BDDAT ;TEST DATA
 2820 013000 022737 000000 001126 CMP #0,@#\$BDDAT ;COMPARE DATA
 2821 013006 001402 BEQ 5\$;BRANCH IF GOOD
 2822 013010 004737 013546 JSR PC,@#ERCS2C ;JUMP TO ERROR FOR CLR (BIT 5)
 2823 ;IN RHCS2

;*CONTROL AND STATUS 2 REGISTER

2824
 2825
 2826
 2827
 2828 013014 012746 000100 5\$: MOV #100,-(SP) ;INCLUDE IR
 2829 013020 053716 001774 BIS @#UNIT,(SP) ;SET UNIT NO.
 2830 013024 012637 001124 MOV (SP)+,@#\$GDDAT ;GOOD DATA FOR TYPE OUT
 2831 013030 013037 001126 MOV @ (R0)+,@#\$BDDAT ;TEST DATA
 2832 013034 023737 001124 001126 CMP @#\$GDDAT,@#\$BDDAT ;COMPARE DATA
 2833 013042 001402 BEQ 6\$;BRANCH IF GOOD
 2834 013044 004737 013546 JSR PC,@#ERCS2C ;JUMP TO ERROR FOR CLR (BIT 5)
 2835 ;IN RHCS2

;*CONTROL AND STATUS 1 REGISTER

2836
 2837
 2838
 2839
 2840
 2841 013050 012737 004276 001124 6\$: MOV #4276,@#\$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
 2842 013056 013037 001126 MOV @ (R0)+,@#\$BDDAT ;TEST DATA
 2843 013062 022737 004276 001126 CMP #4276,@#\$BDDAT ;COMPARE DATA
 2844 013070 001402 BEQ 7\$;BRANCH IF GOOD
 2845 013072 004737 013546 JSR PC,@#ERCS2C ;JUMP TO ERROR FOR CLR (BIT 5)
 2846 ;IN RHCS2

;*ERROR 1 REGISTER

2847
 2848
 2849
 2850
 2851 013076 012737 000000 001124 7\$: MOV #0,@#\$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
 2852 013104 013037 001126 MOV @ (R0)+,@#\$BDDAT ;TEST DATA
 2853 013110 022737 000000 001126 CMP #0,@#\$BDDAT ;COMPARE DATA
 2854 013116 001402 BEQ 10\$;BRANCH IF GOOD
 2855 013120 004737 013546 JSR PC,@#ERCS2C ;JUMP TO ERROR FOR CLR (BIT 5)
 2856 ;IN RHCS2

;*DESIRED SECTOR/TRACK REGISTER

2857
 2858
 2859
 2860
 2861 013124 012737 017437 001124 10\$: MOV #17437,@#\$GDDAT ;GOOD DATA FOR ERROR TYPEOUT


```

2918
2919
2920 013330 012737 000400 001124 16$: MOV #400,@#$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
2921 013336 013037 001126 MOV @ (R0)+,@#$BDDAT ;TEST DATA
2922 013342 022737 000400 001126 CMP #400,@#$BDDAT ;COMPARE DATA
2923 013350 001402 BEQ 17$ ;BRANCH IF GOOD
2924 013352 004737 013546 JSR PC,@#ERCS2C ;JUMP TO ERROR FOR CLR (BIT 5)
2925 ;IN RHCS2
2926
2927 ;*DRIVE STATUS REGISTER
2928
2929 013356 012737 000600 001124 17$: MOV #600,@#$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
2930 013364 013046 MOV @ (R0)+,-(SP) ;GET RHDS1
2931 013366 011637 001126 MOV (SP),@#$BDDAT ;TEST DATA
2932 013372 042716 001100 BIC #VV!PROG,(SP) ;CLEAR VV AND PROG
2933 013376 022726 000600 CMP #600,(SP)+ ;COMPARE DATA
2934 013402 001402 BEQ 20$ ;BRANCH IF GOOD
2935 013404 004737 013546 JSR PC,@#ERCS2C ;JUMP TO ERROR FOR CLR (BIT 5)
2936 ;IN RHCS2
2937
2938 ;*DRIVE TYPE
2939
2940
2941 013410 013737 002010 001124 20$: MOV @#SAVDT,@#$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
2942 013416 013037 001126 MOV @ (R0)+,@#$BDDAT ;TEST DATA
2943 013422 023737 002010 001126 CMP @#SAVDT,@#$BDDAT ;COMPARE DATA
2944 013430 001402 BEQ 21$ ;BRANCH IF GOOD
2945 013432 004737 013546 JSR PC,@#ERCS2C ;JUMP TO ERROR FOR CLR (BIT 5)
2946 ;IN RHCS2
2947
2948
2949 ;*SERIAL NUMBER REGISTER
2950
2951
2952
2953 013436 013737 002012 001124 21$: MOV @#SAVSN,@#$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
2954 013444 013037 001126 MOV @ (R0)+,@#$BDDAT ;TEST DATA
2955 013450 023737 002012 001126 CMP @#SAVSN,@#$BDDAT ;COMPARE DATA
2956 013456 001402 BEQ 22$ ;BRANCH IF GOOD
2957 013460 004737 013546 JSR PC,@#ERCS2C ;JUMP TO ERROR FOR CLR (BIT 5)
2958 ;IN RHCS2
2959
2960
2961 ;*ECC1 POSITION
2962
2963
2964 013464 012737 000000 001124 22$: MOV #0,@#$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
2965 013472 013037 001126 MOV @ (R0)+,@#$BDDAT ;TEST DATA
2966 013476 022737 000000 001126 CMP #0,@#$BDDAT ;COMPARE DATA
2967 013504 001402 BEQ 23$ ;BRANCH IF GOOD
2968 013506 004737 013546 JSR PC,@#ERCS2C ;JUMP TO ERROR FOR CLR (BIT 5)
2969 ;IN RHCS2
2970
2971
2972 ;*ECC2 PATTERN
2973
  
```

```
2974
2975 013512 012737 000000 001124 23$: MOV #0,@#$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
2976 013520 013037 001126 MOV @ (R0)+,@#$BDDAT ;TEST DATA
2977 013524 022737 000000 001126 CMP #0,@#$BDDAT ;COMPARE DATA
2978 013532 001402 BEQ 24$ ;BRANCH IF GOOD
2979 013534 004737 013546 JSR PC,@#ERCS2C ;JUMP TO ERROR FOR CLR (BIT 5)
2980 ;IN RHCS2
2981
2982
2983 ;*LOOK-AHEAD REGISTER
2984
2985 013540 005720 24$: TST (R0)+ ;AS THE LOOK-AHEAD REG. CANNOT BE PREDICTED
2986 ;AFTER AN INIT IT IS NOT CHECKED
2987
2988 ;*CURRENT CYLINDER ADDRESS REGISTER
2989
2990 013542 005720 25$: TST (R0)+ ;AS THE CURRENT REG. CANNOT BE PREDICTED
2991 ;AFTER A INIT IT IS NOT CHECKED
2992
2993 013544 26$: BR TST25 ;BRANCH OVER JSR
2994 013544 000405
2995
2996
2997 013546 014037 042402 ERCS2C: MOV -(R0), @#REGADR ;FAILING REGISTER ADDRESS
2998 013552 104001 ERROR 1 ;CLR (BIT 5) IN RHCS2 DID
2999 ;NOT CLEAR APPROPRIATE BITS
3000 ;OR CLEARED EXTRA BITS
3001 013554 005720 TST (R0)+ ;UNDO -(R0) FOR BAD DATA
3002 013556 000207 RTS PC ;RETURN TO TEST ABOVE
```

```
3003
3004
3005
3006
3007
3008
3009
3010
3011
3012
3013
3014 013560 000004
3015 013562 012706 001000
3016 013566 012737 000025 002032
3017
3018 013574 004737 042732
3019
3020 013600 012777 000001 166052
3021 013606 013777 002102 166024
3022
3023
3024 013614 004037 043424
3025 013620 001632
3026 013622 003160
3027 013624 000023
3028
3029
3030 013626 052777 000001 166004
3031
3032
3033 013634 052737 000100 003210
3034
3035
3036
3037
3038
3039 013642 004037 043424
3040 013646 001632
3041 013650 002114
3042 013652 000023
3043
3044
3045
3046
3047 013654 113737 003205 002141
3048
3049
3050
3051
3052
3053 013662 004037 043626
3054 013666 003160
3055 013670 002114
3056 013672 000023
3057 013674 013702
3058 013676 013702

*****
:*TEST 25      PACK ACKNOWLEDGE COMMAND TEST
*****
:**          THE PACK ACKNOWLEDGE COMMAND WILL BE LOADED INTO RHCS1 WITH GO
:**          THEN ALL REGISTERS WILL BE CHECKED
:**          RH CLEAR WILL BE GIVEN
:**          THEN ALL REGISTERS WILL BE CHECKED
*****
TST25: SCOPE
MOV      #STACK,SP      ;RESET STACK
MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
JSR      PC,@#CLDISK    ;INIT AND SET UP GENERAL CPU/DEVICE
                                ;REGISTER CORRESPONDENCE AND UNIT NO.
MOV      #DMD,@RHMR     ;SET DIAGNOSTIC MODE
MOV      @#PKACK,@RHCS1 ;LOAD 'PACK ACKNOWLEDGE COMMAND' INTO RHCS1

;*SAVE REGISTERS FOR COMPARISON AFTER 'GO' IS ISSUED
JSR      RO,@#SAVER     ;SAVE
RHW      ;FROM
REINTO   ;TO
19.      ;NUMBER OF REGISTERS SAVED

BIS      #GO,@RHCS1     ;ISSUE 'GO' TO PACK ACKNOWLEDGE COMMAND

;*CHANGE SAVED REGISTERS TO EXPECTED VALUES
BIS      #VV,@#REINTO+30 ;SAVED RHDS1

;*AFTER GO HAS BEEN GIVEN TO PACK ACKNOWLEDGE COMMAND
;*SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN
;*BE DONE
JSR      RO,@#SAVER     ;SAVE
RHW      ;FROM
WRF      ;FROM
19.      ;NUMBER OF REGISTERS SAVED

;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
MOVB     @#REINTO+25,@#WRF+25;SAVE UPPER RHAS

;*COMPARE REGISTERS BEFORE PACK ACKNOWLEDGE COMMAND
;*WIT!! AFTER GO
JSR      RO,@#COMPAR    ;COMPARE
REINTO   ;GOOD BUFFER
WRF      ;TEST BUFFER
19.      ;NUMBER
1$       ;RETURN FOR ERROR
1$       ;SAME
```

```
3059 013700 013722          2$:          ;RETURN FOR GOOD COMPARISON
3060
3061 013702 013705 047736    1$:  MOV    @#ERWORD,R5    ;GETTING READY TO INDEX
3062 013706 060505          ADD    R5,R5          ;DOUBLE ERROR WORD
3063 013710 016537 001630 042402  MOV    RHWC-2(R5),@#REGADR ;FAILING REGISTER ADDRESS
3064
3065 013716 104001          ERROR  1              ;IMPROPER REGISTER CHANGE
3066                          ;AFTER PACK ACKNOWLEDGE COMMAND
3067                          ;WITH GO IS GIVEN
3068 013720 000207          RTS    PC              ;RETURN TO COMPARISON
3069
3070 013722          2$:          ;CONTINUE WITH THE NEXT TEST
```

3071
3072
3073
3074
3075
3076
3077
3078
3079
3080
3081
3082
3083
3084
3085
3086
3087
3088
3089
3090
3091
3092
3093
3094
3095
3096
3097
3098
3099
3100
3101
3102
3103
3104
3105
3106
3107
3108
3109
3110
3111
3112
3113
3114
3115
3116
3117
3118
3119
3120
3121
3122
3123
3124
3125
3126

013722 000004
013724 012706 001000
013730 012737 000026 002032
013736 004737 042732

013742 012777 177777 165662
013750 012777 177777 165656
013756 052777 157010 165652
013764 012777 001476 165646
013772 012777 177777 165642
014000 012777 017437 165636
014006 012777 177777 165632
014014 012777 016277 165626
014022 012777 000777 165622
014030 012777 177777 165616
014036 012777 000001 165614
014044 012777 177777 165606

014052 004037 043424
014056 001632
014060 003160
014062 000021

014064 000005
014066 004737 055310
014072 053777 001774 165536

014100 005037 003162
014104 013746 001774
014110 052716 000100
014114 012637 003164
014120 012737 004276 003166
014126 005037 003170
014132 005037 003174
014136 012737 116000 003176
014144 005037 003202
014150 105037 003204
014154 012737 000400 003206

: *TEST 26 UNIBUS INIT TEST

: * ALL POSSIBLE REGISTER BITS ARE FILLED WITH ONES
: * A RESET COMMAND IS GIVEN
: * ALL REGISTERS ARE CHECKED

TST26: SCOPE

MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
JSR PC,@#CLDISK ;INIT AND SET UP GENERAL CPU/DEVICE
;REGISTER CORRESPONDENCE

: *FILL ALL POSSIBLE REGISTER BITS WITH ONES

MOV #177777,@RHWC ;WORD COUNT REGISTER GETS 177777
MOV #177777,@RHBA ;BUS ADDRESS REGISTER GETS 177777
BIS #157010,@RHCS2 ;CONTROL AND STATUS 2 GETS 177430
MOV #1476,@RHCS1 ;CONTROL AND STATUS REGISTER 1 GETS 21476
MOV #177777,@RHER1 ;ERROR REGISTER1 GETS 177777
MOV #17437,@RHDST ;DESIRED SECTOR TRACK
MOV #177777,@RHER2 ;ERROR REGISTER 2
MOV #16277,@RHOF ;OFFSET REGISTER
MOV #777,@RHCA ;DESIRED CYLINDER
MOV #177777,@RHER3 ;ERROR REGISTER 3
MOV #DMD,@RHMR ;MAINTENANCE REGISTER
MOV #177777,@RHMR ;MAINTENANCE REGISTER

: *BEFORE RESET SAVE REGISTERS IN READ INTO BUFFER

JSR RO,@#SAVER ;SAVE
RHWC ;FROM
REINTO ;TO
17. ;NUMBER

: *GIVE RESET AND REINSTATE UNIT NUMBER

RESET
JSR PC,@#STKINT ;INITIALIZE TK
BIS @#UNIT,@RHCS2

: *CHANGE ORIGINAL SAVED REGISTERS TO EXPECTED VALUES AFTER RESET

CLR @#REINTO+2 ;CLEAR SAVED RHBA
MOV @#UNIT,-(SP) ;GET UNIT NUMBER FRO SAVED RHCS2
BIS #IR,(SP) ;INCLUDE IR
MOV (SP)+,@#REINTO+4 ;SAVED RHCS2
MOV #DVA!RDY!76,@#REINTO+6 ;SAVED RHCS1
CLR @#REINTO+10 ;SAVED RHER1
CLR @#REINTO+14 ;SAVED RHER2
MOV #116000,@#REINTO+16 ;SAVED RHOF
CLR @#REINTO+22 ;SAVED RHER3
CLRB @#REINTO+24 ;SAVED RHAS
MOV #400,@#REINTO+26 ;SAVED RHMR

```
3127 ;*CHANGE RHDS1 WITHOUT CHANGING PROG BIT
3128 014162 013746 003210 MOV @#REINTO+30,-(SP) ;GET RHDS1
3129 014166 042716 176777 BIC #^CPROG,(SP) ;CLEAR EVERYTHING EXCEPT PROG
3130 014172 052716 000700 BIS #700,(SP) ;SET EXPECTED BITS - 'DPR', 'DRY' & 'VV'
3131
3132 014176 012637 003210 4$: MOV (SP)+,@#REINTO+30;SAVED RHDS1
3133 014202 005037 003216 CLR @#REINTO+36 ;SAVED RHEC1
3134 014206 005037 003220 CLR @#REINTO+40 ;SAVED RHEC2
3135
3136 ;*AFTER RESET, SAVE REGISTERS FOR COMPARISONS TO BE DONE
3137 014212 004037 043424 JSR R0,@#SAVER ;SAVE
3138 014216 001632 RHC ;FROM
3139 014220 002114 WRFROM ;TO
3140 014222 000021 17. ;NUMBER
3141
3142 ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
3143 ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
3144 ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
3145 014224 113737 003205 002141 MOV @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS
3146
3147 ;*COMPARE REGISTERS BEFORE RESET WITH REGISTERS AFTER RESET
3148 014232 004037 043626 JSR R0,@#COMPAR ;COMPARE
3149 014236 003160 REINTO ;GOOD BUFFER
3150 014240 002114 WRFROM ;TEST BUFFER
3151 014242 000021 17. ;NUMBER
3152 014244 014252 1$ ;RETURN FOR ERROR
3153 014246 014252 1$ ;SAME
3154 014250 014272 2$ ;RETURN FOR GOOD COMPARISON
3155
3156 014252 013705 047736 1$: MOV @#ERWORD,R5 ;GETTING READY TO INDEX
3157 014256 060505 ADD R5,R5 ;DOUBLE ERROR WORD
3158 014260 016537 001630 042402 MOV RHC-2(R5),@#REGADR ;FAILING REGISTER ADDRESS
3159 014266 104001 ERROR 1 ;REGISTER CONTENTS AFTER
3160 ;A RESET THAT IS AN
3161 ;UNIBUS INITIALIZE CAUSED
3162 ;AN IMPROPER REGISTER CHANGE
3163 014270 000207 RTS PC ;RETURN TO COMPARISON
3164 014272 2$: ;RETURN TO POINT ON GOOD COMPARISON
```

3165
3166
3167
3168
3169
3170
3171
3172
3173
3174
3175
3176
3177
3178
3179
3180
3181
3182
3183
3184
3185
3186
3187
3188
3189
3190
3191
3192
3193
3194
3195
3196
3197
3198
3199
3200
3201
3202
3203
3204
3205
3206
3207
3208
3209
3210
3211
3212
3213
3214
3215
3216
3217
3218
3219
3220

.SBTTL SILO TESTS

; *TEST 27 SILO TST 1

; ** THIS TESTS THE SILO BUFFER IN THE RH11 CONTROLLER
; ** A READ IS ATTEMPTED FROM AN EMPTY SILO
; ** DATA LATE (DLT) (RHCS2), TRANSFER ERROR (TRE) (RHCS1),
; ** SPECIAL CONDITION (SC) (RHCS1) SHOULD SET
; ** THEN LOADING "1" INTO TRE SHOULD CLEAR DLT, TRE AND SC

TST27: SCOPE

MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER

; *CHECK TO SEE IF THE PROGRAM IS RUNNING WITH AN RH70

TST @#RH70 ;TEST FLAG FOR RH70 CONTROLLER
BEQ 30\$;IF FLAG = 1, THIS TEST IS SKIPPED
JMP TST30 ;JUMP TO NEXT TEST -----)
30\$: ;IF FLAG = 0, DO THIS TEST

JSR PC,CLDISK ;CLEAR DISK AND LOAD R'S

MOV @RHDB,R0 ;READ FROM EMPTY SILO
MOV @#UNIT,-(SP) ;GET UNIT NO. IN
BIS #DLT!IR,(SP) ;GET DATA LATE BIT AND IR
JSR PC,@#PUTREG ;SAVE REGISTERS
CMP (SP)+,@#CS2 ;IS DATA LATE BIT UP?
BEQ 1\$;IF YES BRANCH
MOV R2,@#\$BDADR ;IF NOT STORE FAILING REG.
ERROR 11 ;RHCS2 DID NOT HAVE DLT
;RHCS2 SHOULD HAVE ONLY
;DLT AND UNIT NUMBER (BIT 0-2)
;ALL OTHER BITS SHOULD
;BE 0

3204 014360 022737 144200 001714 1\$: CMP #SC!TRE!RDY!DVA,@#CS1 ;IS SPECIAL CONDITION, TRANSFER ERROR
;AND READY UP?

BEQ 2\$;IF YES BRANCH
MOV R1,@#\$BDADR ;IF NOT STORE FAILING REG.
ERROR 11 ;RHCS1 DID NOT HAVE SC, DVA
;TRE AND RDY. AFTER A
;READ FROM EMPTY SILO ONLY
;THESE BITS SHOULD BE UP
;ALL OTHERS SHOULD BE 0

3213 014376 012711 040000 2\$: MOV #TRE,@R1 ;CLEAR ERROR BITS BY MOVING
;ONE INTO TRE IN RHCS1

JSR PC,@#PUTREG ;SAVE REGISTERS
3216 014406 022737 004200 001714 CMP #RDY!DVA,@#CS1 ;ALL BITS BUT RDY AND DVA SHOULD
;BE 0

BEQ 3\$;BRANCH IF YES
MOV R1,@#\$BDADR ;STORE FAILING ADDRESS
ERROR 11 ;AFTER A ONE IN TRE ONLY


```
3221  
3222 ;READY AND DVA SHOULD BE SET IN  
3223 014424 013746 001774 3$: MOV @#UNIT,-(SP) ;RHCS1  
3224 014430 052716 000100 BIS #IR,(SP)  
3225 014434 022637 001712 CMP (SP)+,@#CS2 ;RHCS2 SHOULD HAVE IR AND UNIT ONLY  
3226 014440 001403 BEQ TST30 ;BRANCH IF YES  
3227 014442 010237 001122 MOV R2,@#$BDADR ;STORE FAILING ADDR  
3228 014446 104011 ERROR 11 ;AFTER A ONE IN TRE ONLY
```

```
3229
3230
3231
3232
3233
3234
3235
3236
3237
3238
3239
3240
3241
3242
3243
3244
3245 014450 000004
3246 014452 012706 001000
3247 014456 012737 000030 002032
3248
3249
3250
3251 014464 005737 002040
3252 014470 001402
3253 014472 000137 014676
3254 014476
3255
3256 014476 004737 042732
3257
3258 014502 013746 001774
3259 014506 052716 000100
3260 014512 004737 042336
3261 014516 022637 001712
3262 014522 001403
3263 014524 010237 001122
3264 014530 104011
3265
3266
3267 014532 005077 165072
3268 014536 012777 177777 165064
3269 014544 013737 001636 014554
3270 014552 104415
3271 014554 000000
3272 014556 000200
3273 014560 013746 001774
3274 014564 052716 000300
3275 014570 004737 042336
3276 014574 022637 001712
3277 014600 001403
3278 014602 010237 001122
3279 014606 104011
3280
3281
3282 014610 017700 165014
3283 014614 017705 165010
3284 014620 022700 000000
```

```
*****
:TEST 30      SILO TEST 2
*****
:** THIS TESTS THE IR AND 'OR' BITS OF RHCS2
:** AT THE BEGINNING IR SHOULD BE SET AND 'OR' RESET
:** LOADING 0 IN SILO RESETS IR FOR ONLY 2 MICRO SECONDS
:** THIS TIME CANNOT BE CHECKED BUT IT IS CHECKED TO SEE IF
:** IT DOES GO DOWN OR NOT
:** THEN ALL 1 IS LOADED IN SILO 'OR' SHOULD BECOME SET
:** IN 30 MICRO SECONDS AGAIN TIME IS NOT CHECKED
:** 'OR' SHOULD BE SET
:** THE OUTPUT FROM THE SILO SHOULD BE 0 AND ALL ONES
*****
TST30: SCOPE
MOV      #STACK,SP      ;RESET STACK
MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER

;*CHECK TO SEE IF THE PROGRAM IS RUNNING WITH AN RH70
TST      @#RH70          ;TEST FLAG FOR RH70 CONTROLLER
BEQ      30$             ;IF FLAG = 1, THIS TEST IS SKIPPED
JMP      TST31           ;JUMP TO NEXT TEST -----)
30$:
IF FLAG = 0, DO THIS TEST

JSR      PC,CLDISK      ;CLEAR REGISTERS LOAD R'S

MOV      @#UNIT,-(SP)
BIS      #IR,(SP)
JSR      PC,@#PUTREG    ;SAVE REGISTERS
CMP      (SP)+,@#CS2    ;IR SHOULD BE SET 'OR' RESET
BEQ      1$
MOV      R2,@#$BDADR    ;FAILING REGISTER RHCS2
ERROR    11             ;RHCS2 DOES NOT HAVE IR
;SET, UNIT NO. SET AND
;ALL OTHER BITS 0
1$: CLR      @RHDB        ;LOAD DATA BUFFER (SILO) WITH 0
MCM      #-1,@RHDB      ;LOAD SILO WITH ALL ONES
MOV      @#RHCS2,@#2$   ;ADDRESS OF RHCS2
WAT      ;WAIT TRAP
2$: .WORD ;ADDRESS OF RHCS2
OR
3$: MOV      @#UNIT,-(SP) ;
BIS      #OR!IR,(SP)    ;IR AND 'OR'
JSR      PC,@#PUTREG    ;SAVE REGISTERS
CMP      (SP)+,@#CS2    ;IR AND 'OR' SHOULD BE SET
BEQ      4$
MOV      R2,@#$BDADR    ;SAVE RHCS2 ADDR. FAILING REG.
ERROR    11             ;'OR' IN RHCS2 SHOULD BE
;SET TOGETHER WITH IR AND
;UNIT NO.
4$: MOV      @RHDB,R0    ;SAVE SILO DATA SHOULD BE 0
MOV      @RHDB,R5      ;SAVE SILO DATA SHOULD BE ALL 1
CMP      #0,R0          ;FIRST WORD 0? XYZ DO MORE TEST
```

| | | | | | | | | |
|------|--------|--------|--------|--------|-------|-----------------|-----------------|-----------------------------------|
| 3285 | 014624 | 001410 | | | BEQ | 5\$ | | :BRANCH IF YES |
| 3286 | 014626 | 005037 | 001124 | | CLR | @#\$GDDAT | | :GOOD DATA |
| 3287 | | | | | | | | |
| 3288 | 014632 | 010037 | 001126 | | MOV | R0,@#\$BDDAT | | :BAD DATA |
| 3289 | 014636 | 013737 | 001630 | 042402 | MOV | @#RHDB,@#REGADR | | :SAVE RHDB FAILING REG. |
| 3290 | 014644 | 104001 | | | ERROR | 1 | | :SILO DID NOT HAVE THE FIRST WORD |
| 3291 | | | | | | | | : "0" WHEN "OR" WAS SET |
| 3292 | 014646 | 022705 | 177777 | | 5\$: | CMP | #-1,R5 | :SECOND WORD ALL ONES? |
| 3293 | 014652 | 001411 | | | | BEQ | TST31 | :BRANCH IF YES |
| 3294 | 014654 | 012737 | 177777 | 001124 | | MOV | #-1,@#\$GDDAT | :GOOD DATA |
| 3295 | 014662 | 010537 | 001126 | | | MOV | R5,@#\$BDDAT | :BAD DATA |
| 3296 | 014666 | 013737 | 001530 | 042402 | | MOV | @#RHDB,@#REGADR | :SAVE RHDB FAILING REG. |
| 3297 | 014674 | 104001 | | | | ERROR | 1 | :SILO DID NOT HAVE THE SECOND |
| 3298 | | | | | | | | :WORD OF ALL ONES WHEN "OR" |
| 3299 | | | | | | | | :WAS SET |
| 3300 | | | | | | | | |

```
3301
3302
3303
3304
3305
3306
3307
3308
3309
3310
3311 014676 000004
3312 014700 012737 000031 002032
3313
3314
3315
3316 014706 005737 002040
3317 014712 001402
3318 014714 000137 015136
3319 014720
3320
3321
3322 014720 012700 051534
3323 014724 012705 000103
3324 014730 005020
3325 014732 005305
3326 014734 001375
3327 014736 004737 042732
3328 014742 005000
3329 014744 012705 000102
3330 014750 010077 164654
3331 014754 005200
3332 014756 005305
3333 014760 001373
3334 014762 013746 001774
3335 014766 052716 000200
3336 014772 004737 042336
3337 014776 022637 001712
3338 015002 001405
3339 015004 010237 001122
3340 015010 104011
3341 015012 005037 002006
3342 015016 012700 051534
3343 015022 012705 000102
3344 015026 017720 164576
3345 015032 005305
3346 015034 001374
3347 015036 012700 051534
3348 015042 012705 000102
3349 015046 005046
3350 015050 021620
3351 015052 001425
3352 015054 014037 001126
3353 015060 011637 001124
3354 015064 013737 001630 042402
3355 015072 005737 002006
3356 015076 001002
```

```
*****
*TEST 31          SILO TEST 3
*****
**      THIS TESTS SILO BUFFER BY FILLING IT WITH A COUNT FROM
**      0 TO 65 AND THEN CHECKING IF IR IS DOWN AND "OR"
**      IS HIGH AND COMPARING THE SILO OUTPUT.
*****
TST31:  SCOPE
MOV     #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
; *CHECK TO SEE IF THE PROGRAM IS RUNNING WITH AN RH70
TST     @#RH70         ;TEST FLAG FOR RH70 CONTROLLER
BEQ     30$            ;IF FLAG = 1, THIS TEST IS SKIPPED
JMP     TST32         ;JUMP TO NEXT TEST -----)
30$:
;IF FLAG = 0, DO THIS TEST

MOV     #SILOTB,R0    ;TABLE POINTER
MOV     #66.,R5       ;COUNTER
1$:     CLR     (R0)+   ;CLEAR TOTAL TABLE
        DEC     R5     ;COUNT
        BNE    1$     ;BRANCH IF NOT COMPLETELY CLEAR
        JSR    PC,@#CLDISK ;CLEAR ALL REG.
        CLR     R0
        MOV     #66.,R5 ;COUNT
2$:     MOV     R0,@#RHDB ;LOAD SILO WITH COUNT FROM 0 TO 65
        INC     R0     ;NEXT COUNT
        DEC     R5     ;IS 66 LOADS DONE?
        BNE    2$     ;BRANCH IF NOT.
        MOV     @#UNIT,-(SP)
        BIS    #OR,(SP)
        JSR    PC,@#PUTREG ;SAVE REGISTERS
        CMP    (SP)+,@#CS2 ;"OR" SHOULD BE SET IR RESET
        BEQ    3$     ;BRANCH IF YES
        MOV     R2,@#SBDADR ;SAVE RHCS2 ADR. FAILING REG.
        ERROR  11    ;"OR" WAS NOT SET, IR WAS NOT
        CLR    @#ERFLG$ ;RESET AFTER SILO WAS FULL
3$:     MOV     #SILOTB,R0 ;POINTER
        MOV     #66.,R5   ;COUNTER
4$:     MOV     @#RHDB,(R0)+ ;READ SILO
        DEC     R5     ;COUNT
        BNE    4$     ;BRANCH IF 66 NOT DONE
        MOV     #SILOTB,R0 ;POINTER
        MOV     #66.,R5
5$:     CLR    -(SP)
        CMP    (SP),(R0)+
        BEQ    7$     ;BRANCH IF GOOD
        MOV    -(R0),@#SBDAT ;BAD DATA
        MOV    (SP),@#SGDDAT ;GOOD DATA
        MOV    @#RHDB,@#REGADR ;FAILING REG. RHDB
        TST    @#ERFLG$   ;IS THIS FIRST ERROR?
        BNE    6$     ;IF NOT BRANCH
6$:

```

| | | | | | | | | | |
|------|--------|--------|--------|------------|-------------------|--|--|--|--|
| 3357 | 015100 | 104012 | | ERROR | 12 | | | | |
| 3358 | 015102 | 000401 | | BR | 64\$ | | | | |
| 3359 | 015104 | 104013 | | 6\$: ERROR | 13 | | | | |
| 3360 | | | | | | | | | |
| 3361 | | | | | | | | | |
| 3362 | | | | | | | | | |
| 3363 | | | | | | | | | |
| 3364 | | | | | | | | | |
| 3365 | | | | | | | | | |
| 3366 | 015106 | 005720 | | 64\$: TST | (R0)+ | | | | |
| 3367 | | | | | | | | | |
| 3368 | 015110 | 017746 | 164024 | MOV | @SWR,-(SP) | | | | |
| 3369 | 015114 | 042716 | 177577 | BIC | #^CSW07:SW08,(SP) | | | | |
| 3370 | 015120 | 022726 | 000200 | CMP | #SW07,(SP)+ | | | | |
| 3371 | 015124 | 001403 | | BEQ | 10\$ | | | | |
| 3372 | 015126 | 005216 | | 7\$: INC | (SP) | | | | |
| 3373 | 015130 | 005305 | | DEC | R5 | | | | |
| 3374 | 015132 | 001346 | | BNE | 5\$ | | | | |
| 3375 | 015134 | 005726 | | 10\$: TST | (SP)+ | | | | |

:THESE TWO ERROR CALLS ARE FOR
:BRANCH TO AVOID PRINTING NEXT ERROR
:THE SAME TYPEOUT. SILO
:HAD A COUNT WRITTEN IN.
:ON READ OUT AN ERROR WAS
:DETECTED. THE TOTAL SILO
:READOUT IS IN LOCATION
: "SILOTB" TO THE NEXT 65
:WORDS.
:INCREMENT (R0)
:ARE FURTHER COMPARES TO
:BE DONE
:ONLY KEEP SW7 AND SW8
:TEST SW07
:IF NO MORE COMPARE THEN BRANCH
:NEXT GOOD WORD
:COUNT
:BRANCH IF 66 NOT COMPLETE
:POP STACK

```

3376
3377
3378
3379
3380
3381
3382
3383
3384
3385 015136 000004
3386 015140 012737 000032 002032
3387
3388
3389
3390 015146 005737 002040
3391 015152 001402
3392 015154 000137 015262
3393 015160
3394
3395 015160 004737 042732
3396
3397 015164 005000
3398 015166 005200
3399 015170 010077 164434
3400 015174 022700 000103
3401 015200 001401
3402 015202 000771
3403 015204 004737 042336
3404
3405 015210 032737 100000 001712
3406 015216 001003
3407 015220 010237 001122
3408 015224 104011
3409 015226 017737 164376 001126
3410 015234 012737 000001 001124
3411 015242 023737 001124 001126
3412 015250 001404
3413 015252 013737 001630 042402
3414 015260 104012
3415

```

```

*****
:*TEST 32      SILO TEST4
:**          NOW PUT 67 WORDS INTO SILO AND CHECK FOR DLT
:**          EVEN AFTER THE 67TH. WORD INPUT THE FIRST WORD SHOULD NOT CHANGE
*****
TST32: SCOPE
MOV      #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER

;*CHECK TO SEE IF THE PROGRAM IS RUNNING WITH AN RH70

TST      @#RH70        ;TEST FLAG FOR RH70 CONTROLLER
BEQ      30$           ;IF FLAG = 1, THIS TEST IS SKIPPED
JMP      TST33         ;JUMP TO NEXT TEST -----)
30$:
;IF FLAG = 0, DO THIS TEST

JSR      PC,@#CLDISK  ;CLEAR DISK REG.

CLR      R0            ;CLEAR R0
1$:      INC          R0 ;ADD 1
MOV      R0,@#RHDB    ;LOAD SILO
CMP      #67.,R0      ;67 DONE?
BEQ      2$           ;BRANCH IF YES
BR       1$           ;NO SO BRANCH
2$:      JSR          PC,@#PUTREG ;SAVE REGISTERS

BIT      #DLT,@#CS2   ;DLT SET?
BNE      3$           ;BRANCH IF YES
MOV      R2,@#SBDADR  ;FAILING ADDRESS RHCS2
ERROR    11          ;DATA LATE DID NOT SET AT 67TH.
3$:      MOV          @#RHDB,@#SBDDAT ;INPUT TO SILO
MOV      #1,@#SGDDAT ;GOOD DATA
CMP      @#SGDDAT,@#SBDDAT ;COMPARE
BEQ      TST33        ;BRANCH IF GOOD
MOV      @#RHDB,@#REGADR ;FAILING REG. RHDB
ERROR    12          ;WORD IN RHDB CHANGED
;AFTER THE 67TH INPUT.

```

```
3416
3417
3418
3419
3420
3421
3422
3423
3424
3425
3426
3427 015262 000004
3428
3429 015264 012737 000033 002032
3430
3431
3432
3433 015272 005737 002040
3434 015276 001402
3435 015300 000137 015562
3436 015304
3437
3438 015304 004737 042732
3439
3440 015310 013746 001774
3441 015314 052716 000100
3442 015320 004737 042336
3443 015324 022637 001712
3444 015330 001403
3445 015332 010237 001122
3446 015336 104011
3447
3448 015340 013700 001630
3449 015344 005001
3450 015346 010110
3451
3452 015350 005201
3453 015352 022701 000004
3454 015356 103373
3455 015360 013737 001636 015370
3456 015366 104415
3457 015370 000000
3458 015372 000200
3459 015374 004737 042732
3460 015400 013746 001774
3461 015404 052716 000100
3462 015410 004737 042336
3463 015414 022637 001712
3464 015420 001403
3465 015422 010237 001122
3466 015426 104011
3467
3468 015430 013700 001630
3469 015434 012710 000004
3470 015440 011201
3471 015442 011005

*****
*TEST 33      SILO TEST 5
**          THE SILO IS LOADED WITH 0,1,2,3 THEN AFTER
**          'OR' IS UP A CLR IN RHCS2 IS DONE THEN 4,
**          IS LOADED. AFTER 'OR' IS UP 2 READS FROM
**          SILO ARE DONE, ON THE LAST, 'DTL' IN RHCS2 SHOULD BE SET.
*****
TST33: SCOPE
MOV      #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
;*CHECK TO SEE IF THE PROGRAM IS RUNNING WITH AN RH70
TST      @#RH70 ;TEST FLAG FOR RH70 CONTROLLER
BEQ      30$ ;IF FLAG = 1, THIS TEST IS SKIPPED
JMP      TST34 ;JUMP TO NEXT TEST -----)
30$:
IF FLAG = 0, DO THIS TEST
JSR      PC,@#CLDISK ;CLEAR DISK
MOV      @#UNIT,-(SP) ;GET UNIT NO.
BIS      #IR,(SP) ;SET INPUT READY
JSR      PC,@#PUTREG ;SAVE REGISTERS
CMP      (SP)+,@#CS2 ;IR SHOULD BE SET 'OR' CLEARED
BEQ      1$ ;BRANCH IF GOOD
MOV      R2,@#$BDADR ;FAILING REGISTER RHCS2
ERROR    11 ;RHCS2 DOES NOT HAVE IR SET
;AND ALL OTHER BITS 0
1$: MOV      @#RHDB,R0 ;R0 HAS RHDB ADDRESS
CLR      R1 ;DATA
2$: MOV      R1,@R0 ;0, THEN 1 THEN 2 THEN 3
;IN RHDB
INC      R1 ;INCREMENT DATA
CMP      #4,R1 ;IS 4 DONE
BHS      2$ ;BRANCH IF NOT
MOV      @#RHCS2,@#3$
;
3$: .WORD    0 ;WAIT FOR 'OR'
OR ;RHCS2 ADDRESS
;WAIT ON OR.
JSR      PC,@#CLDISK ;CLR IN RHCS2
MOV      @#UNIT,-(SP) ;UNIT NO.
BIS      #IR,(SP)
JSR      PC,@#PUTREG ;SAVE REGISTERS
CMP      (SP)+,@#CS2 ;IR SHOULD BE SET '0'=0
BEQ      4$ ;BRANCH IF GOOD
MOV      R2,@#$BDADR ;FAILING REGISTER RHCS2
ERROR    11 ;RHCS2 DOES NOT HAVE IR SET
;AND ALL OTHER BITS 0
4$: MOV      @#RHDB,R0 ;R0 HAS RHDB ADDRESS
MOV      #4,@R0 ;LOAD 4 IN SILO
MOV      @R2,R1 ;SAVE RHCS2
MOV      @R0,R5 ;READ THE 4 IN SILO
```

```
3472 015444 011003      MOV    @R0,R3      ;READ SILO TO GET DLT
3473 015446 011204      MOV    @R2,R4      ;SAVE RHCS2
3474 015450 032701 000200  BIT    #OR,R1      ;TEST FOR OR IN RHCS2
3475 015454 001424      BEQ    6$          ;IF OR IS NOT SET BRANCH
3476 015456 022705 000004  CMP    #4,R5      ;SILO 4 IS NOW COMPARED
3477 015462 001410      BEQ    5$
3478 015464 010037 042402  MOV    R0,@#REGADR ;SILO ADDRESS
3479 015470 012737 000004 001124  MOV    #4,@#$GDDAT ;GOOD DATA
3480 015476 010537 001126  MOV    R5,@#$BDDAT ;BAD DATA
3481 015502 104001      ERROR 1          ;SILO DID NOT CONTAIN WORD
3482                                ;PUT IN AFTER "OR" WAS UP
3483 015504 005703      5$:  TST    R3      ;IS IT ZERO BECAUSE SILO
3484                                ;IS DESTRUCTIVE READ
3485 015506 001407      BEQ    6$          ;BRANCH IF GOOD
3486 015510 010037 042402  MOV    R0,@#REGADR ;SILO ADDRESS
3487 015514 005037 001124  CLR    @#$GDDAT    ;GOOD DATA
3488 015520 010337 001126  MOV    R3,@#$BDDAT ;BAD DATA
3489 015524 104001      ERROR 1          ;SILO SHOULD BE ZERO
3490                                ;AFTER THE ONE WORD PUT IN
3491                                ;HAS BEEN TAKEN OUT AS
3492                                ;SILO IS A DESTRUCTIVE READ
3493 015526 032704 100000      6$:  BIT    #DLT,R4
3494 015532 001013      BNE    TST34      ;BRANCH IF DLT SET
3495 015534 013746 001774  MOV    @#UNIT,-(SP) ;GET UNIT NO
3496 015540 052716 100300  BIS    #DLT!OR!IR,(SP) ;
3497 015544 012637 001124  MOV    (SP)+,@#$GDDAT ;GOOD DATA
3498 015550 010437 001126  MOV    R4,@#$BDDAT ;BAD DATA
3499 015554 010237 042402  MOV    R2,@#REGADR ;RHCS2 ADDRESS
3500 015560 104001      ERROR 1          ;DATA LATE ERROR
```



```
3501
3502
3503      .SBTTL  MORE REGISTER TESTS
3504
3505
3506      ;:*****
3507      ;*TEST 34      TEST ODD BYTE INSTRUCTION ON RHCS1 - RH11
3508
3509      ;**      RDY (BIT 07) AND DVA (BIT 11) SHOULD ALWAYS BE SET
3510
3511      ;:*****
3512      TST34:  SCOPE
3513
3514      015562  000004      MOV      #TTNO,@#TSTNM      ;THIS SAVES TEST NUMBER
3515
3516      ;*CHECK TO SEE IF THE PROGRAM IS RUNNING WITH AN RH70
3517
3518      015572  005737  002040      TST      @#RH70      ;TEST FLAG FOR RH70 CONTROLLER
3519      015576  001402      BEQ      30$      ;IF FLAG = 1, THIS TEST IS SKIPPED
3520      015600  000137  015720      JMP      TST35      ;JUMP TO NEXT TEST -----)
3521      015604      30$:      ;IF FLAG = 0, DO THIS TEST
3522
3523      015604  012706  001000      MOV      #STACK,SP      ;RESET STACK
3524      015610  004737  042732      JSR      PC,CLDISK      ;CLEAR DISK REG.
3525
3526      015614  012711  003566      MOV      #3566,@R1      ;LOAD RHCS1 WITH ANY NUMBER
3527      015620  010146      MOV      R1,-(SP)      ;GETTING READY TO FORM ODD BYTE
3528      015622  005216      INC      (SP)      ;SP NOW HAS ODD BYTE FOR RHCS1
3529      015624  112736  000005      MOVB    #5,@(SP)+      ;MOVE 5 INTO ODD BYTE FOR RHCS1
3530      015630  011137  001126      MOV      @R1,@#SBDDAT      ;TEST DATA
3531      015634  022737  006766  001126      CMP      #2566!DVA!RDY,@#SBDDAT ;RHCS1 SHOULD HAVE 6766
3532      015642  001406      BEQ      1$      ;BRANCH IF GOOD
3533      015644  012737  006766  001124      MOV      #2566!DVA!RDY,@#GDAT ;GOOD DATA
3534      015652  010137  042402      MOV      R1,@#REGADR      ;FAILING REGISTER RHCS1
3535      015656  104001      ERROR   1      ;MOVING A NUMBER INTO
3536      ;ODD BYTE OF RHCS1 GAVE
3537      ;WRONG RESULTS
3538
3539      015660  112711  000032      1$:      MOVB    #32,@R1      ;MOVE INTO EVEN BYTE
3540      015664  011137  001126      MOV      @R1,@#SBDDAT      ;TEST DATA
3541      015670  022737  006632  001126      CMP      #2432!DVA!RDY,@#SBDDAT ;RHCS1 SHOULD HAVE 6632
3542      015676  001460      BEQ      TST36      ;DO NEXT RH11 TEST IF GOOD -----)
3543      015700  012737  006632  001124      MOV      #2432!DVA!RDY,@#GDAT ;GOOD DATA
3544      015706  010137  042402      MOV      R1,@#REGADR      ;FAILING REGISTER RHCS1
3545      015712  104001      ERROR   1      ;MOVING A NUMBER INTO EVEN
3546      ;BYTE OF RHCS1 GAVE WRONG
3547      ;RESULT
3548
3549      015714  000137  016040      JMP      TST36      ;SKIP RH70 TEST -----)
3550
```

```
3551
3552
3553
3554      ;*****
3555      ;*TEST 35      TEST ODD BYTE INSTRUCTION ON RHCS1 - RH70
3556
3557      ;**      RDY (BIT 07) AND DVA (BIT 11) SHOULD ALWAYS BE SET
3558      ;*****
3559      TST35: SCOPE
3560
3561      015720 000004      MOV      #TTNO,@#TSTNM      ;THIS SAVES TEST NUMBER
3562      015722 012737 000035 002032      MOV      #STACK,SP      ;RESET STACK
3563      015730 012706 001000      JSR      PC,CLDISK      ;CLEAR DISK REG.
3564      015734 004737 042732
3565      015740 012711 003566      MOV      #3566,@R1      ;LOAD RHCS1 WITH ANY NUMBER
3566      015744 010146      MOV      R1,-(SP)      ;GETTING READY TO FORM ODD BYTE
3567      015746 005216      INC      (SP)      ;SP NOW HAS ODD BYTE FOR RHCS1
3568      015750 112736 000005      MOV      #5,@(SP)+      ;MOVE 5 INTO ODD BYTE FOR RHCS1
3569      015754 011137 001126      MOV      @R1,@#SBDDAT      ;TEST DATA
3570
3571      015760 022737 004766 001126      CMP      #566!DVA!RDY,@#SBDDAT ;RHCS1 SHOULD HAVE 4766
3572      015766 001406      BEQ      1$      ;BRANCH IF GOOD
3573      015770 012737 004766 001124      MOV      #566!DVA!RDY,@#SGDDAT ;GOOD DATA
3574      015776 010137 042402      MOV      R1,@#REGADR      ;FAILING REGISTER RHCS1
3575      016002 104001      ERROR   1      ;MOVING A NUMBER INTO
3576      ;ODD BYTE OF RHCS1 GAVE
3577      ;WRONG RESULTS
3578
3579      016004 112711 000032      1$:      MOV      #32,@R1      ;MOVE INTO EVEN BYTE
3580      016010 011137 001126      MOV      @R1,@#SBDDAT      ;TEST DATA
3581      016014 022737 004632 001126      CMP      #432!DVA!RDY,@#SBDDAT ;RHCS1 SHOULD HAVE 4632
3582      016022 001406      BEQ      TST36      ;BRANCH IF GOOD
3583      016024 012737 004632 001124      MOV      #432!DVA!RDY,@#SGDDAT ;GOOD DATA
3584      016032 010137 042402      MOV      R1,@#REGADR      ;FAILING REGISTER RHCS1
3585      016036 104001      ERROR   1      ;MOVING A NUMBER INTO EVEN
3586      ;BYTE OF RHCS1 GAVE WRONG
3587      ;RESULTS
3588
```

```
3589
3590
3591      ;*****
3592      ;*TEST 36      TEST ODD BYTE INSTRUCTION ON RHCS2
3593
3594      ;**      IR (BIT 06) AND THE UNIT SELECT (BIT 0-2) WILL BE SET
3595
3596      ;*****
3597 016040 000004      TST36: SCOPE
3598
3599 016042 012737 000036 002032      MOV      #TTNO,@#TSTNM      ;THIS SAVES TEST NUMBER
3600
3601      ;*CHECK TO SEE IF THE PROGRAM IS RUNNING WITH AN RH70
3602
3603 016050 005737 002040      TST      @#RH70      ;TEST FLAG FOR RH70 CONTROLLER
3604 016054 001402      BEQ      30$      ;IF FLAG = 1, THIS TEST IS SKIPPED
3605 016056 000137 016216      JMP      TST37      ;JUMP TO NEXT TEST -----)
3606 016062      30$:      ;IF FLAG = 0, DO THIS TEST
3607
3608 016062 004737 042732      JSR      PC,@#CLDISK      ;GIVE INIT & SETUP REGISTER CORRES
3609
3610 016066 052712 177000      BIS      #177000,(R2)      ;LOAD RHCS2
3611 016072 010246      MOV      R2,-(SP)      ;GETTING READY FOR ODD BYTE
3612 016074 005216      INC      (SP)      ;SP NOW HAS ODD BYTE FOR RHCS2
3613 016076 105036      CLRB    @#(SP)+      ;CLERR RHCS2 ODD BYTE
3614 016100 013746 001774      MOV      @#UNIT,-(SP)      ;GET UNIT NO.
3615 016104 052716 000100      BIS      #IR,(SP)      ;INPUT READY AS IT IS SET
3616 016110 011237 001126      MOV      @R2,@#SBDDAT      ;TEST DATA
3617 016114 022637 001126      CMP      (SP)+,@#SBDDAT      ;COMPARE TO SEE THAT
3618      ;"CLRB" DID CLEAR
3619 016120 001411      BEQ      1$
3620 016122 013737 001774 001124      MOV      @#UNIT,@#$GDDAT
3621 016130 052737 000100 001124      BIS      #IR,@#$GDDAT      ;GOOD DATA
3622 016136 010237 042402      MOV      R2,@#REGADR      ;FAILING REGISTER RHCS2
3623 016142 104001      ERROR   1      ;CLEARING ODD BYTE OF RHCS2
3624      ;GAVE WRONG RESULTS
3625 016144 013746 001774      1$:      MOV      @#UNIT,-(SP)
3626 016150 052716 000010      BIS      #BAI,(SP)
3627 016154 052712 020000      BIS      #UPE,@R2      ;HAVE UPE AND MPE IN RHCS2
3628      ;BESIDES UNIT SELECT
3629 016160 112612      MOVB    (SP)+,@R2      ;MOVE INTO EVEN BYTE OF RHCS2
3630 016162 013746 001774      MOV      @#UNIT,-(SP)
3631 016166 052716 020110      BIS      #UPE!IR!BAI,(SP)
3632 016172 011637 001124      MOV      (SP),@#$GDDAT      ;GOOD DATA
3633 016176 011237 001126      MOV      @R2,@#SBDDAT      ;TEST DATA
3634 016202 022637 001126      CMP      (SP)+,@#SBDDAT      ;COMPARE TO SEE THAT MOV B DID
3635      ;MOVE EVEN BYTE ONLY
3636 016206 001403      BEQ      TST37      ;;BRANCH IF GOOD
3637 016210 010237 042402      MOV      R2,@#REGADR      ;FAILING REGISTER RHCS2
3638 016214 104001      ERROR   1      ;MOVING A NUMBER INTO EVEN
3639      ;BYTE OF RHCS2 GAVE WRONG
3640      ;RESULTS
3641
```

3642
3643
3644
3645
3646
3647
3648
3649
3650
3651
3652
3653
3654
3655
3656
3657
3658
3659
3660
3661
3662
3663
3664
3665
3666
3667
3668
3669
3670
3671
3672
3673

016216 000004
016220 012737 000037 002032
016226 012706 001000
016232 004737 042732
016236 013704 001632
016242 012714 025252
016246 010446
016250 005216
016252 112736 000377
016256 011437 001126
016262 022737 177652 001126
016270 001406
016272 012737 177652 001124
016300 010437 042402
016304 104001
016306 112714 000123
016312 011437 001126
016316 022737 177523 001126
016324 001406
016326 012737 177523 001124
016334 010437 042402
016340 104001

```
*****  
;*TEST 37 ODD BYTE TEST ON RHWC  
*****  
;** IN THIS REGISTER NO BITS SHOULD BE PERMANENTLY SET  
*****  
TST37: SCOPE  
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER  
MOV #STACK,SP ;RESET STACK  
JSR PC,CLDISK ;CLEAR DISK REGISTERS  
MOV @#RHWC,R4 ;R4 NOW IS WORD COUNT REGISTER  
MOV #25252,@R4 ;LOAD RHWC  
MOV R4,-(SP) ;GETTING READY TO FORM ODD BYTE  
INC (SP) ;SP NOW HAS ODD BYTE FOR RHWC  
MOVB #377,@(SP)+ ;MOVE 377 INTO ODD BYTE OF RHWC  
MOV @R4,@#SBDDAT ;TEST DATA  
CMP #177652,@#SBDDAT ;COMPARE TO SEE IF MOVB DID OK  
BEQ 1$ ;BRANCH IF GOOD  
MOV #177652,@#SGDDAT ;GOOD DATA  
MOV R4,@#REGADR ;REGISTER FAILING RHWC  
ERROR 1 ;MOVING INTO ODD BYTE OF RHWC  
 ;GAVE WRONG RESULTS  
1$: MOVB #123,@R4 ;MOVE INTO EVEN BYTE OF RHWC  
MOV @R4,@#SBDDAT ;TEST DATA  
CMP #177523,@#SBDDAT  
BEQ TST40 ;;BRANCH IF GOOD  
MOV #177523,@#SGDDAT ;GOOD DATA  
MOV R4,@#REGADR ;REGISTER FAILING RHWC  
ERROR 1
```

3674
3675
3676
3677
3678
3679
3680
3681
3682
3683
3684
3685
3686
3687
3688
3689
3690
3691
3692
3693
3694
3695
3696
3697
3698
3699
3700
3701
3702
3703
3704
3705
3706
3707

016342 000004

016344 012706 001000
016350 012737 000040 002032
016356 004737 042732
016362 013704 001634
016366 012714 025253
016372 010446
016374 005216
016376 112736 000377
016402 011437 001126
016406 022737 177652 001126
016414 001406
016416 012737 177652 001124
016424 010437 042402
016430 104001
016432 112714 000125
016436 011437 001126
016442 022737 177524 001126
016450 001406
016452 012737 177524 001124
016460 010437 042402
016464 104001

```
*****  
;*TEST 40 TEST ODD BYTE INSTRUCTION ON RHBA  
** BIT 0 SHOULD ALWAYS BE 0  
*****  
TST40: SCOPE  
MOV #STACK,SP ;RESET STACK  
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER  
JSR PC,CLDISK  
MOV @#RHBA,R4 ;R4 HAS ADDRESS OF RHBA  
MOV #25253,@R4 ;LOAD RHBA  
MOV R4,-(SP) ;GETTING READY FOR ODD BYTE  
INC (SP) ;SP HAS ODD BYTE ADR. OF RHBA  
MOVB #377,@(SP)+ ;LOAD ODD BYTE OF RHBA  
MOV @R4,@#SBDDAT ;TEST DATA  
CMP #177652,@#SBDDAT ;COMPARE MOVB RESULTS  
BEQ 1$ ;BRANCH IF GOOD  
MOV #177652,@#SGDDAT ;GOOD DATA  
MOV R4,@#REGADR ;FAILING REGISTER RHBA  
ERROR 1 ;MOVING INTO ODD BYTE OF  
 ;RHBA GAVE WRONG RESULTS  
1$: MOVB #125,@R4  
MOV @R4,@#SBDDAT ;TEST DATA  
CMP #177524,@#SBDDAT  
BEQ TST41 ;:BRANCH IF GOOD  
MOV #177524,@#SGDDAT ;GOOD DATA  
MOV R4,@#REGADR ;FAILING REGISTER RHBA  
ERROR 1 ;MOVING INTO EVEN BYTE OF  
 ;RHBA GAVE WRONG RESULTS
```

3708
3709
3710
3711
3712
3713
3714
3715
3716
3717
3718
3719
3720
3721
3722
3723
3724
3725
3726
3727
3728
3729
3730
3731
3732
3733
3734
3735
3736
3737
3738
3739
3740
3741
3742
3743
3744
3745
3746
3747
3748
3749
3750
3751
3752
3753
3754
3755
3756
3757
3758
3759
3760
3761
3762
3763

.SBTTL DCL COMMAND TESTS

```
*****  
**      FOUR GENERAL REGISTERS WILL BE RESERVED FOR HARDWARE  
**      R1=RHCS1 CONTROL AND STATUS1  
**      R2=RHCS2 CONTROL AND STATUS2  
**      R3=RHDS1 DRIVE STATUS 1  
**      R4=RHER1 ERROR REGISTER1  
  
**      WHENEVER ANY OTHER USE IS MADE OF THESE REGISTERS  
**      APPROPRIATE SAVING MUST BE DONE  
  
*****
```

```
*****  
**      ERROR REGISTER #01 (RHER1) TEST  
**      BIT #1 (ILLEGAL REGISTER) CANNOT BE TESTED ON PDP11 THIS BIT  
**      IS FOR PDP10 USE ONLY  
  
*****
```

```
*****  
**TEST 41      TEST ILF BIT #0 IN REG. RHER1  
  
**      ALL 3 ILLEGAL FUNCTION CODES SHOULD SET - ATA,ERR,ILF - AND ARE TESTED  
**      A GO WITHOUT CLEARING ILF ERR SHOULD SET - MXF,DLT,TRE - BITS AND THEY ARE ALSO
```

```
*****  
TST41: SCOPE
```

```
MOV      #STACK,SP      ;RESET STACK  
MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER  
JSR      PC,@#CLDISK    ;CLEAR REGISTERS  
MOV      #DMD,@RHMR     ;SET DIAGNOSTIC MODE  
CLR      @#TMPILL       ;GET READY TO MAKE ILLEGAL FUNCTION  
1$: MOV   #FUTABL,RO     ;LOAD FUNCTION CODE TABLE START  
MOV      #17.,R5        ;COUNTER (16 GOOD FUNCTIONS)  
2$: CMP   @#TMPILL,(RO)+ ;IS THIS A LEGAL FUNCTION CODE?  
BNE      3$             ;NO - DECR. FUNCT. CODE CTR  
ADD      #2,@#TMPILL    ;YES MAKE NEXT FUNCTION CODE  
BR       1$             ;TEST NEXT FUNCTION CODE  
3$: DEC   R5             ;MAKE NEXT CODE IF 1ST 16  
                     ;LEGAL FUNCTIONS NOT DONE  
BNE      2$             ;BRANCH IF 16 NOT COMPLETE  
BIT      #100,@#TMPILL  ;ALL BITS UP TO BIT #5 COMPARED?  
BNE      12$            ;YES - EXIT ----->  
MOV      @#TMPILL,@#ILLEG;NO - TEST THE ILLEGAL FUNCTION  
ADD      #2,@#TMPILL    ;TEST NEW FUNCTION CODE NEXT TIME
```

| | | | | | | | | |
|------|--------|--------|--------|--------|-------|-------|------------------|--|
| 3764 | 016576 | 004737 | 042732 | | 4\$: | JSR | PC,@#CLDISK | |
| 3765 | 016602 | 012777 | 000001 | 163050 | | MOV | #DMD,@RHMR | :SET DIAGNOSTIC MODE |
| 3766 | 016610 | 013711 | 002106 | | | MOV | @#ILLEGL,@R1 | :ILLEGAL FUNCTION ----> RHCS1 |
| 3767 | 016614 | 012737 | 016576 | 001110 | | MOV | #4\$,@#SLPERR | :ERROR RETURN POINT |
| 3768 | 016622 | 004737 | 042336 | | | JSR | PC,@#PUTREG | :SAVE REGISTERS |
| 3769 | 016626 | 005737 | 001716 | | | TST | @#RER1 | :THERE SHOULD NOT BE ANY ERROR YET |
| 3770 | 016632 | 001403 | | | | BEQ | 5\$ | :CONTINUE IF RHER1 STILL = 0 |
| 3771 | 016634 | 010437 | 001122 | | | MOV | R4,@#SBDADR | :FAILING REGISTER ADDRESS RHER1 |
| 3772 | 016640 | 104011 | | | | ERROR | 11 | :ALTHOUGH AN ILLEGAL FUNCTION |
| 3773 | | | | | | | | :HAS BEEN MOVED INTO RHCS1 |
| 3774 | | | | | | | | :NO ERRORS SHOULD SHOW TILL |
| 3775 | | | | | | | | :GO IS SET RHER1 SHOULD BE |
| 3776 | | | | | | | | :ALL ZEROS |
| 3777 | | | | | | | | |
| 3778 | 016642 | 052711 | 000001 | | 5\$: | BIS | #GO,@R1 | :GO IN RHCS1 |
| 3779 | 016646 | 004737 | 042336 | | | JSR | PC,@#PUTREG | :SAVE REGISTERS |
| 3780 | 016652 | 022737 | 000001 | 001716 | | CMP | #ILF,@#RER1 | :ILLEGAL FUNCTION BIT SHOULD BE SET |
| 3781 | 016660 | 001403 | | | | BEQ | 6\$ | :IT IS - CONTINUE |
| 3782 | 016662 | 010437 | 001122 | | | MOV | R4,@#SBDADR | :FAILING REGISTER ADDRESS RHER1 |
| 3783 | 016666 | 104011 | | | | ERROR | 11 | :ILLEGAL FUNCTION DID NOT |
| 3784 | | | | | | | | :SET ON AN ILLEGAL FUNCTION |
| 3785 | | | | | | | | :EXECUTION, THE ILLEGAL FUNCTION |
| 3786 | | | | | | | | :BEING EXECUTED IS IN RHCS1 |
| 3787 | | | | | | | | |
| 3788 | 016670 | 013746 | 001736 | | 6\$: | MOV | @#DS1,-(SP) | :GET RHDS1 |
| 3789 | 016674 | 042716 | 001000 | | | BIC | #PROG,(SP) | :MASK PROG |
| 3790 | 016700 | 022726 | 140700 | | | CMP | #ATA!ERR!VV!DPR! | DRY,(SP)+ |
| 3791 | | | | | | | | :ATTENTION (BIT 15) |
| 3792 | | | | | | | | :VOLUME VALID (BIT 6) |
| 3793 | | | | | | | | :COMPOSIT ERROR (BIT 14) |
| 3794 | | | | | | | | :DEVICE READY (BIT 7) SHOULD |
| 3795 | | | | | | | | :BE SET ON RHDS1 |
| 3796 | 016704 | 001404 | | | | BEQ | 7\$ | :THEY ARE - CONTINUE |
| 3797 | 016706 | 013737 | 001662 | 001122 | | MOV | @#RHDS1,@#SBDADR | :FAILING REGISTER ADDRESS RHDS1 |
| 3798 | 016714 | 104011 | | | | ERROR | 11 | :FOLLOWING BITS SHOULD BE SET |
| 3799 | | | | | | | | :WITH AN ILLEGAL FUNCTION |
| 3800 | | | | | | | | :ATTENTION (BIT 15) |
| 3801 | | | | | | | | :COMPOSIT ERROR (BIT 14) |
| 3802 | | | | | | | | :MEDIUM ON LINE (BIT 12) |
| 3803 | | | | | | | | :DEVICE READY (BIT 7) |
| 3804 | | | | | | | | |
| 3805 | 016716 | 004737 | 045172 | | 7\$: | JSR | PC,@#MIDDLE | :GIVE A WRITE HEADER AND |
| 3806 | | | | | | | | :DATA COMMAND WITHOUT |
| 3807 | | | | | | | | :CLEARING THE ERRORS |
| 3808 | | | | | | | | :USING "MIDDLE" SO THAT |
| 3809 | | | | | | | | :IT WILL COME BACK BEFORE |
| 3810 | | | | | | | | :THE END TO FIND OUT ITS |
| 3811 | | | | | | | | :STATE |
| 3812 | 016722 | 010237 | 016730 | | | MOV | R2,@#10\$ | :MOVE RHCS2 ADDRESS |
| 3813 | 016726 | 104415 | | | | WAT | | :WAIT FOR 'MXF' BIT |
| 3814 | 016730 | 000000 | | | 10\$: | .WORD | 0 | :ADDRESS OF RHCS2 |
| 3815 | 016732 | 001000 | | | | MXF | | |
| 3816 | 016734 | 004737 | 042336 | | 11\$: | JSR | PC,@#PUTREG | :SAVE REGISTERS |
| 3817 | | | | | | | | |
| 3818 | 016740 | 032737 | 040000 | 001714 | | BIT | #TRE,@#CS1 | :TRANSFER ERROR (BIT 14) RHCS1 - 'TRE' |
| 3819 | | | | | | | | :SHOULD SET DUE TO 'MXF' |

```
3820 016746 001003          BNE      13$          ;IT IS - CONTINUE
3821 016750 010137 001122    MOV      R1,@#$BDADR ;FAILING REGISTER RHCS1
3822 016754 104011          ERROR    11          ;TRANSFER ERROR (BIT 14) RHCS1 - 'TRE'
3823                                     ;SHOULD BE SET DUE TO 'MXF'
3824                                     ;LOCAL SCOPE RETURN POINT
3825
3826 016756 000660          13$:    BR      1$          ;GO BACK & TEST NEXT FUNCTION CODE
3827
3828 016760 000240          12$:    NOP
3829
```


3830
3831
3832
3833
3834
3835
3836
3837
3838
3839
3840
3841
3842
3843
3844
3845
3846
3847
3848
3849
3850
3851
3852
3853
3854
3855
3856
3857
3858
3859
3860
3861
3862
3863
3864
3865
3866
3867
3868
3869
3870
3871
3872
3873
3874
3875
3876
3877
3878
3879
3880
3881
3882
3883
3884
3885

: *TEST 42 READ IN PRESET

: ** ALL POSSIBLE REGISTERS WILL BE FILLED WITH ONES
: ** THE REGISTER CONTENTS WILL BE SAVED IN REINTO BUFFER
: ** THE READ IN PRESET COMMAND WILL BE GIVEN
: ** ALL REGISTERS WILL BE CHECKED

TST42: SCOPE

```
MOV #STACK,SP ;RESET STACK  
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER  
JSR PC,@#CLDISK ;INIT AND SET GENERAL REGISTERS  
; *FILL ALL POSSIBLE BITS WITH ONES  
MOV #177777,@RHWC ;WORD COUNT REGISTER GETS 177777  
MOV #177777,@RHBA ;BUS ADDRESS REGISTER GETS 177777  
MOV #17437,@RHDST ;DESIRED SECTOR TRACK GETS 17437  
MOV #16377,@RHOF ;OFFSET REGISTER GETS 16277  
MOV #777,@RHCA ;DESIRED CYLINDER GETS 777  
MOV #A16!A17,-(SP) ;GET BIT 9 AND 8  
BIS @#READIN,(SP)  
MOV (SP)+,@RHCS1 ;FILL READ IN PRESET IN RHCS1  
MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE
```

; *THE REGISTERS WILL BE SAVED IN REINTO BUFFER
JSR RO,@#SAVER ;SAVE
RHWC ;FROM
REINTO ;TO
17. ;NUMBER SAVED

; *GIVE READ IN PRESET COMMAND
BIS #GO,@RHCS1 ;INCLUDE GO TO READ IN PRESET

```
; *NOW SAVED REGISTERS WILL BE CHANGED TO EXPECTED VALUE  
CLR @#REINTO+12 ;CLEAR SAVED RHDST  
BIC #FMT22!HCI!ECI,@#REINTO+16 ;CLEAR FMT22,HCI,ECI IN  
; SAVED RHOF  
BIS #VV,@#REINTO+16 ;SET VV IN SAVED RHOF  
CLR @#REINTO+20 ;CLEAR SAVED RHCA
```

; *AFTER A READ IN PRESET COMMAND
; *SAVE REGISTERS AGAIN SO THAT COMPARES CAN BE DONE

```
JSR RO,@#SAVER ;SAVE  
RHWC ;FROM  
WRFROM ;TO  
17. ;NUMBER OF REGISTERS SAVED
```

; *AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
; *OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
; *SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE

```
3886 017140 113737 003205 002141      MOVB   @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS
3887
3888                                     ;*COMPARE REGISTERS BEFORE READ IN PRESET COMMAND
3889                                     ;*WITH AFTER COMMAND
3890
3891 017146 004037 043626      JSR    R0,@#COMPAR      ;COMPARE
3892 017152 003160      REINTO                    ;GOOD BUFFER
3893 017154 002114      WRFROM                    ;TEST BUFFER
3894 017156 000021      17.                      ;NUMBER OF REGISTERS
3895 017160 017166      1$                       ;RETURN FOR ERRGR
3896 017162 017166      1$                       ;SAME
3897 017164 017206      2$                       ;RETURN FOR GOOD COMPARISON
3898
3899 017166 013705 047736      1$:   MOV    @#ERWORD,R5      ;GETTING READY TO INDEX
3900 017172 060505      ADD    R5,R5              ;DOUBLE ERROR WORD
3901 017174 016537 001630 042402  MOV    RHWC-2(R5),@#REGADR ;FAILING REG. ADDRESS
3902 017202 104001      ERROR  1                  ;READ IN PRESET CAUSED IMPROPER
3903                                     ;REGISTER CHANGE
3904 017204 000207      RTS    PC                  ;RETURN FOR FURTHER COMPARISONS
3905
3906 017206      2$:                                     ;NO ERRORS
3907
```

```
3908
3909
3910
3911
3912
3913
3914
3915
3916
3917
3918 017206 000004
3919 017210 012737 000043 002032
3920
3921
3922 017216 004737 042732
3923 017222 012777 000001 162430
3924 017230 013711 002044
3925 017234 012700 001632
3926 017240 012703 001706
3927 017244 012702 000021
3928 017250 013023
3929 017252 005302
3930 017254 001375
3931 017256 013737 001662 017276
3932 017264 010137 017304
3933 017270 052711 000001
3934 017274 104415
3935 017276 000000
3936 017300 000200
3937 017302 104415
3938 017304 000000
3939
3940 017306 000200
3941
3942
3943
3944
3945 017310 004037 043424
3946 017314 001632
3947 017316 002114
3948 017320 000021
3949
3950
3951
3952
3953 017322 113737 001733 002141
3954
3955
3956
3957
3958
3959 017330 004037 043626
3960 017334 001706
3961 017336 002114
3962 017340 000021
3963 017342 017350

;*****
;*TEST 43 NO OPERATION FUNCTION TEST
;** ALL POSSIBLE REGISTERS ARE CLEARED THEN A 'NOP'=0
;** IS GIVEN NO CHANGE SHOULD HAPPEN
;** ALL POSSIBLE REGISTERS ARE FILLED WITH ONES THEN A 'NOP'
;** IS GIVEN NO CHANGE SHOULD HAPPEN
;*****
1ST43: SCOPE
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER

;*START WITH CLR IN RHCS2 (BIT5)
JSR PC,@#CLDISK ;CLEAR ALL POSSIBLE BITS
MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE
MOV @#NOPERA,@R1 ;PUT NOP OPERATION=0 IN RHCS1
MOV #RHWC,R0 ;STARTING ADDRESS OF REG
MOV #WC,R3 ;STARTING ADDRESS OF WHERE SAVED
MOV #RHEC2-RHWC+2/2,R2 ;NUMBER OF REGISTERS
1$: MOV @ (R0)+,(R3)+ ;SAVE HARDWARE REG
DEC R2 ;COUNT
BNE 1$ ;BRANCH IF NOT COMPLETE
MOV @#RHDS1,@#2$ ;GET ADDRESS OF DRIVE STATUS
MOV R1,@#3$ ;GET ADDRESS OF RHCS1
BIS #GO,@R1 ;GO TO RHCS1
WAT ;WAIT FOR DRY IN RHDS1
2$: .WORD 0 ;ADDRESS OF DRIVE STATUS RHDS1
DRY ;DRY WILL BE WAITED ON
WAT ;WAIT FOR RDY IN RHCS1
3$: .WORD 0 ;ADDRESS OF RHCS1 PUT HERE BY AN
;EARLIER MOV
RDY ;RDY WILL BE WAITED ON

;*AFTER A NO OP COMMAND
;*SAVE REGISTERS AGAIN SO THAT COMPARES CAN BE DONE
JSR R0,@#SAVER ;SAVE
RHWC ;FROM
WRFROM ;TO
17. ;NUMBER OF REGISTERS SAVED

;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
MOVB @#AS+1,@#WRFROM+25;SAVE UPPER RHAS

;*COMPARE REGISTERS BEFORE NO OP COMMAND
;*WITH AFTER COMMAND
JSR R0,@#COMPAR ;COMPARE
WC ;GOOD BUFFER
WRFROM ;TEST BUFFER
17. ;NUMBER OF REGISTERS
4$ ;RETURN FOR ERROR
```

```

CZRJGDO,RP04/5/6 DSKLS CTRLR1 MACY11 30A(1052) 10-SEP-79 11:11 PAGE 91 M 7
CZRJGD.P11 10-SEP-79 11:00 T43 NO OPERATION FUNCTION TEST SEQ 0090

3964 017344 017350 4$ :SAME
3965 017346 017370 5$ :RETURN FOR GOOD COMPARISON
3966
3967 017350 013705 047736 4$: MOV @#ERWORD,R5 ;GETTING READY TO INDEX
3968 017354 060505 ADD R5,R5 ;DOUBLE ERROR WORD
3969 017356 016537 001630 042402 MOV RHWC-2(R5),@#REGADR ;FAILING REG. ADDRESS
3970 017364 104001 ERROR 1 ;NO OP COMMAND CAUSED IMPROPER
3971 ;REGISTER CHANGE
3972 017366 000207 RTS PC ;RETURN FOR FURTHER COMPARISONS
3973
3974 017370 5$: ;NO ERRORS
3975
3976
3977
3978 017370 012737 017376 001110 MOV #14$,@#SLPERR ;SET SCOPE LOOP TO 14$
3979 017376 004737 042732 14$: JSR PC,@#CLDISK ;INIT LAST ALL ZERO TEST
3980 017402 012777 000001 162250 MOV #DMD,@#RHMR ;SET DIAGNOSTIC MODE
3981
3982
3983 ;*NOW START WITH ALL ONES IN ALL POSSIBLE REGISTERS
3984
3985 017410 012700 001632 MOV #RHWC,R0 ;ADDRESS OF FIRST REGISTER
3986 017414 012705 000021 MOV #RHEC2-RHWC+2/2,R5 ;NO. OF REGISTERS
3987 017420 012730 177676 6$: MOV #177676,@(R0)+ ;FILL WITH ALL ONES
3988 017424 013777 001774 162204 MOV @#UNIT,@#RHCS2 ;REINSTATE UNIT NUMBER UNDER TEST
3989 ;KEEP INTERRUPT DISABLED
3990 017432 005305 DEC R5 ;COUNT
3991 017434 001371 BNE 6$ ;BRANCH IF INCOMPLETE
3992 017436 013711 002044 MOV @#NOPERA,@R1 ;PUT NOP OPERATION =0 IN RHCS1
3993 017442 012700 001632 MOV #RHWC,R0 ;STARTING ADDRESS OF REG
3994 017446 012703 001706 MOV #WC,R3 ;STARTING ADDRESS OF WHERE SAVED
3995 017452 012702 000021 MOV #RHEC2-RHWC+2/2,R2 ;NUMBER OF REGISTERS
3996 017456 013023 7$: MOV @(R0)+,(R3)+ ;SAVE HARDWARE REG
3997 017460 005302 DEC R2 ;COUNT
3998 017462 001375 BNE 7$ ;BRANCH IF NOT COMPLETE
3999 017464 013737 001662 017504 MOV @#RHDS1,@#10$ ;GET ADDRESS OF DRIVE STATUS
4000 017472 010137 017512 MOV R1,@#11$ ;GET ADDRESS OF RHCS1
4001 017476 052711 000001 BIS #GO,@R1 ;GO TO RHCS1
4002 017502 104415 WAT ;WAIT FOR DRY IN RHDS1
4003 017504 000000 10$: .WORD 0 ;ADDRESS OF DRIVE STATUS RHDS1
4004 017506 000200 DRY ;DRY WILL BE WAITED ON
4005 017510 104415 WAT ;WAIT FOR RDY IN RHCS1
4006 017512 000000 11$: .WORD 0 ;ADDRESS OF RHCS1 PUT HERE BY AN
4007 ;EARLIER MOV.
4008 017514 000200 RDY ;RDY WILL BE WAITED ON
4009
4010 ;*AFTER A NO OP COMMAND
4011 ;*SAVE REGISTERS AGAIN SO THAT COMPARES CAN BE DONE
4012
4013 017516 004037 043424 JSR R0,@#SAVER ;SAVE
4014 017522 001632 RHWC ;FROM
4015 017524 002114 WRFROM ;TO
4016 017526 000021 17. ;NUMBER OF REGISTERS SAVED
4017
4018 ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
4019 ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS

```

```

4020
4021 017530 113737 001733 002141 ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
4022 MOVB @#AS+1,@#WRFROM+25;SAVE UPPER RHAS
4023
4024 ;*COMPARE REGISTERS BEFORE NO OP COMMAND
4025 ;*WITH AFTER COMMAND
4026
4027 017536 004037 043626 JSR RO,@#COMPAR ;COMPARE
4028 017542 001706 WC ;GOOD BUFFER
4029 017544 002114 WRFROM ;TEST BUFFER
4030 017546 000021 17. ;NUMBER OF REGISTERS
4031 017550 017556 12$ ;RETURN FOR ERROR
4032 017552 017556 12$ ;SAME
4033 017554 017576 13$ ;RETURN FOR GOOD COMPARISON
4034
4035 017556 013705 047736 12$: MOV @#ERWORD,R5 ;GETTING READY TO INDEX
4036 017562 060505 ADD R5,R5 ;DOUBLE ERROR WORD
4037 017564 016537 001630 042402 MOV RHWC-2(R5),@#REGADR ;FAILING REG. ADDRESS
4038 017572 104001 ERROR 1 ;NO OP COMMAND CAUSED IMPROPER
4039 ;REGISTER CHANGE
4040 017574 000207 RTS PC ;RETURN FOR FURTHER COMPARISONS
4041
4042 017576 13$: ;NO ERRORS
  
```

4043
4044
4045
4046
4047
4048
4049
4050
4051
4052
4053
4054
4055
4056
4057
4058
4059
4060
4061
4062
4063
4064
4065
4066
4067
4068
4069
4070
4071
4072
4073
4074
4075
4076
4077
4078
4079
4080
4081
4082
4083
4084
4085
4086
4087
4088
4089
4090
4091
4092
4093
4094
4095
4096
4097
4098

```
*****  
: *TEST 44 DRIVE CLEAR  
: ** ALL WRITE BITS OF ALL REGISTERS EXCEPT RHDB ARE FILLED WITH  
: ** ONES EXCEPT FOR BIT #0 AND BIT #6 WHICH ARE "GO" AND  
: ** "ENABLE INTERRUPT" BITS  
: ** THEN A DRIVE CLEAR IS PERFORMED  
: ** THEN ALL REGISTERS EXCEPT RHDB ARE CHECKED  
*****
```

```
TST44: SCOPE  
MOV #STACK,SP ;RESET STACK  
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER  
JSR PC,@#CLDISK ;SET REGISTERS AND CLEAR  
  
: *FILL ALL POSSIBLE BITS WITH ONES  
  
MOV #177777,@RHDB ;BUS ADDRESS REGISTER GETS 177777  
MOV #177777,@RHWC ;WORD COUNT REGISTER GETS 177777  
MOV #177777,@RHBA ;BUS ADDRESS REGISTER GETS 177777  
BIS #157010,@RHCS2 ;CONTROL AND STATUS 2 GETS 157010  
MOV #1476,@RHCS1 ;CONTROL AND STATUS REGISTER/GETS 1476  
MOV #177777,@RHER1 ;ERROR REGISTER1 GETS 177777  
MOV #17437,@RHDS1 ;DESIRED SECTOR TRACK  
MOV #177777,@RHER2 ;ERROR REGISTER 2  
MOV #16277,@RHOF ;OFFSET REGISTER  
MOV #177777,@RHCA ;DESIRED CYLINDER  
MOV #177777,@RHER3 ;ERROR REGISTER 3  
MOV #DMD,@RHMR ;MAINTENANCE REGISTER  
MOV #177777,@RHMR ;MAINTENANCE REGISTER
```

```
: *THIS SETS BITS FOR ALL PRESENT DRIVES  
  
MOV @#TOTALAT,RO ;GET DRIVE PRESENT  
CLR @R2 ;CLEAR RHCS2 AND CARRY BIT  
MOV #8.,R5 ;COUNTER  
30$: ROR RO ;GET BIT INTO CARRY  
BCC 31$ ;BRANCH IF NO UNIT ON THIS BIT  
MOV #-1,@R4 ;MOVE INTO ERROR REGISTER TO SET ATA  
31$: INC @R2 ;INCREMENT RHCS2 - UNIT NO.  
DEC R5 ;COUNT  
BEQ 27$ ;BRANCH IF 8 DONE  
BR 30$ ;CONTINUE THIS ROUTINE  
27$: MOV @#UNIT,-(SP) ;REINSTATE SET BITS  
BIS #157010,(SP)  
MOV (SP)+,@R2 ;
```

```
MOV #DMD,@RHMR ;SET DMD  
MOV @#DCLEAR,@R1 ;DRIVE CLEAR = 10 INTO RHCS1  
BIS #GO,@R1 ;GO  
MOV #RHDB, RO ;RO CONTAINS ADDR. OF ADDR. OF REG.
```

4099
4100 ;*DATA BUFFER REGISTER
4101
4102
4103 020022 012737 177777 001124 28\$: MOV #177777,@#\$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
4104 020030 013037 001126 MOV @\$(RO)+,@#\$BDDAT ;TEST DATA
4105 020034 022737 177777 001126 CMP #177777,@#\$BDDAT ;COMPARE DATA
4106 020042 001402 BEQ 3\$;BRANCH IF GOOD
4107 020044 004737 020702 JSR PC,@#ERCLFC ;JUMP TO ERROR FOR CLR (BIT 5)
4108 ;IN RHCS2
4109

4110 ;*WORD COUNT REGISTER
4111
4112
4113
4114 020050 012737 177777 001124 3\$: MOV #177777,@#\$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
4115 020056 013037 001126 MOV @\$(RO)+,@#\$BDDAT ;TEST DATA
4116 020062 022737 177777 001126 CMP #177777,@#\$BDDAT ;COMPARE DATA
4117 020070 001402 BEQ 4\$;BRANCH IF GOOD
4118 020072 004737 020702 JSR PC,@#ERCLFC ;JUMP TO ERROR FOR CLR (BIT 5)
4119 ;IN RHCS2
4120

4121 ;*BUS ADDRESS REGISTER
4122
4123
4124
4125 020076 012737 177776 001124 4\$: MOV #177776,@#\$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
4126 020104 013037 001126 MOV @\$(RO)+,@#\$BDDAT ;TEST DATA
4127 020110 022737 177776 001126 CMP #177776,@#\$BDDAT ;COMPARE DATA
4128 020116 001402 BEQ 5\$;BRANCH IF GOOD
4129 020120 004737 020702 JSR PC,@#ERCLFC ;JUMP TO ERROR FOR CLR (BIT 5)
4130 ;IN RHCS2
4131

4132 ;*CONTROL AND STATUS 2 REGISTER
4133
4134
4135
4136
4137 020124 012746 000110 5\$: MOV #110,-(SP) ;INCLUDE IR
4138 020130 053716 001774 BIS @#UNIT,(SP) ;SET UNIT NO.
4139 020134 012637 001124 MOV (SP)+,@#\$GDDAT ;GOOD DATA FOR TYPE OUT
4140 020140 013037 001126 MOV @\$(RO)+,@#\$BDDAT ;TEST DATA
4141 020144 023737 001124 001126 CMP @#\$GDDAT,@#\$BDDAT ;COMPARE DATA
4142 020152 001402 BEQ 6\$;BRANCH IF GOOD
4143 020154 004737 020702 JSR PC,@#ERCLFC ;JUMP TO ERROR FOR CLR (BIT 5)
4144 ;IN RHCS2
4145

4146 ;*CONTROL AND STATUS 1 REGISTER
4147
4148
4149
4150 020160 005737 002000 6\$: TST @#NUNIT ;ARE THERE MORE THAN ONE UNIT
4151 020164 001404 BEQ 32\$;BRANCH IF ONLY ONE UNIT
4152 020166 012737 104210 001124 MOV #104210,@#\$GDDAT ;GOOD DATA
4153 020174 000403 BR 33\$
4154 020176 012737 004210 001124 32\$: MOV #4210,@#\$GDDAT ;GOOD DATA

```

4155 020204 013037 001126      33$:  MOV    @ (R0)+, @ $BDDAT ; TEST DATA
4156
4157 020210 023737 001124 001126      CMP    @ $GDDAT, @ $BDDAT ; COMPARE DATA
4158 020216 001402                BEQ    7$                ; BRANCH IF GOOD
4159 020220 004737 020702                JSR    PC, @ $ERCLFC    ; JUMP TO ERROR FOR CLR BIT 5
4160                                ; IN RHCS2
4161
4162                                ; *ERROR 1 REGISTER
4163
4164
4165 020224 012737 000000 001124 7$:  MOV    #0, @ $GDDAT    ; GOOD DATA FOR ERROR TYPEOUT
4166 020232 013037 001126                MOV    @ (R0)+, @ $BDDAT ; TEST DATA
4167 020236 022737 000000 001126      CMP    #0, @ $BDDAT    ; COMPARE DATA
4168 020244 001402                BEQ    10$               ; BRANCH IF GOOD
4169 020246 004737 020702                JSR    PC, @ $ERCLFC    ; JUMP TO ERROR FOR CLR (BIT 5)
4170                                ; IN RHCS2
4171
4172                                ; *DESIRED SECTOR/TRACK REGISTER
4173
4174
4175 020252 012737 017437 001124 10$:  MOV    #17437, @ $GDDAT ; GOOD DATA FOR ERROR TYPEOUT
4176 020260 013037 001126                MOV    @ (R0)+, @ $BDDAT ; TEST DATA
4177 020264 022737 017437 001126      CMP    #17437, @ $BDDAT ; COMPARE DATA
4178 020272 001402                BEQ    11$               ; BRANCH IF GOOD
4179 020274 004737 020702                JSR    PC, @ $ERCLFC    ; JUMP TO ERROR FOR CLR (BIT 5)
4180                                ; IN RHCS2
4181
4182                                ; *ERROR 2 REGISTER
4183
4184
4185 020300 012737 000000 001124 11$:  MOV    #0, @ $GDDAT    ; GOOD DATA FOR ERROR TYPEOUT
4186 020306 013037 001126                MOV    @ (R0)+, @ $BDDAT ; TEST DATA
4187 020312 022737 000000 001126      CMP    #0, @ $BDDAT    ; COMPARE DATA
4188 020320 001402                BEQ    12$               ; BRANCH IF GOOD
4189 020322 004737 020702                JSR    PC, @ $ERCLFC    ; JUMP TO ERROR FOR CLR (BIT 5)
4190                                ; IN RHCS2
4191
4192                                ; *OFFSET REGISTER
4193
4194
4195 020326 012737 116000 001124 12$:  MOV    #116000, @ $GDDAT ; GOOD DATA FOR ERROR TYPEOUT
4196 020334 013037 001126                MOV    @ (R0)+, @ $BDDAT ; TEST DATA
4197 020340 022737 116000 001126      CMP    #116000, @ $BDDAT ; COMPARE DATA
4198 020346 001402                BEQ    13$               ; BRANCH IF GOOD
4199 020350 004737 020702                JSR    PC, @ $ERCLFC    ; JUMP TO ERROR FOR CLR (BIT 5)
4200                                ; IN RHCS2
4201
4202                                ; *DESIRED CYLINDER ADDRESS REGISTER
4203
4204
4205 020354 012737 001777 001124 13$:  MOV    #1777, @ $GDDAT ; GOOD DATA FOR ERROR TYPEOUT
4206 020362 013037 001126                MOV    @ (R0)+, @ $BDDAT ; TEST DATA
4207 020366 022737 001777 001126      CMP    #1777, @ $BDDAT ; COMPARE DATA
4208 020374 001402                BEQ    14$               ; BRANCH IF GOOD
4209 020376 004737 020702                JSR    PC, @ $ERCLFC    ; JUMP TO ERROR FOR CLR (BIT 5)
4210                                ; IN RHCS2
    
```



```

4211
4212
4213
4214
4215 020402 012737 000000 001124 14$: MOV #0,@#$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
4216 020410 013037 001126 001124 :MOV @ (R0)+,@#$BDDAT ;TEST DATA
4217 020414 022737 000000 001126 :CMP #0,@#$BDDAT ;COMPARE DATA
4218 020422 001402 :BEQ 15$ ;BRANCH IF GOOD
4219 020424 004737 020702 :JSR PC,@#ERCLFC ;JUMP TO ERROR FOR CLR (BIT 5)
4220 : ;IN RHCS2
4221
4222
4223
4224 020430 013737 002020 001124 15$: MOV @#TOTALAT,@#$GDDAT;SET ALL BITS OF DRIVE PRESENT IN RHAS
4225 020436 043737 002016 001124 :BIC @#ATTENT,@#$GDDAT ;CLEAR ONLY WORKING DRIVE BIT
4226 020444 013037 001126 001126 :MOV @ (R0)+,@#$BDDAT ;GET RHAS
4227 020450 123737 001124 001126 :CMPB @#$GDDAT,@#$BDDAT ;COMPARE DATA
4228 020456 001402 :BEQ 16$ ;BRANCH IF GOOD
4229 020460 004737 020702 :JSR PC,@#ERCLFC ;JUMP TO ERROR FOR CLR (BIT 5) IN RHCS2
4230
4231
4232
4233
4234 020464 012737 000400 001124 16$: MOV #400,@#$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
4235 020472 013037 001126 001126 :MOV @ (R0)+,@#$BDDAT ;TEST DATA
4236 020476 022737 000400 001126 :CMP #400,@#$BDDAT ;COMPARE DATA
4237 020504 001402 :BEQ 17$ ;BRANCH IF GOOD
4238 020506 004737 020702 :JSR PC,@#ERCLFC ;JUMP TO ERROR FOR CLR (BIT 5)
4239 : ;IN RHCS2
4240
4241
4242
4243
4244 020512 012737 000700 001124 17$: MOV #700,@#$GDDAT ;GOOD DATA FOR PRINTOUT
4245 020520 013046 :MOV @ (R0)+,-(SP) ;GET RHDS1
4246 020522 011637 001126 001126 :MOV (SP),@#$BDDAT ;TEST DATA
4247 020526 042716 001000 :BIC #PROG,(SP) ;CLEAR PROG BIT
4248 020532 022726 000700 :CMP #700,(SP)+ ;COMPARE DATA
4249 020536 001402 :BEQ 20$ ;BRANCH IF GOOD
4250 020540 004737 020702 :JSR PC,@#ERCLFC ;JUMP TO ERROR FOR DRIVE CLEAR
4251
4252
4253
4254 020544 013737 002010 001124 20$: MOV @#SAVDT,@#$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
4255 020552 013037 001126 001126 :MOV @ (R0)+,@#$BDDAT ;TEST DATA
4256 020556 023737 002010 001126 :CMP @#SAVDT,@#$BDDAT ;COMPARE DATA
4257 020564 001402 :BEQ 21$ ;BRANCH IF GOOD
4258 020566 004737 020702 :JSR PC,@#ERCLFC ;JUMP TO ERROR FOR CLR (BIT 5)
4259 : ;IN RHCS2
4260
4261
4262
4263
4264
4265 020572 013737 002012 001124 21$: MOV @#SAVSN,@#$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
4266 020600 013037 001126 001126 :MOV @ (R0)+,@#$BDDAT ;TEST DATA

```

```

4267 020604 023737 002012 001126      CMP    @#SAVSN,@#SBDDAT      ;COMPARE DATA
4268 020612 001402                BEQ    22$                    ;BRANCH IF GOOD
4269 020614 004737 020702                JSR    PC,@#ERCLFC           ;JUMP TO ERROR FOR CLR (BIT 5)
4270                                ;IN RHCS2
4271
4272                                ;*ECC1 POSITION
4273
4274
4275 020620 012737 000000 001124 22$:    MOV    #0,@#SGDDAT           ;GOOD DATA FOR ERROR TYPEOUT
4276 020626 013037 001126                MOV    @ (R0)+,@#SBDDAT      ;TEST DATA
4277 020632 022737 000000 001126                CMP    #0,@#SBDDAT           ;COMPARE DATA
4278 020640 001402                BEQ    23$                    ;BRANCH IF GOOD
4279 020642 004737 020702                JSR    PC,@#ERCLFC           ;JUMP TO ERROR FOR CLR (BIT 5)
4280                                ;IN RHCS2
4281
4282                                ;*ECC2 PATTERN
4283
4284
4285 020646 012737 000000 001124 23$:    MOV    #0,@#SGDDAT           ;GOOD DATA FOR ERROR TYPEOUT
4286 020654 013037 001126                MOV    @ (R0)+,@#SBDDAT      ;TEST DATA
4287 020660 022737 000000 001126                CMP    #0,@#SBDDAT           ;COMPARE DATA
4288 020666 001402                BEQ    24$                    ;BRANCH IF GOOD
4289 020670 004737 020702                JSR    PC,@#ERCLFC           ;JUMP TO ERROR FOR CLR (BIT 5)
4290                                ;IN RHCS2
4291
4292                                ;*LOOK-AHEAD REGISTER
4293
4294
4295 020674 005720                24$:    TST    (R0)+                 ;AS THE LOOK-AHEAD REG. CANNOT BE PREDICTED
4296                                ;IT IS NOT CHECKED AFTER AN INIT
4297
4298                                ;*CURRENT CYLINDER ADDRESS REGISTER
4299
4300 020676 005720                25$:    TST    (R0)+                 ;AS THE CURRENT CYL REG. CANNOT BE PREDICTED
4301                                ;AFTER AN INIT IT IS NOT CHECKED
4302
4303 020700                26$:
4304 020700 000413                BR     TST45                  ;BRANCH OVER JSR
4305
4306 020702 014037 042402      ERCLFC: MOV    -(R0), @#REGADR ;FAILING REGISTER ADDRESS
4307 020706 104001                ERROR 1                       ;CLR FUNCTION = 10 IN RHCS1 DID
4308                                ;NOT CLEAR APPROPRIATE BITS
4309                                ;OR CLEARED EXTRA BITS
4310 020710 005720                TST    (R0)+                 ;UNDO -(R0) FOR ERROR
4311 020712 000207                RTS    PC                     ;RETURN TO ABOVE PROGRAM
4312                                ;OR CLEARED EXTRA BITS
4313 020714 005720                TST    (R0)+                 ;UNDO -(R0) FOR BAD DATA
4314 020716 004737 042766                JSR    PC,@#CHECKT           ;CHECK THAT DVA,RDY,DPR,DRY = 1
4315 020722 104401 062562                TYPE  ,CPHALT                ;AND THAT NO OTHERS = 1. CANNOT CON-
4316                                ;TINUE TESTING IF BOTH AREN'T TRUE
4317 020726 000000                HALT                          ;STOP THE TEST
4318

```

4319
4320
4321
4322
4323
4324
4325
4326
4327
4328
4329
4330
4331
4332
4333
4334
4335
4336
4337
4338
4339
4340
4341
4342
4343
4344
4345
4346
4347
4348
4349
4350
4351
4352
4353
4354
4355
4356
4357
4358
4359
4360
4361
4362
4363
4364
4365
4366
4367
4368
4369
4370
4371
4372
4373
4374

: *TEST 45 SEEK COMMAND TEST

: ** THE SEEK COMMAND WILL BE LOADED INTO RHCS1 WITH GO
: ** THEN ALL REGISTERS WILL BE CHECKED
: ** RH CLEAR WILL BE GIVEN
: ** THEN ALL REGISTERS WILL BE CHECKED

TST45: SCOPE

MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
JSR PC,@#CLDISK ;INIT AND SET UP GENERAL REG. CORRES.
;AND UNIT NUMBER
MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE BIT
;THIS ENABLES COMMANDS WITHOUT MOL
;AND HOLDS RHLA FROM MOVING
CLR @RHDS1 ;MAKE DESIRED SECTOR TRACK LEGAL
MOV @#SEECOM,@RHCS1 ;LOAD SEEK COMMAND INTO CONTROLLER
MOV @RHCC,-(SP) ;GET CURRENT CYLINDER

: *FOLLOWING ARE TWO BLOCKS OF CODE TO LOAD RHCA WITH THE PROPER
: *ADDRESS DEPENDING UPON WHETHER THE DRIVE IS AN RP06 OR RP04

TST @#RP06 ;MOVE DRIVE TYPE FLAG TO ITSELF TO TEST
BEQ 11\$;TREAT THE DRIVE AS AN RP04

9\$: ;TREAT THE DRIVE AS AN RP06
CMP #814.,(SP)+ ;IS CURRENT CYLINDER SAME AS 814. ?
BEQ 9\$;BRANCH IF YES TO MAKE RHCA = 813.
MOV #814.,@#STMP5 ;GET READY TO MAKE RHCA = 814.
BR 10\$;FILL RHCA
MOV #813.,@#STMP5 ;GET READY TO MAKE RHCA = 813.
10\$: MOV @#STMP5,@RHCA ;MAKE DESIRED CYLINDER 814., OR 813.
BR 14\$;SAVE REGISTERS

11\$: ;TREAT THE DRIVE AS AN RP04
CMP #410.,(SP)+ ;IS CURRENT CYLINDER SAME AS 410. ?
BEQ 12\$;BRANCH IF YES TO MAKE RHCA = 409.
MOV #410.,@#STMP5 ;GET READY TO MAKE RHCA = 410.
BR 13\$;FILL RHCA
MOV #409.,@#STMP5 ;GET READY TO MAKE RHCA = 409.
13\$: MOV @#STMP5,@RHCA ;MAKE DESIRED CYLINDER 410., OR 409.

: *SAVE REGISTERS FOR COMPARISON AFTER GO

14\$: JSR R0,@#SAVER ;SAVE
RHW ;FROM
REINTO ;TO
19. ;NUMBER OF REGISTERS SAVED

: *GIVE GO TO COMMAND

```

4375 021102 052777 000001 160530      BIS      #GO,@RHCS1      ;GO TO COMMAND
4376
4377      ;*CHANGE SAVED REGISTERS TO EXPECTED VALUES
4378
4379 021110 052737 000001 003166      BIS      #GO,@#REINT0+6 ;SAVED RHCS1
4380 021116 052737 020000 003210      BIS      #PIP,@#REINT0+30 ;SAVED RHDS1
4381 021124 042737 000200 003210      BIC      #DRY,@#REINT0+30 ;SAVED RHDS1
4382
4383
4384      ;*AFTER GO HAS BEEN GIVEN FOR SEEK COMMAND
4385      ;*SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN
4386      ;*BE DONE
4387
4388 021132 004037 043424      JSR      R0,@#SAVER      ;SAVE
4389 021136 001632      RHC      ;FROM
4390 021140 002114      WRFROM   ;TO
4391 021142 000023      19.      ;NUMBER OF REGISTERS SAVED
4392
4393      ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
4394      ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
4395      ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
4396
4397 021144 113737 003205 002141      MOV      @#REINT0+25,@#WRFROM+25;SAVE UPPER RHAS
4398
4399      ;*COMPARE REGISTERS BEFORE SEEK COMMAND
4400      ;*WITH CONTENTS AFTER GO IS ISSUED
4401
4402 021152 004037 043626      JSR      R0,@#COMPAR     ;COMPARE
4403 021156 003160      REINT0   ;GOOD BUFFER
4404 021160 002114      WRFROM   ;TEST BUFFER
4405 021162 000023      19.      ;NUMBER
4406 021164 021172      1$      ;RETURN FOR ERROR
4407 021166 021172      1$      ;SAME
4408 021170 021212      2$      ;RETURN FOR GOOD COMPARISON
4409
4410 021172 013705 047736      1$:      MOV      @#ERWORD,R5     ;GETTING READY TO INDEX
4411 021176 060505      ADD      R5,R5           ;DOUBLE ERROR WORD
4412 021200 016537 001630 042402      MOV      RHC-2(R5),@#REGADR ;FAILING REGISTER ADDRESS
4413
4414 021206 104001      ERROR    1              ;IMPROPER REGISTER CHANGE
4415      ;AFTER SEEK COMMAND
4416      ;WITH GO IS GIVEN
4417 021210 000207      RTS      PC              ;RETURN TO COMPARISON
4418
4419
4420      ;*NOW GIVE INIT AND GET GO AND PIP DOWN
4421
4422 021212 052712 000040      2$:      BIS      #CLR,@R2        ;RH INITILIZE
4423 021216 013712 001774      MOV      @#UNIT,@R2      ;REINSTATE UNIT NUMBER
4424 021222 012777 000001 160430      MOV      #DMD,@RHMR     ;SET DIAGNOSTIC MODE BIT
4425      ;THIS ENABLES COMMANDS WITHOUT MOL
4426      ;AND HOLDS RHLA FROM MOVING
4427
4428      ;*CHANGE REGISTERS TO EXPECTED VALUE
4429
4430 021230 042737 000001 003166      BIC      #GO,@#REINT0+6 ;SAVED RHCS1
  
```

```

4431 021236 042737 020000 003210      BIC    #PIP,@#REINTO+30 ;SAVED RHDS1
4432 021244 052737 000200 003210      BIS    #DRY,@#REINTO+30 ;SAVED RHDS1
4433 021252 017737 160416 003222      MOV    @RHLA,@#REINTO+42;SAVED RHLA
4434 021260 013737 001210 003224      MOV    @#STMP5,@#REINTO+44 ;SAVED RHCC
4435
4436
4437                                     ;*AFTER INITILIZE SAVE REGISTERS SO THAT
4438                                     ;*COMPARES CAN BE DONE
4439
4440 021266 004037 043424      JSR    RO,@#SAVER          ;SAVE
4441 021272 001632                                     ;FROM
4442 021274 002114      WRFROM                                     ;TO
4443 021276 000023      19.                                     ;NUMBER OF REGISTERS SAVED
4444
4445
4446                                     ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
4447                                     ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
4448                                     ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
4449
4450 021300 113737 003205 002141      MOVB   @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS
4451
4452
4453                                     ;*COMPARE REGISTERS AFTER INITIALIZE
4454
4455 021306 004037 043626      JSR    RO,@#COMPAR        ;COMPARE
4456 021312 003160      REINTO                                     ;GOOD BUFFER
4457 021314 002114      WRFROM                                     ;TEST BUFFER
4458 021316 000023      19.                                     ;NUMBER OF REGISTERS TO BE
4459                                     ;COMPARED
4460 021320 021326      3$                                       ;RETURN POINT FOR ERROR
4461 021322 021326      3$                                       ;SAME
4462 021324 021346      4$                                       ;RETURN POINT FOR GOOD COMPARISON
4463
4464 021326 013705 047736      3$:   MOV    @#ERWORD,R5          ;GETTING READY TO INDEX
4465 021332 060505      ADD    R5,R5                    ;DOUBLE ERROR WORD
4466 021334 016537 001630 042402      MOV    RHWC-2(R5),@#REGADR ;FAILING REGISTER ADDRESS
4467 021342 104001      ERROR 1                          ;IMPROPER REGISTER
4468                                     ;CONTENTS AFTER GIVING AN
4469                                     ;INITILIZE FOLLOWING A
4470                                     ;SEEK COMMAND
4471 021344 000207      RTS    PC                        ;RETURN TO COMPARISON
4472
4473 021346      4$:                                ;GOOD COMPARISON
4474
4475 021346 004737 042732      JSR    PC,@#CLDISK          ;INIT AND SET UP GENERAL REG.
4476                                     ;AND UNIT NUMBER
4477 021352 012777 000001 160300      MOV    #DMD,@#RHMR         ;SET DIAGNOSTIC MODE BIT
4478                                     ;THIS ENABLES COMMANDS WITHOUT MOL
4479                                     ;AND HOLDS RHLA FROM MOVING
4480
4481 021360 013777 002074 160252      MOV    @#SEECOM,@#RHCS1    ;LOAD SEEK COMMAND INTO RHC
4482 021366 005077 160260      CLR    @#RHCA              ;DESIRED CYLINDER ADDRESS
4483
4484                                     ;*SAVE REGISTERS FOR COMPARISON AFTER GO
4485 021372 004037 043424      JSR    RO,@#SAVER          ;SAVE
4486 021376 001632      RHWC                                     ;FROM

```

```

4487 021400 003160 REINTO ;TO
4488 021402 000023 19. ;NUMBER OF REGISTERS SAVED
4489
4490 ;*GIVE GO TO SEEK COMMAND
4491 021404 052777 000001 160226 BIS #GO,@RHCS1 ;GO TO SEEK COMMAND
4492
4493 ;*CHANGE SAVED REGISTERS TO EXPECTED VALUES
4494
4495 021412 052737 000001 003166 BIS #GO,@REINTO+6 ;SAVED RHCS1
4496 021420 052737 020000 003210 BIS #PIP,@REINTO+30 ;SAVED RHDS1
4497 021426 042737 000200 003210 BIC #DRY,@REINTO+30 ;SAVED RHDS1
4498
4499
4500 ;*AFTER GO HAS BEEN GIVEN TO SEEK COMMAND
4501 ;*SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN
4502 ;*BE DONE
4503 021434 004037 043424 JSR RO,@SAVER ;SAVE
4504 021440 001632 RHCW ;FROM
4505 021442 002114 WRFROM ;TO
4506 021444 000023 19. ;NUMBER OF REGISTERS SAVED
4507
4508
4509 ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
4510 ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
4511 ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
4512 021446 113737 003205 002141 MOVB @REINTO+25,@WRFROM+25;SAVE UPPER RHAS
4513
4514
4515 ;*COMPARE REGISTERS BEFORE COMMAND
4516 ;*WITH CONTENTS AFTER GO IS GIVEN
4517
4518 021454 004037 043626 JSR RO,@COMPAR ;COMPARE
4519 021460 003160 REINTO ;GOOD BUFFER
4520 021462 002114 WRFROM ;TEST BUFFER
4521 021464 000023 19. ;NUMBER
4522 021466 021474 5$ ;RETURN FOR ERROR
4523 021470 021474 5$ ;SAME
4524 021472 021514 6$ ;RETURN FOR GOOD COMPARISON
4525 021474 013705 047736 5$: MOV @#ERWORD,R5 ;GETTING READY TO INDEX
4526 021500 060505 ADD R5,R5 ;DOUBLG ERROR WORD
4527 021502 016537 001630 042402 MOV RHCW-2(R5),@#REGADR ;FAILING REGISTER ADDRESS
4528 021510 104001 ERROR 1 ;IMPROPER REGISTER CHANGE
4529 ;AFTER COMMAND
4530 ;WITH GO IS GIVEN
4531 021512 000207 RTS PC ;RETURN TO COMPARISON
4532
4533 ;*NOW GIVE INIT AND GET GO AND PIP DOWN
4534
4535 021514 052712 000040 6$: BIS #CLR,@R2 ;RH INITILIZE
4536 021520 013712 001774 MOV @#UNIT,@R2 ;REINSTATE UNIT NUMBER
4537 021524 012777 000001 160126 MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE BIT
4538 ;THIS ENABLES COMMANDS WITHOUT MOL
4539 ;AND HOLDS RHLA FROM MOVING
4540
4541 ;*CHANGE REGISTERS TO EXPECTED VALUE
4542

```

```

4543 021532 042737 000001 003166      BIC    #GO,@#REINTO+6 ;SAVED RHCS1
4544 021540 042737 020000 003210      BIC    #PIP,@#REINTO+30 ;SAVED RHDS1
4545 021546 052737 000200 003210      BIS    #DRY,@#REINTO+30 ;SAVED RHDS1
4546 021554 017737 160114 003222      MOV    @RHLA,@#REINTO+42;SAVED RHLA
4547 021562 005037 003224      CLR    @#REINTO+44 ;SAVED RHCC
4548
4549
4550
4551      ;*AFTER INITIALIZE SAVE REGISTERS SO THAT
4552      ;*COMPARES CAN BE DONE
4553 021566 004037 043424      JSR    RO,@#SAVER ;SAVE
4554 021572 001632      RHCW ;FROM
4555 021574 002114      WRFROM ;TO
4556 021576 000023      19. ;NUMBER OF REGISTERS SAVED
4557
4558      ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
4559      ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
4560      ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
4561 021600 113737 003205 002141      MOVB  @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS
4562
4563      ;*COMPARE REGISTERS AFTER INITIALIZE
4564
4565
4566 021606 004037 043626      JSR    RO,@#COMPAR ;COMPARE
4567 021612 003160      REINTO ;GOOD BUFFER
4568 021614 002114      WRFROM ;TEST BUFFER
4569 021616 000023      19. ;NUMBER OF REGISTERS TO BE
4570      ;COMPARED
4571 021620 021626      7$ ;RETURN POINT FOR ERROR
4572 021622 021626      7$ ;SAME
4573 021624 021646      8$ ;RETURN POINT FOR GOOD COMPARISON
4574
4575 021626 013705 047736      7$: MOV    @#ERWORD,R5 ;GETTING READY TO INDEX
4576 021632 060505      ADD    R5,R5 ;DOUBLE ERROR WORD
4577 021634 016537 001630 042402      MOV    RHCW-2(R5),@#REGADR ;FAILING REGISTER ADDRESS
4578 021642 104001      ERROR 1 ;# CONTENTS AFTER GIVING AN
4579      ; COMMAND
4580 021644 000207      RTS    PC ;RETURN TO COMPARISON
4581 021646      8$: ;GOOD COMPARISON
  
```

```

4582
4583
4584
4585
4586
4587
4588
4589
4590
4591
4592 021646 000004
4593 021650 012706 001000
4594 021654 012737 000046 002032
4595 021662 004737 042732
4596
4597 021666 012777 000001 157764
4598
4599
4600
4601 021674 013777 002046 157736
4602
4603
4604
4605 021702 004037 043424
4606 021706 001632
4607 021710 003160
4608 021712 000023
4609
4610
4611 021714 052777 000001 157716
4612
4613
4614 021722 052737 000001 003166
4615 021730 052737 020000 003210
4616 021736 042737 000200 003210
4617
4618 021744 005737 001100
4619 021750 001053
4620 021752 032777 020000 157160
4621 021760 001047
4622
4623 021762 104401 021770
4624 021766 000441
4625
4626 022072
4627
4628 022072 013746 001774
4629 022076 104405
4630
4631
4632
4633
4634
4635 022100 004037 043424
4636 022104 001632
4637 022106 002114

```

```

*****
: *TEST 46          UNLOAD COMMAND TEST
*****
: **      THE UNLOAD COMMAND WILL BE LOADED INTO RHCS1 WITH GO
: **      THEN ALL REGISTERS WILL BE CHECKED
: **      RH CLEAR WILL BE GIVEN
*****
TST46:  SCOPE
        MOV      #STACK,SP          ;RESET STACK
        MOV      #TTNO,@#TSTNM     ;THIS SAVES TEST NUMBER
        JSR      PC,@#CLDISK       ;INIT AND SET UP GENERAL REG.
                                         ;AND UNIT NUMBER
        MOV      #DMD,@RHMR        ;SET DIAGNOSTIC MODE BIT
                                         ;THIS ENABLES COMMANDS WITHOUT MOL
                                         ;AND HOLDS RHLA FROM MOVING
        MOV      @#UNLOAD,@RHCS1   ;LOAD UNLOAD COMMAND INTO RH
: *SAVE REGISTERS FOR COMPARISON AFTER GO
        JSR      R0,@#SAVER        ;SAVE
        RHW      ;FROM
        REINTO   ;TO
        19.          ;NUMBER OF REGISTERS SAVED
: *GIVE GO TO UNLOAD COMMAND
        BIS      #GO,@RHCS1       ;GO TO UNLOAD COMMAND
: *CHANGE SAVED REGISTERS TO EXPECTED VALUES
        BIS      #GO,@#REINTO+6    ;SAVED RHCS1
        BIS      #PIP,@#REINTO+30  ;SAVED RHDS1
        BIC      #DRY,@#REINTO+30  ;SAVED RHDS1
        TST      @#$PASS           ;IS THIS FIRST PASS
        BNE      5$                ;BRANCH IF NOT FIRST PASS
        BIT      #SW13,@SWR        ;INHIBIT ERROR PRINT HIGH?
        BNE      5$                ;BRANCH IF SW13 HIGH
        TYPE     ,65$              ;;TYPE ASCIZ STRING
        BR       64$              ;;GET OVER THE ASCIZ
:;65$: .ASCIZ <15><12>/IF DRIVE CONNECTED "STAND BY" LAMP SHOULD BE LIT ON DRIVE NO /
64$:
        MOV      @#UNIT,-(SP)      ;UNIT UNDER TEST
        TYPDS
: *AFTER GO HAS BEEN GIVEN TO UNLOAD COMMAND
: *SAVED REGISTERS AGAIN SO THAT COMPARISONS CAN
: *BE DONE
5$:    JSR      R0,@#SAVER        ;SAVE
        RHW      ;FROM
        WRF      ;TO

```



```

4638 022110 000023          19.          ;NUMBER OF REGISTERS SAVED
4639
4640          ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
4641          ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
4642          ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
4643 022112 113737 003205 002141  MOVB    @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS
4644
4645
4646          ;*COMPARE REGISTERS BEFORE UNLOAD COMMAND
4647          ;*WITH AFTER GO
4648
4649 022120 004037 043626      JSR     R0,@#COMPAR      ;COMPARE
4650 022124 003160            REINTO          ;GOOD BUFFER
4651 022126 002114            WRFROM         ;TEST BUFFER
4652 022130 000023          19.          ;NUMBER
4653 022132 022140          1$          ;RETURN FOR ERROR
4654 022134 022140          1$          ;SAME
4655 022136 022160          2$          ;RETURN FOR GOOD COMPARISON
4656 022140 013705 047736      1$:      MOV     @#ERWORD,R5      ;GETTING READY TO INDEX
4657 022144 060505            ADD     R5,R5          ;DOUBLE ERROR WORD
4658 022146 016537 001630 042402  MOV     RHC-2(R5),@#REGADR ;FAILING REGISTER ADDRESS
4659
4660 022154 104001            ERROR    1          ;IMPROPER REGISTER CHANGE
4661          ;AFTER UNLOAD COMMAND
4662          ;WITH GO IS GIVEN
4663 022156 000207            RTS     PC          ;RETURN TO COMPARISON
4664
4665          ;*NOW GIVE INIT AND GET ALL GO AND PIP DOWN
4666 022160 052712 000040      2$:      BIS     #CLR,@R2      ;RH INITILIZE
4667 022164 013712 001774      MOV     @#UNIT,@R2    ;REINSTATE UNIT NUMBER
4668 022170 012777 000001 157462  MOV     #DMD,@RHMR    ;SET DIAGNOSTIC MODE BIT
4669          ;THIS ENABLES COMMANDS WITHOUT MOL
4670          ;AND HOLDS RHLA FROM MOVING
4671
4672          ;*CHANGE REGISTERS TO EXPECTED VALUE
4673 022176 042737 000001 003166  BIC     #GO,@#REINTO+6 ;SAVED RHCS1
4674 022204 042737 020000 003210  BIC     #PIP,@#REINTO+30 ;SAVED RHDS1
4675 022212 052737 000200 003210  BIS     #DRY,@#REINTO+30 ;SAVED RHDS1
4676 022220 017737 157450 003222  MOV     @RHLA,@#REINTO+42;SAVED RHLA
4677
4678          ;*AFTER INITIALIZE SAVE REGISTERS SO THAT
4679          ;*COMPARES CAN BE DONE
4680
4681 022226 004037 043424      JSR     R0,@#SAVER     ;SAVE
4682 022232 001632            RHC        ;FROM
4683 022234 002114            WRFROM     ;TO
4684 022236 000023          19.          ;NUMBER OF REGISTERS SAVED
4685
4686          ;*COMPARE REGISTERS AFTER INITIALIZE
4687          ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
4688          ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
4689          ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
4690 022240 113737 003205 002141  MOVB    @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS
4691
4692
4693 022246 004037 043626      JSR     R0,@#COMPAR     ;COMPARE
  
```

```
4694 022252 003160 REINTO ;GOOD BUFFER
4695 022254 002114 WRFROM ;TEST BUFFER
4696 022256 000023 19. ;NUMBER OF REGISTERS TO BE
4697 ;COMPARED
4698 022260 022266 3$ ;RETURN POINT FOR ERROR
4699 022262 022266 3$ ;SAME
4700 022264 022306 4$ ;RETURN POINT FOR GOOD COMPARISON
4701
4702 022266 013705 047736 3$: MOV @#ERWORD,R5 ;GETTING READY TO INDEX
4703 022272 060505 ADD R5,R5 ;DOUBLE ERROR WORD
4704 022274 016537 001630 042402 MOV RHW-2(R5),@#REGADR ;FAILING REGISTER ADDRESS
4705 022302 104001 ERROR 1 ;IMPROPER REGISTER
4706 ;CONTENTS AFTER GIVING AN
4707 ;UNLOAD COMMAND
4708 022304 000207 RTS PC ;RETURN TO COMPARISON
4709 022306 4$: ;GOOD COMPARISON
4710
```

```

4711
4712 022306 032777 000020 156624      BIT      #SW4,@SWR      ;TEST FOR NO OFFSET OR RTC
4713 022314 001402                BEQ      6$           ;IF = 0, DO THE NEXT TWO TESTS
4714 022316 000137 023274                JMP      TST51        ;SKIP THE NEXT TWO TESTS -----)
4715 022322                6$:                ;CONTINUE WITH NEXT TWO TESTS
4716
4717
4718
4719
4720
4721
4722
4723
4724
4725
4726
4727
4728 022322 000004                ;*****
4729 022324 012706 001000                ;*TEST 47      OFFSET COMMAND TEST
4730 022330 012737 000047 002032                ;**      THE OFFSET COMMAND WILL BE LOADED INTO RHCS1 WITH GO
4731 022336 004737 042732                ;**      THEN ALL REGISTERS WILL BE CHECKED
4732
4733 022342 012777 000001 157310                ;**      RH CLEAR WILL BE GIVEN
4734
4735
4736
4737
4738 022350 052777 000004 157302                ;**      THEN ALL REGISTERS WILL BE CHECKED
4739 022356 042777 000004 157274                ;*****
4740
4741
4742
4743
4744 022364 017777 157306 157260                TST47:  SCOPE
4745 022372 013711 002074                MOV      #STACK,SP      ;RESET STACK
4746 022376 005211                MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
4747
4748
4749 022400 012700 000004                JSR      PC,@#CLDISK    ;INIT AND SET UP GENERAL REG.
4750 022404 012777 000011 157246 5$:                ;AND UNIT NUMBER
4751 022412 012777 000001 157240                MOV      #DMD,@RHMR     ;SET DIAGNOSTIC MODE BIT
4752 022420 005300                ;THIS ENABLES COMMANDS WITHOUT MOL
4753 022422 001370                ;AND HOLDS RHLA FROM MOVING
4754
4755 022424 004737 042732                ;*GIVE ONE INDEX PULSE TO CLEAR RHLA BEFORE THE START OF THIS TEST
4756 022430 012777 000001 157222                BIS      #MINX,@RHMR    ;SET INDEX PULSE
4757
4758
4759
4760
4761 022436 013777 002076 157174                BIC      #MINX,@RHMR    ;CLEAR INDEX
4762 022444 012777 000001 157176                ;*TO ENABLE LOOP ON THIS TEST THE POSITIONER HAS TO
4763
4764
4765 022452 004037 043424                ;*BE BROUGHT TO CENTER LINE
4766 022456 001632                MOV      @RHCC,@RHCA    ;SET DESIRED CYLINDER TO RHCC
4767
4768
4769
4770
4771
4772
4773
4774
4775
4776
4777
4778
4779
4780
4781
4782
4783
4784
4785
4786
4787
4788
4789
4790
4791
4792
4793
4794
4795
4796
4797
4798
4799

```

```

4767 022460 003160 REINTO ;TO
4768 022462 000023 19. ;NUMBER OF REGISTERS SAVED
4769
4770 ;*GIVE GO TO OFFSET COMMAND
4771 022464 052777 000001 157146 BIS #GO,@RHCS1 ;GO TO OFFSET COMMAND
4772
4773 ;*CHANGE SAVED REGISTERS TO EXPECTED VALUES
4774 022472 052737 000001 003166 BIS #GO,@#REINTO+6 ;SAVED RHCS1
4775 022500 052737 020000 003210 BIS #PIP,@#REINTO+30 ;SAVED RHDS1
4776 022506 042737 000200 003210 BIC #DRY,@#REINTO+30 ;SAVED RHDS1
4777
4778 ;*AFTER GO HAS BEEN GIVEN TO OFFSET COMMAND
4779 ;*SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN
4780 ;*BE DONE
4781
4782 022514 004037 043424 JSR RO,@#SAVER ;SAVE
4783 022520 001632 RHWC ;FROM
4784 022522 002114 WRFROM ;TO
4785 022524 000023 19. ;NUMBER OF REGISTERS SAVED
4786
4787 ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
4788 ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
4789 ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
4790 022526 113737 003205 002141 MOVB @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS
4791
4792
4793 ;*COMPARE REGISTERS BEFORE OFFSET COMMAND
4794 ;*WITH AFTER GO
4795
4796 022534 004037 043626 JSR RO,@#COMPAR ;COMPARE
4797 022540 003160 REINTO ;GOOD BUFFER
4798 022542 002114 WRFROM ;TEST BUFFER
4799 022544 000023 19. ;NUMBER
4800 022546 022554 1$ ;RETURN FOR ERROR
4801 022550 022554 1$ ;SAME
4802 022552 022574 2$ ;RETURN FOR GOOD COMPARISON
4803
4804 022554 013705 047736 1$: MOV @#ERWORD,R5 ;GETTING READY TO INDEX
4805 022560 060505 ADD R5,R5 ;DOUBLE ERROR WORD
4806 022562 016537 001630 042402 MOV RHWC-2(R5),@#REGADR ;FAILING REGISTER ADDRESS
4807 022570 104001 ERROR 1 ;IMPROPER REGISTER CHANGE
4808 ;AFTER OFFSET COMMAND
4809 ;WITH GO IS GIVEN
4810 022572 000207 RTS PC ;RETURN TO COMPARISON
4811
4812 ;*NOW GIVE INIT AND GET ALL GO AND PIP DOWN
4813
4814 022574 052712 000040 2$: BIS #CLR,@R2 ;RH INITILIZE
4815 022600 013712 001774 MOV @#UNIT,@R2 ;REINSTATE UNIT NUMBER
4816 022604 012777 000001 157046 MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE BIT
4817 ;THIS ENABLES COMMANDS WITHOUT M
4818 ;AND HOLDS RHLA FROM MOVING
4819
4820 ;*CHANGE REGISTERS TO EXPECTED VALUE
4821 022612 042737 000001 003166 BIC #GO,@#REINTO+6 ;SAVED RHCS1
4822 022620 042737 000001 003176 BIC #OF25,@#REINTO+16;SAVED RHOF
  
```



```
4860
4861
4862
4863
4864
4865
4866
4867
4868
4869
4870
4871 022730 000004
4872 022732 012706 001000
4873 022736 012737 000050 002032
4874 022744 004737 042732
4875
4876 022750 012777 000001 156702
4877
4878
4879
4880
4881 022756 052777 000004 156674
4882 022764 042777 000004 156666
4883
4884
4885 022772 013777 002100 156640
4886
4887
4888 023000 004037 043424
4889 023004 001632
4890 023006 003160
4891 023010 000023
4892
4893
4894 023012 052777 000001 156620
4895
4896
4897
4898 023020 012700 000004
4899 023024 012777 000011 156626 5$:
4900 023032 012777 000001 156620
4901 023040 005300
4902 023042 001370
4903
4904
4905
4906 023044 052737 000001 003166
4907 023052 052737 020000 003210
4908 023060 042737 000200 003210
4909
4910
4911
4912
4913 023066 004037 043424
4914 023072 001632
4915 023074 002114
```

```
*****
;*TEST 50 RETURN TO CENTER LINE COMMAND TEST
*****
** THE RETURN TO CENTER LINE COMMAND WILL BE LOADED INTO RHCS1 WITH GO
** THEN ALL REGISTERS WILL BE CHECKED
** RH CLEAR WILL BE GIVEN
** THEN ALL REGISTERS WILL BE CHECKED
*****
TST50: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
JSR PC,@#CLDISK ;INIT AND SET UP GENERAL REG.
;AND UNIT NUMBER
MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE BIT
;THIS ENABLES COMMANDS WITHOUT MOL
;AND HOLDS RHLA FROM MOVING

;*GIVE ONE INDEX PULSE TO CLEAR RHLA BEFORE THE START OF THIS TEST
BIS #MINX,@RHMR ;SET INDEX PULSE
BIC #MINX,@RHMR ;CLEAR INDEX

MOV @#RETCL,@RHCS1 ;LOAD RETURN TO CENTER LINE COMMAND INTO RHCS1

;*SAVE REGISTERS FOR COMPARISON AFTER GO
JSR RO,@#SAVER ;SAVE
RHC ;FROM
REINTO ;TO
19. ;NUMBER OF REGISTERS SAVED

;*GIVE GO TO RETURN TO CENTER LINE COMMAND
BIS #GO,@RHCS1 ;GO TO RETURN TO CENTER COMMAND

;*FOUR SECTOR CLOCKS ARE GIVEN TO TAKE POSITIONER TO CENTER LINE
MOV #4,RO ;COUNTER
MOV #MSTCK!DMD,@RHMR ;SET SECTOR CLOCK
MOV #DMD,@RHMR ;RESET SECTOR CLOCK
DEC RO ;COUNT
BNE 5$ ;BRANCH IF NOT COMPLETE

;*CHANGE SAVED REGISTERS TO EXPECTED VALUES
BIS #GO,@#REINTO+6 ;SAVED RHCS1
BIS #PIP,@#REINTO+30 ;SAVED RHDS1
BIC #DRY,@#REINTO+30 ;SAVED RHDS1

;*AFTER GO HAS BEEN GIVEN TO RETURN TO CENTER LINE COMMAND
;*SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN
;*BE DONE
JSR RO,@#SAVER ;SAVE
RHC ;FROM
WRFROM ;TO
```

```
4916 023076 000023 19. ;NUMBER OF REGISTERS SAVED
4917
4918 ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
4919 ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
4920 ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
4921 023100 113737 003205 002141 MOVB @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS
4922
4923
4924 ;*COMPARE REGISTERS BEFORE RETURN TO CENTER LINE COMMAND
4925 ;*WITH AFTER GO
4926 023106 004037 043626 JSR RO,@#COMPAR ;COMPARE
4927 023112 003160 REINTO ;GOOD BUFFER
4928 023114 002114 WRFROM ;TEST BUFFER
4929 023116 000023 19. ;NUMBER
4930 023120 023126 1$ ;RETURN FOR ERROR
4931 023122 023126 1$ ;SAME
4932 023124 023146 2$ ;RETURN FOR GOOD COMPARISON
4933
4934 023126 013705 047736 1$: MOV @#ERWORD,R5 ;GETTING READY TO INDEX
4935 023132 060505 ADD R5,R5 ;DOUBLE ERROR WORD
4936 023134 016537 001630 042402 MOV RHWC-2(R5),@#REGADR ;FAILING REGISTER ADDRESS
4937 023142 104001 ERROR 1 ;IMPROPER REGISTER CHANGE
4938 ;AFTER RETURN TO CENTER LINE COMMAND
4939 ;WITH GO IS GIVEN
4940 023144 000207 RTS PC ;RETURN TO COMPARISON
4941
4942 ;*NOW GIVE INIT AND GET ALL GO AND PIP DOWN
4943
4944 023146 052712 000040 2$: BIS #CLR,@R2 ;RH INITILIZE
4945 023152 013712 001774 MOV @#UNIT,@R2 ;REINSTATE UNIT NUMBER
4946 023156 012777 000001 156474 MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE BIT
4947 ;THIS ENABLES COMMANDS WITHOUT MOL
4948 ;AND HOLDS RHLA FROM MOVING
4949
4950 ;*CHANGE REGISTERS TO EXPECTED VALUE
4951 023164 042737 000001 003166 BIC #GO,@#REINTO+6 ;SAVED RHCS1
4952 023172 042737 020000 003210 BIC #PIP,@#REINTO+30 ;SAVED RHDS1
4953 023200 052737 000200 003210 BIS #DRY,@#REINTO+30 ;SAVED RHDS1
4954 023206 017737 156462 003222 MOV @RHLA,@#REINTO+42;SAVED RHLA
4955
4956 ;*AFTER INITIALIZE SAVE REGISTERS SO THAT
4957 ;*COMPARES CAN BE DONE
4958 023214 004037 043424 JSR RO,@#SAVER ;SAVE
4959 023220 001632 RHWC ;FROM
4960 023222 002114 WRFROM ;TO
4961 023224 000023 19. ;NUMBER OF REGISTERS SAVED
4962
4963 ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
4964 ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
4965 ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
4966 023226 113737 003205 002141 MOVB @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS
4967
4968
4969 ;*COMPARE REGISTERS AFTER INITIALIZE
4970 023234 004037 043626 JSR RO,@#COMPAR ;COMPARE
4971 023240 003160 REINTO ;GOOD BUFFER
```

```

4972 023242 002114      WRFROM      ;TEST BUFFER
4973 023244 000023      19.        ;NUMBER OF REGISTERS TO BE
4974                    ;COMPARED
4975 023246 023254      3$         ;RETURN POINT FOR ERROR
4976 023250 023254      3$         ;SAME
4977 023252 023274      4$         ;RETURN POINT FOR GOOD COMPARISON
4978
4979
4980 023254 013705 047736 3$:   MOV    @#ERWORD,R5    ;GETTING READY TO INDEX
4981 023260 060505      ADD    R5,R5         ;DOUBLE ERROR WORD
4982 023262 016537 001630 042402 MOV    R#WC-2(R5),@#REGADR ;FAILING REGISTER ADDRESS
4983 023270 104001      ERROR   1           ;IMPROPER REGISTER
4984                    ;CONTENTS AFTER GIVING AN
4985                    ;INITIALIZE FOLLOWING A RETURN TO
4986                    ;CENTER LINE COMMAND
4987 023272 000207      RTS    PC           ;RETURN TO COMPARISON
4988
4989 023274              4$:           ;GOOD COMPARISON
4990
  
```


4991
4992
4993
4994
4995
4996
4997
4998
4999
5000
5001
5002 023274 000004
5003 023276 012706 001000
5004 023302 012737 000051 002032
5005 023310 004737 042732
5006
5007 023314 012777 000001 156336
5008
5009
5010
5011
5012
5013
5014 023322 052777 000004 156330
5015 023330 042777 000004 156322
5016
5017 023336 013777 002050 156274
5018
5019
5020 023344 004037 043424
5021 023350 001632
5022 023352 003160
5023 023354 000023
5024
5025
5026 023356 052777 000001 156254
5027
5028
5029
5030 023364 012700 000004
5031 023370 012777 000011 156262 5\$:
5032 023376 012777 000001 156254
5033 023404 005300
5034 023406 001370
5035
5036
5037
5038 023410 052737 000001 003166
5039 023416 052737 020000 003210
5040 023424 042737 000200 003210
5041
5042
5043
5044
5045 023432 004037 043424
5046 023436 001632

:*TEST 51 RECALIBRATE COMMAND TEST

:** THE RECALIBRATE COMMAND WILL BE LOADED INTO RHCS1 WITH GO
:** THEN ALL REGISTERS WILL BE CHECKED
:** RH CLEAR WILL BE GIVEN
:** THEN ALL REGISTERS WILL BE CHECKED

TST51: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
JSR PC,@#CLDISK ;INIT AND SET UP GENERAL REG.
;AND UNIT NUMBER
MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE BIT
;THIS ENABLES COMMANDS WITHOUT MOL
;AND HOLDS RHLA FROM MOVING

;*GIVE ONE INDEX PULSE TO CLEAR RHLA BEFORE THE START OF THIS TEST

BIS #MINX,@RHMR ;SET INDEX PULSE
BIC #MINX,@RHMR ;CLEAR INDEX

MOV @#RECALI,@RHCS1 ;LOAD RECALIBRATE COMMAND INTO RHCS1

;*SAVE REGISTERS FOR COMPARISON AFTER GO

JSR RO,@#SAVER ;SAVE
RHWC ;FROM
REINTO ;TO
19. ;NUMBER OF REGISTERS SAVED

;*GIVE GO TO RECALIBRATE COMMAND

BIS #GO,@RHCS1 ;GO TO RECALIBRATE COMMAND

;*FOUR SECTOR CLOCKS ARE GIVEN TO TAKE POSITIONER TO CYLINDER 0

MOV #4,RO ;COUNTER
MOV #MSTCK!DMD,@RHMR ;SET SECTOR CLOCK
MOV #DMD,@RHMR ;RESET SECTOR CLOCK
DEC RO ;COUNT
BNE 5\$;BRANCH IF NOT COMPLETE

;*CHANGE SAVED REGISTERS TO EXPECTED VALUES

BIS #GO,@#REINTO+6 ;SAVED RHCS1
BIS #PIP,@#REINTO+30 ;SAVED RHDS1
BIC #DRY,@#REINTO+30 ;SAVED RHDS1

;*AFTER GO HAS BEEN GIVEN TO RECALIBRATE COMMAND

;*SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN

;*BE DONE
JSR RO,@#SAVER ;SAVE
RHWC ;FROM

```

5047 023440 002114      WRFROM      ;TO
5048 023442 000023      19.         ;NUMBER OF REGISTERS SAVED
5049
5050      ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
5051      ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
5052      ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
5053 023444 113737 003205 002141  MOVB    @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS
5054
5055
5056      ;*COMPARE REGISTERS BEFORE RECALIBRATE COMMAND
5057      ;*WITH AFTER GO
5058 023452 004037 043626  JSR     R0,@#COMPAR    ;COMPARE
5059 023456 003160      REINTO     ;GOOD BUFFER
5060 023460 002114      WRFROM     ;TEST BUFFER
5061 023462 000023      19.        ;NUMBER
5062 023464 023472      1$         ;RETURN FOR ERROR
5063 023466 023472      1$         ;SAME
5064 023470 023512      2$         ;RETURN FOR GOOD COMPARISON
5065
5066 023472 013705 047736      1$:      MOV     @#ERWORD,R5    ;GETTING READY TO INDEX
5067 023476 060505      ADD     R5,R5         ;DOUBLE ERROR WORD
5068 023500 016537 001630 042402  MOV     RHWC-2(R5),@#REGADR ;FAILING REGISTER ADDRESS
5069 023506 104001      ERROR    1          ;IMPROPER REGISTER CHANGE
5070      ;AFTER RECALIBRATE COMMAND
5071      ;WITH GO IS GIVEN
5072 023510 000207      RTS     PC           ;RETURN TO COMPARISON
5073
5074      ;*NOW GIVE INIT AND GET ALL GO AND PIP DOWN
5075
5076 023512 052712 000040      2$:      BIS     #CLR,@R2      ;RH INITILIZE
5077 023516 013712 001774      MOV     @#UNIT,@R2    ;REINSTATE UNIT NUMBER
5078 023522 012777 000001 156130  MOV     #DMD,@RHMR    ;SET DIAGNOSTIC MODE BIT
5079      ;THIS ENABLES COMMANDS WITHOUT MOL
5080      ;AND HOLDS RHLA FROM MOVING
5081
5082      ;*CHANGE REGISTERS TO EXPECTED VALUE
5083 023530 042737 000001 003166  BIC     #GO,@#REINTO+6 ;SAVED RHCS1
5084 023536 042737 020000 003210  BIC     #PIP,@#REINTO+30 ;SAVED RHDS1
5085 023544 052737 000200 003210  BIS     #DRY,@#REINTO+30 ;SAVED RHDS1
5086 023552 017737 156116 003222  MOV     @RHLA,@#REINTO+42;SAVED RHLA
5087
5088      ;*AFTER INITIALIZE SAVE REGISTERS SO THAT
5089      ;*COMPARES CAN BE DONE
5090 023560 004037 043424  JSR     R0,@#SAVER    ;SAVE
5091 023564 001632      RHWC      ;FROM
5092 023566 002114      WRFROM    ;TO
5093 023570 000023      19.        ;NUMBER OF REGISTERS SAVED
5094
5095      ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
5096      ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
5097      ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
5098 023572 113737 003205 002141  MOVB    @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS
5099
5100
5101      ;*COMPARE REGISTERS AFTER INITIALIZE
5102 023600 004037 043626  JSR     R0,@#COMPAR    ;COMPARE
  
```

```

5103 023604 003160 REINTO ;GOOD BUFFER
5104 023606 002114 WRFROM ;TEST BUFFER
5105 023610 000023 19. ;NUMBER OF REGISTERS TO BE
5106 ;COMPARED
5107 023612 023620 3$ ;RETURN POINT FOR ERROR
5108 023614 023620 3$ ;SAME
5109 023616 023640 4$ ;RETURN POINT FOR GOOD COMPARISON
5110
5111 023620 013705 047736 3$: MOV @#ERWORD,R5 ;GETTING READY TO INDEX
5112 023624 060505 ADD R5,R5 ;DOUBLE ERROR WORD
5113 023626 016537 001630 042402 MOV RHWC-2(R5),@#REGADR ;FAILING REGISTER ADDRESS
5114 023634 104001 ERROR 1 ;IMPROPER REGISTER
5115 ;CONTENTS AFTER GIVING AN
5116 ;INITIALIZE FOLLOWING A
5117 ;RECALIBRATE COMMAND
5118 023636 000207 RTS PC ;RETURN TO COMPARISON
5119
5120 023640 4$: ;GOOD COMPARISON
    
```

5121
5122
5123
5124
5125
5126
5127
5128
5129
5130
5131
5132
5133
5134
5135
5136
5137
5138
5139
5140
5141
5142
5143
5144
5145
5146
5147
5148
5149
5150
5151
5152
5153
5154
5155
5156
5157
5158
5159
5160
5161
5162
5163
5164
5165
5166
5167
5168
5169
5170
5171
5172
5173
5174
5175
5176

023640 000004
023642 012706 001000
023646 012737 000052 002032
023654 004737 042732
023660 012777 000001 155772
023666 013777 002054 155744
023674 004037 043424
023700 001632
023702 003160
023704 000023
023706 052777 000001 155724
023714 052777 000001 155736
023722 052737 000001 003206
023730 017737 155740 003222
023736 004037 043424
023742 001632
023744 002114
023746 000023
023750 113737 003205 002141
023756 004037 043626
023762 003160
023764 002114

```
*****  
: *TEST 52          RELEASE COMMAND TEST  
*****  
: **      THE RELEASE COMMAND WILL BE LOADED INTO RHCS1 WITH GO  
: **      THEN ALL REGISTERS WILL BE CHECKED  
: **      RH CLEAR WILL BE GIVEN  
: **      THEN ALL REGISTERS WILL BE CHECKED  
*****  
TST52:  SCOPE  
MOV      #STACK,SP          ;RESET STACK  
MOV      #TTNO,@#TSTNM     ;THIS SAVES TEST NUMBER  
JSR      PC,@#CLDISK       ;INIT AND SET UP GENERAL REG.  
                          ;AND UNIT NUMBER  
MOV      #DMD,@RHMR        ;SET DIAGNOSTIC MODE BIT  
                          ;THIS ENABLES COMMANDS WITHOUT MOL  
                          ;AND HOLDS RHLA FROM MOVING  
MOV      @#RELEASE,@RHCS1  ;LOAD RELEASE COMMAND INTO RHCS1  
; *SAVE REGISTERS FOR COMPARISON AFTER GO  
JSR      RO,@#SAVER        ;SAVE  
RHW      ;FROM  
REINTO   ;TO  
19.      ;NUMBER OF REGISTERS SAVED  
; *GIVE GO TO RELEASE COMMAND  
BIS      #GO,@RHCS1       ;GO TO RELEASE COMMAND  
BIS      #DMD,@RHMR       ;SET DMD TO HOLD RHLA  
; *CHANGE SAVED REGISTERS TO EXPECTED VALUES  
; *AFTER GO HAS BEEN GIVEN TO RELEASE COMMAND  
BIS      #DMD,@#REINTO+26 ;SAVED RHMR  
MOV      @RHLA,@#REINTO+42 ;SAVED RHLA  
; *SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN  
; *BE DONE  
JSR      RO,@#SAVER        ;SAVE  
RHW      ;FROM  
WRFROM   ;TO  
19.      ;NUMBER OF REGISTERS SAVED  
; *AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT  
; *OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS  
; *SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE  
MOVB    @#REINTO+25,@#WRFROM+25 ;SAVE UPPER RHAS  
; *COMPARE REGISTERS BEFORE RELEASE COMMAND  
; *WITH AFTER GO  
JSR      RO,@#COMPAR      ;COMPARE  
REINTO   ;GOOD BUFFER  
WRFROM   ;TEST BUFFER
```

```
5177 023766 000023      19.              ;NUMBER
5178 023770 023776      1$              ;RETURN FOR ERROR
5179 023772 023776      1$              ;SAME
5180 023774 024016      2$              ;RETURN FOR GOOD COMPARISON
5181
5182 023776 013705 047736 1$:      MOV @#ERWORD,R5 ;GETTING REAYD TO INDEX
5183 024002 060505             ADD R5,R5      ;DOUBLE ERROR WORD
5184 024004 016537 001630 042402 MOV RHC-2(R5),@#REGADR ;FAILING REGISTER ADDRESS
5185 024012 104001             ERROR 1      ;IMPROPER REGISTER CHANGE
5186                          ;AFTER RELEASE COMMAND
5187                          ;WITH GO IS GIVEN
5188 024014 000207             RTS PC       ;RETURN TO COMPARISON
5189 024016
5190
```

5191
5192
5193
5194
5195
5196
5197 024016 000004
5198 024020 012706 001000
5199 024024 012737 000053 002032
5200 024032 004737 042732
5201 024036 012777 000001 155614
5202 024044 004037 045304
5203
5204
5205 024050 000000
5206
5207
5208
5209
5210
5211
5212
5213
5214
5215
5216
5217
5218
5219
5220 024052 000004
5221 024054 012706 001000
5222 024060 012737 000054 002032
5223 024066 004737 042732
5224
5225
5226 024072 012777 000025 155544
5227
5228 024100 005077 155546
5229 024104 012777 010000 155536
5230 024112 013711 002056
5231
5232
5233 024116 004037 043424
5234
5235 024122 001632
5236 024124 003160
5237 024126 000023
5238
5239 024130 004737 042766
5240 024134 104401 062562
5241
5242 024140 000000
5243
5244
5245
5246

*TEST 53 MAKE CURRENT CYLINDER = 0

TST53: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
JSR PC,@#CLDISK ;INIT DRIVE
MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE
JSR RO,@#MAKECYL ;SUBROUTINE TO GIVE A SEEK
;COMMAND FOLOWED BY AN INIT
;THIS SHOULD CHANGE RHCC
;CHANGE RHCC TO 0
0

*TEST 54 LOOK AHEAD REGISTER

** A SEARCH COMMAND IS GIVEN FOR CYLINDER 0, TRACK 0, SECTOR 21.
** THE LOOK AHEAD REGISTER IS CHECKED AFTER INDEX PULSE
** THE EXTENSION FIELD IS CHECKED IN EACH SECTOR AFTER
** 128 BYTES THEN AGAIN AFTER 128 MORE BYTES THEN AGAIN AFTER 256 MORE BYTES
** THE SECTOR COUNT FIELD IS CHECKED AFTER EACH SECTOR
** AT THE END ALL REGISTERS ARE CHECKED

TST54: SCOPE

MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
JSR PC,@#CLDISK ;INIT AND SET UP GENERAL REGISTERS
;*THESE ARE REGULAR SET UPS FOR SEARCH COMMAND
MOV #21.,@RHDS1 ;DESIRED SECTOR/TRACK REGISTER
;TRACK 0 SECTOR 21
CLR @RHCA ;DESIRED CYLINDER =0
MOV #FMT22,@RHOF ;FORMAT BIT=1 (16 BITS PER WORD)
MOV @#SERCH,@R1 ;FILL SEARCH COMMAND IN RHCS1
;*NOW SAVE REGISTERS STARTING FROM RHWC IN WRITE FROM BUFFER
JSR RO,@#SAVER ;SAVE REGISTERS FOR COMPARISON
;AT THE END OF THE SEARCH
RHWC ;START SAVING FROM RHWC
REINTO ;SAVE INTO REINTO
19. ;NUMBER OF REGISTERS SAVED
JSR PC,@#CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-
;TINUE TESTING IF BOTH AREN'T TRUE
HALT ;STOP THE TEST

*NOW THE DIAGNOSTIC MODE BIT WILL BE SET
*AND THE SEARCH OPERATION STARTED

```
5247 024142 005037 001200 CLR @#STMP1 ;THIS WILL HAVE THE EXPECTED
5248 ;VALUE OF RHLA REGISTER
5249
5250 024146 013700 001660 MOV @#RHMR,R0 ;NOW R0 HAS MAINTENANCE REG. ADDR.
5251 024152 017703 155466 MOV @RHDST,R3 ;GET DESIRED SECTOR/TRACK REG.
5252 024156 042703 177400 BIC #177400,R3 ;GET SECTOR ONLY
5253 024162 010337 053662 MOV R3,@#SECTR ;DUPLICATE SECTOR
5254 024166 012710 000001 MOV #DMD,@RO ;S
5255 024172 052777 000001 155440 BIS #GO,@RHCS1 ;GO
5256 024200 052710 000010 BIS #MSTCK,@RO ;SET SECTOR CLOCK
5257 024204 042710 000010 BIC #MSTCK,@RO ;CLEAR SECTOR CLOCK
5258 024210 000240 NOP ;ALLOW TIME BETWEEN SECTOR CLOCKS
5259 024212 052710 000010 BIS #MSTCK,@RO ;SET SECTOR CLOCK
5260 024216 042710 000010 BIC #MSTCK,@RO ;CLEAR SECTOR CLOCK
5261 024222 000240 NOP ;ALLOW TIME BETWEEN SECTOR CLOCKS
5262 024224 052710 000014 BIS #MINX!MSTCK,@RO ;SET INDEX AND SECTOR CLOCK
5263 024230 012710 000001 MOV #DMD,@RO ;RESET INDEX AND SECTOR CLOCK
5264 024234 005703 TST R3 ;IF SECTOR REQUIRED JUMP OUT
5265 024236 001555 BEQ 11$ ;BRANCH OF SECTOR ZERO REQUIRED
5266
5267 ;*AFTER THE INDEX PULSE RHLA WILL BE CHECKED TO BE ZERO
5268 ;*AND $TMP4 WILL BE SET UP TO COUNT BYTES
5269 024240 012737 001140 001206 1$: MOV #608.,@#$TMP4 ;THERE ARE 608 BYTES PER SECTOR
5270 024246 017737 155422 001126 MOV @RHLA,@#$BDDAT ;SAVE RHLA
5271 024254 017737 155364 001720 MOV @RHDST,@#DST ;SAVE DESIRED SECTOR TRACK
5272 024262 023737 001200 001126 CMP @#$TMP1,@#$BDDAT ;RHLA SHOULD BE HAVE EXTENSION
5273 ;FIELD EQUAL TO ZERO
5274 024270 001414 BEQ 2$ ;BRANCH IF GOOD
5275 024272 013737 053662 001202 MOV @#SECTR,@#$TMP2 ;GET SECTOR SOUGHT
5276 024300 160337 001202 SUB R3,@#$TMP2 ;$TMP2 NOW HAS PRESENT SECTOR
5277 024304 012746 001140 MOV #608.,-(SP) ;NUMBER OF BYTES PER SECTOR
5278 024310 163716 001206 SUB @#$TMP4,(SP) ;(SP)HAS PRESENT BYTE NUMBER
5279 024314 012637 001204 MOV (SP)+,@#$TMP3 ;PRESENT BYTE NUMBER
5280 024320 104024 ERROR 24 ;LOOK AHEAD REGISTER AT THE BEGINING OF A
5281 ;SECTOR IS IN ERROR
5282
5283
5284 ;*NOW THE 304 WORDS WILL START
5285 ;*FOR FIRST BYTE CLOCK WILL BE INDEPENDENT OF
5286 ;*SECTOR CLOCK THEN IT WILL COINCIDE FOREVER TILL
5287 ;*THE BEGINNING OF NEXT SECTOR
5288
5289 ;*ONE WORD ONLY THAT IS TWO BYTES
5290
5291 024322 012702 000010 2$: MOV #8.,R2 ;BYTE
5292 024326 012705 000002 MOV #2,R5 ;BYTES PER WORD
5293 024332 000404 BR 4$
5294 024334 052710 000012 3$: BIS #MSTCK!MCLK,@RO ;SET SECTOR AND CLOCK
5295 024340 042710 000012 BIC #MSTCK!MCLK,@RO ;CLEAR SECTOR AND CLOCK
5296 024344 052710 000002 4$: BIS #MCLK,@RO ;SET CLOCK
5297 024350 042710 000002 BIC #MCLK,@RO ;CLEAR CLOCK
5298 024354 005302 DEC R2 ;BYTE COUNTER
5299 024356 001372 BNE 4$ ;BRANCH IF BYTE NOT COMPLETE
5300 024360 005337 001206 DEC @#$TMP4 ;BYTE COUNT DOWN
5301 024364 012702 000007 MOV #7,R2 ;SETUP FOR SECOND BYTE
5302 024370 005305 DEC R5 ;IS WORD COMPLETE?
```

```
5303 024372 001360      BNE      3$      ;BRENCH IF NOT COMPLETE
5304                      ;TO GIVE SECTOR CLOCK AND CLOCK
5305
5306                      ;*NOW 303 WORDS ARE LEFT ALL ARE IDENTICAL
5307                      ;*THAT IS 606 IDENTICAL BYTES WILL BE GIVEN
5308                      ;*RHLA WILL BE CHECKED STAR TO COUNT AFTER
5309                      ;*BEGINNING OF SECTOR PULSE
5310                      ;*AFTER 128 BYTES (2 BYTES ARE ALREADY GIVEN)
5311                      ;*SO 127 MORE
5312                      ;*THEN RHLA WILL BE CHECKED AFTER 128 MORE BYTES
5313                      ;*THEN RHLA WILL BE CHECKED AFTER 256 MORE BYTES
5314                      ;*THEN THE TOTAL OF 608 BYTES WILL BE COMPLETED
5315                      ;*AND RHLA WILL BE MADE READY FOR NEXT SECTOR
5316                      ;*AND RHLA WILL BE CHECKED
5317
5318 024374 012705 000100  MOV      #64.,R5      ;R5 WILL KEEP TRACK WHEN
5319                      ;EXTENSION FIELD IS TO BE CHECKED
5320 024400 012701 000177  MOV      #127.,R1     ;FIRST TIME CHECK EXTENSION FIELD
5321                      ;AFTER 127 MORE BYTES
5322 024404 012702 000007  5$:  MOV      #7,R2      ;CLOCKS PER BYTE COUNTER
5323 024410 052710 000012  BIS      #MSTCK!MCLK,@RO ;SET SECTOR CLOCK AND CLOCK
5324 024414 042710 000012  BIC      #MSTCK!MCLK,@RO ;CLEAR SECTOR CLOCK AND CLOCK
5325 024420 052710 000002  6$:  BIS      #MCLK,@RO   ;SET CLOCK
5326 024424 042710 000002  BIC      #MCLK,@RO   ;RESET CLOCK
5327 024430 005302          DEC      R2          ;COUNT DOWN CLOCKS PER BYTE
5328 024432 001372          BNE      6$          ;BRANCH IF BYTE NOT COMPLETE
5329 024434 005337 001206  DEC      @#$TMP4     ;COUNT DOWN BYTES
5330 024440 001436          BEQ      10$         ;BRANCHOUT IF 608 BYTES DONE
5331 024442 005301          DEC      R1          ;COUNT DOWN NUMBER OF BYTES
5332                      ;TO CHECK EXTENSION FIELD
5333 024444 001357          BNE      5$          ;BRANCH IF EXTENSION FIELD NOT
5334                      ;TO BE CHECKED YET
5335
5336                      ;*NOW THE EXTENSION FIELD OF THE LOOK AHEAD REGISTER
5337                      ;*WILL BE CHECKED
5338
5339 024446 062737 000020 001200  ADD      #20,@#$TMP1   ;GET TO THE NEXT EXTENSION
5340 024454 017737 155214 001126  MOV      @RHLA,@#$BDDAT ;GET RHLA FOR COMPARISON
5341
5342 024462 017737 155156 001720  MOV      @RHDST,@#DST  ;SAVE DESIRED SECTOR TRACK
5343 024470 023737 001200 001126  CMP      @#$TMP1,@#$BDDAT ;CHECK VALUE OF RHLA
5344 024476 001414          BEQ      7$          ;BRANCH IF GOOD
5345 024500 013737 053662 001202  MOV      @#SECTR,@#$TMP2 ;GET SECTOR SOUGHT
5346 024506 160337 001202          SUB      R3,@#$TMP2    ;$TMP2 NOW HAS PRESENT SECTOR
5347 024512 012746 001140          MOV      #608.,-(SP)  ;NUMBER OF BYTES PER SECTOR
5348 024516 163716 001206          SUB      @#$TMP4,(SP) ;(SP) HAS PRESENT BYTE NUMBER
5349 024522 012637 001204          MOV      (SP)+,@#$TMP3 ;PRESENT BYTE NUMBER
5350 024526 104025          ERROR    25         ;LOOK AHEAD ERROR IN THE MIDDLE
5351                      ;OF A SECTOR IS IN ERROR
5352
5353 024530 060505          7$:  ADD      R5,R5         ;GET NEXT STEP TO CHECK EXTENSION FIELD
5354 024532 010501          MOV      R5,R1         ;PUT IN COUNTER
5355 024534 000723          BR       5$          ;BRANCH BACK SECTOR
5356                      ;IS NOT COMPLETE
5357 024536 062737 000020 001200  10$:  ADD      #20,@#$TMP1   ;THESE TWO INSTRUCTIONS GIVE
5358 024544 052710 000010  BIS      #MSTCK,@RO
```



```

5359 024550 042710 000010      BIC    #MSTCK,@RO      ;ONE SECTOR CLOCK EXTRA
5360 024554 000240              NOP                      ;ALLOW TIME BETWEEN SECTOR CLOCK
5361 024556 052710 000010      BIS    #MSTCK,@RO      ;THESE TWO INSTRUCTIONS GIVE
5362 024562 042710 000010      BIC    #MSTCK,@RO      ;ONE SECTOR CLOCK EARLY
5363                                ;BEFORE THE NEXT SECTOR
5364 024566 005303              DEC    R3                ;IS REQUIRED NO OF SECTORS COMPLETE
5365 024570 001223              BNE    1$                ;BRANCH IF NOT
5366
5367                                ;*NOW THE REQUIRED SECTOR IS REACHED
5368                                ;*ONE SECTOR CLOCK WILL BE GIVEN TO GET SECTOR PULSE
5369                                ;*DOWN AND HENCE ATA UP
5370
5371 024572 012702 000010      11$:  MOV    #8.,R2        ;8 CLOCKS
5372 024576 052710 000002      12$:  BIS    #MCLK,@RO      ;SET CLOCK
5373 024602 042710 000002      BIC    #MCLK,@RO      ;CLEAR CLOCKS
5374 024606 005302              DEC    R2                ;COUND DOWN
5375 024610 001372              BNE    12$              ;BRANCH IF 8 NOT DONE
5376 024612 052710 000012      BIS    #MSTCK!MCLK,@RO ;SET SECTOR AND CLOCK
5377 024616 042710 000012      BIC    #MSTCK!MCLK,@RO ;CLEAR SECTOR AND CLOCK
5378
5379                                ;*NOW ALL REGISTERS WILL BE COMPARED
5380                                ;*SO FILL EXPECTED VALUE INTO SAVED LOCATIONS
5381
5382 024622 052737 100000 003166  BIS    #SC,@#REINTO+6   ;INCLUDE SC IN SAVED RHCS1
5383 024630 053737 002016 003204  BIS    @#ATTENT,@#REINTO+24 ;FILL APPROPRIATE ATTENTION
5384                                ;IN SAVED RHAS
5385 024636 052737 000001 003206  BIS    #DMD,@#REINTO+26 ;SET DMD IN RHMR SAVED
5386 024644 052737 100000 003210  BIS    #ATA,@#REINTO+30 ;SET ATA IN RHDS1 SAVED
5387 024652 013737 001200 003222  MOV    @#$TMP1,@#REINTO+42 ;MOVE EXPECTED VALUE
5388                                ;INTO RHLA SAVED
5389
5390                                ;*AFTER SEARCH COMMAND
5391                                ;*SAVE REGISTERS AGAIN SO THAT COMPARES CAN BE DONE
5392
5393 024660 004037 043424      JSR    RO,@#SAVER      ;SAVE
5394 024664 001632              RHC    ;FROM
5395 024666 002114              WRFROM ;TO
5396 024670 000023              19.    ;NUMBER
5397
5398                                ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
5399                                ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
5400                                ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
5401 024672 113737 003205 002141  MOV    @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS
5402
5403
5404                                ;*COMPARE REGISTERS BEFORE SEARCH WITH AFTER
5405
5406 024700 004037 043626      JSR    RO,@#COMPAR     ;COMPAR
5407 024704 003160              REINTO ;GO BUFFER
5408
5409                                ;TEST BUFFER
5410 024710 000022              WRFROM ;NUMBER
5411 024712 024720              18.    ;NUMBER
5412 024714 024720              13$    ;RETURN FOR ERROR
5413 024716 024740              13$    ;SAME
5414 024720 013705 047736      14$    ;RETURN FOR GOOD COMPARISON
5415                                13$:  MOV    @#ERWORD,R5    ;GETTING READY TO INDEX

```

| | | | | | | | | |
|------|--------|--------|--------|--------|-------|---------------------|--|-----------------------|
| 5415 | 024724 | 060505 | | | ADD | R5,R5 | | ;DOUBLE ERROR WORD |
| 5416 | 024726 | 016537 | 001630 | 042402 | MOV | RHWC-2(R5),@#REGADR | | ;FAILING REG. ADDRESS |
| 5417 | 024734 | 104001 | | | ERROR | 1 | | ;CONTENTS OF REGISTER |
| 5418 | 024736 | 000207 | | | RTS | PC | | ;CHANGED AT END OF |
| 5419 | 024740 | | | 14\$: | | | | ;SEARCH |
| 5420 | | | | | | | | |

5421
5422
5423
5424
5425
5426
5427
5428
5429
5430
5431
5432
5433
5434

024740 000004
024742 012706 001000
024746 012737 000055 002032
024754 004737 042732
024760 012777 000001 154672
024766 004037 045304

024772 000000

```
*****  
;*TEST 55 MAKE CURRENT CYLINDER = 0  
*****  
TST55: SCOPE  
MOV #STACK,SP ;RESET STACK  
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER  
JSR PC,@#CLDISK ;INIT DRIVE  
MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE  
JSR RO,@#MAKECYL ;SUBROUTINE TO GIVE A SEEK  
;COMMAND FOLLOVED BY AN INIT  
;THIS SHOULD CHANGE RHCC  
;CHANGE RHCC TO 0  
  
0
```

5435
5436
5437
5438
5439
5440
5441
5442
5443
5444
5445
5446
5447
5448
5449
5450
5451
5452
5453
5454
5455
5456
5457
5458
5459
5460
5461
5462
5463
5464
5465
5466
5467
5468
5469
5470
5471
5472
5473
5474
5475
5476
5477
5478
5479
5480
5481
5482
5483
5484
5485
5486
5487
5488
5489
5490

.SBTTL READ/WRITE ADDRESSING VIA RHMR

*TEST 56 WRITE HEADER AND DATA 1

** WRITE CYLINDER 0, FORMAT 16 BIT PER WORD
** TRACK 0, SECTOR 0, KEYS 0, NUMBER OF WORDS 256 OF 0'S
** AS EVERYTHING IS ZERO THIS PROVES VERY LITTLE
** ANY ERROR LOGIC INDICATIONS ARE NOT CONCLUSIVE ON FIRST PASS
** BECAUSE ERROR LOGIC HAS NOT BEEN CHECKED YET (ON FIRST PASS)

TST56: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
MOV #SECGAP,R0 ;POINTER
MOV #304.,R1 ;COUNTER
1\$: MOV #-1,(R0)+ ;CLEAR 'DISK' AREA TO ALL ONES
DEC R1 ;
BNE 1\$;
JSR PC,CLDISK ;THIS IS USED TO SET UP GENERAL
;REGISTER CORRESPONDENCE

*THESE ARE TO SET UP FOR DISKLESS USE ONLY

MOV #FMT22,@#WCYL ;FORMAT 22=16 BIT WORDS AND
;CYLINDER 0
CLR @#WSECTR ;TRACK=0, SECTOR=0
CLR @#WKEY1 ;KEY1=0
CLR @#WKEY2 ;KEY2=0
MOV #256.,@#FNWORD ;256 DATAWORDS
JSR R5,@#CRC ;GO TO CALCULATE CRC
WCYL
GCRC

*THESE ARE REGULAR SETUPS FOR RH11 & 'WRFROM' OUTPUT BUFFER

MOV #-260.,@RHWC ;256 DATA WORDS 4 HEADER WORDS
MOV #WRFROM,R0 ;BUS ADDRESS TO BE
MOV R0,@RHBA ;BUFFER 'WRFROM'
MOV #259.,R5 ;COUNTER
MOV #FMT22,(R0)+ ;FORMAT =16 BIT WORD
2\$: CLR (R0)+ ;SECTOR=0, TRACK=0,KEYS=0, ALL DATA=0
DEC R5 ;& CYLINDER=0....SO CLEAR ALL 'WRFROM'
BNE 2\$;CONTINUE IF ALL 259 NOT COMPLETE
CLR @RH DST ;TRACK=0, SECTOR=0

JSR PC,@#CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-

5547
5548
5549 025302 000207
5550

RTS PC

:ZEROED OUT
:259 TO 273 TOLERANCE GAP

```
5551
5552
5553      ;*****
5554      ;*TEST 57      WRITE HEADER AND DATA 2
5555      ;**      WRITE CYLINDERO, FORMAT 16 BITS PER WORD
5556      ;**      TRACK 0, SECTOR 1, KEYS 0, NUMBER OF WORDS 256
5557      ;**      OF ALL ONES.
5558      ;**      ANY ERROR LOGIC INDICATIONS ARE NOT CONCLUSIVE ON FIRST PASS
5559      ;**      BECAUSE ERROR LOGIC HAS NOT BEEN CHECKED YET (ON FIRST PASS)
5560
5561      ;*****
5562 025304 000004      TST57: SCOPE
5563 025306 012706 001000      MOV      #STACK,SP      ;RESET STACK
5564 025312 012737 000057 002032      MOV      #TTNO,@#TSTNM      ;THIS SAVES TEST NUMBER
5565
5566 025320 012700 051436      MOV      #SECGAP,RO      ;POINTER
5567 025324 012701 000460      MOV      #304.,R1      ;COUNTER
5568 025330 005020      1$: CLR      (RO)+      ;CLEAR SIMULATED "DISK" AREA IN CORE
5569 025332 005301      DEC      R1      ;
5570 025334 001375      BNE     1$      ;
5571 025336 004737 042732      JSR     PC,@#CLDISK      ;THIS IS USED TO SET GENERAL REGISTERS
5572
5573      ;*THESE ARE TO BE SETUP FOR DISKLESS USE ONLY
5574
5575 025342 012737 010000 052754      MOV     #FMT22,@#WCYL      ;FORMAT 22 = 16 BIT WORDS AND
5576      ;CYLINDER 0
5577 025350 012737 000001 052756      MOV     #1,@#WSECTR      ;TRACK=0, SECTOR=1
5578 025356 005037 052760      CLR     @#WKEY1      ;KEY1=0
5579 025362 005037 052762      CLR     @#WKEY2      ;KEY2=0
5580 025366 012737 000400 053014      MOV     #256.,@#FNWORD      ;256 DATA WORDS
5581 025374 004537 044140      JSR     R5,@#CRC      ;GO TO CALCULATE CRC
5582 025400 052754      WCYL
5583 025402 052764      GCRC
5584
5585      ;*THESE ARE REGULAR SETUPS FOR THE RH11 AND "WRFROM" BUFFER
5586
5587 025404 012777 177374 154220      MOV     #-260.,@RHWC      ;256 DATA WORDS 4 HEADER WORDS
5588 025412 012700 002114      MOV     #WRFROM,RO      ;THESE TWO INSTRUCTIONS GETS
5589 025416 010077 154212      MOV     RO,@RHBA      ;ADDR. OF WRFROM INTO RO AND
5590      ;BUS ADDRESS REGISTER
5591 025422 012720 010000      MOV     #FMT22,(RO)+      ;FORMAT=16 BIT WORDS
5592      ;CYLINDER=0
5593 025426 012720 000001      2$: MOV     #1,(RO)+      ;TRACK=0, SECTOR=1, KEYS=0
5594 025432 005020      CLR     (RO)+      ;KEY1=0
5595 025434 005020      CLR     (RO)+      ;KEY2=0
5596 025436 012705 000400      MOV     #256.,R5      ;COUNTER
5597 025442 012720 177777      3$: MOV     #-1,(RO)+      ;MOVE ALL ONES FOR DATA
5598 025446 005305      DEC     R5
5599 025450 001374      BNE     3$      ;BRANCH IF DATA NOT COMPLETE
5600 025452 012777 000001 154164      MOV     #1,@RHDST      ;TRACK=0 SECTOR=1
5601
5602 025460 004737 042766      JSR     PC,@#CHECKT      ;CHECK THAT DVA,RDY,DPR,DRY = 1
5603 025464 104401 062562      TYPE     ,CPHALT      ;AND THAT NO OTHERS = 1. CANNOT CON-
5604      ;TINUE TESTING IF BOTH AREN'T TRUE
5605 025470 000000      HALT      ;STOP THE TEST
5606
```


5663
5664
5665
5666
5667
5668
5669
5670
5671

025640 000207

RTS PC

:ZEROED
:WORD NOS 259
:IS DATA GAP
:WORD NOS 260 TO 273
:ARE TOLERANCE GAP
:RETURN TO COMPARE

```
5672
5673
5674
5675
5676
5677
5678
5679
5680
5681
5682 025642 000004
5683 025644 012706 001000
5684 025650 012737 000060 002032
5685
5686 025656 012700 051436
5687 025662 012701 000460
5688 025666 012720 000377
5689 025672 005301
5690 025674 001374
5691 025676 004737 042732
5692
5693
5694
5695
5696 025702 012737 010000 052754
5697
5698 025710 012737 000401 052756
5699 025716 005037 052760
5700 025722 005037 052762
5701 025726 012737 000400 053014
5702 025734 004537 044140
5703 025740 052754
5704 025742 052764
5705
5706
5707
5708 025744 012777 177374 153660
5709 025752 012700 002114
5710
5711 025756 010077 153652
5712
5713 025762 012720 010000
5714
5715 025766 012720 000401
5716 025772 005020
5717 025774 005020
5718 025776 012705 000400
5719 026002 012720 052525
5720 026006 005305
5721 026010 001374
5722 026012 012777 000401 153624
5723
5724 026020 004737 042766
5725 026024 104401 062562
5726
5727 026030 000000
```

```
*****
:*TEST 60 WRITE HEADER AND DATA 3
:** WRITE CYLINDER 0 FORMAT 16 BITS PER WORD
:** TRACK 1, SECTOR 1, KEY 0, NUMBER OF WORDS 256
:** ALTERNATE ONES AND ZEROS (052525)
:** ANY ERROR LOGIC INDICATIONS ARE NOT CONCLUSIVE ON FIRST PASS
:** BECAUSE ERROR LOGIC HAS NOT BEEN CHECKED YET (ON FIRST PASS)
*****
TST60: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
MOV #SECGAP,R0 ;POINTER
MOV #304.,R1 ;COUNTER
1$: MOV #377,(R0)+ ;LOAD SIMULATED "DISK" AREA WITH 377
DEC R1 ;
BNE 1$ ;
JSR PC,CLDISK ;THIS IS USED TO SET UP GENERAL
;REGISTER CORRESPONDENCE

:*THESE ARE TO BE SETUP FOR DISKLESS USE ONLY
MOV #FMT22,@#WCYL ;FORMAT 22=16 BIT WORDS AND
;CYLINDER 0
MOV #401,@#WSECTR ;TRACK=1, SECTOR=1
CLR @#WKEY1 ;KEY1=0
CLR @#WKEY2 ;KEY2=0
MOV #256.,@#FNWORD ;256 DATA WORDS
JSR R5,@#CRC ;GO TO CALCULATE CRC
WCYL
GCRC

:*THESE ARE REGULAR SETUPS FOR RH11 AND "WRFROM" BUFFER
MOV #-260.,@RHWC ;256 DATA WORDS 4 HEADER WORDS
MOV #WRFROM,R0 ;THESE TWO INSTRUCTIONS GET
;ADDR. OF WRFROM INTO R0
;AND BUS ADDRESS REGISTER
MOV R0,@RHBA
MOV #FMT22,(R0)+ ;FORMAT=16 BIT WORDS
;CYLINDER=0
2$: MOV #401,(R0)+ ;TRACK=1, SECTOR=1, KEYS=0
CLR (R0)+ ;KEY1=0
CLR (R0)+ ;KEY2=0
MOV #256.,R5 ;COUNTER
3$: MOV #052525,(R0)+ ;MOVE ALTERNATE ONES FOR DATA
DEC R5 ;COUNT
BNE 3$ ;BRANCH IF DATA NOT COMPLETE
MOV #401,@RHDST ;TRACK=1 SECTOR=1
JSR PC,@#CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-
;TINUE TESTING IF BOTH AREN'T TRUE
HALT ;STOP THE TEST
```

```

5728
5729 026032 013711 002066      MOV    @#WRIFOR,@R1      ;GET READY FOR WRITE HEADER
5730                                ;AND DATA WITH 62 IN RHCS1
5731 026036 005037 002006      CLR    @#ERFLG$          ;CLEAR ERROR FLAG
5732 026042 012777 010000 153600  MOV    #FMT22,@RHOF      ;FORMAT BIT=1(16 BIT WORDS
5733 026050 005077 153576      CLR    @RHCA             ;CYLINDER=0
5734 026054 004737 052600      JSR    PC,@#COMWHD       ;WRITE HEADER AND DATA INTO "DISK" CORE
5735
5736                                ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
5737                                ;*FROM THE "COMWHD" ROUTINE THAT MEANS ALL HEADER IN
5738                                ;*"DISK" IS GOOD. DATA IS TO BE CHECKED TO SEE
5739                                ;*IF IT IS ALL 052525 AND WRITE DATA GAP AND
5740                                ;*TOLERANCE GAP TO SEE IF THEY ARE ALL ZEROS.
5741
5742                                ;*RHWC IS CHECKED TO BE = 0
5743
5744 026060 017737 153546 001126  MOV    @RHWC,$BDDAT      ;LOAD AND TEST FOR ZERO
5745 026060 001401                BEQ    6$                ;RHWC SHOULD = 0
5746 026070 104040                ERROR  40                ;RHWC DID NOT = 0 AFTER A WRITE
5747                                ;HEADER AND DATA WAS COMPLETED
5748
5749                                ;*ONLY ECC1 AND ECC2 ARE NOT CHECKED
5750
5751 026072 005737 002006      6$:  TST    @#ERFLG$          ;HAVE ANY ERRORS OCCURED?
5752 026076 001041                BNE    TST61             ;;BRANCH IF YES
5753 026100 004737 043156      JSR    PC,@#CHECKE       ;CHECK THAT BITS = 1
5754 026104 104401 062562      TYPE   ,CPHALT           ;CANNOT CONTINUE TESTING IF THEY DON'T
5755 026110 000000                HALT                    ;STOP THE TEST
5756 026112 005037 052534      CLR    @#WECC1           ;CLEAR ECC
5757 026116 005037 052536      CLR    @#WECC2           ;CLEAR ECC
5758
5759                                ;*FILL "REINTO" BUFFER WITH EXPECTED DATA
5760
5761 026122 004037 042650      JSR    R0,@#CLAREA       ;FILL REINTO BUFFER
5762 026126 003160                REINTO                   ;FROM
5763 026130 004156                REINTO+<255.*2>         ;TO
5764 026132 052525                .WORD  52525             ;DATA
5765 026134 004037 042650      JSR    R0,@#CLAREA       ;FILL REST
5766 026140 004160                REINTO+<256.*2>         ;FROM
5767 026142 004220                REINTO+<272.*2>         ;TO
5768 026144 000000                .WORD  0                 ;DATA
5769 026146 005037 002006      CLR    @#ERFLG$          ;CLEAR ERROR FLAG
5770
5771                                ;*NOW COMPARE "DISK" BUFFER WITH "REINTO" BUFFER IN CORE
5772
5773 026152 004037 043626      JSR    R0,@#COMPAR       ;CHECK
5774 026156 003160                REINTO                   ;GOOD BUFFER
5775 026160 051534                DISK                      ;TEST BUFFER
5776 026162 000421                273.                     ;NUMBER OF WORDS CHECKED
5777 026164 026172                4$                       ;RETURN POINT FOR ERROR HEADER
5778 026166 026176                5$                       ;RETURN POINT FOR ERROR DATA
5779 026170 026202                TST61                    ;RETURN FOR GOOD COMPARISON
5780 026172 104007      4$:  ERROR  7                ;READ ERROR 10 NEXT
5781 026174 000207                RTS    PC                 ;RETURN TO COMPARE
5782 026176 104010      5$:  ERROR  10               ;WORD NOS 1 TO 256 ARE
5783                                ;DATA WORDS

```

5784
5785
5786
5787
5788
5789
5790
5791 026200 000207
5792
5793
5794
5795

RTS PC

;WORD NOS 257 AND 258
;ARE ECC WHICH HAVE BEEN
;ZEROED
;WORD NOS 259
;IS DATA GAP
;WORD NOS 260 TO 273
;ARE TOLERANCE GAP
;RETURN TO COMPARE

```
5796
5797
5798
5799
5800
5801
5802
5803
5804
5805
5806 026202 000004
5807 026204 012706 001000
5808 026210 012737 000061 002032
5809
5810 026216 012700 051436
5811 026222 012701 000460
5812 026226 012720 177777
5813 026232 005301
5814 026234 001374
5815 026236 004737 042732
5816
5817
5818
5819
5820 026242 012737 010000 052754
5821
5822 026250 005037 052756
5823 026254 005037 052760
5824 026260 005037 052762
5825 026264 012737 000400 053014
5826 026272 004537 044140
5827 026276 052754
5828 026300 052764
5829
5830
5831
5832 026302 012777 177374 153322
5833 026310 012700 002114
5834 026314 010077 153314
5835 026320 012705 000403
5836 026324 012720 010000
5837
5838 026330 005020
5839 026332 005305
5840 026334 001375
5841 026336 005077 153302
5842
5843 026342 004737 042766
5844 026346 104401 062562
5845
5846 026352 000000
5847
5848 026354 013711 002066
5849
5850 026360 005037 002006
5851 026364 012777 010000 153256
```

```
*****
*TEST 61 PROGRAM ERROR RHCS2 #10
** WRITE CYLINDER 0, FORMAT 16 BIT PER WORD
** TRACK 0, SECTOR 0, KEYS 0, NUMBER OF WORDS 256 OF 0'S
** WHILE GO BIT IS SET ANOTHER GO IS GIVEN THIS SHOULD SET
** PROGRAM ERROR
*****
1ST61: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER

MOV #SECGAP,R0 ;POINTER
MOV #304.,R1 ;COUNTER
1$: MOV #-1,(R0)+ ;CLEAR DISK AREA TO ALL ONES.
DEC R1 ;
BNE 1$ ;
JSR PC,CLDISK ;THIS IS USED TO SET GENERAL
;REGISTERS

;*THESE ARE TO SET UP FOR DISKLESS USE ONLY
MOV #FMT22,@#WCYL ;FORMAT 22=16 BITWORDS AND
;CYLINDER 0
CLR @#WSECTR ;TRACK=0, SECTOR=0
CLR @#WKEY1 ;KEY1=0
CLR @#WKEY2 ;KEY2=0
MOV #256.,@#FNWORD ;256 DATAWORDS
JSR R5,@#CRC ;GO TO CALCULATE CRC
WCYL
GCRC

;*THESE ARE REGULAR SETUPS
MOV #-260.,@RHWC ;256 DATA WORDS 4 HEADER WORDS
MOV #WRFROM,R0 ;FROM BUFFER 'WRFROM'
MOV R0,@RHBA ;IN BUS ADDRESS
MOV #259.,R5 ;COUNTER
MOV #FMT22,(R0)+ ;FORMAT =16 BIT WORD
;CYLINDER=0
2$: CLR (R0)+ ;SECTOR=0, TRACK=0,KEYS=0, ALL DATA=0
DEC R5 ;COUNT
BNE 2$ ;BRANCH IF ALL 259 NOT COMPLETE
CLR @RH DST ;TRACK=0, SECTOR=0

JSR PC,@#CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-
;TINUE TESTING IF BOTH AREN'T TRUE
HALT ;STOP THE TEST

MOV @#WRIFOR,@R1 ;GET READY FOR WRITE HEADER
;AND DATA WITH 62 IN RHCS1
CLR @#ERFLG$ ;CLEAR ERROR FLAG
MOV #FMT22,@RHOF ;FORMAT BIT=1,16 BIT WORDS
```

| | | | | | | | | |
|------|--------|--------|--------|--------|-------|-------------------|--|-------------------------------------|
| 5852 | 026372 | 005077 | 153254 | | CLR | @RHCA | | ;CYLINDER 0 |
| 5853 | 026376 | 012777 | 000001 | 153254 | MOV | #DMD,@RHMR | | ;SET DIAGNOSTIC MODE |
| 5854 | 026404 | 052777 | 000001 | 153226 | BIS | #GO,@RHCS1 | | ;GO |
| 5855 | 026412 | 000240 | | | NOP | | | |
| 5856 | 026414 | 052777 | 000001 | 153216 | BIS | #GO,@RHCS1 | | ;THIS GO SHOULD SET PGE |
| 5857 | | | | | | | | |
| 5858 | 026422 | 004737 | 042336 | | JSR | PC,@#PUTREG | | ;SAVE REGISTERS |
| 5859 | 026426 | 032737 | 002000 | 001714 | BIT | #PGE,@#CS1 | | ;IS PGE SET |
| 5860 | 026434 | 001404 | | | BEQ | 3\$ | | ;BRANCH IF GOOD |
| 5861 | 026436 | 013737 | 001636 | 001122 | MOV | @#RHCS2,@#\$BDADR | | |
| 5862 | 026444 | 104037 | | | ERROR | 37 | | ;PGE DID NOT SET WHEN A WRITE |
| 5863 | | | | | | | | ;WAS ATTEMPTED WITH ONE IN PROGRESS |
| 5864 | 026446 | | | | | | | |

3\$:

5865
5866
5867
5868
5869
5870
5871
5872
5873
5874
5875
5876
5877
5878
5879
5880
5881
5882
5883
5884
5885
5886

```
*****  
** THESE TESTS ARE THROUGH THE MAINTAINABILITY REGISTER - RHMR  
  
** THE SECTOR GAP AND SYNC BYTE ARE ALWAYS READ AS  
** ZEROS AND 144000 NO MATTER WHAT IS IN THE SIMULATED DISK AREA  
** TAGGED SECGAP: AND WSSYNC:  
  
** THE HEADER CONSISTING OF CYLINDER ADDRESS, SECTOR,  
** TRACK AND THE KEYS ARE READ FROM LOCATION  
** CYL:, SECTOR:, KEY1:, AND KEY2 AND NOT FROM  
** HEADER: ON SIMULATED DISK  
  
** CRC IS READ FROM SIMULATED DISK LOCATION WCRC:  
** HEADER GAP IS ALWAYS READ AS ZEROS NO MATTER  
** WHAT IS ON THE SIMULATED DISK AREA  
  
** THE DATA SYNC IS READ FROM HDWSYN:  
** ON SIMULATED DISK  
  
** ALL DATA IS READ FROM SIMULATED DISK DISK:  
*****
```

```
5887
5888
5889
5890
5891
5892      ;*****
5893      ;*TEST 62      READ HEADER AND DATA 1
5894      ;**      READ CYLINDER 0 FORMAT 16 BITS PER WORD
5895      ;**      TRACK 0, SECTOR 0, KEYS 0, 256 WORDS OF 0
5896      ;**      ANY ERROR LOGIC INDICATIONS ARE NOT CONCLUSIVE ON FIRST PASS
5897      ;**      BECAUSE ERROR LOGIC HAS NOT BEEN CHECKED YET (ON FIRST PASS)
5898      ;*****
5899      026446 000004      TST62: SCOPE
5900      026450 012706 001000      MOV      #STACK,SP      ;RESET STACK
5901      026454 012737 000062 002032      MOV      #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
5902
5903
5904      ;*      SETUP FOR WHAT IS TO BE READ
5905      ;*      HEADER CRC IS RESTORED FROM A SUBROUTINE
5906
5907      026462 012746 000000      MOV      #0,-(SP)      ;DATA TO BE READ
5908      026466 012705 000400      MOV      #256.,R5      ;COUNTER
5909      026472 012700 051534      MOV      #DISK,R0      ;START OF SIMULATED DISK DATA
5910      026476 011620      1$: MOV      (SP),(R0)+    ;MOVE IN DATA ON TO SIMULATED DISK
5911      026500 005305      DEC      R5      ;COUNT
5912      026502 001375      BNE     1$      ;BRANCH IF 256 NOT COMPLETE
5913      026504 005726      TST     (SP)+    ;UNDO -(SP)
5914      026506 012705 000021      MOV      #17.,R5      ;2 ECC WORDS
5915      ;1 DATA GAP
5916      ;14 TOLERANCE GAP
5917      026512 005020      2$: CLR      (R0)+    ;CLEAR ECC, DATA GAP, AND
5918      026514 005305      DEC      R5      ;TOLERANCE GAP
5919      026516 001375      BNE     2$      ;BRANCH IF NOT COMPLETE
5920
5921
5922      ;*THESE ARE TO SETUP FOR DISKLESS USE ONLY
5923
5924      026520 012737 010000 047616      MOV      #0!FMT22,@#CYL ;16 BITS PER WORD
5925      ;CYLINDER 0, FORMAT 16 BITS
5926      026526 112737 000000 047621      MOV      #0,@#SECOTR+1 ;TRACK 0
5927      026534 112737 000000 047620      MOV      #0,@#SECOTR   ;SECTOR 0
5928      026542 012737 000000 047622      MOV      #0,@#KEY1     ;KEY1=0
5929      026550 012737 000000 047624      MOV      #0,@#KEY2     ;KEY2=0
5930      026556 012737 000400 047676      MOV      #256.,@#DAWORD ;NO. OF DATA WORDS
5931      026564 005037 047626      CLR      @#X           ;THIS IS A READ COMMAND
5932      026570 004537 044140      JSR     R5,@#CRC      ;GO TO CALCULATE CRC
5933      026574 047616      CYL
5934      026576 051516      WCRC
5935
5936      ;*THESE ARE REGULAR SETUPS FOR RH11 AND "REINTO"
5937
5938      026600 004737 042732      JSR     PC,@#CLDISK   ;SETUP GENERAL REGISTERS
5939      026604 012777 177374 153020      MOV      #-256.-4.,@RHWC ;256. DATA 4 HEADER WORDS
5940      026612 012777 003160 153014      MOV      #REINTO,@RHBA ;STARTING ADDRESS OF READ BUFFER
5941      026620 112746 000000      MOV      #0,-(SP)     ;IN LOWER BYTE GET SECTOR
5942      026624 112766 000000 000001      MOV      #0,1(SP)    ;GET TRACK IN HIGHER BYTE
```



```

5943 026632 012677 153006      MOV    (SP)+,@RHDST    ;TRACK/SECTOR IN RHDST
5944 026636 012777 014000 153004  MOV    #FMT22!ECI,@RHOF ;16 BITS PER WORD
5945                                     ;ECC CORRECTION INHIBIT
5946                                     ;BECAUSE ECC IS NOT GOING
5947                                     ;TO BE CHECKED
5948 026644 005077 153002      CLR    @RHCA          ;CYLINDER 0
5949
5950 026650 004737 042766      JSR    PC,@#CHECKT    ;CHECK THAT DVA,RDY,DPR,DRY = 1
5951 026654 104401 062562      TYPE  ,CPHALT        ;AND THAT NO OTHERS = 1. CANNOT CON-
5952                                     ;TINUE TESTING IF BOTH AREN'T TRUE
5953 026660 000000      HALT                ;STOP THE TEST
5954
5955 026662 013711 002072      MOV    @#REFOR,@R1    ;READ HEADER AND DATA=72
5956 026666 005037 002006      CLR    @#ERFLG$      ;CLEAR ERROR FLAG
5957 026672 004737 047456      JSR    PC,@#COMHD     ;READ HEADER AND DATA
5958
5959
5960                                     ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
5961                                     ;*FROM THE "COMHD" ROUTINE THAT MEANS SECTOR GAP,
5962                                     ;*FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
5963                                     ;*SYNC BYTE HAVE GONE BY AND SYNCs WERE CORRECTLY
5964                                     ;*DETECTED.
5965
5966
5967                                     ;*RHWc IS CHECKED TO BE = 0 AFTER THE READ OPERATION
5968
5969 026676 017737 152730 001126  MOV    @RHWc,$BDDAT   ;LOAD AND TEST RHWc
5970 026704 001401      BEQ    20$            ;SHOULD = 0
5971 026706 104040      ERROR  40            ;RHWc DOES NOT = 0 AFTER A READ
5972                                     ;HEADER AND DATA
5973
5974                                     ;*HEADER AND DATA ARE TO BE CHECKED.
5975                                     ;*IN CHECKING READ DATA THE WRITE FROM BUFFER
5976                                     ;*"WRFROM" IS FILLED WITH EXPECTED DATA AND
5977                                     ;*COMPARISONS ARE MADE.
5978
5979 026710 005737 002006      20$: TST    @#ERFLG$      ;ANY ERRORS ALREADY THERE
5980 026714 001046      BNE    TST63         ;BRANCH IF YES
5981 026716 004737 043156      JSR    PC,@#CHECKE   ;CHECK THAT BITS = 1
5982 026722 104401 062562      TYPE  ,CPHALT        ;CANNOT CONTINUE TESTING IF THEY DON'T
5983 026726 000000      HALT                ;STOP THE TEST
5984 026730 012700 002114      MOV    #WRFROM,R0    ;GETTING READY TO FILL EXPECTED DATA
5985 026734 012720 010000      MOV    #0!FMT22,(R0)+ ;CYLINDER 0
5986 026740 112746 000000      MOVB  #0,-(SP)       ;IN LOWER BYTE GET SECTOR
5987 026744 112766 000000 000001  MOVB  #0,1(SP)       ;GET TRACK IN HIGHER BYTE
5988 026752 012620      MOV    (SP)+,(R0)+   ;GET TRACK/SECTOR IN BUFFER
5989 026754 012720 000000      MOV    #0,(R0)+     ;KEY1 IN BUFFER
5990 026760 012720 000000      MOV    #0,(R0)+     ;KEY2 IN BUFFER
5991 026764 012701 000400      MOV    #256.,R1     ;DATA WORD COUNTER
5992 026770 012702 000000      MOV    #0,R2        ;DATA
5993 026774 010220      3$: MOV    R2,(R0)+     ;DATA INTO BUFFER
5994 026776 005301      DEC    R1            ;COUNT
5995 027000 001375      BNE    3$           ;BRANCH IF 256 NOT DONE
5996
5997                                     ;*NOW READ DATA BUFFER WILL BE CHECKED
5998

```

| | | | | | | |
|------|--------|--------|--------|--------|-------------|--------------------------------|
| 5999 | 027002 | 004037 | 043626 | JSR | RO,@#COMPAR | :CHECK |
| 6000 | 027006 | 002114 | | WRFROM | | :GOOD BUFFER |
| 6001 | 027010 | 003160 | | REINTO | | :TEST BUFFER |
| 6002 | 027012 | 000404 | | 4+256. | | :NUMBER OF WORDS CHECKED |
| 6003 | 027014 | 027022 | | 4\$ | | :RETURN POINT FOR ERROR HEADER |
| 6004 | 027016 | 027026 | | 5\$ | | :RETURN POINT FOR ERROR DATA |
| 6005 | 027020 | 027032 | | TST63 | | :RETURN FOR GOOD COMPARISON |
| 6006 | 027022 | 104004 | 4\$: | ERROR | 4 | :READ NEXT ERROR |
| 6007 | 027024 | 000207 | | RTS | PC | :RETURN TO "COMPAR" |
| 6008 | 027026 | 104005 | 5\$: | ERROR | 5 | :WORD NOS 1 TO 4 ARE |
| 6009 | | | | | | :HEADER WORDS |
| 6010 | | | | | | :5 TO 260 ARE DATA WORDS |
| 6011 | 027030 | 000207 | | RTS | PC | :RETURN TO "COMPAR" |
| 6012 | | | | | | |
| 6013 | | | | | | |
| 6014 | | | | | | |
| 6015 | | | | | | |
| 6016 | | | | | | |

```
6017
6018
6019
6020
6021
6022
6023
6024
6025
6026 027032 000004
6027 027034 012706 001000
6028 027040 012737 000063 002032
6029
6030
6031
6032
6033
6034 027046 012746 177777
6035 027052 012705 000400
6036 027056 012700 051534
6037 027062 011620
6038 027064 005305
6039 027066 001375
6040 027070 005726
6041 027072 012705 000021
6042
6043
6044 027076 005020
6045 027100 005305
6046 027102 001375
6047
6048
6049
6050
6051 027104 012737 010000 047616
6052
6053 027112 112737 000000 047621
6054 027120 112737 000001 047620
6055 027126 012737 000000 047622
6056 027134 012737 000000 047624
6057 027142 012737 000400 047676
6058 027150 005037 047626
6059 027154 004537 044140
6060 027160 047616
6061 027162 051516
6062
6063
6064
6065 027164 004737 042732
6066 027170 012777 177374 152434
6067 027176 012777 003160 152430
6068 027204 112746 000001
6069 027210 112766 000000 000001
6070 027216 012677 152422
6071 027222 012777 014000 152420
6072
```

```
*****
*TEST 63 READ HEADER AND DATA 2
** READ CYLINDER 0 FORMAT 16 BITS PER WORD
** TRACK 0, SECTOR 1, KEYS 0, 256 WORDS OF 177777
** ANY ERROR LOGIC INDICATIONS ARE NOT CONCLUSIVE ON FIRST PASS
** BECAUSE ERROR LOGIC HAS NOT BEEN CHECKED YET (ON FIRST PASS)
*****
TST63: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER

;* SETUP FOR WHAT IS TO BE READ
;* HEADER CRC IS RESTORED FROM A SUBROUTINE
MOV #-1,-(SP) ;DATA TO BE READ
MOV #256.,R5 ;COUNTER
MOV #DISK,R0 ;START OF SIMULATED DISK DATA
1$: MOV (SP),(R0)+ ;MOVE IN DATA ON TO SIMULATED DISK
DEC R5 ;COUNT
BNE 1$ ;BRANCH IF 256 NOT COMPLETE
TST (SP)+ ;UNDO -(SP)
MOV #17.,R5 ;2 ECC WORDS
;1 DATA GAP
;14 TOLERANCE GAP
2$: CLR (R0)+ ;CLEAR ECC, DATA GAP, AND
DEC R5 ;TOLERANCE GAP
BNE 2$ ;BRANCH IF NOT COMPLETE

;*THESE ARE TO SETUP FOR DISKLESS USE ONLY
MOV #0!FMT22,@#CYL ;16 BITS PER WORD
;CYLINDER 0, FORMAT 16 BITS
MOVB #0,@#SECOTR+1 ;TRACK 0
MOVB #1,@#SECOTR ;SECTOR 1
MOV #0,@#KEY1 ;KEY1=0
MOV #0,@#KEY2 ;KEY2=0
MOV #256.,@#DAWORD ;NO. OF DATA WORDS
CLR @#X ;THIS IS A READ COMMAND
JSR R5,@#CRC ;GO TO CALCULATE CRC
CYL
WCRC

;*THESE ARE REGULAR SETUPS FOR RH11 AND "REINTO"
JSR PC,@#CLDISK ;SETUP GENERAL REGISTERS
MOV #-256.-4.,@RHWC ;256. DATA 4 HEADER WORDS
MOV #REINTO,@RHBA ;STARTING ADDRESS OF READ BUFFER
MOVB #1,-(SP) ;IN LOWER BYTE GET SECTOR
MOVB #0,1(SP) ;GET TRACK IN HIGHER BYTE
MOV (SP)+,@RHDST ;TRACK/SECTOR IN RHDST
MOV #FMT22!EC1,@RHOF ;16 BITS PER WORD
;ECC CORRECTION INHIBIT
```

```
6073 ;BECAUSE ECC IS NOT GOING
6074 ;TO BE CHECKED
6075 027230 005077 152416 CLR @RHCA ;CYLINDER 0
6076
6077 027234 004737 042766 JSR PC,@#CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
6078 027240 104401 062562 TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-
6079 ;TINUE TESTING IF BOTH AREN'T TRUE
6080 027244 000000 HALT ;STOP THE TEST
6081
6082 027246 013711 002072 MOV @#REFOR,@R1 ;READ HEADER AND DATA=72
6083 027252 005037 002006 CLR @#ERFLG$ ;CLEAR ERROR FLAG
6084 027256 004737 047456 JSR PC,@#COMHD ;READ HEADER AND DATA
6085
6086
6087 ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
6088 ;*FROM THE "COMHD" ROUTINE THAT MEANS SECTOR GAP,
6089 ;*FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
6090 ;*SYNC BYTE HAVE GONE BY AND SYNCs WERE CORRECTLY
6091 ;*DETECTED.
6092
6093
6094 ;*RHWC IS CHECKED TO BE = 0 AFTER THE READ OPERATION
6095
6096 027262 017737 152344 001126 MOV @RHWC,$BDDAT ;LOAD AND TEST RHWC
6097 027270 001401 BEQ 20$ ;SHOULD = 0
6098 027272 104040 ERROR 40 ;RHWC DOES NOT = 0 AFTER A READ
6099 ;HEADER AND DATA
6100
6101 ;*HEADER AND DATA ARE TO BE CHECKED.
6102 ;*IN CHECKING READ DATA THE WRITE FROM BUFFER
6103 ;*"WRFROM" IS FILLED WITH EXPECTED DATA AND
6104 ;*COMPARISONS ARE MADE.
6105
6106 027274 005737 002006 20$: TST @#ERFLG$ ;ANY ERRORS ALREADY THERE
6107 027300 001046 BNE TST64 ;BRANCH IF YES
6108 027302 004737 043156 JSR PC,@#CHECKE ;CHECK THAT BITS = 1
6109 027306 104401 062562 TYPE ,CPHALT ;CANNOT CONTINUE TESTING IF THEY DON'T
6110 027312 000000 HALT ;STOP THE TEST
6111 027314 012700 002114 MOV #WRFROM,R0 ;GETTING READY TO FILL EXPECTED DATA
6112 027320 012720 010000 MOV #0:FMT22,(R0)+ ;CYLINDER 0
6113 027324 112746 000001 MOV #1,-(SP) ;IN LOWER BYTE GET SECTOR
6114 027330 112766 000000 000001 MOV #0,1(SP) ;GET TRACK IN HIGHER BYTE
6115 027336 012620 MOV (SP)+,(R0)+ ;GET TRACK/SECTOR IN BUFFER
6116 027340 012720 000000 MOV #0,(R0)+ ;KEY1 IN BUFFER
6117 027344 012720 000000 MOV #0,(R0)+ ;KEY2 IN BUFFER
6118 027350 012701 000400 MOV #256,R1 ;DATA WORD COUNTER
6119 027354 012702 177777 MOV #-1,R2 ;DATA
6120 027360 010220 3$: MOV R2,(R0)+ ;DATA INTO BUFFER
6121 027362 005301 DEC R1 ;COUNT
6122 027364 001375 BNE 3$ ;BRANCH IF 256 NOT DONE
6123
6124 ;*NOW READ DATA BUFFER WILL BE CHECKED
6125
6126 027366 004037 043626 JSR R0,@#COMPAR ;CHECK
6127 027372 002114 WRFROM ;GOOD BUFFER
6128 027374 003160 REINTO ;TEST BUFFER
```

6129 027376 000404
6130 027400 027406
6131 027402 027412
6132 027404 027416
6133 027406 104004
6134 027410 000207
6135 027412 104005
6136
6137
6138 027414 000207
6139
6140
6141
6142

4+256.
4\$
5\$
TST64
4\$: ERROR 4
RTS PC
5\$: ERROR 5
RTS PC

:NUMBER OF WORDS CHECKED
:RETURN POINT FOR ERROR HEADER
:RETURN POINT FOR ERROR DATA
:RETURN FOR GOOD COMPARISON
:READ NEXT ERROR
:RETURN TO "COMPAR"
:WORD NOS 1 TO 4 ARE
:HEADER WORDS
:5 TO 260 ARE DATA WORDS
:RETURN TO "COMPAR"

```
6143
6144
6145
6146
6147
6148
6149
6150
6151
6152 027416 000004
6153 027420 012706 001000
6154 027424 012737 000064 002032
6155
6156
6157
6158
6159
6160 027432 012746 052525
6161 027436 012705 000400
6162 027442 012700 051534
6163 027446 011620
6164 027450 005305
6165 027452 001375
6166 027454 005726
6167 027456 012705 000021
6168
6169
6170 027462 005020
6171 027464 005305
6172 027466 001375
6173
6174
6175
6176
6177 027470 012737 010000 047616
6178
6179 027476 112737 000001 047621
6180 027504 112737 000001 047620
6181 027512 012737 000000 047622
6182 027520 012737 000000 047624
6183 027526 012737 000400 047676
6184 027534 005037 047626
6185 027540 004537 044140
6186 027544 047616
6187 027546 051516
6188
6189
6190
6191 027550 004737 042732
6192 027554 012777 177374 152050
6193 027562 012777 003160 152044
6194 027570 112746 000001
6195 027574 112766 000001 000001
6196 027602 012677 152036
6197 027606 012777 014000 152034
6198
```

```
*****
;*TEST 64 READ HEADER AND DATA 3
** READ CYLINDER 0 FORMAT 16 BITS PER WORD
** TRACK 1, SECTOR 1, KEYS 0, 256 WORDS OF 052525
** ANY ERROR LOGIC INDICATIONS ARE NOT CONCLUSIVE ON FIRST PASS
** BECAUSE ERROR LOGIC HAS NOT BEEN CHECKED YET (ON FIRST PASS)
*****
TST64: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER

;* SETUP FOR WHAT IS TO BE READ
;* HEADER CRC IS RESTORED FROM A SUBROUTINE

MOV #052525,-(SP) ;DATA TO BE READ
MOV #256.,R5 ;COUNTER
MOV #DISK,R0 ;START OF SIMULATED DISK DATA
1$: MOV (SP),(R0)+ ;MOVE IN DATA ON TO SIMULATED DISK
DEC R5 ;COUNT
BNE 1$ ;BRANCH IF 256 NOT COMPLETE
TST (SP)+ ;UNDO -(SP)
MOV #17.,R5 ;2 ECC WORDS
;1 DATA GAP
;14 TOLERANCE GAP
2$: CLR (R0)+ ;CLEAR ECC, DATA GAP, AND
DEC R5 ;TOLERANCE GAP
BNE 2$ ;BRANCH IF NOT COMPLETE

;*THESE ARE TO SETUP FOR DISKLESS USE ONLY
MOV #0!FMT22,@#CYL ;16 BITS PER WORD
;CYLINDER 0, FORMAT 16 BITS
MOVB #1,@#SECOTR+1 ;TRACK 1
MOVB #1,@#SECOTR ;SECTOR 1
MOV #0,@#KEY1 ;KEY1=0
MOV #0,@#KEY2 ;KEY2=0
MOV #256.,@#DAWORD ;NO. OF DATA WORDS
CLR @#X ;THIS IS A READ COMMAND
JSR R5,@#CRC ;GO TO CALCULATE CRC
CYL
WCRC

;*THESE ARE REGULAR SETUPS FOR RH11 AND "REINTO"
JSR PC,@#CLDISK ;SETUP GENERAL REGISTERS
MOV #-256.-4.,@RHWC ;256. DATA 4 HEADER WORDS
MOV #REINTO,@RHBA ;STARTING ADDRESS OF READ BUFFER
MOVB #1,-(SP) ;IN LOWER BYTE GET SECTOR
MOVB #1,1(SP) ;GET TRACK IN HIGHER BYTE
MOV (SP)+,@RHDST ;TRACK/SECTOR IN RHDST
MOV #FMT22!ECI,@RHOF ;16 BITS PER WORD
;ECC CORRECTION INHIBIT
```

```
6199 ;BECAUSE ECC IS NOT GOING
6200 ;TO BE CHECKED
6201 027614 005077 152032 CLR @RHCA ;CYLINDER 0
6202
6203 027620 004737 042766 JSR PC,@#CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
6204 027624 104401 062562 TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-
6205 ;TINUE TESTING IF BOTH AREN'T TRUE
6206 027630 000000 HALT ;STOP THE TEST
6207
6208 027632 013711 002072 MOV @#REFOR,@R1 ;READ HEADER AND DATA=72
6209 027636 005037 002006 CLR @#ERFLG$ ;CLEAR ERROR FLAG
6210 027642 004737 047456 JSR PC,@#COMHD ;READ HEADER AND DATA
6211
6212
6213 ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
6214 ;*FROM THE "COMHD" ROUTINE THAT MEANS SECTOR GAP,
6215 ;*FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
6216 ;*SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
6217 ;*DETECTED.
6218
6219
6220 ;*RHWC IS CHECKED TO BE = 0 AFTER THE READ OPERATION
6221
6222 027646 017737 151760 001126 MOV @RHWC,$BDDAT ;LOAD AND TEST RHWC
6223 027654 001401 BEQ 20$ ;SHOULD = 0
6224 027656 104040 ERROR 40 ;RHWC DOES NOT = 0 AFTER A READ
6225 ;HEADER AND DATA
6226
6227 ;*HEADER AND DATA ARE TO BE CHECKED.
6228 ;*IN CHECKING READ DATA THE WRITE FROM BUFFER
6229 ;*"WRFROM" IS FILLED WITH EXPECTED DATA AND
6230 ;*COMPARISONS ARE MADE.
6231
6232 027660 005737 002006 20$: TST @#ERFLG$ ;ANY ERRORS ALREADY THERE
6233 027664 001046 BNE TST65 ;BRANCH IF YES
6234 027666 004737 043156 JSR PC,@#CHECKE ;CHECK THAT BITS = 1
6235 027672 104401 062562 TYPE ,CPHALT ;CANNOT CONTINUE TESTING IF THEY DON'T
6236 027676 000000 HALT ;STOP THE TEST
6237 027700 012700 002114 MOV #WRFROM,R0 ;GETTING READY TO FILL EXPECTED DATA
6238 027704 012720 010000 MOV #0!FMT22,(R0)+ ;CYLINDER 0
6239 027710 112746 000001 MOVB #1,-(SP) ;IN LOWER BYTE GET SECTOR
6240 027714 112766 000001 000001 MOVB #1,1(SP) ;GET TRACK IN HIGHER BYTE
6241 027722 012620 MOV (SP)+,(R0)+ ;GET TRACK/SECTOR IN BUFFER
6242 027724 012720 000000 MOV #0,(R0)+ ;KEY1 IN BUFFER
6243 027730 012720 000000 MOV #0,(R0)+ ;KEY2 IN BUFFER
6244 027734 012701 000400 MOV #256,R1 ;DATA WORD COUNTER
6245 027740 012702 052525 MOV #052525,R2 ;DATA
6246 027744 010220 3$: MOV R2,(R0)+ ;DATA INTO BUFFER
6247 027746 005301 DEC R1 ;COUNT
6248 027750 001375 BNE 3$ ;BRANCH IF 256 NOT DONE
6249
6250 ;*NOW READ DATA BUFFER WILL BE CHECKED
6251
6252 027752 004037 043626 JSR R0,@#COMPAR ;CHECK
6253 027756 002114 WRFROM ;GOOD BUFFER
6254 027760 003160 REINTO ;TEST BUFFER
```

6255 027762 000404
6256 027764 027772
6257 027766 027776
6258 027770 030002
6259 027772 104004
6260 027774 000207
6261 027776 104005
6262
6263
6264 030000 000207
6265
6266
6267
6268

4+256.
4\$
5\$
TST65
4\$: ERROR 4
RTS PC
5\$: ERROR 5
RTS PC

:NUMBER OF WORDS CHECKED
:RETURN POINT FOR ERROR HEADER
:RETURN POINT FOR ERROR DATA
:RETURN FOR GOOD COMPARISON
:READ NEXT ERROR
:RETURN TO "COMPAR"
:WORD NOS 1 TO 4 ARE
:HEADER WORDS
:5 TO 260 ARE DATA WORDS
:RETURN TO "COMPAR"


```
6269
6270
6271
6272
6273
6274
6275
6276
6277
6278
6279 030002 000004
6280
6281 030004 012737 000065 002032
6282
6283
6284 030012 012706 001000
6285 030016 012737 000065 002032
6286
6287 030024 004037 042650
6288 030030 051534
6289 030032 052560
6290 030034 000000
6291
6292
6293
6294 030036 012737 010000 047616
6295
6296 030044 112737 000000 047621
6297 030052 112737 000000 047620
6298 030060 005037 047622
6299 030064 005037 047624
6300 030070 012737 000400 047664
6301 030076 012737 000001 047626
6302 030104 004537 044140
6303 030110 047616
6304 030112 051516
6305
6306
6307
6308 030114 004037 042650
6309 030120 002114
6310 030122 003114
6311 030124 000377
6312 030126 004737 042732
6313 030132 012777 177400 151472
6314 030140 012777 002114 151466
6315 030146 012746 000000
6316 030152 112766 000000 000001
6317 030160 012677 151460
6318 030164 012777 010000 151456
6319 030172 012777 000000 151452
6320 030200 004737 042766
6321 030204 104401 062562
6322
6323 030210 000000
6324 030212 013711 002064
```

```
*****
;TEST 65 WRITE DATA
;
;** WRITE CYLINDER 0, FORMAT 16 BITS PER WORD
;** TRACK 0, SECTOR 0, KEYS 0, NUMBER OF WORDS 256 OF 377
;** ANY ERROR LOGIC INDICATION IS NOT CONCLUSIVE ON FIRST PASS
;** BECAUSE ERROR LOGIC HAS NOT BEEN CHECKED
*****
TST65: SCOPE
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
JSR RO,@#CLAREA ;CLEAR SIMULATED DISK
.WORD DISK ;FROM
.WORD TOLGAP+16 ;TO
.WORD 0 ;DATA
;*THESE ARE SETUP FOR DISKLESS USE ONLY
MOV #0!FMT22,@#CYL;CYLINDER 0
;16 BITS PER WORD
MOVB #0,@#SECOTR+1;TRACK 0
MOVB #0,@#SECOTR ;SECTOR 0
CLR @#KEY1 ;KEY1 0
CLR @#KEY2 ;KEY2 0
MOV #256,@#NOWORD ;NO OF DATA WORDS
MOV #1,@#X ;WRITE DATA
JSR R5,@#CRC ;GO TO CALCULATE CRC
CYL
WCRC
;*THESE ARE REGULAR SETUPS
JSR RO,@#CLAREA ;FILL WRITE BUFFER WITH 377
WRFROM ;FROM LOCATION
WRFROM+<256.*2> ;TO LOCATION
377 ;DATA
JSR PC,@#CLDISK ;SETUP GENERAL REGISTERS
MOV #-256,@#RHWC ;256. DATA WORDS
MOV #WRFROM,@#RHBA ;STARTING ADDRESS OF WRITE BUFFER
MOV #0,-(SP) ;SECTOR 0
MOVB #0,1(SP) ;TRACK 0
MOV (SP)+,@#RHDST ;SECTOR 0 TRACK 0
MOV #FMT22,@#RHOF ;16 BITS PER WORD FORMAT
MOV #0,@#RHCA ;CYLINDER 0
JSR PC,@#CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-
;TINUE TESTING IF BOTH AREN'T TRUE
HALT ;STOP THE TEST
MOV @#WRIDAT,@R1 ;WRITE DATA=60
```

6325 030216 005037 002006
6326 030222 004737 047456
6327
6328
6329
6330
6331
6332
6333
6334 030226 004737 042336
6335 030232 005737 002006
6336 030236 001041
6337 030240 012700 000377
6338 030244 012701 051534
6339 030250 012702 000400
6340
6341 030254 012737 000401 047736 1\$:
6342 030262 020021
6343 030264 001424
6344 030266 010037 001124
6345 030272 014137 001126
6346 030276 160237 047736
6347 030302 005737 002006
6348 030306 001002
6349 030310 104004
6350 030312 000401
6351
6352 030314 104005 2\$:
6353 030316 005721 64\$:
6354 030320 017746 150614
6355 030324 042716 177177
6356 030330 022726 000200
6357 030334 001402
6358 030336 005302 3\$:
6359 030340 001345
6360
6361
6362
6363

CLR @#ERFLG\$;CLEAR ERROR FLAG
JSR PC,@#COMHD ;WRITE DATA

;*IF THE PROGRAM COMES BACY HERE WITHOUT ERROR PRINTOUTS
;*FROM THE "COMHD" ROUTINE IT MEANS SECTOR GAP, SYNC BYTE
;*HEADER, HEADER CRC, HEADER GAP AND SYNC BYTE HAVE GONE BY
;*AND SYNCs WERE CORRECTLY DETECTED, DATA IS TO BE CHECKED.

JSR PC,@#PUTREG ;SAVE REGISTERS
TST @#ERFLG\$;HAVE ANY ERRORS OCCURED?
BNE TST66 ;BRANCH IF YES
MOV #377,R0 ;GOOD DATA
MOV #DISK,R1 ;DATA WRITTEN INTO "DISK"
MOV #256.,R2 ;COUNTER
MOV #256.+1,@#ERWORD ;FOR ERROR WORD
CMP R0,(R1)+ ;COMPARE GOOD DATA WITH DATA ON DISK
BEQ 3\$;BRANCH IF GOOD
MOV R0,@#\$GDDAT ;GOOD DATA
MOV -(R1),@#\$BDDAT ;BAD DATA
SUB R2,@#ERWORD ;ERROR WORD NO
TST @#ERFLG\$;ANY ERRORS ALREADY THERE?
BNE 2\$;BRANCH IF YES
ERROR 4 ;ERROR ON WRITE DATA COMMAND
BR 64\$;BRANCH TO AVOID PRINTING NEXT ERROR
ERROR 5 ;WORD NO GIVES WORD IN ERROR
TST (R1)+ ;UNDO -(R1) FOR BAD DATA
MOV @SWR,-(SP) ;GET SWITCH SETTING
BIC #177177,(SP) ;KEEP ONLY SWITCH 7 AND 8
CMP #SW07,(SP)+ ;IS 7 SET AND 8 RESET
BEQ TST66 ;BRANCH OUT IF YES
DEC R2 ;IF NOT COUNT 256 WORDS
BNE 1\$;BRANCH IF 256. NOT DONE

```
6364
6365
6366
6367
6368
6369
6370
6371
6372
6373
6374 030342 000004
6375 030344 012706 001000
6376
6377 030350 012737 000066 002032
6378 030356 004037 042650
6379 030362 051534
6380 030364 052532
6381 030366 177400
6382
6383 030370 004037 042650
6384 030374 003160
6385 030376 004156
6386 030400 000000
6387
6388
6389
6390 030402 012737 010000 047616
6391 030410 105037 047621
6392 030414 112737 000001 047620
6393 030422 005037 047622
6394 030426 005037 047624
6395 030432 012737 000012 047676
6396 030440 005037 047626
6397 030444 004537 044140
6398 030450 047616
6399 030452 051516
6400
6401
6402
6403 030454 004737 042732
6404 030460 013711 002070
6405 030464 012777 177766 151140
6406 030472 012777 003160 151134
6407 030500 112746 000001
6408 030504 112766 000000 000001
6409 030512 012677 151126
6410 030516 012777 014000 151124
6411
6412
6413 030524 005077 151122
6414 030530 004737 042766
6415 030534 104401 062562
6416
6417 030540 000000
6418 030542 005037 002006
6419 030546 004737 047456

:*****
:*TEST 66 READ DATA
:** READ CYLINDERO, FORMAT 16 BITS PER WORD
:** TRACK0, SECTOR 1, KEYS 0, 10 WORDS OF 177400
:** ANY ERROR LOGIC INDICATION IS NOT CONCLUSIVE
:** BECAUSE ERROR LOGIC HAS NOT BEEN CHECKED YET (ON FIRST PASS)
:*****
1ST66: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
JSR RO,@#CLAREA ;CLEAR SIMULATED DISK
.WORD DISK ;FROM
.WORD DISK+776 ;TO
.WORD 177400 ;DATA
JSR RO,@#CLAREA ;CLEAR READ INTO BUFFER
.WORD REINTO ;FROM
.WORD REINTO+776 ;TO
.WORD 0 ;DATA
:*THESE ARE TO SETUP FOR DISKLESS USE ONLY
MOV #FMT22,@#CYL ;CYLINDER 0 16 BITS PER WORD FORMAT
CLRB @#SECOTR+1 ;TRACK 0
MOV #1,@#SECOTR ;SECTOR 1
CLR @#KEY1 ;KEY1=0
CLR @#KEY2 ;KEY2=0
MOV #10,@#DAWORD ;NO. OF DATA WORDS
CLR @#X ;THIS IS A READ COMMAND
JSR R5,@#CRC ;GO TO CALCULATE CRC
CYL
WCRC
:*THESE ARE REGULAR SETUPS
JSR PC,@#CLDISK ;SETUP GENERAL REGISTERS
MOV @#READAT,@R1 ;READ DATA INTO RHCS1=70
MOV #-10,@#RHWC ;10 DATA WORDS
MOV @#REINTO,@#RHBA ;STARTING ADDRESS OF READ BUFFER
MOVB #1,-(SP) ;IN LOWER BYTE GET SECTOR 1
MOVB #0,1(SP) ;GET TRACK0 IN UPPER BYTE
MOV (SP)+,@#RHDST ;TRACK/SECTOR IN RHDST
MOV #FMT22!ECI,@#RHOF ;16 BITS PER WORD
;ECC CORRECTION INHIBIT BECAUSE
;ECC IS NOT CHECKED HERE
CLR @#RHCA ;CYLINDER 0
JSR PC,@#CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-
;TINUE TESTING IF BOTH AREN'T TRUE
HALT ;STOP THE TEST
CLR @#ERFLG$ ;CLEAR ERROR FLAG
JSR PC,@#COMHD ;READ DATA
```

```
6420
6421 ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUT
6422 ;*FROM "COMHD" ROUTINE IN MEANS DATA IS TO BE CHECKED
6423
6424 ;*NOW THE DATA READ INTO "REINTO" BUFFER WILL
6425 ;*BE CHECKED, ONLY 10 WORDS SHOULD BE CHANGED
6426 ;*ALL OTHER WORDS SHOULD REMAIN UNCHANGED
6427 ;*THE "WRFROM" BUFFER IS FILLED WITH EXPECTED DATA AND CHECKED
6428
6429 030552 005737 002006 TST @#ERFLG$ ;HAVE ANY ERRORS OCCURED?
6430 030556 001053 BNE TST67 ;:BRANCH IF YES
6431 030560 004037 042650 JSR RO,@#CLAREA ;CLEAR BUFFER
6432 030564 002114 WRFROM ;FROM
6433 030566 003112 WRFROM+776 ;TO
6434 030570 000000 0 ;DATA
6435
6436 030572 004037 042650 JSR RO,@#CLAREA ;FILL EXPECTED DATA
6437 030576 002114 WRFROM ;FROM
6438 030600 002136 WRFROM+22 ;TO
6439 030602 177400 177400 ;DATA
6440
6441 ;*NOW READ DATA BUFFER IS CHECKED
6442
6443 030604 012700 002114 MOV #WRFROM,R0 ;GOOD DATA
6444 030610 012701 003160 MOV #REINTO,R1 ;DATA READ
6445 030614 012702 000400 MOV #256.,R2 ;COUNTER
6446 030620 012737 000401 047736 1$: MOV #257.,@#ERWORD ;FOR ERROR WORD NO
6447 030626 022021 CMP (R0)+,(R1)+ ;COMPARE GOOD WITH READ BUFFER
6448 030630 001424 BEQ 2$ ;BRANCH IF GOOD
6449 030632 014037 001124 MOV -(R0),@#SGDDAT ;GOOD DATA
6450 030636 014137 001126 MOV -(R1),@#SBDDAT ;BAD DATA
6451 030642 160237 047736 SUB R2,@#ERWORD ;ERROR WORD NO
6452 030646 005737 002006 TST @#ERFLG$ ;ANY ERRORS ALREADY THERE
6453 030652 001002 BNE 3$ ;IF YES BRANCH DO NOT TYPE HEADER
6454 030654 104004 ERROR 4 ;ERROR ON READ DATA
6455 030656 000401 BR 4$ ;BRANCH TO AVOID PRINTING NEXT ERROR
6456 030660 104005 3$: ERROR 5 ;WORD NO 1-10 ARE DATA
6457 ;WORDS
6458 ;WORD NOS 11-256 HAVE NOT BEEN
6459 ;READ AND BUFFER SHOULD BE
6460 ;ZERO IF OTHER THAN ZERO
6461 ;WRONG NUMBER OF WORDS HAVE
6462 ;BEEN READ IN THE DISK NOW
6463 ;CONTAINS 177400 ALL 256
6464 ;WORDS BUT ONLY 10 WORDS
6465 ;SHOULD BE READ IN
6466
6467 030662 022021 4$: CMP (R0)+,(R1)+ ;UNDO -(R0) AND -(R1) FOR ERROR
6468 030664 017746 150250 MOV @SWR,-(SP) ;GET SWITCH SETTING
6469 030670 042716 177177 BIC #177177,(SP) ;KEEP ONLY SWITCH 7 AND 8
6470 030674 022726 000200 CMP #SW07,(SP)+ ;IS 7 SET AND 8 RESET
6471 030700 001402 BEQ TST67 ;BRANCH OUT IF YES
6472 030702 005302 2$: DEC R2 ;COUNT
6473 030704 001345 BNE 1$ ;BRANCH IF NOT COMPLETE
6474
6475
```

CZRJGDO,RP04/5/6 DSKLS CTRLR1
CZRJGD.P11 10-SEP-79 11:00

MACY11 30A(1052) 10-SEP-79 11:11 E 12
T66 READ DATA PAGE 148

SEQ 0147

6476

6477
6478
6479
6480
6481
6482
6483
6484
6485
6486
6487
6488
6489
6490
6491
6492
6493
6494
6495
6496
6497
6498
6499
6500
6501
6502
6503
6504
6505
6506
6507
6508
6509
6510
6511
6512
6513
6514
6515
6516
6517
6518
6519
6520
6521
6522
6523
6524
6525
6526
6527
6528
6529
6530
6531
6532

030706 000004

030710 012706 001000
030714 012737 000067 002032

030722 012701 003160
030726 012721 010000
030732 012721 000401
030736 005021
030740 005021
030742 004037 042650
030746 003170
030750 003236
030752 070707

030754 012700 177776
030760 012701 003240
030764 010021
030766 000261
030770 006100
030772 103774

030774 004037 042650
031000 003300
031002 004156
031004 000000

031006 004037 042650
031012 051534
031014 051602
031016 070707

031020 012700 177776
031024 012701 051604
031030 010021
031032 000261

```
;*
:*****
:*TEST 67      WRITE CHECKY HEADER AND DATA
:
:**          WRITE CHECK CYLINDER 0, FORMAT 16 BITS PER WORD
:**          TRACK 1, SECTOR 1, KEYS 0, 36 WORDS AS SHOWN BELOW
:**          ANY DEVICE LOGIC ERROR INDICATION IS NOT CONCLUSIVE ON FIRST PASS
:**          BECAUSE ERROR LOGIC HAS NOT BEEN CHECKED YET (ON FIRST PASS)
:**          ONLY RH WRITE CHECK ERROR (RHCS2 BIT 14) IS TESTED HERE
:*****
TST67: SCOPE
:
:*          DATA TABLE
:*          20 WORDS OF 070707
:*          THEN 16 WORDS WITH ZERO FLOATING FROM RIGHT
:*          TO LEFT (EG. 177776,177775,177773 ETC)
:
MOV          #STACK,SP          ;RESET STACK
MOV          #TTNO,@#TSTNM      ;THIS SAVES TEST NUMBER
:
;*SET UP "REINTO" FOR WHAT IS TO BE READ
MOV          #REINTO,R1         ;STARTING ADDRESS
MOV          #FMT22,(R1)+       ;CYLINDER 0 FORMAT 16 BIT WORDS
MOV          #401,(R1)+         ;TRACK=1, SECTOR=1
CLR          (R1)+              ;KEY1=0
CLR          (R1)+              ;KEY2=0
JSR          RO,@#CLAREA        ;FILL "REINTO" BUFFER
.WORD       REINTO+<4*2>        ;FROM
.WORD       REINTO+<23.*2>     ;TO
.WORD       070707             ;DATA
:
MOV          #177776,RO         ;GETTING READY TO FLOAT 0
MOV          #REINTO+<24.*2>,R1 ;STARTING ADDRESS WHERE 177776 GOES
1$: MOV          RO,(R1)+        ;MOVE IN FLOATING 0
SEC          ;SET CARRY
ROL          RO                 ;GET 0 ONE BIT LEFT
BCS         1$                 ;BRANCH IF 16 NOT DONE
:
JSR          RO,@#CLAREA        ;FILL THE REST OF BUFFER WITH 0
.WORD       REINTO+<40.*2>     ;FROM
.WORD       REINTO+776         ;TO
.WORD       0                  ;DATA
:
;*SET UP SIMULATED DISK WITH WHAT IS TO BE READ
JSR          'RO,@#CLAREA      ;FILL "DISK" BUFFER
.WORD       DISK               ;FROM
.WORD       DISK+<19.*2>      ;TO
.WORD       070707            ;DATA
:
MOV          #177776,RO         ;GETTING READY TO FLOAT ZEROS
MOV          #DISK+<20.*2>,R1  ;STARTING ADDRESS WHERE 177776 GOES
2$: MOV          RO,(R1)+        ;MOVE IN FLOATING 0
SEC          ;SET CARRY
```

```
6533 031034 006100 ROL RO ;GET 0 ONE BIT LEFT
6534 031036 103774 BCS 2$ ;BRANCH IF 16 NOT DONE
6535
6536 031040 004037 042650 JSR RO,@#CLAREA ;FILL THE REST OF BUFFER WITH 177777
6537 031044 051644 .WORD DISK+<36.*2> ;FROM
6538 031046 052532 .WORD DISK+776 ;TO
6539 031050 177777 .WORD 177777 ;DATA
6540
6541 031052 004737 043456 JSR PC,@#WRCHHD ;WRITE CHECK HEADER AND DATA
6542 ;CYLINDER 0, TRACK 1, SECTOR 1
6543
6544 ;*IF THE PROGRAM COMES BACK HERE THEN WRITE CHECK
6545 ;*HAS BEEN COMPLETED NOW WRITE CHECK ERROR BIT IS TO BE TESTED
6546
6547 031056 013746 001774 MOV @#UNIT,-(SP) ;GET UNIT NUMBER
6548 031062 052716 000100 BIS #IR,(SP) ;ONLY BIT 6 SHOULD BE SET
6549 031066 004737 042336 JSR PC,@#PUTREG ;SAVE REGISTERS
6550 031072 022637 001712 CMP (SP)+,@#CS2 ;COMPARE RHCS2
6551 031076 001406 BEQ 4$ ;BRANCH IF GOOD
6552 031100 032712 040000 BIT #WCE,@R2 ;WRITE CHECK ERROR HIGH?
6553 031104 001402 BEQ 3$ ;BRANCH IF ERROR NOT DUE TO "WCE"
6554 031106 104017 ERROR 17 ;RHDB CONTAINS FAILING WORD
6555 031110 000401 BR 4$ ;RHBA CONTAINS ADDRESS+2
6556 ;OF THE WORD IN MEMORY FROM
6557 ;THE DISK THAT DID NOT COMPARE
6558 ;TRE AND SC WILL BE SET DUE TO
6559 ;WCE
6560 031112 104017 3$: ERROR 17 ;WCE CORRECTLY WAS NOT SET BUT SOME
6561 ;BITS OTHER THAN IR
6562 ;AND UNIT NO. WAS SET
6563
6564 ;*NOW CHECK MEMORY TO SEE IF NOTHING GOT DESTROYED
6565 ;*FILL "WRFROM" WITH WHAT SHOULD BE IN "REINTO" THEN CHECK
6566
6567 031114 012700 002114 4$: MOV #WRFROM,RO ;STARTING ADDRESS
6568 031120 012720 010000 MOV #FMT22,(RO)+ ;CYLINDER
6569 031124 012720 000401 MOV #401,(RO)+ ;TRACK=1, SECTOR=1
6570 031130 005020 CLR (RO)+ ;KEY1=0
6571 031132 005020 CLR (RO)+ ;KEY2=0
6572
6573 031134 004037 042650 JSR RO,@#CLAREA ;FILL "WRFROM" BUFFER
6574 031140 002124 .WORD WRFROM+<4*2> ;FROM
6575 031142 002172 .WORD WRFROM+<23.*2> ;TO
6576 031144 070707 .WORD 070707 ;DATA
6577
6578 031146 012700 177776 MOV #177776,RO ;GETTING READY TO FLOAT 0
6579 031152 012701 002174 5$: MOV #WRFROM+<24.*2>,R1 ;STARTING ADDRESS WHERE 177776 GOES
6580 031156 010021 MOV RO,(R1)+ ;MOVE IN FLOATING 0
6581 031160 000261 SEC ;SET CARRY
6582 031162 006100 ROL RO ;GET 0 ONE BIT LEFT
6583 031164 103774 BCS 5$ ;BRANCH IF 16 NOT DONE
6584
6585 031166 004037 042650 JSR RO,@#CLAREA ;FILL THE REST OF BUFFER WITH 0
6586 031172 002234 .WORD WRFROM+<40.*2> ;FROM
6587 031174 003112 .WORD WRFROM+776 ;TO
6588 031176 000000 .WORD 0 ;DATA
```

| | | | | | | | | | |
|------|--------|--------|--------|-------|-------|----|--|--|--|
| 6589 | | | | | | | | | |
| 6590 | | | | | | | | | |
| 6591 | 031200 | 005037 | 002006 | | | | | | |
| 6592 | | | | | | | | | |
| 6593 | 031204 | 004037 | 043626 | | | | | | |
| 6594 | 031210 | 002114 | | | | | | | |
| 6595 | 031212 | 003160 | | | | | | | |
| 6596 | 031214 | 000400 | | | | | | | |
| 6597 | 031216 | 031224 | | | | | | | |
| 6598 | 031220 | 031230 | | | | | | | |
| 6599 | 031222 | 031236 | | | | | | | |
| 6600 | 031224 | 104004 | | 6\$: | ERROR | 4 | | | |
| 6601 | 031226 | 000207 | | | RTS | PC | | | |
| 6602 | 031230 | 104005 | | 7\$: | ERROR | 5 | | | |
| 6603 | | | | | | | | | |
| 6604 | | | | | | | | | |
| 6605 | | | | | | | | | |
| 6606 | | | | | | | | | |
| 6607 | 031232 | 000207 | | | RTS | PC | | | |
| 6608 | | | | | | | | | |
| 6609 | 031234 | 000240 | | 10\$: | NOP | | | | |

```

;*NOW THE READ BUFFER WILL BE CHECKED
CLR @#ERFLG$ ;CLEAR ERROR FLAG

JSR RO,@#COMPAP ;CHECK
WRFROM ;GOOD BUFFER
REINTO ;TEST BUFFER
256. ;NUMBER OF WORDS CHECKED
6$ ;RETURN POINT FOR ERROR HEADER
7$ ;RETURN POINT FOR ERROR DATA
;RETURN FOR GOOD COMPARISON
;READ NEXT ERROR 5
;RETURN TO COMPARISON SUBROUTINE
;DATA IN REINTO BUFFER GOT
;CHANGED AFTER A WRITE
;CHECK HEADER AND DATA COMMAND
;WORD NO CONTAINS THE WORD
;NUMBER THAT GOT CHANGED
;RETURN TO COMPARISON SUBROUTINE

```


6610
6611
6612
6613
6614
6615
6616
6617
6618
6619
6620
6621
6622
6623
6624
6625
6626
6627
6628
6629
6630
6631
6632
6633
6634
6635
6636
6637
6638
6639
6640
6641
6642
6643
6644
6645
6646
6647
6648
6649
6650
6651
6652
6653
6654
6655
6656
6657
6658
6659
6660
6661
6662
6663
6664
6665

031236 000004

031240 012706 001000
031244 012737 000070 002032

031252 012700 000001
031256 012701 003160
031262 010021
031264 006100
031266 103375
031270 012700 177776
031274 012701 003220
031300 010021
031302 000261
031304 006100
031306 103774

031310 004037 042650
031314 003260
031316 004156
031320 000001

031322 012700 000001
031326 012701 051534
031332 010021
031334 006100
031336 103375

031340 012700 177776
031344 012701 051574
031350 010021
031352 000261
031354 006100
031356 103774

```
*****  
:*TEST 70 WRITE CHECK DATA  
  
:** WRITE CHECK DATA CYLINDER 0 FORMAT 16 BITS PER WORD  
:** TRACK 1, SECTOR 1, KEYS 0, 32 WORDS OF DATA  
:** ANY DEVICE LOGIC ERROR INDICATIONS ARE NOT CONCLUSIVE ON FIRST PASS  
:** BECAUSE ERROR LOGIC HAS NOT YET (ON FIRST PASS) BEEN CHECKED  
:** ONLY RH WRITE CHECK ERROR IS TESTED  
  
*****  
TST70: SCOPE  
  
;*DATA TABLE  
;*TOTAL OF 32 WORDS CONSISTING OF  
;*16 WORDS OF FLOATING ONES (EG. 1, 2, 4, 10)  
;*16 WORDS OF FLOATING ZEROS (EG. 177776, 177775)  
  
MOV #STACK,SP ;RESET STACK  
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER  
  
;*SET UP "REINTO" FOR WHAT IS TO BE READ  
  
MOV #1,R0 ;GETTING READY TO FLOAT 1  
MOV #REINTO,R1 ;STARTING ADDRESS WHERE 1 GOES  
1$: MOV RO,(R1)+ ;MOVE FLOATING 1  
ROL RO ;GET 1 ONE BIT LEFT  
BCC 1$ ;BRANCH IF 16 NOT DONE  
MOV #177776,R0 ;GETTING READY TO FLOAT 0  
MOV #REINTO+<16.*2>,R1 ;STARTING ADDRESS WHERE 177776 GOES  
2$: MOV RO,(R1)+ ;MOVE IN FLOATING 0  
SEC ;SET CARRY  
ROL RO ;GET 0 ONE BIT LEFT  
BCS 2$ ;BRANCH IF 16 NOT DONE  
  
JSR RO,@#CLAREA ;FILL REST OF BUFFER WITH 1  
.WORD REINTO+<32.*2> ;FROM  
.WORD REINTO+776 ;TO  
.WORD 1 ;WITH DATA  
  
;*SET UP SIMULATED DISK WITH WHAT IS TO BE READ  
  
MOV #1,R0 ;GETTING READY TO FLOAT 1  
MOV #DISK,R1 ;STARTING ADDRESS WHERE 1 GOES  
3$: MOV RO,(R1)+ ;MOVE FLOATING 1  
ROL RO ;GET 1 ONE BIT LEFT  
BCC 3$ ;BRANCH IF 16 NOT DONE  
  
MOV #177776,R0 ;GETTING READY TO FLOAT 0  
MOV #DISK+<16.*2>,R1 ;STARTING ADDRESS WHERE 177776 GOES  
4$: MOV RO,(R1)+ ;MOVE FLOATING 0  
SEC ;SET CARRY  
ROL RO ;GET 0 ONE BIT LEFT  
BCS 4$ ;BRANCH IF 16 NOT DONE
```

```
6666 031360 004037 042650 JSR RO,@#CLAREA ;FILL REST OF BUFFER WITH 0
6667 031364 051634 .WORD DISK+<32.*2> ;FROM
6668 031366 052532 .WORD DISK+776 ;TO
6669 031370 000000 .WORD 0 ;WITH DATA
6670
6671 031372 004737 043770 JSR PC,@#WRCHDA ;WRITE CHECK DATA
6672 ;CYLINDER 0, TRACK 1, SECTOR 1
6673 ;KEYS 0, 32 WORDS.
6674
6675 ;*IF THE PROGRAM COMES BACK HERE THEN WRITE CHECK
6676 ;*HAS BEEN COMPLETED NOW WRITE CHECK ERROR BIT IS TESTED
6677
6678 031376 013746 001774 MOV @#UNIT,-(SP) ;GET UNIT NUMBER
6679 031402 052716 000100 BIS #IR,(SP) ;ONLY BIT 6 SHOULD BE SET
6680 031406 004737 042336 JSR PC,@#PUTREG ;SAVE REGISTERS
6681 031412 022637 001712 CMP (SP)+,@#CS2 ;COMPARE RHCS2
6682 031416 001407 BEQ 6$ ;BRANCH IF GOOD
6683 031420 032737 040000 001712 BIT #WCE,@#CS2 ;WRITE CHECK ERROR HIGH?
6684 031426 001402 BEQ 5$ ;BRANCH IF ERROR NOT DUE TO "WCE"
6685 031430 104017 ERROR 17 ;RHDB CONTAINS FAILING WORD
6686 031432 000401 BR 6$ ;RHBA CONTAINS ADDRESS+2
6687 ;OF THE WORD IN MEMORY FROM
6688 ;THE DISK THAT DID NOT COMPARE
6689 ;TRE AND SC WILL BE SET DUE TO WCE
6690 031434 104017 5$: ERROR 17 ;WCE WAS CORRECTLY NOT SET
6691 ;BUT SOME BITS OTHER THAN
6692 ;IR AND UNIT NO. WERE SET
6693
6694 ;*NOW CHECK MEMORY TO SEE IF ANYTHING GOT DESTROYED
6695 ;*FILL "WRFROM" WITH WHAT SHOULD BE IN REINTO THEN CHECK IT
6696
6697 031436 005037 002006 6$: CLR @#ERFLG$ ;CLEAR ERROR FLAG
6698 031442 012700 000001 MOV #1,RO ;GETTING READY TO FLOAT 1
6699 031446 012701 002114 MOV #WRFROM,R1 ;START ADDRESS WHERE 1 GOES
6700 031452 010021 7$: MOV RO,(R1)+ ;MOVE FLOATING 1
6701 031454 006100 ROL RO ;GET 1 ONE BIT LEFT
6702 031456 103375 BCC 7$ ;BRANCH IF 16 NOT DONE
6703
6704 031460 012700 177776 MOV #177776,RO ;GETTING READY TO FLOAT 0
6705 031464 012701 002154 MOV #WRFROM+<16.*2>,R1 ;STARTING ADDRESS WHERE 177776 GOES
6706 031470 010021 10$: MOV RO,(R1)+ ;MOVE IN FLOATING 0
6707 031472 000261 SEC ;SET CARRY
6708 031474 006100 ROL RO ;GET 0 ONE BIT LEFT
6709 031476 103774 BCS 10$ ;BRANCH IF CARRY SET
6710
6711 031500 004037 042650 JSR RO,@#CLAREA ;FILL REST OF BUFFER WITH 1
6712 031504 002214 .WORD WRFROM+<32.*2> ;FROM
6713 031506 003112 .WORD WRFROM+776 ;TO
6714 031510 000001 .WORD 1 ;WITH DATA
6715
6716 ;*NOW THE READ BUFFER WILL BE CHECKED
6717
6718 031512 004037 043626 JSR RO,@#COMPAR ;CHECK
6719 031516 002114 WRFROM ;GOOD BUFFER
6720 031520 003160 REINTO ;TEST BUFFER
6721 031522 000400 256. ;NUMBER OF WORDS CHECKED
```


6739
6740
6741
6742
6743
6744
6745
6746
6747
6748
6749
6750
6751
6752
6753
6754
6755
6756
6757
6758
6759
6760
6761
6762
6763
6764
6765
6766
6767
6768
6769
6770
6771
6772
6773
6774
6775
6776
6777
6778
6779
6780
6781
6782
6783
6784
6785
6786
6787
6788
6789
6790
6791
6792
6793
6794

.SBTTL ERROR BIT FUNCTIONAL TESTS

*TEST 71 ATTENTION WITH ERROR TEST

** THIS TESTS THE SETTING OF ATA BIT BOTH IN THE RHAS
** AND THE RHDS1 REGISTERS WITH THE SETTING OF EACH
** ERROR BIT ON THE THREE ERROR REGISTERS.
** IN EACH OF THE ABOVE CASES ERR IN RHDS1 SHOULD
** ALSO SET.

** "GO" SHOULD CLEAR ERR, ATA IN RHDS1 AND RHAS BUT NOT ERROR REG.
** PUTTING "1" IN RHAS DRIVE POSITION CLEARS DRIVE BIT IN ATA IN RHDS1
** UPPER BYTE OF RHAS IS INVALID

TST71: SCOPE

MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
JSR PC,@#CLDISK ;CLEAR DISK REGISTERS
JSR PC,@#CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-
;TINUE TESTING IF BOTH AREN'T TRUE
HALT ;STOP THE TEST

MOV #REINTO,R0 ;BUFFER STARTING FOR 3 ERROR
;REGISTERS
MOV @#RHER1,(R0)+ ;RHER1 STORED IN REINTO
MOV #0,(R0)+ ;BITS NOT TO BE CHECKED IN RHER1
MOV @#RHER2,(R0)+ ;RHER2 STORED IN REINTO+4
MOV #0,(R0)+ ;BITS NOT TO BE CHECKED IN RHER2
MOV @#RHER3,(R0)+ ;RHER3 STORED IN REINTO+10
MOV #0,(R0)+ ;BITS NOT TO BE CHECKED IN RHER3

MOV @#RHAS,R4 ;R4 HAS RHAS
MOV @#ATTENT,R5 ;R5 HAS ATA BIT IN RHAS
MOV #2,@#SLPERR ;THAT SHOULD SET WITH ERROR
;RETURN POINT TO ERROR
MOV #3,@#STMP1 ;ERROR REGISTER COUNTER
MOV #REINTO,R0 ;REGISTER BUFFER POINTER

1\$: MOV (R0)+,R2 ;R2 HAS ADDRESS OF ERROR REG
MOV #BIT0,R1 ;R1 WILL HAVE BIT UNDER TEST
2\$: BIS #CLR,@RHCS2 ;CLEAR RHCS2
MOV @#UNIT,@RHCS2 ;REINSTATE UNIT NO.
MOV R1,@R2 ;SET ERROR BIT
JSR PC,@#PUTREG ;READ AND SAVE REGISTERS
CMPB R5,@#AS ;ONLY THE BIT IN R5 SHOULD BE
;SET IN RHAS

031544 000004
031546 012706 001000
031552 012737 000071 002032
031560 004737 042732
031564 004737 042766
031570 104401 062562
031574 000000
031576 012700 003160
031602 013720 001642
031606 012720 000000
031612 013720 001646
031616 012720 000000
031622 013720 001654
031626 012720 000000
031632 013704 001656
031636 013705 002016
031642 012737 031670 001110
031650 012737 000003 001200
031656 012700 003160
031662 012002 1\$:
031664 012701 000001
031670 052777 000040 147740 2\$:
031676 013777 001774 147732
031704 010112
031706 004737 042336
031712 120537 001732

```
6795 031716 001401 BEQ 3$ ;LOOK @ RHDS1 IF GOOD
6796 031720 104020 ERROR 20 ;WITH THE SETTING OF ONE
6797 ;ERROR BIT IN AN ERROR
6798 ;REGISTER, THE CORRESPONDING
6799 ;RHAS BIT DID NOT SET
6800
6801 031722 013746 001736 3$: MOV @#DS1,-(SP) ;GET RHDS1
6802 031726 042716 001100 BIC #VV!PROG,(SP) ;REMOVE VV AND PROG
6803 031732 022726 140600 CMP #ATA!ERR!DPR!DRY,(SP)+;THESE BITS PLUS VV SHOULD BE IN RHDS1
6804 031736 001401 BEQ 4$ ;CHECK 'GO' NEXT, IF THIS WAS OK
6805 031740 104020 ERROR 20 ;WITH THE SETTING OF ONE
6806 ;ERROR BIT, COMPOSITE ERROR
6807 ;OR ATTENTION ACTIVE, OR
6808 ;ONE OF THE OTHER
6809 ;PERMANENT BITS DID NOT SET
6810
6811 031742 012777 000001 147670 4$: MOV #GO,@RHCS1 ;GIVE NO-OP
6812 031750 004737 042336 JSR PC,@#PUTREG ;SAVE REGISTERS
6813 031754 020112 CMP R1,@R2 ;GO SHOULD NOT CLEAR ERROR
6814 031756 001410 BEQ 5$ ;FURTHER CHECK OF 'GO' FUNCTIONALITY
6815 031760 010237 042402 MOV R2,@#REGADR ;FAILING REGISTER
6816 031764 010137 001124 MOV R1,@#$GDDAT ;GOOD DATA
6817 031770 013737 001712 001126 MOV @#CS2,@#$BDDAT ;BAD DATA
6818 031776 104001 ERROR 1 ;'GO' WITH NO-OP CHANGED
6819 ;ERROR REGISTER
6820
6821 032000 013746 001736 5$: MOV @#DS1,-(SP) ;GET RHDS1
6822 032004 042716 001100 BIC #VV!PROG,(SP) ;CLEAR VV AND PROG
6823 032010 022726 140600 CMP #ATA!ERR!DPR!DRY,(SP)+;GO SHOULD NOT CLEAR ANY BITS
6824 032014 001401 BEQ 7$ ;CHECK NEXT ERROR BIT IF A-OK
6825 032016 104020 ERROR 20 ;'GO' WITH NO-OP SHOULD NOT CLEAR
6826 ;ATA AND/OR ERR
6827
6828
6829 ;*THIS IS THE MAIN BIT TESTING CONTROL LOGIC
6830
6831
6832 032020 006301 7$: ASL R1 ;GET NEXT BIT TO THE LEFT
6833 032022 103403 BCS 10$ ;GO ON TO NEXT REGISTER IF DONE
6834 032024 031001 BIT (R0),R1 ;IS THIS BIT TO BE TESTED ?
6835 032026 001374 BNE 7$ ;IF NOT, GET NEXT ONE
6836 032030 000717 BR 2$ ;IF TO BE TESTED, GO DO IT !
6837
6838 032032 005720 10$: TST (R0)+ ;ADVANCE R0 TO NEXT ERROR REG.
6839 032034 005337 001200 DEC @#$TMP1 ;REGISTER COUNTER
6840 032040 001310 BNE 1$ ;DO NEXT ONE, IF 3 NOT COMPLETE
6841
6842
6843 ;*NOW AFTER SETTING ATA IN RHDS1 '1' IN RHAS AT THE
6844 ;*DRIVE POSITION SHOULD CLEAR ATA IN RHDS1
6845
6846 032042 004737 042732 11$: JSR PC,@#CLDISK ;CLEAR
6847 032046 012737 032042 001110 MOV #11$,$LPERR ;ERROR RETURN
6848 032054 012714 177777 MOV #-1,@R4 ;SET BIT IN RHAS AND ATA IN RHDS1
6849 032060 013777 002016 147570 MOV @#ATTENT,@RHAS ;WRITE 1 INTO DRIVE BIT POSITION
6850 032066 004737 042336 JSR PC,@#PUTREG ;SAVE REGISTERS
```

```

6851 032072 105737 001732      TSTB  @#AS      ;THIS SHOULD BE ZERO
6852 032076 001401              BEQ    12$
6853 032100 104020              ERROR  20      ;MOVING A "1" INTO RHAS
6854                                ;AT THE DRIVE BIT POSITION
6855                                ;DID NOT CLEAR IT
6856
6857 032102 013746 001736      12$:  MOV    @#DS1,-(SP) ;GET RHDS1
6858 032106 042716 001000      BIC    #PROG,(SP) ;MASK PROGRAMABLE
6859 032112 022726 040700      CMP    #ERR!VV!DPR!DRY,(SP)+
6860                                ;RHDS1 SHOULD HAVE THESE BITS
6861                                ;BUT ATA SHOULD BE CLEARED
6862 032116 001401              BEQ    13$      ;CHECK RHER1 IF GOOD
6863 032120 104020              ERNOR  20      ;MOVING "1" INTO RHAS AT THE
6864                                ;DRIVE BIT POSITION DID NOT
6865                                ;CLEAR ATA IN RHDS1
6866
6867 032122 022737 177777 001716 13$:  CMP    #-1,@#ER1 ;RHER1 SHOULD NOT CHANGE
6868                                ;BY CLEARING RHAS
6869 032130 001401              BEQ    TST72   ;BRANCH IF GOOD
6870 032132 104020              ERROR  20      ;RHER1 WAS CHANGED BY CLEARING
6871                                ;RHAS BY MOVING "1" INTO
6872                                ;THE DRIVE BIT POSITION
6873
6874
6875
6876
6877
  
```

```
6878
6879
6880
6881
6882
6883
6884
6885
6886
6887
6888
6889 032134 000004
6890 032136 012706 001000
6891
6892 032142 012737 000072 002032
6893 032150 004037 042650
6894 032154 051534
6895 032156 052532
6896 032160 177400
6897
6898 032162 004037 042650
6899 032166 003160
6900 032170 004156
6901 032172 000000
6902
6903
6904
6905 032174 012737 010000 047616
6906 032202 105037 047621
6907 032206 112737 000001 047620
6908 032214 005037 047622
6909 032220 005037 047624
6910 032224 012737 000012 047676
6911 032232 005037 047626
6912 032236 004537 044140
6913 032242 047616
6914 032244 051516
6915
6916
6917
6918 032246 004737 042732
6919 032252 013711 002070
6920 032256 012777 177766 147346
6921 032264 012777 003160 147342
6922 032272 112746 000001
6923 032276 112766 000000 000001
6924 032304 012677 147334
6925 032310 012777 014000 147332
6926
6927
6928 032316 005077 147330
6929 032322 004737 042766
6930 032326 104401 062562
6931
6932 032332 000000
6933 032334 052777 000010 147274
```

```
*****
*TEST 72      BUS ADDRESS INHIBIT
**          READ CYLINDERO, FORMAT 16 BITS PER WORD
**          TRACK0, SECTOR 1, KEYS 0, 10 WORDS OF 177400
**          THIS IS DONE WITH BUS ADDRESS INHIBIT SET
**          ANY ERROR LOGIC INDICATION IS NOT CONCLUSIVE ON FIRST PASS
**          BECAUSE ERROR LOGIC HAS NOT BEEN CHECKED YET (ON FIRST PASS)
*****
TST72:  SCOPE
        MOV      #STACK,SP      ;RESET STACK
        MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
        JSR      RO,@#CLAREA    ;CLEAR SIMULATED DISK
        .WORD   DISK           ;FROM
        .WORD   DISK+776      ;TO
        .WORD   177400        ;DATA
        JSR      RO,@#CLAREA    ;CLEAR READ INTO BUFFER
        .WORD   REINTO        ;FROM
        .WORD   REINTO+776    ;TO
        .WORD   0             ;DATA
        ;*THESE ARE TO SETUP FOR DISKLESS USE ONLY
        MOV      #FMT22,@#CYL   ;CYLINDER 0 16 BITS PER WORD FORMAT
        CLRB    @#SECOTR+1     ;TRACK 0
        MOV     #1,@#SECOTR    ;SECTOR 1
        CLR     @#KEY1        ;KEY1=0
        CLR     @#KEY2        ;KEY2=0
        MOV     #10,@#DAWORD   ;NO. OF DATA WORDS
        CLR     @#X           ;THIS IS A READ COMMAND
        JSR     R5,@#CRC       ;GO TO CALCULATE CRC
        CYL
        WCRC
        ;*THESE ARE REGULAR SETUPS
        JSR     PC,@#CLDISK    ;SETUP GENERAL REGISTERS
        MOV     @#READAT,@R1   ;READ DATA INTO RHCS1=70
        MOV     #-10,@#RHWC    ;10 DATA WORDS
        MOV     #REINTO,@#RHBA ;STARTING ADDRESS OF READ BUFFER
        MOV     #1,-(SP)       ;IN LOWER BYTE GET SECTOR 1
        MOV     #0,1(SP)      ;GET TRACK0 IN UPPER BYTE
        MOV     (SP)+,@#RHDST  ;TRACK/SECTOR IN RHDST
        MOV     #FMT22!ECI,@#RHOF ;16 BITS PER WORD
        ;ECC CORRECTION INHIBIT BECAUSE
        ;ECC IS NOT CHECKED HERE
        CLR     @#RHCA        ;CYLINDER 0
        JSR     PC,@#CHECKT    ;CHECK THAT DVA,RDY,DPR,DRY = 1
        TYPE    ,CPHALT      ;AND THAT NO OTHERS = 1. CANNOT CON-
        ;TINUE TESTING IF BOTH AREN'T TRUE
        HALT
        BIS     #BA1,@#RHCS2  ;SET BUS ADDRESS INHIBIT
```

```

6934 032342 005037 002006 CLR @#ERFLG$ ;CLEAR ERROR FLAG
6935 032346 004737 047456 JSR PC,@#COMHD ;READ DATA
6936
6937 ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUT
6938 ;*FROM "COMHD" ROUTINE IN MEANS DATA IS TO BE CHECKED
6939
6940 ;*NOW THE DATA READ INTO "REINTO" BUFFER WILL
6941 ;*BE CHECKED, ONLY ONE WORD SHOULD BE CHANGED
6942 ;*ALL OTHER WORDS SHOULD REMAIN UNCHANGED
6943 ;*THE "WRFROM" BUFFER IS FILLED WITH EXPECTED DATA AND CHECKED
6944
6945 032352 005037 002006 CLR @#ERFLG$ ;CLEAR FLAG
6946 032356 004037 042650 JSR RO,@#CLAREA ;CLEAR BUFFER
6947 032362 002114 WRFROM ;FROM
6948 032364 003112 WRFROM+776 ;TO
6949 032366 000000 0 ;DATA
6950
6951 ;*EXPECTED DATA IS 177400 IN FIRST LOCATION ONLY
6952 032370 012737 177400 002114 MOV #177400,@#WRFROM ;EXPECTED DATA
6953
6954 ;*NOW READ DATA BUFFER IS CHECKED
6955
6956 032376 012700 002114 MOV #WRFROM,R0 ;GOOD DATA
6957 032402 012701 003160 MOV #REINTO,R1 ;DATA READ
6958 032406 012702 000400 MOV #256.,R2 ;COUNTER
6959 032412 012737 000401 047736 1$: MOV #257.,@#ERWORD ;FOR ERROR WORD NO
6960 032420 022021 CMP (R0)+,(R1)+ ;COMPARE GOOD WITH READ BUFFER
6961 032422 001424 BEQ 2$ ;BRANCH IF GOOD
6962 032424 014037 001124 MOV -(R0),@#$GDDAT ;GOOD DATA
6963 032430 014137 001126 MOV -(R1),@#$BDDAT ;BAD DATA
6964 032434 160237 047736 SUB R2,@#ERWORD ;ERROR WORD NO
6965 032440 005737 002006 TST @#ERFLG$ ;ANY ERRORS ALREADY THERE
6966 032444 001002 BNE 3$ ;IF YES BRANCH DO NOT TYPE HEADER
6967 032446 104004 ERROR 4 ;ERROR ON READ DATA
6968 032450 000401 BR 4$ ;BRANCH TO AVOID PRINTING NEXT ERROR
6969 032452 104005 3$: ERROR 5 ;WORD NO 1-10 ARE DATA
6970 ;WORDS
6971 ;WORD NOS 11-256 HAVE NOT BEEN
6972 ;READ AND BUFFER SHOULD BE
6973 ;ZERO IF OTHER THAN ZERO
6974 ;WRONG NUMBER OF WORDS HAVE
6975 ;BEEN READ IN THE DISK NOW
6976 ;CONTAINS 177400 ALL 256
6977 ;WORDS BUT ONLY 10 WORDS
6978 ;SHOULD BE READ IN
6979
6980 032454 022021 4$: CMP (R0)+,(R1)+ ;UNDO -(R0) AND -(R1) FOR ERROR
6981 032456 017746 146456 MOV @SWR,-(SP) ;GET SWITCH SETTING
6982 032462 042716 177177 BIC #177177,(SP) ;KEEP ONLY SWITCH 7 AND 8
6983 032466 022726 000200 CMP #SW07,(SP)+ ;IS 7 SET AND 8 RESET
6984 032472 001402 BEQ TST73 ;BRANCH OUT IF YES
6985 032474 005302 2$: DEC R2 ;COUNT
6986 032476 001345 BNE 1$ ;BRANCH IF NOT COMPLETE
6987
6988
6989

```


6990
6991

```
6992
6993
6994
6995
6996
6997
6998
6999
7000
7001
7002
7003 032500 000004
7004 032502 012706 001000
7005 032506 012737 000073 002032
7006
7007
7008
7009 032514 005737 002040
7010 032520 001402
7011 032522 000137 033044
7012 032526
7013
7014
7015 032526 004037 042650
7016 032532 051534
7017 032534 052532
7018 032536 177400
7019
7020
7021
7022
7023 032540 012737 010000 047616
7024 032546 105037 047621
7025 032552 112737 000001 047620
7026 032560 005037 047622
7027 032564 005037 047624
7028 032570 012737 000001 047676
7029 032576 005037 047626
7030 032602 004537 044140
7031 032606 047616
7032 032610 051516
7033
7034
7035
7036 032612 004737 042732
7037 032616 013711 002070
7038 032622 012777 177777 147002
7039 032630 012777 160000 146776
7040 032636 052711 001400
7041 032642 112746 000001
7042 032646 112766 000000 000001
7043 032654 012677 146764
7044 032660 012777 014000 146762
7045
7046
7047 032666 005077 146760
```

```
*****
*TEST 73      RHCS2 - BIT # 11 - NEM
**          READ CYLINDER0, FORMAT 16 BITS PER WORD
**          TRACK0, SECTOR 1, KEYS 0, 1 WORD OF 177400
**          THIS IS DONE WITH BUS ADDRESS INHIBIT SET
**          BUS ADDRESS USED IS 760000 THIS IS ALWAYS NON EXISTANT
**          THIS SHOULD SET NEM
*****
TST73:  SCOPE
        MOV      #STACK,SP      ;RESET STACK
        MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
        ;*CHECK TO SEE IF THE PROGRAM IS RUNNING WITH AN RH70
        TST      @#RH70         ;TEST FLAG FOR RH70 CONTROLLER
        BEQ      30$            ;IF FLAG = 1, THIS TEST IS SKIPPED
        JMP      TST74         ;JUMP TO NEXT TEST -----)
        ;IF FLAG = 0, DO THIS TEST
30$:
        JSR      R0,@#CLAREA    ;CLEAR SIMULATED DISK
        .WORD   DISK            ;FROM
        .WORD   DISK+776        ;TO
        .WORD   177400          ;DATA
        ;*THESE ARE TO SETUP FOR DISKLESS USE ONLY
        MOV      #FMT22,@#CYL   ;CYLINDER 0, 16 BITS PER WORD FORMAT
        CLRB    @#SECOTR+1      ;TRACK 0
        MOV     #1,@#SECOTR     ;SECTOR 1
        CLR     @#KEY1          ;KEY1=0
        CLR     @#KEY2          ;KEY2=0
        MOV     #1,@#DAWORD     ;NO. OF DATA WORDS
        CLR     @#X             ;THIS IS A READ COMMAND
        JSR     R5,@#CRC        ;GO TO CALCULATE CRC
        CYL
        WCRC
        ;*THESE ARE REGULAR SETUPS
        JSR     PC,@#CLDISK     ;SETUP GENERAL REGISTERS
        MOV     @#READAT,@R1    ;READ DATA INTO RHCS1=70
        MOV     #-1,@#RHWC      ;10 DATA WORDS
        MOV     #160000,@#RHBA  ;STARTING ADDRESS OF READ BUFFER
        BIS     #A16:A17,@R1   ;IS 760000
        MOV     #1,-(SP)        ;IN LOWER BYTE GET SECTOR 1
        MOV     #0,1(SP)        ;GET TRACK0 IN UPPER BYTE
        MOV     (SP)+,@#RHDST   ;TRACK/SECTOR IN RHDST
        MOV     #FMT22!EC1,@#RHOF ;16 BITS PER WORD
        ;ECC CORRECTION INHIBIT BECAUSE
        ;ECC IS NOT CHECKED HERE
        CLR     @#HCA          ;CYLINDER 0
```

```

7048 032672 004737 042766 JSR PC,@#CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
7049 032676 10440i 062562 TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-
7050 ;TINUE TESTING IF BOTH AREN'T TRUE
7051 032702 000000 HALT ;STOP THE TEST
7052 032704 052777 000010 146724 BIS #BAI,@RHCS? ;SET BUS ADDRESS INHIBIT
7053 032712 005037 002006 CLR @#ERFLG$ ;CLEAR ERROR FLAG
7054 032716 004737 047456 JSR PC,@#COMHD ;READ DATA
7055
7056
7057
7058 032722 011137 001126 1$: MOV @R1,@#$BDDAT ;TEST DATA
7059
7060 032726 022737 145670 001126 CMP #SC!TRE!DVA!A16!A17!RDY!70,@#$BDDAT ;COMPARE RHCS1
7061 032734 001406 BEQ 2$ ;BRANCH IF GOOD
7062 032736 012737 144270 001124 MOV #SC!TRE!DVA!RDY!70,@#$GDDAT ;GOOD DATA
7063 032744 010137 042402 MOV R1,@#REGADR ;REGISTER RHCS1
7064 032750 104001 ERROR 1 ;REFERENCE NON EXISTANT
7065 ;MEMORY DID NOT SET
7066 ;REQUIRED BITS
7067 032752 013746 001774 2$: MOV @#UNIT,-(SP) ;GET UNIT NUMBER
7068 032756 052716 004110 BIS #NEM!IR!BAI,(SP) ;INCLUDE NEM BAI AND IR
7069 032762 012637 001124 MOV (SP)+,@#$GDDAT ;
7070 032766 011237 001126 MOV @R2,@#$BDDAT ;TEST DATA
7071 032772 023737 001124 001126 CMP @#$GDDAT,@#$BDDAT;COMPARE RHCS2
7072 033000 001403 BEQ 3$
7073 033002 010237 042402 MOV R2,@#REGADR ;REGISTER ADDRESS
7074 033006 104001 ERROR 1 ;REFRENCING NONEXISTANT MEMORY
7075 ;CAUSED AN ERROR SHOULD SET NEM
7076 033010 017737 146620 001126 3$: MOV @RHBA,@#$BDDAT ;TEST DATA
7077
7078 033016 022737 160000 001126 CMP #160000,@#$BDDAT ;COMPARE RHBA
7079 033024 001407 BEQ 4$ ;BRANCH IF GOOD
7080 033026 012737 160000 001124 MOV #160000,@#$GDDAT;GOOD DATA
7081 033034 013737 001634 042402 MOV @#RHBA,@#REGADR ;REGISTER ADDRESS RHBA
7082 033042 104001 ERROR 1 ;AFTER A NON EXISTANT MEMORY ERROR
7083 ;RHBA DOES NOT HAVE 160002
7084 033044 4$:
7085
7086
7087
7088
7089
7090
7091
  
```

```
7092
7093
7094
7095
7096
7097
7098
7099
7100
7101
7102
7103
7104
7105 033044 000004
7106
7107
7108
7109
7110
7111
7112
7113 033046 012706 001000
7114 033052 012737 000074 002032
7115 033060 004737 042732
7116
7117
7118
7119 033064 012700 000001
7120 033070 012701 003160
7121 033074 010021
7122 033076 006100
7123 033100 103375
7124 033102 012700 177776
7125 033106 012701 003220
7126 033112 010021
7127 033114 000261
7128 033116 006100
7129 033120 103774
7130
7131 033122 004037 042650
7132 033126 003260
7133 033130 004156
7134 033132 000001
7135
7136
7137
7138 033134 012700 000001
7139 033140 012701 051534
7140 033144 010021
7141 033146 006100
7142 033150 103375
7143
7144 033152 012700 177776
7145 033156 012701 051574
7146 033162 010021
7147 033164 000261

;*****
;*TEST 74 WRITE CHECK ERROR
;
; ** WRITE CHECK DATA CYLINDER 0 FORMAT 16 BITS PER WORD
; ** TRACK 1, SECTOR 1, KEYS 0, 32 WORDS OF DATA
; ** FIFTH WORD IS CHANGED ON DISK TO GIVE WRITE CHECK ERROR
; ** ANY DEVICE LOGIC ERROR INDICATIONS ARE NOT CONCLUSIVE
; ** ON FIRST PASS
; ** BECAUSE ERROR LOGIC HAS NOT YET BEEN CHECKED
; ** ONLY RM WRITE CHECK ERROR IS TESTED
;*****
TST74: SCOPE
;
; *DATA TABLE
; *TOTAL OF 32 WORDS CONSISTING OF
; *16 WORDS OF FLOATING ONES (EG. 1, 2, 4, 10)
; *16 WORDS OF FLOATING ZEROS (EG. 177776, 177775)
;
MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
JSR PC,@#CLDISK ;INIT AND SET UP GENERAL REGISTERS
;
; *SET UP "REINTO" FOR WHAT IS TO BE READ
;
MOV #1,R0 ;GETTING READY TO FLOAT 1
MOV #REINTO,R1 ;STARTING ADDRESS WHERE 1 GOES
1$: MOV RO,(R1)+ ;MOVE FLOATING 1
ROL RO ;GET 1 ONE BIT LEFT
BCC 1$ ;BRANCH IF 16 NOT DONE
MOV #177776,R0 ;GETTING READY TO FLOAT 0
MOV #REINTO+<16.*2>,R1 ;STARTING ADDRESS WHERE 177776 GOES
2$: MOV RO,(R1)+ ;MOVE IN FLOATING 0
SEC ;SET CARRY
ROL RO ;GET 0 ONE BIT LEFT
BCS 2$ ;BRANCH IF 16 NOT DONE
;
JSR RO,@#CLAREA ;FILL REST OF BUFFER WITH 1
.WORD REINTO+<32.*2> ;FROM
.WORD REINTO+776 ;TO
.WORD 1 ;WITH DATA
;
; *SET UP SIMULATED DISK WITH WHAT IS TO BE READ
;
MOV #1,R0 ;GETTING READY TO FLOAT 1
MOV #DISK,R1 ;STARTING ADDRESS WHERE 1 GOES
3$: MOV RO,(R1)+ ;MOVE FLOATING 1
ROL RO ;GET 1 ONE BIT LEFT
BCC 3$ ;BRANCH IF 16 NOT DONE
;
MOV #177776,R0 ;GETTING READY TO FLOAT 0
MOV #DISK+<16.*2>,R1 ;STARTING ADDRESS WHERE 177776 GOES
4$: MOV RO,(R1)+ ;MOVE FLOATING 0
SEC ;SET CARRY
```

```

7148 033166 006100          ROL    R0          ;GET 0 ONE BIT LEFT
7149 033170 103774          BCS    4$          ;BRANCH IF 16 NOT DONE
7150
7151 033172 004037 042650   JSR    R0,@#CLAREA ;FILL REST OF BUFFER WITH 0
7152 033176 051634          .WORD  DISK+<32.*2> ;FROM
7153 033200 052532          .WORD  DISK+776    ;TO
7154 033202 000000          .WORD  0           ;WITH DATA
7155
7156                          ;*CHANGE FIFTH WORD TO 0 ON DISK
7157
7158 033204 005037 051544   CLR    @#DISK+10   ;CLEAR FIFTH WORD ON DISK
7159 033210 005037 002006   CLR    @#ERFLG$   ;CLEAR ERROR FLAG
7160 033214 004737 043770   JSR    PC,@#WRCHDA ;WRITE CHECK DATA
7161                          ;CYLINDER 0, TRACK 1, SECTOR 1
7162                          ;KEYS 0, 32 WORDS.
7163
7164                          ;*IF THE PROGRAM COMES BACK HERE THEN WRITE CHECK
7165                          ;*HAS BEEN COMPLETED, NOW WRITE CHECK ERROR BIT IS TESTED
7166                          ;*ALONG WITH RHWC FOR PROPER WORD COUNT AND RHBA FOR ADDRESS
7167
7168 033220 013746 001774   MOV    @#UNIT,-(SP) ;GET UNIT NUMBER
7169 033224 052716 040300   BIS    #IR!OR!WCE,(SP) ;ONLY BIT 6 SHOULD BE SET
7170 033230 004737 042336   JSR    PC,@#PUTREG ;SAVE REGISTERS
7171 033234 022637 001712   CMP    (SP)+,@#CS2 ;COMPARE RHCS2
7172 033240 001407          BEQ    6$          ;BRANCH IF GOOD
7173 033242 032737 040000 001712  BIT    #WCE,@#CS2 ;WRITE CHECK ERROR HIGH?
7174 033250 001002          BNE    5$          ;BRANCH IF ERROR NOT DUE TO "WCE"
7175 033252 104017          ERROR  17         ;RHDB CONTAINS FAILING WORD
7176 033254 000401          BR     6$          ;RHBA CONTAINS ADDRESS+2
7177                          ;OF THE WORD IN MEMORY FROM
7178                          ;THE DISK THAT DID NOT COMPARE
7179                          ;TRE AND SC WILL BE SET DUE TO WCE
7180 033256 104017          5$:  ERROR  17         ;WCE WAS CORRECTLY NOT SET
7181                          ;BUT SOME BITS OTHER THAN
7182                          ;IR AND UNIT NO. WERE SET
7183
7184 033260 005737 002040          6$:  TST    @#RH70    ;TEST FOR RH70 CONTROLLER
7185 033264 001414          BEQ    16$         ;SKIP RH70 CODE AND DO RH11 IF NOT
7186
7187 033266 022737 177750 001706  CMP    #-24.,@#WC ;COMPARE RHWC AFTER A FORCED
7188                          ;WRITE CHECK ERROR
7189 033274 001402          BEQ    17$         ;CHECK RHBA IF GOOD
7190 033276 104017          ERROR  17         ;WORD COUNT REGISTER IN ERROR AFTER A
7191                          ;FORCED WRITE CHECK ERROR ON FIFTH WORD
7192 033300 000421          BR     15$         ;BRANCH TO CONTINUE TEST
7193
7194 033302 022737 003200 001710 17$:  CMP    #REINTO+<8.*2>,@#BA ;COMPARE RHBA AFTER A FORCED
7195                          ;WRITE CHECK ERROR IN FIFTH WORD
7196 033310 001415          BEQ    15$         ;CONTINUE IF GOOD
7197 033312 104017          ERROR  17         ;BUS ADDRESS REGISTER IN ERROR AFTER
7198                          ;FORCED WRITE CHECK ERROR ON FIFTH WORD
7199 033314 000413          BR     15$         ;SKIP RH11 CODE AND CONTINUE WITH TEST
7200
7201 033316 022737 177745 001706 16$:  CMP    #-27.,@#WC ;COMPARE RHWC AFTER A FORCED
7202                          ;WRITE CHECK ERROR
7203 033324 001402          BEQ    14$         ;CHECK RHBA IF GOOD
  
```

```

7204 033326 104017          ERROR 17          ;WORD COUNT REGISTER IN ERROR AFTER A
7205                                ;FORCED WRITE CHECK ERROR ON FIFTH WORD
7206 033330 000405          BR      15$          ;BRANCH TO CONTINUE TEST
7207
7208 033332 022737 003172 001710 14$: CMP    #REINTO+<5.*2>,@#BA;COMPARE RHBA AFTER FORCED
7209                                ;WRITE CHECK ERROR IN FIFTH WORD
7210 033340 001401          BEQ     15$          ;CONTINUE IF GOOD
7211 033342 104017          ERROR 17          ;BUS ADDRESS REGISTER IN ERROR AFTER
7212                                ;FORCED WRITE CHECK ERROR ON FIFTH WORD
7213
7214                                ;*NOW CHECK MEMORY TO SEE IF ANYTHING GOT DESTROYED
7215                                ;*FILL "WRFROM" WITH WHAT SHOULD BE IN REINTO THEN CHECK
7216
7217 033344 005037 002006      15$: CLR    @#ERFLG$          ;CLEAR ERROR FLAG
7218 033350 012700 000001      MOV    #1,RO          ;GETTING READY TO FLOAT 1
7219 033354 012701 002114      MOV    #WRFROM,R1     ;START ADDRESS WHERE 1 GOES
7220 033360 010021          7$: MOV    RO,(R1)+        ;MOVE FLOATING 1
7221 033362 006100          ROL    RO             ;GET 1 ONE BIT LEFT
7222 033364 103375          BCC    7$            ;BRANCH IF 16 NOT DONE
7223
7224 033366 012700 177776      MOV    #177776,RO     ;GETTING READY TO FLOAT 0
7225 033372 012701 002154      MOV    #WRFROM+<16.*2>,R1 ;STARTING ADDRESS WHERE 177776 GOES
7226 033376 010021          10$: MOV   RO,(R1)+      ;MOVE IN FLOATING 0
7227 033400 000261          SEC                     ;SET CARRY
7228 033402 006100          ROL    RO             ;GET 0 ONE BIT LEFT
7229 033404 103774          BCS    10$          ;BRANCH IF CARRY SET
7230
7231 033406 004037 042650      JSR    RO,@#CLAREA    ;FILL REST OF BUFFER WITH 1
7232 033412 002214          .WORD WRFROM+<32.*2> ;FROM
7233 033414 003112          .WORD WRFROM+776     ;TO
7234 033416 000001          .WORD 1              ;WITH DATA
7235
7236                                ;*NOW THE READ BUFFER WILL BE CHECKED
7237
7238 033420 004037 043626      JSR    RO,@#COMPAR    ;CHECK
7239 033424 002114          WRFROM                ;GOOD BUFFER
7240 033426 003160          REINTO                ;TEST BUFFER
7241 033430 000400          256.                 ;NUMBER OF WORDS CHECKED
7242 033432 033440          11$                  ;RETURN POINT FOR ERROR HEADER
7243 033434 033444          12$                  ;RETURN POINT FOR ERROR DATA
7244
7245 033436 033452          TST75                 ;RETURN FOR GOOD COMPARISON
7246
7247 033440 104004          11$: ERROR 4          ;READ NEXT ERROR 5
7248 033442 000207          RTS    PC             ;RETURN TO COMPARISON SUBROUTINE
7249 033444 104005          12$: ERROR 5          ;DATA IN REINTO BUFFER GOT
7250                                ;CHANGED AFTER A WRITE
7251                                ;CHECK DATA COMMAND
7252                                ;WORD NO CONTAINS THE WORD
7253                                ;NUMBER THAT GOT CHANGED
7254 033446 000207          RTS    PC             ;RETURN TO COMPARISON SUBROUTINE
7255
7256 033450 000240          13$: NOP              ;ONLY A BRANCH POINT
7257
7258
7259
  
```

```
7260
7261
7262
7263
7264
7265
7266
7267
7268
7269
7270 033452 000004
7271
7272
7273 033454 012706 001000
7274
7275 033460 012737 000075 002032
7276
7277
7278
7279
7280 033466 012746 125252
7281 033472 012705 000400
7282 033476 012700 051534
7283 033502 011620
7284 033504 005305
7285 033506 001375
7286 033510 005726
7287 033512 012705 000021
7288
7289
7290 033516 005020
7291 033520 005305
7292 033522 001375
7293
7294
7295
7296
7297 033524 012737 000000 047616
7298
7299 033532 112737 000001 047621
7300 033540 112737 000000 047620
7301 033546 012737 000000 047622
7302 033554 012737 000000 047624
7303 033562 012737 000004 047676
7304 033570 005037 047626
7305 033574 004537 044140
7306 033600 047616
7307 033602 051516
7308
7309
7310
7311 033604 004737 042732
7312 033610 012777 177770 146014
7313 033616 012777 003160 146010
7314 033624 112746 000000
7315 033630 112766 000001 000001
```

```
*****
:*TEST 75      ERROR REGISTER #1-BIT 4 -FORMAT ERROR
:**          THE SIMULATED DISK IS FILLED WITH CYLINGER 0 TRACK 1
:**          SECTOR 0 FORMAT=18 BITS PER WORD AND 4 WORDS
:**          OF 125252, A READ HEADER AND DATA COMMAND IS GIVEN WITH 16 BITS
:**          PER WORD FORMAT, FER=BIT4 SHOULD SET BUT THE
:**          READ SHOULD BE COMPLETE
*****
1ST75: SCOPE
MOV      #STACK,SP      ;RESET STACK
MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
:*      SETUP FOR WHAT IS TO BE READ
:*      HEADER CRC IS RESTORED FROM A SUBROUTINE
MOV      #125252,-(SP)  ;DATA TO BE READ
MOV      #256.,R5       ;COUNTER
MOV      #DISK,R0       ;START OF SIMULATED DISK DATA
1$:      MOV      (SP),(R0)+ ;MOVE IN DATA ON TO SIMULATED DISK
DEC      R5              ;COUNT
BNE      1$             ;BRANCH IF 256 NOT COMPLETE
TST      (SP)+          ;UNDO -(SP)
MOV      #17.,R5       ;2 ECC WORDS
                          ;1 DATA GAP
                          ;14 TOLERANCE GAP
2$:      CLR      (R0)+   ;CLEAR ECC, DATA GAP, AND
DEC      R5              ;TOLERANCE GAP
BNE      2$             ;BRANCH IF NOT COMPLETE

:*THESE ARE TO SETUP FOR DISKLESS USE ONLY
MOV      #0!0,@#CYL ;16 BITS PER WORD
                          ;CYLINDER 0, FORMAT 16 BITS
MOVB     #1,@#SECOTR+1 ;TRACK 1
MOVB     #0,@#SECOTR   ;SECTOR 0
MOV      #0,@#KEY1     ;KEY1=0
MOV      #0,@#KEY2     ;KEY2=0
MOV      #4.,@#DAWORD  ;NO. OF DATA WORDS
CLR      @#X           ;THIS IS A READ COMMAND
JSR      R5,@#CRC      ;GO TO CALCULATE CRC
CYL
WCRC

:*THESE ARE REGULAR SETUPS FOR RH11 AND "REINTO"
JSR      PC,@#CLDISK  ;SETUP GENERAL REGISTERS
MOV      #-4.-4.,@RHWC ;4. DATA 4 HEADER WORDS
MOV      #REINTO,@RHBA ;STARTING ADDRESS OF READ BUFFER
MOVB     #0,-(SP)      ;IN LOWER BYTE GET SECTOR
MOVB     #1,1(SP)      ;GET TRACK IN HIGHER BYTE
```

```

7316 033636 012677 146002      MOV      (SP)+,@RHDST      ;TRACK/SECTOR IN RHDST
7317 033642 012777 014000 146000  MOV      #FMT22!EC1,@RHOF ;16 BITS PER WORD
7318                                     ;ECC CORRECTION INHIBIT
7319                                     ;BECAUSE ECC IS NOT GOING
7320                                     ;TO BE CHECKED
7321 033650 005077 145776      CLR      @RHCA             ;CYLINDER 0
7322
7323 033654 004737 042766      JSR      PC,@#CHECKT      ;CHECK THAT DVA,RDY,DPR,DRY = 1
7324 033660 104401 062562      TYPE    ,CPHALT          ;AND THAT NO OTHERS = 1. CANNOT CON-
7325                                     ;TINUE TESTING IF BOTH AREN'T TRUE
7326 033664 000000      HALT
7327                                     ;STOP THE TEST
7328 033666 013711 002072      MOV      @#REFOR,@R1      ;READ HEADER AND DATA=72
7329 033672 005037 002006      CLR      @#ERFLG$        ;CLEAR ERROR FLAG
7330 033676 004737 047456      JSR      PC,@#COMHD       ;READ HEADER AND DATA
7331
7332

```

```

;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
;*FROM THE "COMHD" ROUTINE THAT MEANS SECTOR GAP,
;*FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
;*SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
;*DETECTED.

```

```

;*RHWC IS CHECKED TO BE = 0 AFTER THE READ OPERATION

```

```

7342 033702 017737 145724 001126  MOV      @RHWC,$BDDAT     ;LOAD AND TEST RHWC
7343 033710 001401      BEQ      20$              ;SHOULD = 0
7344 033712 104040      ERROR   40               ;RHWC DOES NOT = 0 AFTER A READ
7345                                     ;HEADER AND DATA
7346

```

```

;*HEADER AND DATA ARE TO BE CHECKED.
;*IN CHECKING READ DATA THE WRITE FROM BUFFER
;*"WRFROM" IS FILLED WITH EXPECTED DATA AND
;*COMPARISONS ARE MADE.

```

```

7352 033714 005737 002006      20$:  TST      @#ERFLG$        ;ANY ERRORS ALREADY THERE
7353 033720 001055      BNE     TST76             ;BRANCH IF YES
7354 033722 004737 043156      JSR      PC,@#CHECKE     ;CHECK THAT BITS = 1
7355 033726 104401 062562      TYPE    ,CPHALT          ;CANNOT CONTINUE TESTING IF THEY DON'T
7356 033732 000000      HALT                    ;STOP THE TEST
7357 033734 012700 002114      MOV      #WRFROM,R0      ;GETTING READY TO FILL EXPECTED DATA
7358 033740 012720 000000      MOV      #0,(R0)+        ;CYLINDER 0
7359 033744 112746 000000      MOV      #0,-(SP)        ;IN LOWER BYTE GET SECTOR
7360 033750 112766 000001 000001  MOV      #1,1(SP)        ;GET TRACK IN HIGHER BYTE
7361 033756 012620      MOV      (SP)+,(R0)+     ;GET TRACK/SECTOR IN BUFFER
7362 033760 012720 000000      MOV      #0,(R0)+        ;KEY1 IN BUFFER
7363 033764 012720 000000      MOV      #0,(R0)+        ;KEY2 IN BUFFER
7364 033770 012701 000400      MOV      #256,R1         ;DATA WORD COUNTER
7365 033774 012702 125252      MOV      #125252,R2      ;DATA
7366 034000 010220      3$:  MOV      R2,(R0)+        ;DATA INTO BUFFER
7367 034002 005301      DEC     R1                ;COUNT
7368 034004 001375      BNE     3$                ;BRANCH IF 256 NOT DONE
7369

```

```

;*NOW READ DATA BUFFER WILL BE CHECKED

```

7370
7371


```
7372 034006 004037 043626 JSR RO,@#COMPAR ;CHECK
7373 034012 002114 WRFROM ;GOOD BUFFER
7374 034014 003160 REINTO ;TEST BUFFER
7375 034016 000010 4+4. ;NUMBER OF WORDS CHECKED
7376 034020 034026 4$ ;RETURN POINT FOR ERROR HEADER
7377 034022 034032 5$ ;RETURN POINT FOR ERROR DATA
7378 034024 034036 6$ ;RETURN FOR GOOD COMPARISON
7379 034026 104004 4$: ERROR 4 ;READ NEXT ERROR
7380 034030 000207 RTS PC ;RETURN TO "COMPAR"
7381 034032 104005 5$: ERROR 5 ;WORD NOS 1 TO 4 ARE
7382 ;HEADER WORDS
7383 ;5 TO 260 ARE DATA WORDS
7384 034034 000207 RTS PC ;RETURN TO "COMPAR"
7385
7386
7387 ;*NOW SEE THAT FORMAT ERROR BIT GOT SET
7388
7389 034036 004737 042336 6$: JSR PC,@#PUTREG ;SAVE REGISTERS
7390
7391 034042 022737 100020 001716 CMP #FER!DCK,@#ER1 ;FORMAT ERROR SHOULD BE SET
7392 034050 001401 BEQ TST76 ;BRANCH IF GOOD
7393 034052 104020 ERROR 20 ;A 16 BIT PER WORD READ WAS ATTEMPTED
7394 ;WHEN THE DISK HAD
7395 ;THE FORMAT BIT=0= 18 BITS PER
7396 ;WORD THE READ WAS
7397 ;COMPLETED BUT ERROR REG
7398 ;WAS NOT RIGHT
7399 ;NOTE DCK WILL BE SET BECAUSE
7400 ;ECC HAS NOT BEEN GENERATED
7401
7402
```

```
7403
7404
7405          ;*****
7406          ;*TEST 76      ERROR REGISTER #1-BIT 4 -FORMAT ERROR
7407
7408          ;**      THE SIMULATED DISK HEADER IS FILLED WITH CYLINDER 0
7409          ;**      TRACK 0, SECTOR 0 FORMAT 18 BITS PER WORD
7410          ;**      A WRITE DATA COMMAND IS GIVEN WITH SAME HEADER
7411          ;**      EXCEPT FORMAT BIT.  THE DATA SHOULD NOT BE WRITTEN.
7412          ;*****
7413 034054 000004 1ST76: SCOPE
7414          ;*NOW A WRITE DATA WILL BE ATTEMPTED WITH
7415          ;*WRONG FORMAT BIT
7416
7417 034056 012706 001000      MOV      #STACK,SP      ;RESET STACK
7418
7419 034062 012737 000076 002032      MOV      #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
7420
7421 034070 012737 177777 047732      MOV      #-1,@#NOSYNC ;SET FLAG SO THAT DATA SYNC
7422          ;AND DATA IS NOT READ
7423 034076 004037 042650      FRMAT1: JSR      RO,@#CLAREA ;CLEAR SIMULATED DISK
7424 034102 051534          .WORD    DISK          ;FROM
7425 034104 052560          .WORD    TOLGAP+16    ;TO
7426 034106 000000          .WORD    0            ;DATA
7427          ;*THESE ARE SETUP FOR DISKLESS USE ONLY
7428 034110 005037 047616      CLR      @#CYL          ;CYLINDER 0, FORMAT 18 BIT WORDS
7429 034114 105037 047621      CLR     @#SECOTR+1    ;TRACK 0
7430 034120 105037 047620      CLR     @#SECOTR     ;SECTOR 0
7431 034124 005037 047622      CLR      @#KEY1       ;KEY1 0
7432 034130 005037 047624      CLR      @#KEY2       ;KEY2 0
7433 034134 012737 000004 047664      MOV      #4,@#NOWORD  ;NO OF DATA WORDS
7434 034142 012737 000001 047626      MOV      #1,@#X        ;WRITE DATA
7435 034150 004537 044140      JSR      R5,@#CRC      ;GO TO CALCULATE CRC
7436 034154 052754          WCYL
7437 034156 052764          GCRC
7438
7439          ;*THESE AER REGULAR SETUPS
7440
7441 034160 004037 042650      JSR      RO,@#CLAREA  ;FILL WRITE FROM BUFFER WITH 125252
7442 034164 002114          WRFROM          ;FROM
7443 034166 002122          WRFROM+6       ;TO
7444 034170 125252          125252        ;DATA
7445 034172 004737 042732      JSR      PC,@#CLDISK  ;SETUP GENERAL REGISTERS
7446 034176 012777 177774 145426      MOV      #-4,@#RHWC   ;256 DATA WORDS
7447 034204 012777 002114 145422      MOV      #WRFROM,@#RHBA ;STARTING ADDRESS OF WRITE BUFFER
7448 034212 005077 145426      CLR      @#RHDS       ;TRACK=0 SECTOR=0
7449 034216 012777 010000 145424      MOV      #FMT22,@#RHOF ;16 BITS PER WORD FORMAT
7450 034224 005077 145422      CLR      @#RHCA       ;CYLINDER 0
7451 034230 004737 042766      JSR      PC,@#CHECKT  ;CHECK THAT DVA,RDY,DPR,DRY = 1
7452 034234 104401 062562      TYPE     ,CPHALT     ;AND THAT NO OTHERS = 1.  CANNOT CON-
7453          ;TINUE TESTING IF BOTH AREN'T TRUE
7454 034240 000000          HALT           ;STOP THE TEST
7455 034242 013711 002064      MOV      @#WRIDAT,@R1 ;WRITE DATA=60
7456 034246 005037 002006      CLR      @#ERFLG$    ;CLEAR ERROR FLAG
7457 034252 004737 047456      JSR      PC,@#COMHD   ;WRITE DATA
7458          ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
```

```

7459 ;*FROM THE "COMHD" ROUTINE IT MEANS SECTOR GAP, SYNC BYTE
7460 ;*HEADER, HEADER CRC, HEADER GAP AND SYNC BYTE HAVE GONE BY
7461 ;*AND SYNC'S WERE CORRECTLY DETECTED
7462 ;*DATA IS TO BE CHECKED
7463 034256 004737 042336 JSR PC,@#PUTREG ;SAVE REGISTERS
7464 034262 005737 002006 TST @#ERFLG$ ;HAS ANY ERRORS OCCURED?
7465 034266 001041 BNE 4$ ;BRANCH IF YES
7466 034270 012700 000000 MOV #0,R0 ;GOOD DATA
7467 034274 012701 051534 MOV #DISK,R1 ;DATA WRITTEN INTO "DISK"
7468 034300 012702 000004 MOV #4,R2 ;COUNTER
7469 034304 012737 000005 047736 1$: MOV #5,@#ERWORD ;FOR ERROR WORD
7470 034312 020021 CMP R0,(R1)+ ;COMPARE GOOD DATA WITH DATA ON DISK
7471 034314 001424 BEQ 3$ ;BRANCH IF GOOD
7472 034316 010037 001124 MOV R0,@#$GDDAT ;GOOD DATA
7473 034322 014137 001126 MOV -(R1),@#$BDDAT ;BAD DATA
7474 034326 160237 047736 SUB R2,@#ERWORD ;ERROR WORD NO
7475 034332 005737 002006 TST @#ERFLG$ ;ANY ERRORS ALREADY THERE?
7476 034336 001002 BNE 2$ ;BRANCH IF YES
7477 034340 104004 ERROR 4 ;ERROR ON WRITE DATA COMMAND
7478 ;ON A WRITE DATA WITH
7479 ;WRONG FORMAT NO DATA
7480 ;SHOULD BE WRITTEN
7481 ;WORD NO GIVES WORD IN ERROR
7482 034342 000401 BR 5$ ;BRANCH TO AVOID PRINTING NEXT ERROR
7483 034344 104005 2$: ERROR 5
7484 034346 005721 5$: TST (R1)+ ;UNDO -(R1) FOR BAD DATA
7485 034350 017746 144564 MOV @SWR,-(SP) ;GET SWITCH SETTING
7486 034354 042716 177177 BIC #177177,(SP) ;KEEP ONLY SWITCH 7 AND 8
7487 034360 022726 000200 CMP #SW07,(SP)+ ;IS 7 SET AND 8 RESET.
7488 034364 001402 BEQ 4$ ;BRANCH IF YES
7489 034366 005302 3$: DEC R2 ;IF NOT COUNT 256 WORDS
7490 034370 001345 BNE 1$ ;BRANCH IF 256 NOT DONE
7491
7492 ;*NOW CHECK TO SEE THAT FORMAT ERROR BIT GOT SET
7493
7494 034372 022737 000020 001716 4$: CMP #FER,@#ER1 ;FORMAT ERROR SHOULD BE SET
7495 034400 001401 BEQ TST77 ;BRANCH IF GOOD
7496 034402 104020 ERROR 20 ;A 16 BIT PER WORD WRITE DATA
7497 ;WAS ATTEMPTED WHEN THE DISK
7498 ;HAD THE FORMAT BIT =0=18
7499 ;BITS PER WORD THE WRITE
7500 ;WAS CORRECTLY ABORTED
7501 ;BUT ERROR REG. 1 WAS WRONG
7502

```

7503
7504
7505
7506
7507
7508
7509
7510
7511
7512
7513
7514
7515
7516
7517
7518
7519
7520
7521
7522
7523
7524
7525
7526
7527
7528
7529
7530
7531
7532
7533
7534
7535
7536
7537
7538
7539
7540
7541
7542
7543
7544
7545
7546
7547
7548
7549
7550
7551
7552
7553
7554
7555
7556
7557
7558

*TEST 77 RHER1 - BIT #2 - REG. MODIFICATION REFUSED

** IN THIS TEST THE REGISTERS ARE IN TWO GROUPS
** FIRST - RHCS1,RHDST,RHOF,RHCA,RHER1,RHER2,RHER3 - SETS RMR
** SECOND - RHMR,RHAS - DOES NOT SET RMR
** IF WRITING IS ATTEMPTED DURING AN OPERATION

** ONLY ONE REGISTER IS WRITTEN INTO THAT IS RHCA

** 1 THE REGISTERS CONTENTS ARE SAVED IN "REINTO" BUFFER
** 2 WRITE HEADER AND DATA IS STARTED
** 3 ATTEMPT IS MADE TO WRITE INTO REGISTERS
** 4 ALL REGISTERS ARE COMPARED

TST77: SCOPE

MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
JSR PC,@#CLDISK ;CLEAR DISK
JSR PC,@#CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-
;TINUE TESTING IF BOTH AREN'T TRUE
;STOP THE TEST

1\$: MOV #RHCA,R0
2\$: MOV (R0)+,R5 ;R5 HAS ADDRESS OF REG. UNDER TEST
BIS #CLR,@RHCS2
MOV @#UNIT,@RHCS2 ;REINSTATE UNIT NO.

;*SET UP FOR AN OPERATION (WRITE HEADER AND DATA)

MOV @#WRIFOR,@RHCS1 ;WRITE HEADER AND DATA=62
;IN RHCS1
MOV #-10.,@RHWC ;10 WORDS
MOV #WRFROM,@RHBA ;BUS ADDRESS = WRFROM
MOV #10,@RHDST ;DESIRED TRACK=0, SECTOR=10
BIS #BA1,@RHCS2 ;BUS ADDRESS INCREMENT INHIBIT
MOV #FMT22,@RHOF ;FORMAT 16 BIT WORDS
CLR @RHCA ;CYLINDER =0

;*SAVE REGISTERS

JSR R0,@#SAVER ;SAVE
RHCS1 ;FROM
REINTO ;TC
14. ;NUMBER OF REGISTERS SAVED

;*NOW THE COMMAND IS GIVEN TO
;*WRITE HEADER AND DATA FOR CYL=0, SECTOR=10
;*TRACK=0 IT COMES BACK AFTER ONE SECTOR
;*HAS PASSED

C 14

CZRJGDO,RP04/5/6 DSKLS CTRLR1 MACY11 30A(1052) 10-SEP-79 11:11 PAGE 172
 CZRJGD.P11 10-SEP-79 11:00 177 RHER1 - BIT #2 - REG. MODIFICATION REFUSED SEQ 0171

```

7559 034542 012777 000001 145110      MOV    #DMD,@RHMR      ;SET DIAGNOSTIC MODE
7560 034550 005277 145064              INC    @RHCS1          ;GO TO RHCS1 WITH 62
7561 034554 012715 177672              MOV    #177672,@R5    ;TRY WRITING ALL BITS EXCEPT
7562                                ;GO, RMR, IE
7563 034560 052737 000001 003200      BIS    #DMD,@#REINTO+20 ;SET DMD IN SAVED REGISTER RHMR
7564 034566 052737 000004 003162      BIS    #RMR,@#REINTO+2 ;SET RMR IN SAVED REG. RHER1
7565 034574 042737 000200 003202      BIC    #DRY,@#REINTO+22 ;CLEAR DRY IN RHDS1
7566 034602 052737 040000 003202      BIS    #ERR,@#REINTO+22 ;SET ERR IN RHDS1
7567 034610 052737 000001 003160      BIS    #GO,@#REINTO    ;SET GO IN SAVED REG. RHCS1
7568 034616 042737 000200 003160      BIC    #RDY,@#REINTO   ;CLEAR RDY BIT
7569
7570                                ;*AFTER AN ATTEMPT TO WRITE INTO A REGISTER
7571                                ;*SAVE REGISTERS AGAIN SO THAT COMPARES CAN BE DONE
7572
7573 034624 004037 043424              JSR    RO,@#SAVER      ;SAVE
7574 034630 001640                      RHCS1                    ;FROM
7575 034632 002114                      WRFROM                  ;TO
7576 034634 000016                      14.                    ;NUMBER
7577
7578                                ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
7579                                ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
7580                                ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
7581 034636 113737 003177 002133      MOVB   @#REINTO+17,@#WRFROM+17;SAVE UPPER RHAS
7582
7583
7584                                ;*COMPARE REGISTERS BEFORE ATTEMPTED WRITE WITH AFTER
7585
7586 034644 004037 043626              JSR    RO,@#COMPAR     ;COMPAR
7587 034650 003160                      REINTO                  ;GO BUFFER
7588 034652 002114                      WRFROM                  ;TEST BUFFER
7589 034654 000016                      14.                    ;NUMBER
7590 034656 034664                      4$                      ;RETURN FOR ERROR
7591 034660 034664                      4$                      ;SAME
7592 034662 034704                      5$                      ;RETURN FOR GOOD COMPARISON
7593 034664 013705 047736              4$: MOV    @#ERWORD,R5   ;GETTING READY TO INDEX
7594 034670 060505                      ADD    R5,R5            ;DOUBLE ERROR WORD
7595 034672 016537 001636 042402      MOV    RHCS1-2(R5),@#RECADR ;FAILING REG. ADDRESS
7596 034700 104001                      ERROR 1                 ;CONTENTS OF REGISTER
7597 034702 000207                      RTS    PC               ;CHANGED WITH
7598                                ;AN ATTEMPT TO WRITE
7599                                ;DURING AN OPERATION
7600                                ;*THE FOLLOWING CLEAR MAY SET THE ATA BIT BECAUSE GO IS HIGH
7601                                ;*
7602 034704 004737 042732              5$: JSR    PC,@#CLDISK  ;CLEAR DISK
7603
7604

```

```
7605
7606
7607
7608
7609
7610 034710 000004
7611 034712 012706 001000
7612 034716 012737 000100 002032
7613 034724 004737 042732
7614 034730 012777 000001 144722
7615 034736 004037 045304
7616
7617
7618 034742 000001
7619
7620
7621
7622
7623
7624
7625
7626
7627
7628
7629
7630
7631
7632
7633
7634
7635 034744 000004
7636
7637
7638 034746 012706 001000
7639 034752 012737 000101 002032
7640 034760 005037 047732
7641
7642
7643
7644
7645 034764 004737 044414
7646
7647
7648
7649 034770 004037 042650
7650 034774 003160
7651 034776 004160
7652 035000 000000
7653
7654
7655
7656 035002 012700 002114
7657 035006 012720 010000
7658 035012 012720 000401
7659 035016 012720 000001
7660 035022 012720 000001
```

```
*****
:*TEST 100 MAKE CURRENT CYLINDER = 1
*****
TST100: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
JSR PC,@#CLDISK ;INIT DRIVE
MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE
JSR RO,@#MAKECYL ;SUBROUTINE TO GIVE A SEEK
;COMMAND FOLOWED BY AN INIT
;THIS SHOULD CHANGE RHCC
;CHANGE RHCC TO 1

*****
:*TEST 101 ERROR REG1 - BIT #7 - HEADER COMPARE ERROR
** THE SIMULATED DISK IS SET TO READ CYLINDER=0, TRACK=1
** SECTOR=1, KEYS=1, 256 WORDS OF 177400
** A READ HEADER AND DATA COMMAND IS GIVEN TO READ
** CYLINDER=1, TRACK=1, SECTOR=1, KEY1=1, KEY2=1
** REINTO BUFFER IS FILLED WITH 0
** WRFROM IS FILLED WITH 10000,401,1,1,1, AND ALL 177400
** AFTER THE READ THE REINTO BUFFER IS EXPECTED TO
** HAVE WHAT IS IN WRFROM - 10000,401,1,1 AND ALL 177400
*****
TST101: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
CLR @#NOSYNC ;SET FLAG SO THAT DATA SYNC
;AND DATA IS READ
;*FILL SIMULATED DISK
JSR PC,@#SETDSK ;SET UP SIMULATED DISK
;*FILL REINTO BUFFER WITH 0
JSR RO,@#CLAREA ;FILL REINTO BUFFER
REINTO ;FROM LOCATION
REINTO+<256.*2> ;TO LOCATION
0 ;DATA
;*FILL WRFROM WITH 10000,401,1,1, AND ALL 177400
MOV #WRFROM,RO
MOV #FMT22,(RO)+ ;10000 INTO WRFROM
MOV #401,(RO)+ ;401=TRACK1,SECTOR1
MOV #1,(RO)+ ;1 INTO WRFROM+
MOV #1,(RO)+ ;1 INTO WRFROM+6
```

7661
7662
7663
7664
7665
7666
7667
7668
7669
7670
7671
7672
7673
7674
7675
7676
7677
7678
7679
7680
7681
7682
7683
7684
7685
7686

035026 004037 042650
035032 002124
035034 003114
035036 177400

035040 004037 044542
035044 000072
035046 000001
035050 000001
035052 000001
035054 177400
035056 003160
035060 000000

035062 000001
035064 000240

```
;*FILL ALL 0  
JSR RO,@#CLAREA ;FILL WRFROM  
WRFROM+10 ;FROM  
WRFROM+<256.*2> ;TO  
177400 ;DATA  
  
;*NOW GIVE A READ HEADER AND DATA COMMAND  
;*CYLINDER=1  
;*TRACK = 1  
;*SECTOR = 1  
  
JSR RO,@#HCCRCE ;READ HEADER AND DATA  
72 ;CYLINDER  
1 ;SECTOR  
1 ;TRACK  
-256. ;WORD COUNT  
REINTO ;RHBA BUFFER  
0 ;READ  
  
1 ;HEADER COMPARE  
1$: NOP ;RETURN POINT FROM HCCRCE
```

7687
7688
7689
7690
7691
7692
7693
7694
7695
7696
7697
7698
7699
7700
7701
7702
7703
7704
7705
7706
7707
7708
7709
7710
7711
7712
7713
7714
7715
7716
7717
7718
7719
7720
7721
7722
7723
7724
7725
7726
7727
7728
7729
7730
7731
7732
7733
7734
7735
7736
7737
7738
7739
7740
7741
7742

```
*****  
:*TEST 102 MAKE CURRENT CYLINDER = 0  
*****  
TST102: SCOPE  
MOV #STACK,SP ;RESET STACK  
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER  
JSR PC,@#CLDISK ;INIT DRIVE  
MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE  
JSR RO,@#MAKECYL ;SUBROUTINE TO GIVE A SEEK  
;COMMAND FOLOWED BY AN INIT  
;THIS SHOULD CHANGE RHCC  
;CHANGE RHCC TO 0  
0  
  
*****  
:*TEST 103 ERROR REG1 - BIT #7 - HEADER COMPARE ERROR  
  
:** THE SIMULATED DISK IS SET TO READ CYLINDER=0, TRACK=1  
:** SECTOR=1, KEYS=1, 256 WORDS OF 177400  
:** A READ HEADER AND DATA COMMAND IS GIVEN TO READ  
:** CYLINDER=0, TRACK=0, SECTOR=1, KEY1=1, KEY2=1  
:** REINTO BUFFER IS FILLED WITH 0  
:** WRFROM IS FILLED WITH 10000,401,1,1,1, AND ALL 177400  
:** AFTER THE READ THE REINTO BUFFER IS EXPECTED TO  
:** HAVE WHAT IS IN WRFROM - 10000,401,1,1 AND ALL 177400  
  
*****  
TST103: SCOPE  
  
MOV #STACK,SP ;RESET STACK  
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER  
CLR @#NOSYNC ;SET FLAG SO THAT DATA SYNC  
;AND DATA IS READ  
  
;*FILL SIMULATED DISK  
JSR PC,@#SETDSK ;SET UP SIMULATED DISK  
  
;*FILL REINTO BUFFER WITH 0  
JSR RO,@#CLAREA ;FILL REINTO BUFFER  
REINTO ;FROM LOCATION  
REINTO+<256.*2> ;TO LOCATION  
0 ;DATA  
  
;*FILL WRFROM WITH 10000,401,1,1, AND ALL 177400  
MOV #WRFROM,RO  
MOV #FMT22,(RO)+ ;10000 INTO WRFROM  
MOV #401,(RO)+ ;401=TRACK1,SECTOR1  
MOV #1,(RO)+ ;1 INTO WRFROM+  
MOV #1,(RO)+ ;1 INTO WRFROM+6
```

002032
144544

002032


```
7743 ;*FILL ALL 0
7744
7745 035204 004037 042650 JSR RO,@#CLAREA ;FILL WRFROM
7746 035210 002124 WRFROM+10 ;FROM
7747 035212 003114 WRFROM+<256.*2> ;TO
7748 035214 177400 177400 ;DATA
7749
7750 ;*NOW GIVE A READ HEADER AND DATA COMMAND
7751 ;*CYLINDER=0
7752 ;*TRACK = 0
7753 ;*SECTOR = 1
7754
7755 035216 004037 044542 JSR RO,@#HCCRCE
7756 035222 000072 72 ;READ HEADER AND DATA
7757 035224 000000 0 ;CYLINDER
7758 035226 000001 1 ;SECTOR
7759 035230 000000 0 ;TRACK
7760 035232 177400 -256. ;WORD COUNT
7761 035234 003160 REINTO ;RHBA BUFFER
7762 035236 000000 0 ;READ
7763
7764 035240 000001 1 ;HEADER COMPARE
7765 035242 000240 1$: NOP ;RETURN POINT FROM HCCRCE
7766
7767
```

```
7768
7769
7770
7771
7772 035244 000004
7773 035246 012706 001000
7774 035252 012737 000104 002032
7775 035260 004737 042732
7776 035264 012777 000001 144366
7777 035272 004037 045304
7778
7779
7780 035276 000001
7781
7782
7783
7784
7785
7786
7787
7788
7789
7790
7791
7792
7793
7794
7795
7796
7797 035300 000004
7798
7799
7800 035302 012706 001000
7801 035306 012737 000105 002032
7802
7803 035314 012737 177777 047732
7804
7805
7806
7807
7808 035322 004737 044414
7809
7810
7811
7812 035326 004037 042650
7813 035332 002114
7814 035334 003114
7815 035336 125252
7816
7817
7818
7819
7820
7821 035340 004037 042650
7822 035344 003160
7823 035346 004160

*****
*TEST 104 MAKE CURRENT CYLINDER = 1
*****
TST104: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
JSR PC,@#CLDISK ;INIT DRIVE
MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE
JSR RO,@#MAKECYL ;SUBROUTINE TO GIVE A SEEK
;COMMAND FOLOWED BY AN INIT
;THIS SHOULD CHANGE RHCC
;CHANGE RHCC TO 1
1

*****
*TEST 105 ERROR REG.1 - BIT #7 - HEADER COMPARE ERROR
*****
** THE SIMULATED DISK IS SET UP FOR CYLINDER=0, TRACK=1
** SECTOR=1, KEYS=1, 256 WORDS OF 177400
** A WRITE DATA COMMAND IS GIVEN TO WRITE CYLINDER=1
** TRACK=1, SECTOR=1, KEY1=1, KEY2=1
** WRFROM BUFFER IS FILLED WITH 125252
** REINTO BUFFER IS FILLED WITH 177400
** AFTER THE WRITE COMMAND THE DISK IS EXPECTED TO
** HAVE 177400
*****
TST105: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
MOV #-1,@#NOSYNC ;SET FLAG SO THAT DATA SYNC
;AND DATA IS NOT READ
;*FILL SIMULATED DISK
JSR PC,@#SETDSK ;SETUP SIMULATED DISK
;*FILL WRFROM WITH 125252
JSR RO,@#CLAREA ;FILL WRFROM BUFFER
WRFROM ;FROM LOCATION
WRFROM+<256.*2> ;TO LOCATION
125252 ;DATA
;*FILL REINTO WITH 256 WORDS OF 177400
;*THIS IS WHAT IS EXPECTED TO BE ON DISK EVEN AFTER
;*AN ATTEMPT TO WRITE 125252
JSR RO,@#CLAREA ;FILL REINTO BUFFER
REINTO ;FROM LOCATION
REINTO+<256.*2> ;TO
```

CZRJGDO,RP04/5/6 DSKLS CTRLR1
CZRJGD.P11 10-SEP-79 11:00

MACY11 30A(1052) 10-SEP-79 11:11 PAGE 178
T105 ERROR REG.1 - BIT #7 - HEADER COMPARE ERROR

I 14

SEQ 0177

```
7824 035350 177400 177400
7825
7826 ;*NOW GIVE A WRITE DATA COMMAND
7827 ;*CYLINDER = 1,
7828 ;*TRACK = 1
7829 ;*SECTOR = 1
7830
7831 035352 004037 044542 JSR RO,@#HCCRCE
7832 035356 000060 60 ;WRITE DATA
7833 035360 000001 1 ;CYLINDER
7834 035362 000001 1 ;SECTOR
7835 035364 000001 1 ;TRACK
7836 035366 177400 -256. ;WORD COUNT
7837 035370 002114 WRFROM ;RHBA BUFFER
7838 035372 000001 1 ;WRITE
7839
7840 035374 000001 1 ;HEADER COMPARE
7841 035376 000240 1$: NOP ;RETURN POINT FROM HCCRCE
7842
```

```
7843
7844
7845
7846
7847
7848
7849 035400 000004
7850 035402 012706 001000
7851 035406 012737 000106 002032
7852 035414 004737 042732
7853 035420 012777 000001 144232
7854 035426 004037 045304
7855
7856
7857 035432 000000
7858
7859
7860
7861
7862
7863
7864
7865
7866
7867
7868
7869
7870
7871
7872
7873
7874 035434 000004
7875
7876
7877 035436 012706 001000
7878 035442 012737 000107 002032
7879
7880 035450 012737 177777 047732
7881
7882
7883
7884
7885 035456 004737 044414
7886
7887
7888
7889 035462 004037 042650
7890 035466 002114
7891 035470 003114
7892 035472 125252
7893
7894
7895
7896
7897
7898 035474 004037 042650

:*****
:*TEST 106 MAKE CURRENT CYLINDER = 0
:*****
TST106: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
JSR PC,@#CLDISK ;INIT DRIVE
MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE
JSR RO,@#MAKECYL ;SUBROUTINE TO GIVE A SEEK
;COMMAND FOLOWED BY AN INIT
;THIS SHOULD CHANGE RHCC
;CHANGE RHCC TO 0

:*****
:*TEST 107 ERROR REG.1 - BIT #7 - HEADER COMPARE ERROR
:*****
:** THE SIMULATED DISK IS SET UP FOR CYLINDER=0, TRACK=1
:** SECTOR=1, KEYS=1, 256 WORDS OF 177400
:** A WRITE DATA COMMAND IS GIVEN TO WRITE CYLINDER=0
:** TRACK=0, SECTOR=1, KEY1=1, KEY2=1
:** WRFROM BUFFER IS FILLED WITH 125252
:** REINTO BUFFER IS FILLED WITH 177400
:** AFTER THE WRITE COMMAND THE DISK IS EXPECTED TO
:** HAVE 177400
:*****
TST107: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
MOV #-1,@#NOSYNC ;SET FLAG SO THAT DATA SYNC
;AND DATA IS NOT READ
;*FILL SIMULATED DISK
JSR PC,@#SETDSK ;SETUP SIMULATED DISK
;*FILL WRFROM WITH 125252
JSR RO,@#CLAREA ;FILL WRFROM BUFFER
WRFROM ;FROM LOCATION
WRFROM+<256.*2> ;TO LOCATION
125252 ;DATA
;*FILL REINTO WITH 256 WORDS OF 177400
;*THIS IS WHAT IS EXPECTED TO BE ON DISK EVEN AFTER
;*AN ATTEMPT TO WRITE 125252
JSR RO,@#CLAREA ;FILL REINTO BUFFER
```

CZRJGDO,RP04/5/6 DSKLS CTRLR1
CZRJGD.P11 10-SEP-79 11:00

MACY11 30A(1052) 10-SEP-79 11:11 PAGE 180
T107 ERROR REG.1 - BIT #7 - HEADER COMPARE ERROR

SEQ 0179

```

7899 035500 003160          REINTO          ;FROM LOCATION
7900 035502 004160          REINTO+<256.*2> ;TO
7901 035504 177400          177400
7902
7903                          ;*NOW GIVE A WRITE DATA COMMAND
7904                          ;*CYLINDER = 0,
7905                          ;*TRACK = 0
7906                          ;*SECTOR      = 1
7907
7908 035506 004037 044542    JSR      RO,@#HCCRCE
7909 035512 000060          60          ;WRITE DATA
7910 035514 000000          0          ;CYLINDER
7911 035516 000001          1          ;SECTOR
7912 035520 000000          0          ;TRACK
7913 035522 177400          -256.    ;WORD COUNT
7914 035524 002114          WRFROM    ;RHBA BUFFER
7915 035526 000001          1          ;WRITE
7916
7917 035530 000001          1          ;HEADER COMPARE
7918 035532 000240          1$:      NOP    ;RETURN POINT FROM HCCRCE
7919
```

7920
7921
7922
7923
7924
7925
7926
7927
7928
7929
7930
7931
7932
7933
7934
7935
7936
7937
7938
7939
7940
7941
7942
7943
7944
7945
7946
7947
7948
7949
7950
7951
7952
7953
7954
7955
7956
7957
7958
7959
7960
7961
7962
7963
7964
7965
7966
7967
7968
7969
7970
7971
7972
7973
7974
7975

035534 000004

035536 012706 001000
035542 012737 000110
035550 005037 047732

035554 004737 044414
035560 005137 051516

035564 004037 042650
035570 003160
035572 004160
035574 000000

035576 012700 002114
035602 012720 010000
035606 012720 000401
035612 012720 000001
035616 012720 000001

035622 004037 042650
035626 002124
035630 003114
035632 177400

035634 004037 044542

002032

```
*****  
*TEST 110 RHER1 - BIT #8 - CRC ERROR (READING)  
*****  
** THE SIMULATED DISK IS SET TO READ CYLINDER=0, TRACK=1  
** SECTOR=1, KEYS=1, 256 WORDS OF 177400  
** A READ HEADER AND DATA COMMAND IS GIVEN TO READ  
** CYLINDER=0, TRACK=1, SECTOR=1, KEY1=1, KEY2=1  
** REINTO BUFFER IS FILLED WITH 0  
** WRFROM IS FILLED WITH 10000,401,1,1,1, AND ALL 177400  
** AFTER THE READ THE REINTO BUFFER IS EXPECTED TO  
** HAVE WHAT IS IN WRFROM - 10000,401,1,1 AND ALL 177400  
*****  
TST110: SCOPE  
  
MOV #STACK,SP ;RESET STACK  
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER  
CLR @#NOSYNC ;SET FLAG SO THAT DATA SYNC  
;AND DATA IS READ  
  
;*FILL SIMULATED DISK  
JSR PC,@#SETDSK ;SET UP SIMULATED DISK  
COM @#WCRC ;CHANCE CRC TO GIVE HCRC  
  
;*FILL REINTO BUFFER WITH 0  
JSR R0,@#CLAREA ;FILL REINTO BUFFER  
REINTO ;FROM LOCATION  
REINTO+<256.*2> ;TO LOCATION  
0 ;DATA  
  
;*FILL WRFROM WITH 10000,401,1,1, AND ALL 177400  
MOV #WRFROM,R0  
MOV #FMT22,(R0)+ ;10000 INTO WRFROM  
MOV #401,(R0)+ ;401=TRACK1,SECTOR1  
MOV #1,(R0)+ ;1 INTO WRFROM+  
MOV #1,(R0)+ ;1 INTO WRFROM+6  
  
;*FILL ALL 0  
JSR R0,@#CLAREA ;FILL WRFROM  
WRFROM+10 ;FROM  
WRFROM+<256.*2> ;TO  
177400 ;DATA  
  
;*NOW GIVE A READ HEADER AND DATA COMMAND  
;*CYLINDER=0  
;*TRACK = 1  
;*SECTOR = 1  
JSR R0,@#HCCRCE
```

CZRJGDO,RP04/5/6 DSKLS CTRLR1
CZRJGD.P11 10-SEP-79 11:00

M 14
MACY11 30A(1052) 10-SEP-79 11:11 PAGE 182
T110 RHER1 - BIT #8 - CRC ERROR (READING)

SEQ 0181

| | | | | | |
|------|--------|--------|----------|--|---------------------------|
| 7976 | 035640 | 000072 | 72 | | :READ HEADER AND DATA |
| 7977 | 035642 | 000000 | 0 | | :CYLINDER |
| 7978 | 035644 | 000001 | 1 | | :SECTOR |
| 7979 | 035646 | 000001 | 1 | | :TRACK |
| 7980 | 035650 | 177400 | -256. | | :WORD COUNT |
| 7981 | 035652 | 003160 | REINTO | | :RHBA BUFFER |
| 7982 | 035654 | 000000 | 0 | | :READ |
| 7983 | | | | | |
| 7984 | 035656 | 000000 | 0 | | :CRC ERROR |
| 7985 | 035660 | 000240 | 1\$: NOP | | :RETURN POINT FROM HCCRCE |
| 7986 | | | | | |
| 7987 | | | | | |

7988
7989
7990
7991
7992
7993
7994
7995
7996
7997
7998
7999
8000
8001
8002
8003
8004
8005
8006
8007
8008
8009
8010
8011
8012
8013
8014
8015
8016
8017
8018
8019
8020
8021
8022
8023
8024
8025
8026
8027
8028
8029
8030
8031
8032
8033
8034
8035
8036
8037
8038
8039
8040
8041
8042
8043

035662 000004
035664 012706 001000
035670 012737 000111 002032
035676 012737 177777 047732
035704 004737 044414
035710 005137 051516
035714 004037 042650
035720 002114
035722 003114
035724 125252
035726 004037 042650
035732 003160
035734 004160
035736 177400
035740 004037 044542
035744 000060
035746 000000
035750 000001
035752 000001
035754 177400

```
*****  
*TEST 111 RHER1 - BIT 8 - CRC ERROR (WRITING)  
** THE SIMULATED DISK IS SET UP FOR CYLINDER=0, TRACK=1  
** SECTOR=1, KEYS=1, 256 WORDS OF 177400  
** A WRITE DATA COMMAND IS GIVEN TO WRITE CYLINDER=0  
** TRACK=1, SECTOR=1, KEY1=1, KEY2=1  
** WRFROM BUFFER IS FILLED WITH 125252  
** REINTO BUFFER IS FILLED WITH 177400  
** AFTER THE WRITE COMMAND THE DISK IS EXPECTED TO  
** HAVE 177400  
*****  
TST111: SCOPE  
MOV #STACK,SP ;RESET STACK  
MOV #TNO,@#TSTNM ;THIS SAVES TEST NUMBER  
MOV #-1,@#NOSYNC ;SET FLAG SO THAT DATA SYNC  
;AND DATA IS NOT READ  
;*FILL SIMULATED DISK  
JSR PC,@#SETDSK ;SETUP SIMULATED DISK  
COM @#WCRC ;CHANGE CRC TO GIVE HCRC  
;*FILL WRFROM WITH 125252  
JSR RO,@#CLAREA ;FILL WRFROM BUFFER  
WRFROM ;FROM LOCATION  
WRFROM+<256.*2> ;TO LOCATION  
125252 ;DATA  
;*FILL REINTO WITH 256 WORDS OF 177400  
;*THIS IS WHAT IS EXPECTED TO BE ON DISK EVEN AFTER  
;*AN ATTEMPT TO WRITE 125252  
JSR RO,@#CLAREA ;FILL REINTO BUFFER  
REINTO ;FROM LOCATION  
REINTO+<256.*2> ;TO  
177400  
;*NOW GIVE A WRITE DATA COMMAND  
;*CYLINDER = 0,  
;*TRACK = 1  
;*SECTOR = 1  
JSR RO,@#HCCRCE ;WRITE DATA  
60 ;CYLINDER  
0 ;SECTOR  
1 ;TRACK  
1 ;TRACK  
-256. ;WORD COUNT
```


| | | | | | | |
|------|--------|--------|------|--------|--|---------------------------|
| 8044 | 035756 | 002114 | | WRFROM | | ;RHBA BUFFER |
| 8045 | 035760 | 000001 | | 1 | | ;WRITE |
| 8046 | | | | | | |
| 8047 | 035762 | 000000 | | 0 | | ;CRC ERROR |
| 8048 | 035764 | 000240 | 1\$: | NOP | | ;RETURN POINT FROM HCCRCE |
| 8049 | | | | | | |
| 8050 | | | | | | |

8051
8052
8053
8054
8055 035766 005737 002036
8056 035772 001401
8057 035774 000402
8058 035776 000137 036452
8059 036002
8060
8061
8062
8063
8064
8065
8066
8067 036002 000004
8068 036004 012706 001000
8069 036010 012737 000112 002032
8070 036016 004737 042732
8071 036022 012777 000001 143630
8072 036030 004037 045304
8073
8074
8075 036034 001456
8076
8077
8078
8079
8080
8081
8082
8083
8084
8085
8086
8087
8088
8089
8090
8091
8092 036036 000004
8093
8094
8095 036040 012706 001000
8096 036044 012737 000113 002032
8097
8098 036052 004037 042650
8099 036056 051534
8100 036060 052560
8101 036062 000000
8102
8103
8104
8105 036064 012737 011456 047616
8106

;*SET UP FOR THE TWO LAST SECTOR TRANSFERRED TESTS FOLLOWING

```
::*****  
TST @#RP06 ;MOVE RP06 FLAG TO ITSELF TO TEST  
BEQ 2$ ;IF = 0 TREAT DRIVE AS RP04  
BR 3$ ;TREAT AS RP06 - DO NEXT 'MAKECL' & TEST  
2$: JMP @#DOG ;DO SECOND FOLLOWING 'MAKECL' AND TEST  
3$:  
::*****
```

```
::*****  
;*TEST 112 MAKE CURRENT CYLINDER = 814.  
::*****
```

```
TST112: SCOPE  
MOV #STACK,SP ;RESET STACK  
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER  
JSR PC,@#CLDISK ;INIT DRIVE  
MOV #DMD,@#RHRM ;SET DIAGNOSTIC MODE  
JSR RO,@#MAKECYL ;SUBROUTINE TO GIVE A SEEK  
;COMMAND FOLLOED BY AN INIT  
;THIS SHOULD CHANGE RHCC  
;CHANGE RHCC TO 814.  
814.
```

```
::*****  
;*TEST 113 RHDS1 (BIT #10) - LAST SECTOR TRANSFERRED, 'LST'
```

```
;** WRITE CYLINDER 814., FORMAT 16 BITS PER WORD  
;** TRACK 18., SECTOR 21., KEYS 0, NUMBER OF WORDS  
;** 256., OF 377  
;** 'LST' BIT # 10 - RHDS1, SHOULD SET AFTER WRITE  
;** IS COMPLETE.
```

```
::*****  
TST113: SCOPE
```

```
MOV #STACK,SP ;RESET STACK  
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER  
JSR RO,@#CLAREA ;CLEAR SIMULATED DISK  
.WORD DISK ;FROM  
.WORD TOLGAP+16 ;TO  
.WORD 0 ;DATA
```

;*THESE ARE SETUP FOR DISKLESS USE ONLY

```
MOV #814.!FMT22,@#CYL;CYLINDER 814.  
;16 BITS PER WORD
```

| | | | | | | | | |
|------|--------|--------|--------|--------|------|-----------------|------------------|--------------------------------------|
| 8107 | 036072 | 112737 | 000022 | 047621 | | MOVB | #18.,@#SECOTR+1; | TRACK 18. |
| 8108 | 036100 | 112737 | 000025 | 047620 | | MOVB | #21.,@#SECOTR | ;SECTOR 21. |
| 8109 | 036106 | 005037 | 047622 | | | CLR | @#KEY1 | ;KEY1 0 |
| 8110 | 036112 | 005037 | 047624 | | | CLR | @#KEY2 | ;KEY2 0 |
| 8111 | 036116 | 012737 | 000400 | 047664 | | MOV | #256.,@#NOWORD | ;NO OF DATA WORDS |
| 8112 | 036124 | 012737 | 000001 | 047626 | | MOV | #1,@#X | ;WRITE DATA |
| 8113 | 036132 | 004537 | 044140 | | | JSR | R5,@#CRC | ;GO TO CALCULATE CRC |
| 8114 | 036136 | 047616 | | | | CYL | | |
| 8115 | 036140 | 051516 | | | | WCRC | | |
| 8116 | | | | | | | | |
| 8117 | | | | | | | | |
| 8118 | | | | | | | | |
| 8119 | 036142 | 004037 | 042650 | | | JSR | R0,@#CLAREA | ;FILL WRITE BUFFER WITH 377 |
| 8120 | 036146 | 002114 | | | | WRFROM | | ;FROM LOCATION |
| 8121 | 036150 | 003114 | | | | WRFROM+<256.*2> | | ;TO LOCATION |
| 8122 | 036152 | 000377 | | | | 377 | | ;DATA |
| 8123 | 036154 | 004737 | 042732 | | | JSR | PC,@#CLDISK | ;SETUP GENERAL REGISTERS |
| 8124 | 036160 | 012777 | 177400 | 143444 | | MOV | #-256.,@RHWC | ;256. DATA WORDS |
| 8125 | 036166 | 012777 | 002114 | 143440 | | MOV | #WRFROM,@RHBA | ;STARTING ADDRESS OF WRITE BUFFER |
| 8126 | 036174 | 012746 | 000025 | | | MOV | #21.,-(SP) | ;SECTOR 21. |
| 8127 | 036200 | 112766 | 000022 | 000001 | | MOVB | #18.,1(SP) | ;TRACK 18. |
| 8128 | 036206 | 012677 | 143432 | | | MOV | (SP)+,@RHDS1 | ;SECTOR 21. TRACK 18. |
| 8129 | 036212 | 012777 | 010000 | 143430 | | MOV | #FMT22,@RHOF | ;16 BITS PER WORD FORMAT |
| 8130 | 036220 | 012777 | 001456 | 143424 | | MOV | #814.,@RHCA | ;CYLINDER 814. |
| 8131 | 036226 | 004737 | 042766 | | | JSR | PC,@#CHECKT | ;CHECK THAT DVA,RDY,DPR,DRY = 1 |
| 8132 | 036232 | 104401 | 062562 | | | TYPE | ,CPHALT | ;AND THAT NO OTHERS = 1. CANNOT CON- |
| 8133 | | | | | | | | ;TINUE TESTING IF BOTH AREN'T TRUE |
| 8134 | 036236 | 000000 | | | | HALT | | ;STOP THE TEST |
| 8135 | 036240 | 013711 | 002064 | | | MOV | @#WRIDAT,@R1 | ;WRITE DATA=60 |
| 8136 | 036244 | 005037 | 002006 | | | CLR | @#ERFLG\$ | ;CLEAR ERROR FLAG |
| 8137 | 036250 | 004737 | 047456 | | | JSR | PC,@#COMHD | ;WRITE DATA |
| 8138 | | | | | | | | |
| 8139 | | | | | | | | |
| 8140 | | | | | | | | |
| 8141 | | | | | | | | |
| 8142 | | | | | | | | |
| 8143 | | | | | | | | |
| 8144 | | | | | | | | |
| 8145 | 036254 | 004737 | 042336 | | | JSR | PC,@#PUTREG | ;SAVE REGISTERS |
| 8146 | 036260 | 005737 | 002006 | | | TST | @#ERFLG\$ | ;HAVE ANY ERRORS OCCURED? |
| 8147 | 036264 | 001062 | | | | BNE | 5\$ | ;BRANCH IF YES |
| 8148 | 036266 | 012700 | 000377 | | | MOV | #377,R0 | ;GOOD DATA |
| 8149 | 036272 | 012701 | 051534 | | | MOV | #DISK,R1 | ;DATA WRITTEN INTO "DISK" |
| 8150 | 036276 | 012702 | 000400 | | | MOV | #256.,R2 | ;COUNTER |
| 8151 | | | | | | | | |
| 8152 | 036302 | 012737 | 000401 | 047736 | 1\$: | MOV | #256.+1,@#ERWORD | ;FOR ERROR WORD |
| 8153 | 036310 | 020021 | | | | CMP | R0,(R1)+ | ;COMPARE GOOD DATA WITH DATA ON DISK |
| 8154 | 036312 | 001424 | | | | BEQ | 3\$ | ;BRANCH IF GOOD |
| 8155 | 036314 | 010037 | 001124 | | | MOV | R0,@#\$GDDAT | ;GOOD DATA |
| 8156 | 036320 | 014137 | 001126 | | | MOV | -(R1),@#\$BDDAT | ;BAD DATA |
| 8157 | 036324 | 160237 | 047736 | | | SUB | R2,@#ERWORD | ;ERROR WORD NO |
| 8158 | 036330 | 005737 | 002006 | | | TST | @#ERFLG\$ | ;ANY ERRORS ALREADY THERE? |
| 8159 | 036334 | 001002 | | | | BNE | 2\$ | ;BRANCH IF YES |
| 8160 | 036336 | 104004 | | | | ERROR | 4 | ;ERROR ON WRITE DATA COMMAND |
| 8161 | 036340 | 000401 | | | | BR | 64\$ | ;BRANCH TO AVOID PRINTING NEXT ERROR |
| 8162 | | | | | | | | |

*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
 *FROM THE "COMHD" ROUTINE IT MEANS SECTOR GAP, SYNC BYTE
 *HEADER, HEADER CRC, HEADER GAP AND SYNC BYTE HAVE GONE BY
 *AND SYNC'S WERE CORRECTLY DETECTED, DATA IS TO BE CHECKED.

```
8163 036342 104005      2$:  ERROR 5          ;WORD NO GIVES WORD IN ERROR
8164 036344 005721      64$: TST (R1)+       ;UNDO -(R1) FOR BAD DATA
8165 036346 017746 142566  MOV @SWR,-(SP)      ;GET SWITCH SETTING
8166 036352 042716 177177  BIC #177177,(SP)   ;KEEP ONLY SWITCH 7 AND 8
8167 036356 022726 000200  CMP #SW07,(SP)+    ;IS 7 SET AND 8 RESET
8168 036362 001402      BEQ 4$             ;BRANCH OUT IF YES
8169 036364 005302      3$:  DEC R2        ;IF NOT COUNT 256 WORDS
8170 036366 001345      BNE 1$            ;BRANCH IF 256. NOT DONE
8171
8172
8173 036370 013746 001736      4$:  MOV @#DS1,-(SP)   ;GET RHDS1
8174 036374 042716 001000      BIC #PROG,(SP)     ;CLEAR PROG
8175 036400 022726 002700      CMP #LST!DPR!DRY!VV,(SP)+;IS 'LST' HIGH ?
8176 036404 001412      BEQ 5$             ;BRANCH IF GOOD
8177 036406 013737 001662 042402  MOV @#RHDS1,@#REGADR ;FAILING REG. ADDRESS
8178 036414 012737 002700 001124  MOV #LST!DPR!DRY!VV,@#SGDDAT ;GOOD DATA
8179 036422 013737 001736 001126  MOV @#DS1,@#BDDAT  ;BAD DATA
8180 036430 104001      ERROR 1           ;'LST' DID NOT SET AFTER
8181                                     ;LAST SECTOR ON LAST TRACK
8182                                     ;ON LAST CYLINDER WAS
8183                                     ;WRITTEN
8184                                     ;VV BIT #6 MAY OR MAY NOT BE HIGH
8185 036432 013737 001640 036442  5$:  MOV @#RHCS1,@#6$   ;SET UP 'WAT' SUBROUTINE
8186 036440 104115      WAT
8187 036442 000000      6$:  0              ;RHCS1 ADDRESS
8188 036444 000200      RDY             ;WAIT FOR READY
8189
8190 036446 000137 037116      JMP @#CAT         ;DON'T DO THE RP04 'LST' TEST FOLLOWING
8191
```

8192
8193
8194
8195 036452
8196
8197
8198
8199 036452 000004
8200 036454 012706 001000
8201 036460 012737 000114 002032
8202 036466 004737 042732
8203 036472 012777 000001 143160
8204 036500 004037 045304
8205
8206
8207 036504 000632
8208
8209
8210
8211
8212
8213
8214
8215
8216
8217
8218
8219
8220
8221
8222 036506 000004
8223
8224
8225 036510 012706 001000
8226 036514 012737 000115 002032
8227
8228 036522 004037 042650
8229 036526 051534
8230 036530 052560
8231 036532 000000
8232
8233
8234
8235 036534 012737 010632 047616
8236
8237 036542 112737 000022 047621
8238 036550 112737 000025 047620
8239 036556 005037 047622
8240 036562 005037 047624
8241 036566 012737 000400 047664
8242 036574 012737 000001 047626
8243 036602 004537 044140
8244 036606 047616
8245 036610 051516
8246
8247

DOG:

```
*****  
;*TEST 114 MAKE CURRENT CYLINDER = 410.  
*****  
TST114: SCOPE  
MOV #STACK,SP ;RESET STACK  
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER  
JSR PC,@#CLDISK ;INIT DRIVE  
MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE  
JSR RO,@#MAKECYL ;SUBROUTINE TO GIVE A SEEK  
;COMMAND FOLOWED BY AN INIT  
;THIS SHOULD CHANGE RHCC  
;CHANGE RHCC TO 410.  
410.
```

```
*****  
;*TEST 115 RHDS1 (BIT #10) - LAST SECTOR TRANSFERRED, 'LST'  
*****  
;** WRITE CYLINDER 410., FORMAT 16 BITS PER WORD  
;** TRACK 18., SECTOR 21., KEYS 0, NUMBER OF WORDS  
;** 256., OF 377  
;** 'LST' BIT #10 - RHDS1 SHOULD SET AFTER THE  
;** WRITE IS COMPLETED
```

```
*****  
TST115: SCOPE  
MOV #STACK,SP ;RESET STACK  
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER  
JSR RO,@#CLAREA ;CLEAR SIMULATED DISK  
.WORD DISK ;FROM  
.WORD TOLGAP+16 ;TO  
.WORD 0 ;DATA  
;*THESE ARE SETUP FOR DISKLESS USE ONLY  
MOV #410.!FMT22,@#CYL;CYLINDER 410.  
;16 BITS PER WORD  
MOVB #18.,@#SECOTR+1;TRACK 18.  
MOVB #21.,@#SECOTR ;SECTOR 21.  
CLR @#KEY1 ;KEY1 0  
CLR @#KEY2 ;KEY2 0  
MOV #256.,@#NOWORD ;NO OF DATA WORDS  
MOV #1,@#X ;WRITE DATA  
JSR R5,@#CRC ;GO TO CALCULATE CRC  
CYL  
WCRC  
;*THESE ARE REGULAR SETUPS
```

```
8248
8249 036612 004037 042650 JSR RO,@#CLAREA ;FILL WRITE BUFFER WITH 377
8250 036616 002114 WRFROM ;FROM LOCATION
8251 036620 003114 WRFROM+<256.*2> ;TO LOCATION
8252 036622 000377 377 ;DATA
8253 036624 004737 042732 JSR PC,@#CLDISK ;SETUP GENERAL REGISTERS
8254 036630 012777 177400 142774 MOV #-256.,@RHWC ;256. DATA WORDS
8255 036636 012777 002114 142770 MOV #WRFROM,@RHBA ;STARTING ADDRESS OF WRITE BUFFER
8256 036644 012746 000025 MOV #21.,-(SP) ;SECTOR 21.
8257 036650 112766 000022 000001 MOV #18.,1(SP) ;TRACK 18.
8258 036656 012677 142762 MOV (SP)+,@RHDS1 ;SECTOR 21. TRACK 18.
8259 036662 012777 010000 142760 MOV #FMT22,@RHOF ;16 BITS PER WORD FORMAT
8260 036670 012777 000632 142754 MOV #410.,@RHCA ;CYLINDER 410.
8261 036676 004737 042766 JSR PC,@#CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
8262 036702 104401 062562 TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-
8263 ;TINUE TESTING IF BOTH AREN'T TRUE
8264 036706 000000 HALT ;STOP THE TEST
8265 036710 013711 002064 MOV @#WRIDAT,@R1 ;WRITE DATA=60
8266 036714 005037 002006 CLR @#ERFLG$ ;CLEAR ERROR FLAG
8267 036720 004737 047456 JSR PC,@#COMHD ;WRITE DATA
8268
8269
8270
8271 ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
8272 ;*FROM THE "COMHD" ROUTINE IT MEANS SECTOR GAP, SYNC BYTE
8273 ;*HEADER, HEADER CRC, HEADER GAP AND SYNC BYTE HAVE GONE BY
8274 ;*AND SYNC'S WERE CORRECTLY DETECTED, DATA IS TO BE CHECKED.
8275 036724 004737 042336 JSR PC,@#PUTREG ;SAVE REGISTERS
8276 036730 005737 002006 TST @#ERFLG$ ;HAVE ANY ERRORS OCCURED?
8277 036734 001062 BNE 5$ ;BRANCH IF YES
8278 036736 012700 000377 MOV #377,R0 ;GOOD DATA
8279 036742 012701 051534 MOV #DISK,R1 ;DATA WRITTEN INTO "DISK"
8280 036746 012702 000400 MOV #256.,R2 ;COUNTER
8281
8282 036752 012737 000401 047736 1$: MOV #256.+1,@#ERWORD ;FOR ERROR WORD
8283 036760 020021 CMP R0,(R1)+ ;COMPARE GOOD DATA WITH DATA ON DISK
8284 036762 001424 BEQ 3$ ;BRANCH IF GOOD
8285 036764 010037 001124 MOV R0,@#$GDDAT ;GOOD DATA
8286 036770 014137 001126 MOV -(R1),@#$BDDAT ;BAD DATA
8287 036774 160237 047736 SUB R2,@#ERWORD ;ERROR WORD NO
8288 037000 005737 002006 TST @#ERFLG$ ;ANY ERRORS ALREADY THERE?
8289 037004 001002 BNE 2$ ;BRANCH IF YES
8290 037006 104004 ERROR 4 ;ERROR ON WRITE DATA COMMAND
8291 037010 000401 BR 64$ ;BRANCH TO AVOID PRINTING NEXT ERROR
8292
8293 037012 104005 2$: ERROR 5 ;WORD NO GIVES WORD IN ERROR
8294 037014 005721 64$: TST (R1)+ ;UNDO -(R1) FOR BAD DATA
8295 037016 017746 142116 MOV @SWR,-(SP) ;GET SWITCH SETTING
8296 037022 042716 177177 BIC #177177,(SP) ;KEEP ONLY SWITCH 7 AND 8
8297 037026 022726 000200 CMP #SW07,(SP)+ ;IS 7 SET AND 8 RESET
8298 037032 001402 BEQ 4$ ;BRANCH OUT IF YES
8299 037034 005302 3$: DEC R2 ;IF NOT COUNT 256 WORDS
8300 037036 001345 BNE 1$ ;BRANCH IF 256. NOT DONE
8301
8302
8303 037040 013746 001736 4$: MOV @#DS1,-(SP) ;GET RHDS1
```

CZRJGDO,RP04/5/6 DSKLS CTRLR1
CZRJGD.P11 10-SEP-79 11:00

MACY11 30A(1052) 10-SEP-79 11:11 PAGE 190
1115 RHDS1 (BIT #10) - LAST SECTOR TRANSFERRED, 'LST'

SEQ 0189

H 15

| | | | | | | | | |
|------|--------|--------|--------|--------|------|-------|---------------------------|--------------------------------------|
| 8304 | 037044 | 042716 | 001000 | | | BIC | #PROG,(SP) | ;CLEAR PROG BIT |
| 8305 | 037050 | 022726 | 002700 | | | CMP | #LST!DPR!DRY!VV,(SP)+ | ;IS 'LST' HIGH ? |
| 8306 | 037054 | 001412 | | | | BEQ | 5\$ | ;WAIT FOR 'RDY' IF GOOD |
| 8307 | 037056 | 013737 | 001662 | 042402 | | MOV | @#RHDS1,@#REGADR | ;FAILING REG. ADDRESS |
| 8308 | 037064 | 012737 | 002700 | 001124 | | MOV | #LST!DPR!DRY!VV,@#\$GDDAT | ;GOOD DATA |
| 8309 | 037072 | 013737 | 001736 | 001126 | | MOV | @#DS1,@#\$BDDAT | ;BAD DATA |
| 8310 | 037100 | 104001 | | | | ERROR | 1 | ; 'LST' DID NOT SET AFTER |
| 8311 | | | | | | | | ; LAST SECTOR ON LAST TRACK ON LAST |
| 8312 | | | | | | | | ; CYLINDER WAS WRITTEN - 'VV' BIT #6 |
| 8313 | | | | | | | | ; MAY OR MAY NOT BE HIGH |
| 8314 | | | | | | | | |
| 8315 | 037102 | 013737 | 001640 | 037112 | 5\$: | MOV | @#RHCS1,@#6\$ | ;SET UP 'WAT' SUBROUTINE |
| 8316 | 037110 | 104415 | | | | WAT | | |
| 8317 | 037112 | 000000 | | | 6\$: | O | | ;RHCS1 ADDRESS |
| 8318 | 037114 | 000200 | | | | RDY | | ;WAIT FOR 'RDY' BIT |

8319
8320
8321 037116
8322
8323
8324
8325
8326
8327
8328
8329
8330
8331
8332
8333
8334
8335
8336
8337 037116 000004
8338 037120 012706 001000
8339 037124 012737 000116 002032
8340 037132 004737 042732
8341 037136 004037 042650
8342 037142 051534
8343 037144 052560
8344 037146 000000
8345
8346
8347
8348
8349 037150 005737 002036
8350 037154 001404
8351
8352 037156 012737 011456 047616
8353 037164 000403
8354
8355 037166 012737 010632 047616 10\$:
8356
8357
8358 037174 112737 000022 047621 11\$:
8359 037202 112737 000025 047620
8360 037210 005037 047622
8361 037214 005037 047624
8362 037220 012737 000400 047664
8363 037226 012737 000001 047626
8364 037234 004537 044140
8365 037240 047616
8366 037242 051516
8367
8368
8369
8370 037244 004037 042650
8371 037250 002114
8372 037252 003114
8373 037254 000377
8374 037256 004737 042732

```
CAT:
:*****
:*TEST 116      ERROR REGISTER 1 - BIT #9 AOE

:**          A WRITE DATA COMMAND IS GIVEN TO CYLINDER 410./814.
:**          SECTOR 21 TRACK 18, KEYS 0, DATA 377
:**          WORD COUNT REGISTER FOR 522 (256+66+200) WORDS

:**          AFTER 256 WORDS HAVE BEEN WRITTEN
:**          'AOE' SHOULD COME UP

:**          RHWC WILL SHOW 200 BECAUSE THE SILO IS 66 WORDS AND
:**          256 WORDS HAVE BEEN WRITTEN - TOTAL 322
:**          THIS IS 200 SHORT OF 522
:*****

TST116: SCOPE
MOV      #STACK,SP      ;RESET STACK
MOV      #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
JSR      PC,@#CLDISK   ;INIT AND SET UP GENERAL REG. CORRES.
JSR      RO,@#CLAREA   ;CLEAR SIMULATED DISK
.WORD    DISK          ;FROM
.WORD    TOLGAP+16     ;TO
.WORD    0             ;DATA

;*THESE ARE TO SETUP FOR DISKLESS USE ONLY
;*AND WILL HANDLE RP04 OR RP06 DRIVES

TST      @#RP06 ;MOVE RP06 FLAG TO ITSELF TO TEST
BEQ      10$     ;TREAT DRIVE AS RP04 IF = 0

MOV      #814.!FMT22,@#CYL;CYLINDER 814., 16 BITS PER WORD
BR       11$     ;TREAT DRIVE AS RP06

MOV      #410.!FMT22,@#CYL;CYLINDER 410., 16 BITS PER WORD
;TREAT DRIVE AS RP04

MOVB     #18.,@#SECOTR+1 ;TRACK 18.
MOVB     #21.,@#SECOTR   ;SECTOR 21.
CLR      @#KEY1          ;KEY1 0
CLR      @#KEY2          ;KEY2 0
MOV      #256.,@#NOWORD ;NO OF DATA WORDS
MOV      #1,@#X          ;WRITE DATA
JSR      R5,@#CRC        ;GO TO CALCULATE CRC
CYL
WCRC

;*THESE ARE REGULAR SETUPS

JSR      RO,@#CLAREA     ;FILL WRITE BUFFER WITH 377
WRFROM   ;FROM
WRFROM+<256.*2>        ;TO
377      ;DATA
JSR      PC,@#CLDISK   ;SETUP GENERAL REGISTERS
```



```

8375 037262 012777 176766 142342 MOV #-522.,@RHWC ;522. DATA WORDS
8376 037270 012777 002114 142336 MOV #WRFROM,@RHBA ;STARTING ADDRESS OF WRITE BUFFER
8377 037276 012746 000025 MOV #21.,-(SP) ;SECTOR 21.
8378 037302 112766 000022 000001 MOV #18.,1(SP) ;TRACK 18.
8379 037310 012677 142330 MOV (SP)+,@RHDST ;SECTOR 21. TRACK 18.
8380 037314 012777 010000 142326 MOV #FMT22,@RHOF ;16 BITS PER WORD FORMAT
8381
8382 ;*CHECK TO SEE WHAT TYPE OF DRIVE IS BEING TESTED
8383 ;*AND LOAD CYLINDER ADDRESS REGISTER WITH THE PROPER NUMBER
8384
8385 037322 005737 002036 TST @#RP06 ;MOVE FLAG TO ITSELF TO TEST
8386 037326 001404 BEQ 12$ ;TREAT AS RP04 IF = 0
8387 037330 012777 001456 142314 MOV #814.,@RHCA ;CYLINDER 814.
8388 037336 000403 BR 13$ ;TREAT AS RP06
8389 037340 012777 000632 142304 12$: MOV #410.,@RHCA ;CYLINDER 410.
8390 037346 13$:
8391
8392 037346 004737 042766 JSR PC,@#CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
8393 037352 104401 062562 TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-
8394 ;TINUE TESTING IF BOTH AREN'T TRUE
8395 037356 000000 HALT ;STOP THE TEST
8396 037360 013711 002064 MOV @#WRIDAT,@R1 ;WRITE DATA=60
8397 037364 005037 002006 CLR @#ERFLG$ ;CLEAR ERROR FLAG
8398
8399 ;*THE REGISTERS WILL BE SAVED IN REINTO BUFFER
8400
8401 037370 004037 043424 JSR RO,@#SAVER ;SAVE
8402 037374 001632 RHWC ;FROM
8403 037376 003160 REINTO ;TO
8404 037400 000023 19. ;NUMBER SAVED
8405
8406 ;*GIVE WRITE DATA COMMAND
8407
8408 037402 004737 047456 JSR PC,@#COMHD ;WRITE DATA COMMAND
8409
8410 ;*CHANGE SAVED REGISTERS TO EXPECTED VALUES
8411
8412 037406 005737 002040 TST @#RH70 ;CHECK FOR RH70 CONTROLLER
8413 037412 001407 BEQ JP1 ;SKIP RH70 CODE AND DC RH11 IF NOT
8414
8415 037414 012737 177376 003160 VAR1: MOV #-258.,@#REINTO ;SAVED RHWC SHOULD BE = 258.
8416 037422 012737 003134 003162 VAR2: MOV #WRFROM+<2*256.>+<2*8.>,@#REINTO+2
8417 ;SAVED RHBA SHOULD BE WRFROM+256+8
8418 037430 000406 BR JP2 ;SKIP NEXT RH11 CODE
8419
8420 037432 012737 177470 003160 JP1: MOV #-200.,@#REINTO ;SAVED RHWC SHOULD BE = 200.
8421 037440 012737 003320 003162 MOV #WRFROM+<2*256.>+<2*66.>,@#REINTO+2
8422 ;SAVED RHBA SHOULD BE WRFROM+256+66
8423
8424 037446 052737 000200 003164 JP2: BIS #OR,@#REINTO+4 ;SAVED RHCS2
8425 037454 042737 000100 003164 BIC #IR,@#REINTO+4 ;SAVED RHCS2
8426 037462 052737 140000 003166 BIS #SC!TRE,@#REINTO+6;SAVED RHCS1 SHOULD HAVE 'SC' & 'TRE'
8427 037470 012737 001000 003170 MOV #AOE,@#REINTO+10 ;SAVED RHER1 SHOULD HAVE 'AOE'
8428 037476 017737 142142 003172 MOV @RHDST,@#REINTO+12;SAVED RHDST SHOULD HAVE=
8429 ;RHDST IS UNDEFINED
8430

```

```

8431 ;*CHECK TO SEE WHAT TYPE OF DRIVE IS BEING TESTED
8432 ;*AND SET UP CYLINDER ADDRESS ACCORDINGLY
8433
8434 037504 005737 002036 TST @#RP06 ;MOVE RP06 FLAG TO ITSELF TO TEST
8435 037510 001404 BEQ 14$ ;TREAT AS RP04 IF = 0
8436 037512 012737 001457 003200 MOV #815.,@#REINTO+20;SAVED DESIRED CYLINDER ADDRESS
8437 037520 000403 BR 15$ ;TREAT AS RP06
8438 037522 012737 000633 003200 14$: MOV #411.,@#REINTO+20;SAVED DESIRED CYLINDER ADDRESS
8439
8440 037530 013737 002016 003204 15$: MOV @#ATTENT,@#REINTO+24 ;SAVED RHAS SHOULD HAVE APPRO. BIT
8441 037536 052737 000001 003206 BIS #DMD,@#REINTO+26;SAVED RHMR
8442 037544 052737 142000 003210 BIS #ATA!ERR!LST,@#REINTO+30 ;SAVED RHDS!
8443
8444 ;*AFTER A WRITE DATA COMMAND WITH 'AOE' ERROR
8445 ;*SAVE REGISTERS AGAIN SO THAT COMPARES CAN BE DONE
8446
8447 037552 004037 043424 JSR R0,@#SAVER ;SAVE
8448 037556 001632 RHC ;FROM
8449 037560 002114 WRFROM ;TO
8450 037562 000021 17. ;NUMBER OF REGISTERS SAVED
8451
8452 ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
8453 ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
8454 ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
8455
8456 037564 113737 003205 002141 MOVB @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS
8457
8458 ;*COMPARE REGISTERS BEFORE WRITE DATA COMMAND
8459 ;*WITH AFTER COMMAND
8460
8461 037572 004037 043626 JSR R0,@#COMPAR ;COMPARE
8462 037576 003160 REINTO ;GOOD BUFFER
8463 037600 002114 WRFROM ;TEST BUFFER
8464 037602 000021 17. ;NUMBER OF REGISTERS
8465 037604 037612 1$ ;RETURN FOR ERROR
8466 037606 037612 1$ ;SAME
8467 037610 037632 2$ ;RETURN FOR GOOD COMPARISON
8468
8469 037612 013705 047736 1$: MOV @#ERWORD,R5 ;GETTING READY TO INDEX
8470 037616 060505 ADD R5,R5 ;DOUBLE ERROR WORD
8471 037620 016537 001630 042402 MOV RHC-2(R5),@#REGADR ;FAILING REG. ADDRESS
8472 037626 104001 ERROR 1 ;FORCED AOE ERROR CAUSED IMPROPER
8473 ;REGISTER CHANGE
8474 037630 000207 RTS PC ;RETURN FOR FURTHER COMPARISONS
8475 ;NO ERRORS
8476 037632 005037 002006 2$: CLR @#ERFLG$ ;CLEAR ERROR FLAG
8477
8478 ;*DATA IS TO BE CHECKED HERE
8479
8480
8481 037636 004737 042336 JSR PC,@#PUTREG ;SAVE REGISTERS
8482 037642 012700 000377 MOV #377,R0 ;GOOD DATA
8483 037646 012701 051534 MOV #DISK,R1 ;DATA WRITTEN INTO "DISK"
8484 037652 012702 000400 MOV #256.,R2 ;COUNTER
8485 037656 012737 000400 047736 3$: MOV #256.,@#ERWORD ;FOR ERROR WORD
8486 037664 020021 CMP R0,(R1)+ ;COMPARE GOOD DATA WITH DATA ON DISK
  
```

| | | | | | | | |
|------|--------|--------|--------|------------|-----------------|--|--|
| 8487 | 037666 | 001424 | | BEQ | 6\$ | | ;BRANCH IF GOOD |
| 8488 | 037670 | 010037 | 001124 | MOV | R0,@#\$GDDAT | | ;GOOD DATA |
| 8489 | 037674 | 014137 | 001126 | MOV | -(R1),@#\$BDDAT | | ;BAD DATA |
| 8490 | 037700 | 160237 | 047736 | SUB | R2,@#ERWORD | | ;ERROR WORD NO |
| 8491 | 037704 | 005737 | 002006 | TST | @#ERFLG\$ | | ;ANY ERRORS ALREADY THERE? |
| 8492 | 037710 | 001002 | | BNE | 4\$ | | ;BRANCH IF YES |
| 8493 | 037712 | 104004 | | ERROR | 4 | | ;ERROR ON WRITE DATA COMMAND WITH FORCED 'AOE' |
| 8494 | 037714 | 000401 | | BR | 5\$ | | ;BRANCH TO AVOID PRINTING NEXT ERROR |
| 8495 | 037716 | 104005 | | 4\$: ERROR | 5 | | ;WORD NO. GIVES WORD IN ERROR |
| 8496 | 037720 | 005721 | | 5\$: TST | (R1)+ | | ;UNDO -(R1) FOR BAD DATA |
| 8497 | 037722 | 017746 | 141212 | MOV | @SWR,-(SP) | | ;GET SWITCH SETTING |
| 8498 | 037726 | 042716 | 177177 | BIC | #177177,(SP) | | ;KEEP ONLY SWITCH 7 AND 8 |
| 8499 | 037732 | 022726 | 000200 | CMP | #SW07,(SP)+ | | ;IS 7 SET AND 8 RESET |
| 8500 | 037736 | 001402 | | BEQ | 7\$ | | ;BRANCH OUT IF YES -----> |
| 8501 | | | | | | | |
| 8502 | 037740 | 005302 | | 6\$: DEC | R2 | | ;IF NOT COUNT 256 WORDS |
| 8503 | 037742 | 001345 | | BNE | 3\$ | | ;BRANCH IF 256. NOT DONE |
| 8504 | | | | | | | |
| 8505 | 037744 | | | 7\$: | | | |

```
8506
8507
8508
8509
8510 037744 000004
8511 037746 012706 001000
8512 037752 012737 000117 002032
8513 037760 004737 042732
8514 037764 012777 000001 141666
8515 037772 004037 045304
8516
8517
8518 037776 000000
8519
8520
8521
8522
8523
8524
8525
8526
8527
8528
8529
8530
8531
8532 040000 000004
8533 040002 012706 001000
8534 040006 012737 000120 002032
8535 040014 004737 042732
8536
8537
8538 040020 012777 000001 141632
8539 040026 052777 000004 141624
8540 040034 042777 000004 141616
8541
8542
8543
8544
8545 040042 012777 177400 141562
8546 040050 012700 003160
8547 040054 010077 141554
8548
8549 040060 012720 010000
8550
8551 040064 012720 012000
8552 040070 005020
8553 040072 005020
8554 040074 012705 000400
8555 040100 012720 177777 1$:
8556 040104 005305
8557 040106 001374
8558 040110 012777 012000 141526
8559
8560 040116 004737 042766
8561 040122 104401 062562
```

```
*****
;*TEST 117 MAKE CURRENT CYLINDER = 0
*****
TST117: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
JSR PC,@#CLDISK ;INIT DRIVE
MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE
JSR RO,@#MAKECYL ;SUBROUTINE TO GIVE A SEEK
;COMMAND FOLOWED BY AN INIT
;THIS SHOULD CHANGE RHCC
;CHANGE RHCC TO 0

*****
;*TEST 120 ERROR REGISTER 1 - BIT #10 'IAE'
; ** A READ HEADER AND DATA IS GIVEN TO TRACK 20, SECTOR 0
; ** AN INDEX PULSE IS GIVEN TO GET RHLA TO 0
; ** IAE BIT SHOULD SET

*****
TST120: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
JSR PC,@#CLDISK ;CLEAR REGISTERS AND SET UNIT NO.

;*GIVE INDEX PULSE
MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE
BIS #MINX,@RHMR ;SET INDEX
BIC #MINX,@RHMR ;CLEAR INDEX

;*THESE ARE REGULAR SETUPS
MOV #-256,@RHWC ;256 DATA WORDS 4 HEADER WORDS
MOV #REINTO,RO ;THESE TWO INSTRUCTIONS GETS
MOV RO,@RHBA ;ADDR, OF WRFROM INTO RO AND
;BUS ADDRESS REGISTER
MOV #FMT22,(RO)+ ;FORMAT=16 BIT WORDS
;CYLINDER=0
MOV #12000,(RO)+ ;TRACK=20 SECTOR=0 KEYS=0
CLR (RO)+ ;KEY1=0
CLR (RO)+ ;KEY2=0
MOV #256,R5 ;COUNTER
1$: MOV #-1,(RO)+ ;MOVE ALL ONES FOR DATA
DEC R5
BNE 1$ ;BRANCH IF DATA NOT COMPLETE
MOV #12000,@RHDST ;TRACK=20 SECTOR=0
JSR PC,@#CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-
```

```
8562  
8563 040126 000000 HALT ;TINUE TESTING IF BOTH AREN'T TRUE  
8564 ;STOP THE TEST  
8565 040130 013711 002072 MOV @#REFOR,@R1 ;GET READY FOR WRITE HEADER AND  
8566 ;DATA WITH 62 IN RHCS1  
8567 040134 005037 002006 CLR @#ERFLG$ ;CLEAR ERROR FLAG  
8568 040140 012777 010000 141502 MOV #FMT22,@RHOF ;FORMAT BIT=1 (16 BIT WORDS)  
8569 040146 005077 141500 CLR @RHCA ;CYLINDER =0  
8570  
8571 ;*THE REGISTERS WILL BE SAVED IN REINTO BUFFER  
8572 040152 004037 043424 JSR RO,@#SAVER ;SAVE  
8573 040156 001632 RHWC ;FROM  
8574 040160 003160 REINTO ;TO  
8575 040162 000023 19. ;NUMBER SAVED  
8576  
8577 ;*GO TO WRITE HEADER AND DATA  
8578  
8579 040164 013700 001660 MOV @#RHMR,RO ;NOW RO WAS MAINTENANCE REG. ADDR.  
8580 040170 012710 000001 MOV #DMD,@RO ;SET DIAGNOSTIC MODE  
8581 040174 052777 000001 141436 BIS #GO,@RHCS1 ;GO  
8582  
8583 ;*CHANGE SAVED REGISTERS TO EXPECTED VALUE  
8584 040202 052737 140000 003166 BIS #SC!TRE,@#REINTO+6 ;SAVED RHCS1  
8585 040210 012737 002000 003170 MOV #IAE,@#REINTO+10 ;SAVED RHER1  
8586 040216 012737 012001 003172 MOV #12001,@#REINTO+12 ;SAVED RHDST  
8587 040224 013737 002016 003204 MOV @#ATTENT,@#REINTO+24 ;SAVED RHAS  
8588 040232 052737 000001 003206 BIS #DMD,@#REINTO+26 ;SAVED RHMR  
8589 040240 052737 140000 003210 BIS #ATA!ERR,@#REINTO+30 ;SAVED RHDS1  
8590  
8591 ;*SAVE REGISTERS AGAIN SO THAT COMPARES CAN BE DONE  
8592 040246 004037 043424 JSR RO,@#SAVER ;SAVE  
8593 040252 001632 RHWC ;FROM  
8594 040254 002114 WRFROM ;TO  
8595 040256 000023 19. ;NUMBER OF REGISTERS SAVED  
8596  
8597 ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT  
8598 ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS  
8599 ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE  
8600 040260 113737 003205 002141 MOVB @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS  
8601  
8602  
8603 ;*COMPARE REGISTERS BEFORE READ IN PRESET COMMAND  
8604 ;*WITH AFTER COMMAND  
8605 040266 004037 043626 JSR RO,@#COMPAR ;COMPARE  
8606 040272 003160 REINTO ;GOOD BUFFER  
8607 040274 002114 WRFROM ;TEST BUFFER  
8608 040276 000021 17. ;NUMBER OF REGISTERS  
8609 040300 040306 2$ ;RETURN FOR ERROR  
8610 040302 040306 2$ ;SAME  
8611 040304 040326 3$ ;RETURN FOR GOOD COMPARISON  
8612  
8613 040306 013705 047736 2$: MOV @#ERWORD,R5 ;GETTING READY TO INDEX  
8614 040312 060505 ADD R5,R5 ;DOUBLE ERROR WORD  
8615 040314 016537 001630 042402 MOV RHWC-2(R5),@#REGADR ;FAILING REG. ADDRESS  
8616 040322 104001 ERROR 1 ;FORCED IAE CAUSED IMPROPER  
8617 ;REGISTER CHANGE
```

B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Q
R
S
T
U
V
W
X
Y
Z

CZRJGDO,RP04/5/6 DSKLS CTRLR1
CZRJGD.P11 10-SEP-79 11:00

MACY11 30A(1052) 10-SEP-79 11:11 B 16
T120 ERROR REGISTER 1 - BIT #10 'IAE' PAGE 197

SEQ 0196

| | | | | | | |
|------|--------|--------|--------|------|-----|---------------------------------|
| 8618 | 040324 | 000207 | | RTS | PC | ;RETURN FOR FURTHER COMPARISONS |
| 8619 | | | | | | |
| 8620 | | | | | | ;NO ERRORS |
| 8621 | | | | | | |
| 8622 | 040326 | 004737 | 042732 | 3\$: | JSR | PC,@#CLDISY ;CLEAR GO BIT |
| 8623 | | | | | | |

8624
8625
8626
8627
8628
8629
8630
8631
8632
8633
8634
8635
8636
8637
8638
8639
8640
8641
8642
8643
8644
8645
8646
8647
8648
8649
8650
8651
8652
8653
8654
8655
8656
8657
8658
8659
8660
8661
8662
8663
8664
8665
8666
8667
8668
8669
8670
8671
8672
8673
8674
8675
8676
8677
8678
8679

```
*****  
;*TEST 121      ERROR REGISTER 1- BIT #10 'IAE'  
  
**      A WRITE HEADER AND DATA IS GIVEN TO SECTOR 22  
**      TRACK 0 CYLINDER 0  
  
**      WORD COUNT IS SET TO 256.  
  
**      AN INDEX PULSE IS GIVEN TO GET RHLA TO 0  
  
**      IAE BIT SHOULD SET
```

```
TST121: SCOPE  
MOV      #STACK,SP      ;RESET STACK  
MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER  
JSR      PC,@#CLDISK    ;CLEAR REGISTERS AND SET UNIT NO.
```

```
;*GIVE INDEX PULSE  
MOV      #DMD,@RHMR     ;SET DIAGNOSTIC MODE  
BIS      #MINX,@RHMR    ;SET INDEX  
BIC      #MINX,@RHMR    ;CLEAR INDEX
```

```
;*THESE ARE REGULAR SETUPS
```

```
MOV      #-256.,@RHWC   ;256 DATA WORDS 4 HEADER WORDS  
MOV      #WRFROM,RO     ;THESE TWO INSTRUCTIONS GETS  
MOV      RO,@RHBA       ;ADDR. OF WRFROM INTO RO AND  
;BUS ADDRESS REGISTER  
MOV      #FMT22,(RO)+   ;FORMAT=16 BIT WORDS  
;CYLINDER=0  
MOV      #22.,(RO)+    ;TRACK=0, SECTOR=22, KEYS=0  
CLR      (RO)+          ;KEY1=0  
CLR      (RO)+          ;KEY2=0  
MOV      #256.,R5       ;COUNTER  
1$: MOV      #-1,(RO)+   ;MOVE ALL ONES FOR DATA  
DEC      R5  
BNE      1$             ;BRANCH IF DATA NOT COMPLETE  
MOV      #22.,@RHDST    ;TRACK=0 SECTOR=22  
  
JSR      PC,@#CHECKT    ;CHECK THAT DVA,RDY,DPR,DRY = 1  
TYPE     ,CPHALT        ;AND THAT NO OTHERS = 1. CANNOT CON-  
;TINUE TESTING IF BOTH AREN'T TRUE  
HALT     ;STOP THE TEST
```

```
MOV      @#WRIFOR,@R1   ;GET READY FOR WRITE HEADER AND  
;DATA WITH 62 IN RHCS1  
CLR      @#ERFLG$       ;CLEAR ERROR FLAG  
MOV      #FMT22,@RHOF   ;FORMA BIT=1 (16 BIT WORDS)  
CLR      @RHCA          ;CYLINDER =0
```

```
;*AS EXCEPTION IS ASSERTED BEFORE RUN IS  
;*LATCHED RHWC,RHBA,RHCS1,RHCS2 CANNOT BE CHECKED  
;*BECAUSE RHWC WILL VARY DEPENDING UPON GATE DELAYS  
;*ON DIFFERENT UNITS
```

002032

141300

141272

141264

141230

002114

141222

010000

000026

000400

177777

141174

141150

141146

1\$:

```
8680
8681
8682 040504 004037 043424
8683 040510 001642
8684 040512 003160
8685 040514 000015
8686
8687
8688
8689 040516 013700 001660
8690 040522 012710 000001
8691 040526 052777 000001 141104
8692
8693
8694 040534 012737 002000 003160
8695 040542 012737 000027 003162
8696 040550 013737 002016 003174
8697 040556 052737 000001 003176
8698 040564 052737 140000 003200
8699
8700
8701 040572 004037 043424
8702 040576 001642
8703 040600 002114
8704 040602 000015
8705
8706
8707
8708
8709 040604 113737 003175 002131
8710
8711
8712
8713
8714 040612 004037 043626
8715 040616 003160
8716 040620 002114
8717 040622 000015
8718 040624 040632
8719 040626 040632
8720 040630 040652
8721
8722 040632 013705 047736 2$:
8723 040636 060505
8724 040640 016537 001640 042402
8725 040646 104001
8726
8727 040650 000207
8728
8729
8730
8731 040652 004737 042732 3$:
8732
8733
```

;
*THE REGISTERS WILL BE SAVED IN REINTO BUFFER
JSR RO,@#SAVER ;SAVE
RHER1 ;FROM
REINTO ;TO
13. ;NUMBER SAVED
;
*GO TO WRITE HEADER AND DATA
MOV @#RHMR,RO ;NOW RO HAS MAINTENANCE REG. ADDR.
MOV #DMD,@RO ;SET DIAGNOSTIC MODE
BIS #GO,@RHCS1 ;GO
;
*CHANGE SAVED REGISTERS TO EXPECTED VALUE
MOV #IAE,@#REINTO ;SAVED RHER1
MOV #23.,@#REINTO+2;SAVED RHDST
MOV @#ATTENT,@#REINTO+14 ;SAVED RHAS
BIS #DMD,@#REINTO+16 ;SAVED RHMR
BIS #ATA!ERR,@#REINTO+20 ;SAVED RHDS1
;
*SAVE REGISTERS AGAIN SO THAT COMPARES CAN BE DONE
JSR RO,@#SAVER ;SAVE
RHER1 ;FROM
WRFROM ;TO
13. ;NUMBER OF REGISTERS SAVED
;
*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
MOVB @#REINTO+15,@#WRFROM+15;SAVE UPPER RHAS
;
*COMPARE REGISTERS BEFORE READ IN PRESET COMMAND
*WITH AFTER COMMAND
JSR RO,@#COMPAR ;COMPARE
REINTO ;GOOD BUFFER
WRFROM ;TEST BUFFER
13. ;NUMBER OF REGISTERS
2\$;RETURN FOR ERROR
2\$;SAME
3\$;RETURN FOR GOOD COMPARISON
;
*GETTING READY TO INDEX
MOV @#ERWORD,R5 ;DOUBLE ERROR WORD
ADD R5,R5 ;FAILING REG. ADDRESS
MOV RHER1-2(R5),@#REGADR ;FORCED IAE CAUSED IMPROPER
ERROR 1 ;REGISTER CHANGE
RTS PC ;RETURN FOR FURTHER COMPARISONS
;
*NO ERRORS
;
*CLEAR GO BIT


```
8734  
8735  
8736  
8737  
8738  
8739  
8740  
8741  
8742  
8743  
8744  
8745  
8746  
8747  
8748 040656 000004  
8749 040660 012706 001000  
8750 040664 012737 000122 002032  
8751 040672 004737 042732  
8752  
8753  
8754 040676 012777 000001 140754  
8755 040704 052777 000004 140746  
8756 040712 042777 000004 140740  
8757  
8758  
8759  
8760 040720 012777 177400 140704  
8761 040726 012700 002114  
8762 040732 010077 140676  
8763  
8764 040736 012705 000400  
8765 040742 012720 177777 1$:  
8766 040746 005305  
8767 040750 001374  
8768 040752 012777 000000 140664  
8769  
8770 040760 004737 042766  
8771 040764 104401 062562  
8772  
8773 040770 000000  
8774  
8775 040772 013711 002064  
8776  
8777 040776 005037 002006  
8778 041002 012777 010000 140640  
8779  
8780 041010 005737 002036  
8781 041014 001404  
8782  
8783 041016 012777 001457 140626  
8784 041024 000403  
8785  
8786 041026 012777 000633 140616 4$:  
8787  
8788  
8789
```

```
*****  
; *TEST 122 ERROR REGISTER 1- BIT #10 'IAE'  
  
; ** A WRITE DATA IS GIVEN TO SECTOR 0  
; ** TRACK 0 CYLINDER 411  
  
; ** WORD COUNT IS SET TO 256.  
; ** AN INDEX PULSE IS GIVEN TO GET RHLA TO 0  
  
; ** IAE BIT SHOULD SET  
  
*****  
1ST122: SCOPE  
MOV #STACK,SP ;RESET STACK  
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER  
JSR PC,@#CLDISK ;CLEAR REGISTERS AND SET UNIT NO.  
  
; *GIVE INDEX PULSE  
MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE  
BIS #MINX,@RHMR ;SET INDEX  
BIC #MINX,@RHMR ;CLEAR INDEX  
  
; *THESE ARE REGULAR SETUPS  
  
MOV #-256.,@RHWC ;256 DATA WORDS 4 HEADER WORDS  
MOV #WRFROM,R0 ;THESE TWO INSTRUCTIONS GETS  
MOV R0,@RHBA ;ADDR. OF WRFROM INTO R0 AND  
;BUS ADDRESS REGISTER  
MOV #256.,R5 ;COUNTER  
1$: MOV #-1,(R0)+ ;MOVE ALL ONES FOR DATA  
DEC R5  
BNE 1$ ;BRANCH IF DATA NOT COMPLETE  
MOV #0.,@RHDST ;TRACK=0 SECTOR=0  
  
JSR PC,@#CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1  
TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-  
;TINUE TESTING IF BOTH AREN'T TRUE  
;STOP THE TEST  
  
MOV @#WRIDAT,@R1 ;GET READY FOR WRITE  
;DATA WITH 60 IN RHCS1  
CLR @#ERFLG$ ;CLEAR ERROR FLAG  
MOV #FMT22,@RHOF ;FORMA BIT=1 (16 BIT WORDS)  
  
TST @#RP06 ;MOVE FLAG TO ITSELF TO TEST  
BEQ 4$ ;TREAT DRIVE AS RP04 IF FLAG = 0  
  
MOV #815.,@RHCA ;CYLINDER = 815 (ONE TOO MANY)  
BR 5$ ;TREAT DRIVE AS RP06  
  
MOV #411.,@RHCA ;CYLINDER = 411 (ONE TOO MANY)  
;TREAT DRIVE AS RP04  
  
; *AS EXCEPTION IS ASSERTED BEFORE RUN IS
```

```
8790 ;*LATCHED RHWC,RHBA,RHCS1,RHCS2 CANNOT BE CHECKED
8791 ;*BECAUSE RHWC WILL VARY DEPENDING UPON GATE DELAYS
8792 ;*ON DIFFERENT UNITS
8793
8794 ;*THE REGISTERS WILL BE SAVED IN REINTO BUFFER
8795 041034 004037 043424 5$: JSR RO,@#SAVER ;SAVE
8796 041040 001642 RHER1 ;FROM
8797 041042 003160 REINTO ;TO
8798 041044 000015 13. ;NUMBER SAVED
8799
8800 ;*GO TO WRITE HEADER AND DATA
8801
8802 041046 013700 001660 MOV @#RHMR,RO ;NOW RO HAS MAINTENANCE REG. ADDR.
8803 041052 012710 000001 MOV #DMD,@RO ;SET DIAGNOSTIC MODE
8804 041056 052777 000001 140554 BIS #GO,@RHCS1 ;GO
8805
8806 ;*CHANGE SAVED REGISTERS TO EXPECTED VALUE
8807 041064 012737 002000 003160 MOV #IAE,@#REINTO ;SAVED RHER1
8808 041072 012737 000001 003162 MOV #1,@#REINTO+2;SAVED RHDST
8809 041100 013737 002016 003174 MOV @#ATTENT,@#REINTO+14 ;SAVED RHAS
8810 041106 052737 000001 003176 BIS #DMD,@#REINTO+16 ;SAVED RHMR
8811 041114 052737 140000 003200 BIS #ATA!ERR,@#REINTO+20 ;SAVED RHDS1
8812
8813 ;*SAVE REGISTERS AGAIN SO THAT COMPARES CAN BE DONE
8814 041122 004037 043424 JSR RO,@#SAVER ;SAVE
8815 041126 001642 RHER1 ;FROM
8816 041130 002114 WRFROM ;TO
8817 041132 000015 13. ;NUMBER OF REGISTERS SAVED
8818
8819 ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
8820 ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
8821 ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
8822 041134 113737 003175 002131 MOVB @#REINTO+15,@#WRFROM+15;SAVE UPPER RHAS
8823
8824
8825 ;*COMPARE REGISTERS BEFORE READ IN PRESET COMMAND
8826 ;*WITH AFTER COMMAND
8827 041142 004037 043626 JSR RO,@#COMPAR ;COMPARE
8828 041146 003160 REINTO ;GOOD BUFFER
8829 041150 002114 WRFROM ;TEST BUFFER
8830 041152 000015 13. ;NUMBER OF REGISTERS
8831 041154 041162 2$ ;RETURN FOR ERROR
8832 041156 041162 2$ ;SAME
8833 041160 041202 3$ ;RETURN FOR GOOD COMPARISON
8834
8835 041162 013705 047736 2$: MOV @#ERWORD,R5 ;GETTING READY TO INDEX
8836 041166 060505 ADD R5,R5 ;DOUBLE ERROR WORD
8837 041170 016537 001640 042402 MOV RHER1-2(R5),@#REGADR ;FAILING REG. ADDRESS
8838 041176 104001 ERROR 1 ;FORCED IAE CAUSED IMPROPER
8839 ;REGISTER CHANGE
8840 041200 000207 RTS PC ;RETURN FOR FURTHER COMPARISONS
8841
8842 ;NO ERRORS
8843
8844 041202 004737 042732 3$: JSR PC,@#CLDISK ;CLEAR GO BIT
```

```
8845
8846
8847
8848
8849
8850
8851
8852
8853
8854 041206 000004
8855 041210 012737 000001 001212
8856 041216 012737 000000 177776
8857 041224 104401 041232
8858 041230 000425
8859
8860 041304
8861 041304 013746 001774
8862 041310 104405
8863 041312 104401 041320
8864 041316 000402
8865
8866 041324
8867 041324 013746 001112
8868 041330 104405
8869 041332 005037 001112
8870 041336 005037 001102
8871 041342 005737 002002
8872
8873 041346 001413
8874
8875 041350 005237 001100
8876 041354 104401 041537
8877 041360 013746 001100
8878 041364 104405
8879 041366 104401 041534
8880 041372 000137 007656
8881
8882 041376 005337 001776
8883 041402 001413
8884 041404 013700 001774
8885 041410 012701 001754
8886 041414 022100
8887 041416 001401
8888 041420 000775
8889 041422 011137 001774
8890 041426 000137 007656
8891
8892
8893
8894
8895
8896
8897
8898
8899
8900

*****
*TEST 123 END OF DRIVE
** THIS IS THE END OF TEST FOR ONE DRIVE
** IF THERE ARE MORE DRIVES THEN THE PROGRAM
** JUMPS TO TEST 5 FOR NEXT DRIVE TEST
** END PASS IS REACHED ONLY AFTER ALL DRIVES ARE COMPLETE
*****
TST123: SCOPE
MOV #1,$TIMES ;;DO 1 ITERATION
MOV #0,PS ;REINSTATE PS TO 0
TYPE ,65$ ;;TYPE ASCIZ STRING
BR 64$ ;;GET OVER THE ASCIZ
;;65$: .ASCIZ <15><12>/TOTAL ERRORS ON THIS PASS ON UNIT NO. /
64$: MOV @#UNIT,-(SP) ;GET READY TO TYPE UNIT NUMBER
TYPDS
TYPE ,67$ ;;TYPE ASCIZ STRING
BR 66$ ;;GET OVER THE ASCIZ
;;67$: .ASCIZ /= /
66$: MOV @#SERTTL,-(SP) ;GET READY TO TYPE NUMBER OF ERRORS
TYPDS
CLR @#SERTTL ;CLEAR TOTAL NUMBER OF ERRORS
CLR @#STSTNM ;CLEAR TEST NUMBER
TST @#SELECT ;STARTING FROM 210 ?
BEQ 3$ ;TEST NEXT DRIVE IF NOT
;CONTINUE ON THIS ONE IF SO
INC @#$PASS ;INCREASE PASS COUNT
TYPE ,SENDMG ;TYPE END PASS #
MOV @#$PASS,-(SP)
TYPDS
TYPE ,$ENULL
JMP @#TST5 ;DO NEXT TESTS ----->
3$: DEC @#NCUNITS ;NO. OF UNITS PRESENT DECREMENTED
BEQ $EOP ;BRANCH IF ALL DRIVES COMPLETE
MOV @#UNIT,R0 ;UNIT UNDER TEST
MOV #UNITS,R1 ;TABLE
1$: CMP (R1)+,R0 ;IS THIS UNIT JUST TESTED
BEQ 2$ ;BRANCH IF YES
BR 1$ ;BRANCH IF NO
2$: MOV (R1),@#UNIT ;THIS IS NEXT UNIT
JMP @#TST5 ;GO FOR NEXT TESTS ----->

.SBTTL
.SBTTL ***SUBROUTINES***
.SBTTL

.SBTTL END OF PASS ROUTINE
*****
```

```
8901      ;*INCREMENT THE PASS NUMBER ($PASS)
8902      ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
8903      ;*IF THERES A MONITOR GO TO IT
8904      ;*IF THERE ISN'T JUMP TO TST1
8905
8906 041432 $EOP:
8907 041432 000004          SCOPE
8908 041434 005037 001102  CLR      $TSTNM          ;;ZERO THE TEST NUMBER
8909 041440 005037 001212  CLR      $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
8910 041444 005237 001100  INC      $PASS          ;;INCREMENT THE PASS NUMBER
8911 041450 042737 100000 001100  BIC      #100000,$PASS  ;;DON'T ALLOW A NEG. NUMBER
8912 041456 005327          DEC      (PC)+          ;;LOOP?
8913 041460 000001          $EOPCT: .WORD 1
8914 041462 003022          BGT      $DOAGN          ;;YES
8915 041464 012737          MOV      (PC)+,@(PC)+  ;;RESTORE COUNTER
8916 041466 000001          $ENDCT: .WORD 1
8917 041470 041460          $EOPCT
8918 041472 104401 041537  TYPE      , $SENDMG      ;;TYPE "END PASS #"
8919 041476 013746 001100  MOV      $PASS,-(SP)    ;;SAVE $PASS FOR TYPEOUT
8920 041502 104405          TYPDS          ;;GO TYPE--DECIMAL ASCII WITH SIGN
8921 041504 104401 041534  TYPE      , $ENULL       ;;TYPE A NULL CHARACTER
8922 041510 013700 000042  $GET42: MOV      @#42,R0  ;;GET MONITOR ADDRESS
8923 041514 001405          BEQ      $DOAGN         ;;BRANCH IF NO MONITOR
8924 041516 000005          RESET          ;;CLEAR THE WORLD
8925 041520 004710          $ENDAD: JSR      PC,(R0)  ;;GO TO MONITOR
8926 041522 000240          NGP          ;;SAVE ROOM
8927 041524 000240          NOP          ;;FOR
8928 041526 000240          NOP          ;;ACT11
8929 041530          $DOAGN:
8930 041530 000137          JMP      @(PC)+          ;;RETURN
8931 041532 005504          $RTNAD: .WORD  TST1
8932 041534 377 377 000  $ENULL: .BYTE  -1,-1,0  ;;NULL CHARACTER? STRING
8933 041537 015 042412 042116  $ENDMG: .ASCIZ  <15><12>/END PASS #/
8934 041544 050040 051501 020123
8935 041552 000043
8936
8937
8938      ;*****
8939      ;**HERE IS A DETAILED EXPLANATION OF HOW THE LOOP ON ERROR WORKS.
8940      ;**ON HITTING AN ERROR IF THE LOOP ON ERROR SWITCH IS SET, THE
8941      ;**PROGRAM GOES BACK - USUALLY BACK TO THE BEGINNING OF THE TEST.
8942
8943      ;**WHEN THIS OPERATOR SELECTABLE SCOPE LOOP IS USED THEN THE POINT
8944      ;**THE PROGRAM GOES BACK TO CAN BE CHANGED.
8945      ;**THE RESTRICTIONS TO THE POINT WHERE THE PROGRAM CAN GO ARE: -
8946      ;**1. IT MUST BE WITHIN THE TEST UNDER CONSIDERATION
8947      ;**2. LOOP ON ERROR SWITCH MUST BE SET
8948      ;**3. THE ERROR MUST OCCUR WITHIN THE TEST UNDER CONSIDERATION
8949      ;**IF THE ERROR DOES NOT OCCUR WITHIN THE TEST UNDER CONSIDERATION
8950      ;**THE PROGRAM WILL REVERT TO NORMAL OPERATION. HOWEVER, IF LOOP ON
8951      ;**TEST SWITCH IS SET AND THIS OPERATOR SELECTABLE SCOPE LOOP IS USED
8952      ;**THEN THE PROGRAM WILL LOOP BACK TO THE SELECTED POINT WHEN IT
8953      ;**COMES TO THE END OF THE TEST UNDER CONSIDERATION.
8954      ;**
8955      ;**AFTER LOOPING FOR SOME TIME IF THE LOOP SWITCH IS PUT DOWN THEN
8956      ;**NORMAL OPERATION WILL CONTINUE.
```

```
8957      ;:*****
8958
8959
8960 041554 000000      TESTAD: 0      ;FIRST ADDRESS OF TEST
8961
8962 041556      OPERSEL:
8963 041556 005037 177776      CLR      PS      ;MAKE PROCESSOR STATUS ZERO
8964 041562 104401 041570      TYPE      ,65$      ;;TYPE ASCIZ STRING
8965 041566 000421      BR      64$      ;;GET OVER THE ASCIZ
8966      ;;65$: .ASCIZ <15><12>/THE PROGRAM WAS IN TEST NUMBER /
8967 041632      64$:
8968 041632 013746 002032      MOV      @#TSTNM,-(SP) ;GET READY TO TYPE TEST
8969 041636 104402      TYPOC      ;NUMBER
8970 041640 104401 041646      TYPE      ,67$      ;;TYPE ASCIZ STRING
8971 041644 000414      BR      66$      ;;GET OVER THE ASCIZ
8972      ;;67$: .ASCIZ <15><12>/THE LOOP BACK PC WAS /
8973 041676      66$:
8974 041676 013746 001110      MOV      @#LPERR,-(SP) ;GET READY TO TYPE LOOP BACK PC
8975 041702 104402      TYPOC
8976 041704 104401 001223      TYPE      ,SCLRF
8977 041710 104401 041716      TYPE      ,69$      ;;TYPE ASCIZ STRING
8978 041714 000430      BR      68$      ;;GET OVER THE ASCIZ
8979      ;;69$: .ASCIZ <15><12>/SET SWITCH FOR LOOP ON ERROR OR LOOP ON TEST/
8980 041776      68$:
8981 041776 104401 042004      TYPE      ,71$      ;;TYPE ASCIZ STRING
8982 042002 000430      BR      70$      ;;GET OVER THE ASCIZ
8983      ;;71$: .ASCIZ <15><12>/TYPE THE FIRST PC OF THE TEST TO BE LOOPED ON/
8984 042064      70$:
8985 042064 104401 042072      TYPE      ,73$      ;;TYPE ASCIZ STRING
8986 042070 000422      BR      72$      ;;GET OVER THE ASCIZ
8987      ;;73$: .ASCIZ <15><12>/ FOLLOWED BY A CARRIAGE RETURN /<15><12>
8988 042136      72$:
8989 042136 104412      RDOCT
8990 042140 062716 000002      ADD      #2,(SP)      ;GET LPADR
8991 042144 012637 001106      MOV      (SP)+,@#LPADR
8992 042150 104401 042156      TYPE      ,75$      ;;TYPE ASCIZ STRING
8993 042154 000417      BR      74$      ;;GET OVER THE ASCIZ
8994      ;;75$: .ASCIZ <15><12>/TYPE THE PC WHERE YOU WANT/
8995 042214      74$:
8996 042214 104401 042222      TYPE      ,77$      ;;TYPE ASCIZ STRING
8997 042220 000440      BR      76$      ;;GET OVER THE ASCIZ
8998      ;;77$: .ASCIZ <15><12>/ THE PROGRAM TO LOOP BACK TO FOLLOWED BY A CARRIAGE RETURN /<15>
8999 042322      76$:
9000 042322 104412      RDOCT
9001 042324 012637 001110      MOV      (SP)+,@#LPERR ;GET LPERR
9002 042330 013746 001106      MOV      @#LPADR,-(SP)
9003 042334 000002      RTI
9004
9005
9006
9007      .SBTTL SAVE REGISTERS ROUTINE
9008
9009
9010      ;*THIS SAVES THE CONTENTS OF ALL HARDWARE REGISTERS
9011      ;*IN MEMORY LOCATIONS TAGED FROM "WC" TO "EC2"
9012      ;*
```

```
9013
9014
9015
9016
9017
9018 042336
9019 042336 010046
9020 042340 010146
9021 042342 010246
9022 042344 012700 001632
9023 042350 012701 001706
9024 042354 012702 000023
9025 042360 013021
9026 042362 005302
9027 042364 001375
9028 042366 012602
9029 042370 012601
9030 042372 012600
9031 042374 000207
9032
9033
9034
9035
9036
9037
9038
9039
9040
9041
9042
9043 042376 000000
9044 042400 000000
9045 042402 000000
9046
9047 042404 012537 042376
9048 042410 012504
9049 042412 010437 042402
9050 042416 010537 042400
9051 042422 062705 000004
9052 042426 012703 000001
9053 042432 004737 042454
9054 042436 004737 042454
9055 042442 000241
9056 042444 006103
9057 042446 005703
9058 042450 001370
9059 042452 000205
9060 042454 005103
9061 042456 012737 042464 042712
9062 042464 032777 001000 136446
9063 042472 001411
9064 042474 105737 001103
9065 042500 001406
9066 042502 000005
9067 042504 013777 001774 137124
9068 042512 004737 055310

; *
; * THIS IS DONE SO THAT COMPARES ARE DONE WITH SAVED LOCATIONS
; * AND NOT THE REGISTERS THEMSELVES. THIS WILL MAKE
; * ERROR PRINTOUTS FOR GOOD AND BAD DATA ALWAYS DIFFRENT

PUTREG:
MOV R0,-(SP) ;;PUSH R0 ON STACK
MOV R1,-(SP) ;;PUSH R1 ON STACK
MOV R2,-(SP) ;;PUSH R2 ON STACK
MOV #RHC,R0 ;STARTING ADDRESS OF REG
MOV #WC,R1 ;STARTING ADDRESS OF WERE SAVED
MOV #RHCC-RHC+2/2,R2 ;NUMBER OF REG. INTO R2
10$: MOV @ (R0)+,(R1)+ ;SAVE HARDWARE REG.
DEC R2
BNE 10$
MOV (SP)+,R2 ;;POP STACK INTO R2
MOV (SP)+,R1 ;;POP STACK INTO R1
MOV (SP)+,R0 ;;POP STACK INTO R0
RTS PC

.SBTTL FLOAT 1 AND 0

; *FLOAT A ONE AND A ZERO THRU A DESIGNATED REGISTER
; *ABSOLUTE ADDRESS OF REG. UNDER TEST IS IN R4
MASK: 0 ;BITS UNDER TEST
LERR: 0 ;ERROR HLT ADDRESS
REGADR: 0
BITST: MOV (R5)+, MASK ;FETCH DATA MASK
MOV (R5)+, R4 ;GET ADDRESS OF REG. UNDER TEST
MOV R4, REGADR
MOV R5, LERR ;GET ERROR RETURN ADDR.
ADD #4, R5 ;MODIFY RETURN ADDR. TO JUMP OVER RTS
MOV #1, R3 ;INITIALIZE DATA PATTERN
BLT1: JSR PC, BLT2 ;OUTPUT FLOATING ZERO
JSR PC, BLT2 ;OUTPUT FLOATING ONE
CLC
ROL R3 ;SHIFT PATTERN
TST R3
BNE BLT1 ;BRANCH IF NOT COMPLETE
RTS R5 ;RETURN TO TEST
BLT2: COM R3 ;COMPLEMENT PATTERN
MOV #BLT3, @#LAD ;SET SCOPE LOOP
BLT3: BIT #SW09,@SWR ;LOOP ON ERROR
BEQ 4$ ;BRANCH IF NO
TSTB @#SERFLG ;ANY ERRORS
BEQ 4$ ;BRANCH IF NO
RESET ;START WITH AN INIT
MOV @#UNIT,@RHCS2 ;SET UNIT NUMBER UNDER TEST
JSR PC,@#TKINT ;INITILIZE TK
```

```
9069                                     ;INIT FOR SCOPING LOOPS
9070 042516 010337 001124 4$: MOV R3,@#SGDDAT ;STORE GOOD DATA
9071 042522 005137 042376 COM @#MASK ;AND MASK WITH PATTERN
9072 042526 043737 042376 001124 BIC @#MASK,@#SGDDAT ;CLEAR THE REST
9073 042534 005137 042376 COM @#MASK ;RESTORE MASK
9074 042540 013714 001124 MOV @#SGDDAT,(R4) ;OUTPUT TO REGISTER
9075 042544 011437 001126 MOV (R4),@#SBDDAT ;INPUT FROM REGISTER
9076 042550 005137 042376 COM @#MASK
9077 042554 043737 042376 001126 BIC @#MASK,@#SBDDAT ;AND MASK OUT RECEIVED DATA
9078 042562 005137 042376 COM @#MASK ;RESTORE MASK
9079 042566 023737 001124 001126 CMP @#SGDDAT,@#SBDDAT ;IS DATA CORRECT
9080 042574 001424 BEQ 1$ ;BRANCH IF GOOD
9081 042576 011437 001126 MOV (R4),@#SBDDAT
9082 042602 023704 001640 CMP @#RHCS1,R4 ;REGISTER UNDER TEST RHCS1?
9083 042606 001004 BNE 2$ ;BRANCH IF NOT
9084 042610 052737 004200 001124 BIS #RDY!DVA,@#SGDDAT;SET RDY AND DVA
9085 042616 000410 BR 3$
9086 042620 023704 001636 2$: CMP @#RHCS2,R4 ;REGISTER UNDER TEST RHCS2?
9087 042624 001005 BNE 3$ ;BRANCH IF NOT
9088 042626 011446 MOV @R4,-(SP) ;GET RHCS2
9089 042630 042716 177477 BIC #^C<IR!OR>,(SP) ;KEEP IR AND OR BIT
9090 042634 052637 001124 BIS (SP)+,@#SGDDAT ;SET IR OR BITS IF NEEDED
9091 042640 004777 177534 3$: JSR PC,@LERR ;GO TO REPORT ERROR
9092 042644 000240 NOP ;REPLACE BY 104420 FOR LOCAL SCOPE LOOP
9093 042646 000207 1$: RTS PC
```

```
9094
9095
9096 .SBTTL CLEAR MEMORY ROUTINE
9097
9098 ;* THIS CLEARS ANY BLOCK OF MEMORY
9099 ;* FILLING IT WITH ANY DATA
9100 ;*
9101 ;* CALL
9102 ;* JSR RO,CLAREA
9103 ;* X ;STARTING ADDRESS OF BLOCK
9104 ;* Y
9105 ;* Z ;DATA TO BE FILLED
9106 ;*R1 WILL HAVE STARTING ADDRESS OF BLOCK TO BE FILLED
9107 ;*R2 AFTER SUBTRACTION WILL HAVE TWICE NUMBER OF LOCATIONS
9108 ;*R3 WILL HAVE DATA TO BE FILLED
9109 ;*TO AVOID DIVIDE ROUTINE TWO DECREMENT R2 WILL BE USED
```

```
9110
9111
9112 042650 CLAREA:
9113 042650 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
9114 042652 010246 MOV R2,-(SP) ;;PUSH R2 ON STACK
9115 042654 010346 MOV R3,-(SP) ;;PUSH R3 ON STACK
9116 042656 012001 MOV (R0)+,R1 ;FROM
9117 042660 012002 MOV (R0)+,R2 ;TO
9118 042662 012003 MOV (R0)+,R3 ;DATA
9119 042664 160102 SUB R1,R2 ;NO. OF LOCATIONS MINUS TWO
9120 042666 062702 000002 ADD #2,R2 ;GET TWICE NO OF LOCATIONS
9121 042672 010321 1$: MOV R3,(R1)+ ;MOVE IN DATA
9122 042674 005302 DEC R2
9123 042676 005302 DEC R2
9124 042700 001374 BNE 1$ ;BRANCH IF NOT COMPLETE
```

CZRJGDO,RP04/5/6 DSKLS CTRLR1
CZRJGD.P11 10-SEP-79 11:00

MACY11 30A(1052) 10-SEP-79 11:11 L 16 PAGE 207
CLEAR MEMORY ROUTINE

SEQ 0206

| | | | | | |
|------|--------|--------|-----|----------|---------------------|
| 9125 | 042702 | 012603 | MOV | (SP)+,R3 | ::POP STACK INTO R3 |
| 9126 | 042704 | 012602 | MOV | (SP)+,R2 | ::POP STACK INTO R2 |
| 9127 | 042706 | 012601 | MOV | (SP)+,R1 | ::POP STACK INTO R1 |
| 9128 | 042710 | 000200 | RTS | RC | :RETURN |


```

9129
9130 042712 000000          LAD:      0
9131
9132 042714 032777 001000 136216 T.SCOPI: BIT    #SW09, @SWR
9133 042722 001402          BEQ      1$
9134 042724 013716 042712          MOV     @#LAD, (SP)
9135 042730 000002          1$:     RTI
9136
9137          ;*EXAMPLE OF THE USE OF THE ABOVE
9138          ;*THIS WILL LOOP BETWEEN X: AND SCOP1 PROVIDED THERE IS NO 'NEWTST'
9139          ;*MOV     #X,      @#LAD
9140          ;*X:     ---      ---
9141          ;*       ---      ---
9142          ;*       ---      ---
9143          ;*       SCOP1
9144
9145          .SBTTL  CLEAR DISK ROUTINE
9146
9147          CLDISK: MOV     @#RHCS1,      R1      ;R1 WILL BE CONTROL AND STATUS1
9148 042732 013701 001640          MOV     @#RHCS2,      R2      ;R2 WILL BE CONTROL AND STATUS2
9149 042736 013702 001636          MOV     @#RHDS1,      R3      ;R3 WILL BE DISK STATUS REGISTER1
9150 042742 013703 001662          MOV     @#RHER1,      R4      ;R4 WILL BE ERROR REGISTER #1
9151 042746 013704 001642
9152
9153 042752 012712 000040          MOV     #CLR,@R2          ;CLEAR ALL REG.
9154 042756 013712 001774          MOV     @#UNIT,@R2      ;REINSTATE UNIT NO.
9155 042762 005011          CLR     @R1              ;CLEAR FUNCTION BITS
9156 042764 000207          RTS     PC
  
```

9157
9158
9159
9160
9161
9162
9163
9164
9165
9166
9167
9168
9169
9170
9171
9172
9173
9174
9175
9176
9177
9178
9179
9180
9181
9182
9183
9184
9185
9186
9187
9188
9189
9190
9191
9192
9193
9194
9195
9196
9197
9198
9199
9200
9201
9202
9203
9204
9205
9206
9207
9208
9209
9210
9211
9212

.SBTTL CHECK DISK STATUS ROUTINES

;*THIS CHECKS THAT DEVICE AVAILABLE (DVA) AND READY (RDY) IN RHCS1 = 1
;*AND CHECKS THAT DEVICE PRESENT (DPR), DEVICE READY (DRY) IN RHDS1 = 1
;*IT ALSO CHECKS THAT NO OTHER BITS IN THESE REGISTERS = 1

```

CHECKT: MOV      (SP),@#PCJSR      ;SAVE PC OF JSR+4
          SUB      #4,@#PCJSR      ;GET PC OF JSR
          JSR      PC,@#PUTREG     ;SAVE REGISTERS
          CMP      #DVA!RDY,@#CS1  ;RHCS1 SHOULD HAVE DEVICE AVAILABLE
          BEQ      3$              ;AND BE READY
          ;BRANCH IF GOOD

          BIT      #DVA,@#CS1      ;BAD SO TEST DEVICE AVAILABLE
          BNE      1$              ;BRANCH IF DVA THERE
          MOV      R1,@#$BDADR     ;ADDRESS OF BAD REGISTER (RHCS1)
          ERROR   26              ;RHCS1 DID NOT HAVE DEVICE
          ;AVAILABLE AT START OF TEST
          BR       3$              ;BRANCH TO NEXT COMPARE

          BIT      #RDY,@#CS1      ;TEST READY
          BNE      2$              ;IF RDY THERE BRANCH
          MOV      R1,@#$BDADR     ;ADDRESS OF BAD REGISTER (RHCS1)
          ERROR   26              ;RHCS1 DID NOT HAVE READY
          ;RIGHT AT START OF TEST
          BR       3$              ;BRANCH TO NEXT COMPARE

          MOV      R1,@#$BDADR     ;ADDRESS OF BAD REGISTER (RHCS1)
          ERROR   26              ;RHCS1 HAD SOME BITS OTHER
          ;THAN DVA AND RDY SET
          ;ALL OTHER BITS SHOULD BE 0

          MOV      @#DS1,-(SP)     ;GET RHDS1
          BIC      #VV!PROG,(SP)   ;CLEAR VV AND PROGRAMABLE BIT
          CMP      #DPR!DRY,(SP)+;RHDS1 SHOULD HAVE THESE SET
          BEQ      8$              ;BRANCH IF THEY ARE

          BIT      #DPR,@#DS1      ;TEST DRIVE PRESENT
          BNE      5$              ;CONTINUE IF THERE
          MOV      R3,@#$BDADR     ;ADDRESS OF BAD REGISTER (RHDS1)
          ERROR   26              ;RHDS1 DOES NOT HAVE DPR
          BR       7$              ;BRANCH OUT

          BIT      #DRY,@#DS1      ;TEST DRIVE READY
          BNE      6$              ;IF DPR WAS THERE, BRANCH IF GOOD
          MOV      R3,@#$BDADR     ;ADDRESS OF BAD REGISTER (RHDS1)
          ERROR   26              ;RHDS1 DOES NOT HAVE DRY
          BR       7$              ;BRANCH OUT

          MOV      R3,@#$BDADR     ;ADDRESS OF BAD REGISTER (RHDS1)
          ERROR   26              ;RHDS1 HAS SOME BITS OTHER
          ;THAN MOL, DRY, DPR, SET
          ;ALL OTHER BITS SHOULD BE 0
          ;RETURN TO TEST AND HALT

          RTS      PC              ;RETURN TO TEST AND HALT

          ADD      #6,(SP)         ;ADJUST STACK TO GET OVER HALT IN TEST
          RTS      PC              ;RETURN TO TEST AND CONTINUE TESTING
    
```

CZRJGDD,RP04/5/6 DSKLS CTRLR1
CZRJGD.P11 10-SEP-79 11:00

MACY11 30A(1052) 10-SEP-79 11:11
CHECK DISK STATUS ROUTINES

C 1
PAGE 210

SEQ 0209

9213

```
9214
9215
9216
9217
9218 043156 011637 002014
9219 043162 162737 000004 002014
9220 043170 004737 042336
9221 043174 032737 000200 001714
9222
9223 043202 001004
9224 043204 010137 001122
9225 043210 104026
9226
9227 043212 000427
9228 043214 032737 004000 001714 1$:
9229
9230 043222 001004
9231 043224 010137 001122
9232 043230 104026
9233
9234 043232 000417
9235 043234 032737 000200 001736 2$:
9236 043242 001004
9237 043244 010337 001122
9238 043250 104026
9239 043252 000407
9240 043254 032737 000400 001736 3$:
9241 043262 001004
9242 043264 010337 001122
9243 043270 104026
9244
9245 043272 000207 4$:
9246
9247 043274 062716 000006 5$:
9248 043300 000207
```

;*THIS CHECKS THAT DEVICE AVAILABLE (DVA) AND READY (RDY) IN RHCS1 = 1
;*AND CHECKS THAT DEVICE PRESENT (DPR), DEVICE READY (DRY) IN RHDS1 = 1

CHECKE: MOV (SP),@#PCJSR ;SAVE PC OF JSR+4
SUB #4,@#PCJSR ;GET PC OF JSR
JSR PC,@#PUTREG ;READ & SAVE REGISTERS
BIT #RDY,@#CS1 ;RHCS1 SHOULD HAVE DEVICE AVAILABLE
;AND BE READY
BNE 1\$;BRANCH IF GOOD
MOV R1,@#\$BDADR ;FAILING REGISTER
ERROR 26 ;RHCS1 IS IN ERROR
;DOES NOT HAVE DVA, RDY
BR 4\$;BRANCH OUT
BIT #DVA,@#CS1 ;RHCS1 SHOULD HAVE DEVICE AVAILABLE
;AND BE READY
BNE 2\$;BRANCH IF GOOD
MOV R1,@#\$BDADR ;FAILING REGISTER
ERROR 26 ;RHCS1 IS IN ERROR
;DOES NOT HAVE DVA, RDY
BR 4\$;BRANCH OUT
BIT #DRY,@#DS1 ;RHDS1 SHOULD HAVE DPR, DRY
BNE 3\$;BRANCH IF THERE
MOV R3,@#\$BDADR ;FAILING REGISTER RHDS1
ERROR 26 ;RHDS1 DOES NOT HAVE DPR, DRY
BR 4\$;BRANCH OUT
BIT #DPR,@#DS1 ;RHDS1 SHOULD HAVE DPR, DRY
BNE 5\$;BRANCH OUT AND CONTINUE IF THERE
MOV R3,@#\$BDADR ;FAILING REGISTER RHDS1
ERROR 26 ;RHDS1 DOES NOT HAVE DPR, DRY
RTS PC ;RETURN TO TEST AND HALT
ADD #6,(SP) ;ADJUST STACK TO GET OVER HALT IN TEST
RTS PC ;RETURN TO TEST AND CONTINUE TESTING

```
9249
9250
9251      ;*      WAIT LOOP
9252      ;*      ONE LOOP OR ONE COUNT = 5.15 MICROSEC WITH BIPOLAR MEMORY (MIN)
9253      ;*      ONE LOOP OR ONE COUNT = 11.86 MICROSEC WITH CORE (MIN)
9254      ;*      WITH CORE ERROR IS INDICATED AFTER ABOUT 650 MILLISEC (MIN)
9255
9256 043302 177777      TIMCNT: 177777      ;WAITING COUNT
9257 043304 010046      WAIT.T: MOV      R0,-(SP)      ;SAVE R0
9258 043306 016600 000002      MOV      2(SP),R0      ;GET ADRESS OF REG. ADDERSS
9259 043312 010037 001204      MOV      R0,@#STMP3      ;WAT PC+2 IN $TMP3
9260 043316 162737 000002 001204      SUB      #2,@#STMP3      ;WAT PC FOR TYPEOUT
9261 043324 012037 001176      MOV      (R0)+,@#STMP0      ;WAIT REGISTER ADDRESS
9262 043330 012037 001200      MOV      (R0)+,@#STMP1      ;WAIT ON BIT
9263 043334 010066 000002      MOV      R0,2(SP)      ;RESTORE RETURN ON STACK
9264 043340 012600      MOV      (SP)+,R0      ;RESTORE R0
9265 043342 013737 043302 001202      MOV      @#TIMCNT,@#STMP2      ;TEMPORARY COUNT
9266 043350 033777 001200 135620 1$:      BIT      @#STMP1,@#STMP0      ;IS REQUIRED BIT THERE?
9267 043356 001021      BNE      2$      ;BRANCH IF YES
9268 043360 005337 001202      DEC      @#STMP2      ;COUNT
9269 043364 001371      BNE      1$      ;BRANCH IF NOT TIME UP
9270 043366 013737 043302 001202      MOV      @#TIMCNT,@#STMP2      ;TEMPORARY COUNT
9271 043374 033777 001200 135574 3$:      BIT      @#STMP1,@#STMP0      ;IS REQUIRED BIT THERE?
9272 043402 001007      BNE      2$      ;BRANCH IF YES
9273 043404 005337 001202      DEC      @#STMP2      ;COUNT
9274 043410 001371      BNE      3$      ;BRANCH IF NOT TIME UP
9275 043412 017737 135560 001126      MOV      @#STMP0,@#SBDDAT      ;REGISTER CONTENTS
9276 043420 104016      ERROR   16      ;WAITED ON BIT FAILED TO SET
9277 043422 000002      2$:      RTI
9278
9279
9280
9281      ;*      CALL FOR THE ABOVE WAITLOOP IS
9282      ;*
9283      ;*      MOV      @A,@#X$      ;A CONTAINS REGISTER ADDRESS
9284      ;*      -      -      -      ;HENCE X$ WILL HAVE ABSOLUTE REG. ADR.
9285      ;*      -      -      -
9286      ;*      -      -      -
9287      ;*      WAT
9288      ;*      0      ;ABSOLUTE REG. ADDRESS UNDER WAIT
9289      ;*      .WORD 0      ;BIT WAITED FOR
9290      ;*
9291      ;*
```

9292
 9293
 9294
 9295
 9296
 9297
 9298
 9299
 9300
 9301
 9302
 9303 043424
 9304 043424 010146
 9305 043426 010246
 9306 043430 010346
 9307 043432 012001
 9308 043434 012002
 9309 043436 012003
 9310 043440 013122
 9311 043442 005303
 9312 043444 001375
 9313 043446 012603
 9314 043450 012602
 9315 043452 012601
 9316 043454 000200
 9317
 9318
 9319
 9320
 9321
 9322
 9323
 9324
 9325
 9326
 9327 043456 012737 010000 047616
 9328 043464 112737 000001 047621
 9329 043472 112737 000001 047620
 9330 043500 005037 047622
 9331 043504 005037 047624
 9332 043510 012737 000044 047676
 9333 043516 005037 047626
 9334 043522 004537 044140
 9335 043526 047616
 9336 043530 051516
 9337
 9338
 9339
 9340 043532 004737 042732
 9341
 9342 043536 012777 177730 136066
 9343 043544 012777 003160 136062
 9344 043552 112746 000001
 9345 043556 112766 000001 000001
 9346 043564 012677 136054
 9347 043570 012777 014000 136052

.SBTTL SAVE ROUTINE

;*THIS IS A SUBROUTINE TO SAVE REGISTERS
 ;*IN THE REGISTER TABLE TO ANY LOCATION
 ;*THE CALL IS
 ;*JSR R0,@#SAVER
 ;*FROM
 ;*TO
 ;*NUMBER OF WORDS SAVED

SAVER:

```

MOV R1,-(SP)    ;;PUSH R1 ON STACK
MOV R2,-(SP)    ;;PUSH R2 ON STACK
MOV R3,-(SP)    ;;PUSH R3 ON STACK
MOV (R0)+,R1    ;FROM
MOV (R0)+,R2    ;TO
MOV (R0)+,R3    ;NUMBER
1$:MOV @R1+,(R2)+;SAVE REGISTER CONTENTS
DEC R3          ;COUNT
BNE 1$         ;BRANCH IF NOT DONE
MOV (SP)+,R3   ;;POP STACK INTO R3
MOV (SP)+,R2   ;;POP STACK INTO R2
MOV (SP)+,R1   ;;POP STACK INTO R1
RTS R0
  
```

.SBTTL WRITE CHECK ROUTINE

;*THIS IS A SUBROUTINE TO DO WRITE CHECK HEADER AND DATA
 ;*CYLINDER 0, TRACK 1, SECTOR 1, KEYS 0

WRCHHD:

```

;*THESE ARE TO SET UP FOR DISKLESS USE ONLY
MOV #FMT22,@#CYL ;CYLINDER 0 FORMAT 16 BIT WORDS
MOVB #1,@#SECOTR+1 ;TRACK=1
MOVB #1,@#SECOTR  ;SECTOR=1
CLR @#KEY1        ;KEY1=0
CLR @#KEY2        ;KEY2=0
MOV #36.,DAWORD  ;NO OF DATA WORDS
CLR @#X           ;THIS IS A READ OPERATION
JSR R5,@#CRC      ;GO TO CALCULATE CRC
CYL
WCRC
  
```

;*THESE ARE REGULAR SETUPS

```

JSR PC,@#CLDISK  ;SET UP GENERAL REGISTERS
;AND CLEAR DISK REGISTERS
MOV #-40.,@RHWC  ;36 DATA WORDS 4 HEADER WORDS
MOV #REINTO,@RHBA ;STARTING ADDRESS OF READ BUFFER
MOVB #1,-(SP)    ;SECTOR=1
MOVB #1,1(SP)    ;TRACK=1 IN UPPER BYTE
MOV (SP)+,@RHDST ;TRACK=1, SECTOR=1 IN RHDST
MOV #FMT22!ECI,@RHOF ;16 BIT WORDS
  
```


9363
9364
9365
9366
9367
9368
9369
9370
9371
9372
9373
9374
9375
9376
9377
9378
9379
9380
9381
9382
9383
9384
9385
9386
9387
9388
9389
9390
9391
9392
9393
9394
9395
9396
9397
9398
9399
9400
9401
9402
9403
9404
9405
9406
9407
9408
9409
9410
9411
9412
9413
9414
9415
9416
9417
9418

.SBTTL COMPARE ROUTINE

;*THIS IS A SUBROUTINE TO COMPARE TWO BLOCKS IN MEMORY
;*R1 HAS GOOD DATA BUFFER ADDRESS
;*R2 HAS TEST DATA BUFFER ADDRESS
;*\$TMP0 HAS ADDRESS OF RETURN ON ERROR TO PRINT HEADER
;*\$TMP1 HAS ADDRESS OF RETURN ON ERROR TO PRINT DATA
;*R3 HAS NUMBER OF WORDS TO BE COMPARED
;*R4 HAS ONE MORE THAN NUMBER OF WORDS TO BE COMPARED

COMPAR:

```

MOV R1,-(SP)      ;;PUSH R1 ON STACK
MOV R2,-(SP)      ;;PUSH R2 ON STACK
MOV R3,-(SP)      ;;PUSH R3 ON STACK
MOV R4,-(SP)      ;;PUSH R4 ON STACK
MOV R5,-(SP)      ;;PUSH R5 ON STACK
MOV (R0)+,R1      ;ADDRESS OF GOOD DATA BUFFER
MOV (R0)+,R2      ;ADDRESS OF TEST DATA BUFFER
MOV (R0)+,R3      ;NO OF WORDS TO BE COMPARED
MOV (R0)+,$TMP0   ;RETURN ON ERROR TO PRINT HEADER
MOV (R0)+,$TMP1   ;RETURN ON ERROR TO PRINT DATA
MOV (R0),R0       ;RETURN ON NO ERROR
MOV R3,R4        ;NO OF WORDS TO BE COMPARED
INC R4
1$: MOV R4,@#ERWORD ;FOR ERROR WORD NO
    CMP (R1)+,(R2)+ ;COMPARE GOOD WITH TEST DATA
    BEQ 3$         ;BRANCH IF GOOD

MOV -(R1),@#$GDDAT ;GOOD DATA
MOV -(R2),@#$BDDAT ;BAD DATA
SUB R3,@#ERWORD    ;ERROR WORD NO.
TST @#ERFLG$      ;ANY ERRORS ALREAY THERE
BNE 2$            ;BRANCH IF YES
JSR PC,@$TMP0     ;RETURN TO PRINT HEADER
BR 5$            ;BRANCH TO AVOID PRINTING NEXT ERROR
2$: JSR PC,@$TMP1 ;RETURN TO PRINT DATA
5$: CMP (R1)+,(R2)+ ;UNDO -(R1) AND -(R2) FOR ERRORS
    MOV @SWR,-(SP) ;GET SWITCH SETTING
    BIC #^C600,(SP) ;KEEP ONLY SWITCH 7 AND 8
    CMP #SW07,(SP)+ ;IS 7 SET AND 8 RESET
    BEQ 4$         ;BRANCH OUT IF YES
3$: DEC R3        ;COUNT
    BNE 1$        ;BRANCH IF ALL NOT DEVICE
4$:

MOV (SP)+,R5     ;;POP STACK INTO R5
MOV (SP)+,R4     ;;POP STACK INTO R4
MOV (SP)+,R3     ;;POP STACK INTO R3
MOV (SP)+,R2     ;;POP STACK INTO R2
MOV (SP)+,R1     ;;POP STACK INTO R1
RTS R0           ;RETURN TO MAIN PROGRAM
    
```

001176
001200
047736
001124
001126
047736
002006
135254
135250
135202
177177
000200


```

9419
9420      ;*THIS IS A SUBROUTINE TO DO WRITE CHECK DATA
9421      ;*CYLINDER 0, TRACK 1, SECTOR 1, KEYS 0
9422
9423      ;*THESE ARE TO SET UP FOR DISKLESS USE ONLY
9424
9425 043770 012737 010000 047616 WRCHDA: MOV    #FMT22,@#CYL    ;CYLINDER 0 FORMAT 16 BIT WORDS
9426 043776 112737 000001 047621      MOVB   #1,@#SECOTR+1 ;TRACK=1
9427 044004 112737 000001 047620      MOVB   #1,@#SECOTR   ;SECTOR=1
9428 044012 005037 047622      CLR    @#KEY1        ;KEY1=0
9429 044016 005037 047624      CLR    @#KEY2        ;KEY2=0
9430 044022 012737 000040 047676      MOV    #32.,@#DAWORD ;NO OF DATA WORDS
9431 044030 005037 047626      CLR    @#X           ;THIS IS A READ OPERATION
9432
9433 044034 004537 044140      JSR    R5,@#CRC      ;GO TO CALCULATE CRC
9434 044040 047616
9435 044042 051516
9436
9437      ;*THESE ARE REGULAR SETUPS
9438
9439 044044 004737 042732      JSR    PC,@#CLDISK  ;SET UP GENERAL REGISTERS
9440                                ;AND CLEAR DISK REGISTERS
9441
9442 044050 012777 177740 135554      MOV    #-32.,@RHWC   ;36 DATA WORDS 4 HEADER WORDS
9443 044056 012777 003160 135550      MOV    #REINTO,@RHBA ;STARTING ADDRESS OF READ BUFFER
9444 044064 112746 000001          MOVB   #1,-(SP)      ;SECTOR=1
9445 044070 112766 000001 000001      MOVB   #1,1(SP)     ;TRACK=1 IN UPPER BYTE
9446 044076 012677 135542          MOV    (SP)+,@RHDST ;TRACK=1, SECTOR=1 IN RHDST
9447 044102 012777 014000 135540      MOV    #FMT22!ECI,@RHOF ;16 BIT WORDS
9448                                ;ECC CORRECTION INHIBIT BECAUSE
9449                                ;ECC LOGIC IS NOT CHECKED YET
9450 044110 005077 135536      CLR    @RHCA        ;CYLINDER=0
9451 044114 004737 042766      JSR    PC,@#CHECKT  ;CHECK THAT DVA,RDY,DPR,DRY = 1
9452 044120 104401 062562      TYPE   ,CPHALT     ;AND THAT NO OTHERS = 1. CANNOT CON-
9453                                ;TINUE TESTING IF BOTH AREN'T TRUE
9454 044124 000000          HALT                    ;STOP THE TEST
9455 044126 013711 002060      MOV    @#WRCHK, @R1 ;WRITE CHECK DATA=50 INTO RHCS1
9456 044132 004737 047456      JSR    PC,@#COMHD  ;WRITE CHECK HEADER AND DATA
9457                                ;SAME AS READ HEADER AND DATA
9458
9459 044136 000207      RTS    PC           ;RETURN TO WRITE CHECK TEST

```

```

9460
9461          .SBTTL  CRC GENERATION ROUTINE
9462
9463
9464          ;*THIS IS A SUBROUTINE TO CALCULATE CRC FOR THE FOUR
9465          ;*HEADER WORDS AND STORE THEM IN "WCRC" AND "GCRC"
9466          ;*R1 - REGISTER FOR CRC, INCREMENTED CRC VALUE IS HERE
9467          ;*R2 - THIS HAS BIT POSITION 2 VALUE C
9468          ;*R3 - THIS HAS BIT POSITION 16 I.E. OUTPUT BIT VALUE B
9469          ;*R4 - THIS HAS BIT POSITION 15 VALUE E
9470          ;*$TMP0 - NUMBER OF WORDS
9471          ;*$TMP2 - NUMBER OF BITS PER WORD = 16
9472          ;*$TMP3 - TEMPORARY REG.
9473          ;*$TMP4 - TEMPORARY REG TO TRANSFER CARRY
9474          ;*$TMP5 - THIS HAS DATA BIT VALUE D
9475
9476          ;*FETCH DATA BIT D
9477          ;*B = D XOR 16
9478          ;*C = B XOR 2
9479          ;*E = B XOR 15
9480          ;*ROTATE RIGHT ONE POSITION
9481          ;*B GOES TO POSITION 1
9482          ;*C GOES TO POSITION 3
9483          ;*E GOES TO POSITION 16
9484          ;*REPET 64 TIMES
9485          ;*CALL JSR      R5,@#CRC
9486          ;*X      ;FIRST LOCATION AT
9487          ;*Y      ;PUT CRC IN WCRC FOR READ GCRC FOR WRITE
9488
9489 044140          CRC:
9490 044140 010046  MOV      R0,-(SP)          ;;PUSH R0 ON STACK
9491 044142 012500  MOV      (R5)+,R0          ;GET POINTER TO CYL NO.
9492 044144 010146  MOV      R1,-(SP)          ;;PUSH R1 ON STACK
9493 044146 010246  MOV      R2,-(SP)          ;;PUSH R2 ON STACK
9494 044150 010346  MOV      R3,-(SP)          ;;PUSH R3 ON STACK
9495 044152 010446  MOV      R4,-(SP)          ;;PUSH R4 ON STACK
9496 044154 005001  CLR      R1                ;CLEAR WORKING LOCATION
9497 044156 005037 001210  CLR      @#$TMP5
9498 044162 012737 000004 001176  MOV      #4,@#$TMP0        ;WORD COUNT
9499 044170 012037 001204 16$:  MOV      (R0)+,@#$TMP3    ;TEMPORARY WORD STORAGE
9500 044174 012737 000020 001202  MOV      #16,,$TMP2       ;BIT COUNT
9501 044202 013737 001204 001206  MOV      @#$TMP3,@#$TMP4  ;TEMPORARY WORD STORAGE
9502 044210 006037 001204 15$:  ROR      @#$TMP3          ;GET LSB INTO "C"
9503 044214 006037 001210  ROR      @#$TMP5          ;GET ABOVE "C" INTO $TMP5
9504 044220 032701 000001  BIT      #BIT0,R1         ;IS POSITION 15 HIGH
9505 044224 001403  BEQ      1$                ;BRANCH IF POSITION 16 LOW
9506 044226 012703 100000  MOV      #BIT15,R3        ;GET POSITION 16
9507 044232 000401  BR      2$
9508 044234 005003 1$:  CLR      R3                ;GET POSITION 16
9509 044236 063703 001210 2$:  ADD      @#$TMP5,R3        ;XOR POSITION 16 WITH D
9510                                ;TO GIVE B
9511 044242 032701 040000  BIT      #BIT14,R1        ;IS POSITION 2 HIGH
9512 044246 001403  BEQ      3$                ;BRANCH IF POSITION 2 LOW
9513 044250 012702 100000  MOV      #BIT15,R2        ;GET POSITION 2
9514 044254 000401  BR      4$
9515 044256 005002 3$:  CLR      R2                ;GET POSITION 2

```

```

9516 044260 060302      4$:  ADD      R3,R2      ;XOR B WITH POSITION 2
9517                                ;TO GIVE C
9518 044262 032701 000002  BIT      #BIT1,R1      ;IS POSITION 15 HIGH
9519 044266 001403      BEQ      5$            ;BRANCH IF POSITION 15 LOW
9520 044270 012704 100000  MOV      #BIT15,R4     ;GET POSITION 15
9521 044274 000401      BR       6$
9522 044276 005004      5$:  CLR      R4            ;GET POSITION 15
9523 044300 060304      6$:  ADD      R3,R4      ;XOR POSITION 15 WITH B
9524                                ;TO GIVE E
9525 044302 006037 001206  ROR      @#$TMP4      ;GET LSB INTO "C"
9526 044306 006001      ROR      R1            ;GET ABOVE C INTO R1
9527 044310 005703      TST      R3            ;TEST B
9528 044312 100403      BMI      7$            ;BRANCH IF B=1
9529 044314 042701 100000  BIC      #BIT15,R1     ;SET B IN POSITION 1
9530 044320 000402      BR       10$
9531 044322 052701 100000  7$:  BIS      #BIT15,R1   ;SET B IN POSITION 1
9532 044326 005702      10$: TST      R2            ;TEST C
9533 044330 100403      BMI      11$          ;BRANCH IF C=1
9534 044332 042701 020000  BIC      #BIT13,R1     ;GET C IN POSITION 3
9535 044336 000402      BR       12$
9536 044340 052701 020000  11$: BIS      #BIT13,R1   ;GET C IN POSITION 3
9537 044344 005704      12$: TST      R4            ;TEST E
9538 044346 100403      BMI      13$          ;BRANCH IF E=1
9539 044350 042701 000001  BIC      #BIT0,R1     ;GET E IN POSITION 16
9540 044354 000402      BR       14$
9541 044356 052701 000001  13$: BIS      #BIT0,R1     ;GET E IN POSITION 16
9542 044362 005337 001202  14$: DEC      @#$TMP2    ;BIT COUNTER
9543 044366 001310      BNE      15$          ;BRANCH IF 16 NOT DONE
9544 044370 005337 001176  DEC      @#$TMP0      ;WORD COUNTER
9545 044374 001275      BNE      16$          ;BRANCH IF 4 NOT DONE
9546 044376 010135      MOV      R1,@(R5)+    ;PUT CRC WHERE DESIRED
9547 044400 012604      MOV      (SP)+,R4     ;;POP STACK INTO R4
9548 044402 012603      MOV      (SP)+,R3     ;;POP STACK INTO R3
9549 044404 012602      MOV      (SP)+,R2     ;;POP STACK INTO R2
9550 044406 012601      MOV      (SP)+,R1     ;;POP STACK INTO R1
9551 044410 012600      MOV      (SP)+,R0     ;;POP STACK INTO R0
9552 044412 000205      RTS      R5

```

```

9553
9554
9555
9556
9557                                ;*THIS IS A SUBROUTINE TO SET UP THE SIMULATOR DISK FOR
9558                                ;*CYLINDER 0 (16 BITS PER WORD)
9559                                ;*TRACK 1, SECTOR 1
9560                                ;*KEY1 1
9561                                ;*KEY2 1
9562                                ;*CRC THROUGH THE JSR R5,@#CRC
9563                                ;*256 WORDS OF 177400
9564
9565                                ;*CALL JSR PC,@#SETDSK
9566

```

```

9567 044414      SETDSK:  MOV      R0,-(SP)      ;;PUSH R0 ON STACK
9568 044414 010046  MOV      R1,-(SP)      ;;PUSH R1 ON STACK
9569 044416 010146  MOV      R2,-(SP)      ;;PUSH R2 ON STACK
9570 044420 010246  MOV      #177400,R0    ;DATA IN THE DISK
9571 044422 012700 177400

```

```
9572 044426 012701 000400      MOV      #256.,R1      ;COUNTER
9573 044432 012702 051534      MOV      #DISK,R2     ;START OF SIMULATOR DISK
9574 044436 010022             1$: MOV      R0,(R2)+    ;MOVE IN DATA
9575 044440 005301             DEC      R1           ;COUNT FOR 256
9576 044442 001375             BNE     1$           ;BRANCH IF 256 NOT COMPLETE
9577 044444 012701 000021      MOV      #17.,R1     ;2 ECC WORDS, 1 DATA GAP
9578                                ;14 TOLERANCE GAP
9579 044450 005022             2$: CLR      (R2)+    ;CLEAR ECC,DATA GAP AND
9580                                ;TOLERANCE GAP
9581 044452 005301             DEC      R1           ;COUNT
9582 044454 001375             BNE     2$           ;BRANCH IF NOT COMPLETE
9583
9584                                ;*NOW SET UP FOR DISKLESS USE
9585
9586 044456 012737 010000 047616  MOV      #FMT22,@#CYL ;CYLINDER 0 (16 BIT WORDS)
9587 044464 112737 000001 047621  MOVVB   #1,@#SECOTR+1 ;TRACK=1
9588 044472 112737 000001 047620  MOVVB   #1,@#SECOTR  ;SECTOR=1
9589 044500 012737 000001 047622  MOV      #1,@#KEY1   ;KEY1=1
9590 044506 012737 000001 047624  MOV      #1,@#KEY2   ;KEY2=1
9591 044514 013737 000400 047676  MOV      256.,@#DAWORD ;NO. OF DATA WORDS
9592 044522 004537 044140      JSR     R5,@#CRC     ;GO TO CALCULATE CRC
9593 044526 047616             CYL           ;FIRST CRC WORD
9594 044530 051516             WCRC        ;PUT CALCULATED CRC
9595 044532 012602             MOV      (SP)+,R2   ;;POP STACK INTO R2
9596 044534 012601             MOV      (SP)+,R1   ;;POP STACK INTO R1
9597 044536 012600             MOV      (SP)+,R0   ;;POP STACK INTO R0
9598 044540 000207             RTS      PC
```

```

9599
9600      ;*THIS IS A SUBROUTINE TO CHECK HEADER COMPARE ERROR
9601      ;*(BIT #7) AND CRC ERROR (BIT #8)
9602
9603      ;*CALL JSR RO,@#HCCRCE
9604
9605      ;*      COM      ;COMMAND-READ HEADER AND DATA
9606      ;              -WRITE DATA
9607      ;*      C      ;CYLINDER
9608      ;*      S      ;SECTOR
9609      ;*      T      ;TRACK
9610      ;*      -N.    ;WORD COUNT
9611      ;*      B      ;RHBA BUFFER START
9612      ;*      X      ;1=WRITE DATA 0=READ
9613      ;*      H      ;H=1 HEADER CHECK, H=0 CRC CHECK
9614
9615 044542 010037 002014      HCCRCE: MOV      RO,@#PCJSR      ;SAVE PC OF JSR+4
9616 044546 162737 000004 002014 SUB      #4,@#PCJSR      ;GET PC OF JSR
9617 044554 004737 042732 JSR      PC,@#CLDISK     ;INIT AND SETUP GENERAL REG.
9618 044560 004737 042766 JSR      PC,@#CHECKT     ;CHECK THAT DVA,RDY,DPR,DRY = 1
9619 044564 104401 062562 TYPE      ,CPHALT        ;AND THAT NO OTHERS = 1. CANNOT CON-
9620                                     ;TINUE TESTING IF BOTH AREN'T TRUE
9621 044570 000000 HALT                                     ;STOP THE TEST
9622 044572 011037 001210 MOV      (RO),@#$TMP5     ;SAVE COMMAND
9623 044576 012011 MOV      (RO)+,@R1        ;COMMAND
9624 044600 012077 135046 MOV      (RO)+,@RHCA     ;CYLINDER
9625 044604 112046 MOV      (RO)+,-(SP)     ;SECTOR
9626 044606 105720 TSTB     (RO)+           ;UP DATE RO
9627 044610 112066 000001 MOV      (RO)+,1(SP)     ;TRACK
9628 044614 105720 TSTB     (RO)+           ;UPDATE RO
9629 044616 012677 135022 MOV      (SP)+,@RHDST    ;TRACK SECTOR
9630 044622 012077 135004 MOV      (RO)+,@RHWC     ;NO. OF DATA WORDS +4 HEADER
9631                                     ;IF A READ HEADER AND DATA
9632 044626 012077 135002 MOV      (RO)+,@RHBA     ;STARTING ADDRESS OF BUFFER
9633 044632 012037 047626 MOV      (RO)+,@#X      ;X=0 READ HEADER AND DATA
9634                                     ;X=1 WRITE DATA
9635 044636 012777 014000 135004 MOV      #FMT22!ECI,@RHOF ;16 BITS PER WORD
9636                                     ;ECC CORRECTION INHIBIT
9637 044644 005037 002006 CLR      @#ERFLG$        ;CLEAR ERROR FLAG
9638 044650 004737 047456 JSR      PC,@#COMHD     ;COMMAND
9639
9640      ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
9641      ;*FROM THE "COMHD" ROUTINE THAT MEANS SECTOR GAP,
9642      ;*FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
9643      ;*SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
9644      ;*DETECTED
9645      ;*HEADER AND DATA ARE TO BE CHECKED.
9646
9647 044654 004737 042336 JSR      PC,@#PUTREG     ;SAVE REGISTERS
9648 044660 005737 002006 TST      @#ERFLG$        ;ANY ERRORS ALREADY THERE
9649 044664 001034 BNE      10$            ;BRANCH IF YES
9650 044666 005737 047626 TST      @#X            ;IS THIS A READ
9651 044672 001015 BNE      3$            ;IF A WRITE DATA BRANCH

```

```

9652
9653
9654
9655
9656
9657
9658
9659 044674 004037 043626 JSR RO,@#COMPAR ;CHECK
9660 044700 002114 WRFROM ;GOOD DATA
9661 044702 003160 REINTO ;TEST BUFFER
9662 044704 000400 256. ;4 HEADER 252 DATA
9663 044706 044714 1$ ;RETURN POINT FOR ERROR HEADER
9664 044710 044720 2$ ;RETURN POINT FOR ERROR DATA
9665 044712 044756 10$ ;RETURN FOR GOOD COMPARISON
9666 044714 104004 1$: ERROR 4 ;READ NEXT ERROR 5
9667 044716 000207 RTS PC ;RETURN TO COMPARISON SUBROUTINE
9668 044720 104005 2$: ERROR 5 ;WORD NO 1 THRU 4 ARE
9669 ;HEADER WORDS AND HENCE
9670 ;SHOULD BE READ AS WRITTEN ON
9671 ;DISK, WORD NOS. 5 ONWARDS
9672 ;SHOULD NOT BE READ AND HENCE
9673 ;READ INTO BUFFER
9674 ;SHOULD BE UNCHANGED
9675 044722 000207 RTS PC ;RETURN TO COMPARISON
9676
9677 044724 000414 BR 10$ ;JUMP OUT
9678
9679 ;*NOW THE DISK WILL BE CHECKED
9680 ;*NO DATA SHOULD BE WRITTEN
9681 ;*REINTO BUFFER HAS BEEN FILLED WITH EXPECTED DATA
9682 ;*DISK HAS BEEN FILLED WITH 177400
9683 ;*WRFROM HAS BEEN FILLED WITH 125252
9684
9685 044726 004037 043626 3$: JSR RO,@#COMPAR ;CHECK
9686 044732 003160 REINTO ;GOOD DATA BUFFER
9687 044734 051534 DISK ;TEST BUFFER
9688 044736 000400 256.
9689 044740 044746 4$ ;RETURN POINT FOR ERROR HEADER
9690 044742 044752 5$ ;RETURN POINT FOR ERROR DATA
9691 044744 044756 10$ ;RETURN POINT FOR GOOD COMPARISON
9692 044746 104004 4$: ERROR 4 ;READ NEXT ERROR 5
9693 044750 000207 RTS PC ;RETURN TO COMPARISON SUBROUTINE
9694 044752 104005 5$: ERROR 5 ;WORD NO ARE ALL DATA
9695 ;WORDS THE SHOULD NOT
9696 ;HAVE BEEN CHANGED BY THE
9697 ;WRITE COMMAND
9698 044754 000207 RTS PC ;RETURN TO COMPARISON SUBROUTINE
9699 044756 005720 10$: TST (R0)+ ;IS THIS A HCRC ON HCE CHECK?
9700 044760 001442 BEQ 6$ ;BRANCH IF HCRC
9701 044762 022737 000072 001210 CMP #72,@#$TMP5 ;IS THIS A READ COMMAND
9702 044770 001417 BEQ 11$ ;BRANCH IF YES
9703 044772 017737 134644 001126 MOV @RHER1,@#$BDDAT ;TEST DATA
9704 045000 022737 000200 001126 CMP #HCE,@#$BDDAT ;ONLY HEADER COMPARE BIT?
9705 ;SHOULD BE SET
9706 045006 001470 BEQ 7$ ;BRANCH IF GOOD
9707 045010 013737 001642 042402 MOV @RHER1,@#REGADR ;REGISTER ADDRESS RHER1

```

```

9708 045016 012737 000200 001124      MOV    #HCE,@#SGDDAT ;GOOD DATA
9709 045024 104027                    ERROR  27             ;AFTER AN ERROR ON THE
9710                                     ;HEADER ONLY HCE SHOULD
9711 045026 000460                    BR     7$             ;BE SET
9712 045030                                     11$:
9713 045030 017737 134606 001126      MOV    @RHER1,@#SBDDAT ;TEST DATA
9714 045036 022737 100200 001126      CMP    #DCK!HCE,@#SBDDAT ;ONLY HEADER COMPARE BIT?
9715                                     ;SHOULD BE SET
9716                                     ;DCK IS SET BECAUSE ECC IS NOT READ
9717 045044 001451                    BEQ    7$             ;BRANCH IF GOOD
9718 045046 013737 001642 042402      MOV    @#RHER1,@#REGADR ;REGISTER ADDRESS RHER1
9719 045054 012737 100200 001124      MOV    #DCK!HCE,@#SGDDAT ;GOOD DATA
9720 045062 104027                    ERROR  27             ;AFTER AN ERROR ON THE
9721                                     ;HEADER ONLY HCE SHOULD
9722 045064 000441                    BR     7$             ;BE SET
9723 045066 022737 000072 001210 6$:    CMP    #72,@#STMP5    ;IS THIS A READ COMMAND?
9724 045074 001417                    BEQ    12$            ;BRANCH IF A READ
9725 045076 017737 134540 001126      MOV    @RHER1,@#SBDDAT ;TEST DATA
9726 045104 022737 000400 001126      CMP    #HCRC,@#SBDDAT ;ONLY CRC ERROR SHOULD BE THERE
9727 045112 001426                    BEQ    7$
9728 045114 013737 001642 042402      MOV    @#RHER1,@#REGADR ;REG. ADDR = RHER1
9729 045122 012737 000400 001124      MOV    #HCRC,@#SGDDAT ;GOOD DATA
9730 045130 104027                    ERROR  27             ;AFTER A CRC ERROR ONLY CRC
9731                                     ;SHOULD BE SET
9732 045132 000416                    BR     7$             ;BRANCH OUT
9733 045134 017737 134502 001126 12$:  MOV    @RHER1,@#SBDDAT ;TEST DATA
9734
9735 045142 022737 100400 001126      CMP    #DCK!HCRC,@#SBDDAT;HCRC AND DCK SHOULD BE SET
9736                                     ;DCK IS SET BECAUSE ECC IS NOT READ
9737 045150 001407                    BEQ    7$             ;BRANCH IF GOOD
9738 045152 012737 100400 001124      MOV    #DCK!HCRC,@#SGDDAT;GOOD DATA
9739 045160 013737 001642 042402      MOV    @#RHER1,@#REGADR;FAILING REGISTER RHER1
9740 045166 104027                    ERROR  27             ;AFTER A CRC ERROR ON A READ
9741                                     ;DCK AND HCRC SHOULD BE SET
9742                                     ;DCK IS SET BECAUSE ECC IS NOT READ
9743 045170 000200                    7$:   RTS     R0           ;RETURN TO MAIN TEST
9744
9745
9746
9747                                     ;*THIS IS A SUBROUTINE TO LEAVE AT THE MIDDLE OF
9748                                     ;*A WRITE HEADER AND DATA COMMAND
9749                                     ;*IT TRYS TO GET SECTOR 10, TRACK 0, CYLINDER 0
9750                                     ;*BUT COMES OUT AFTER ONE SECTOR
9751                                     ;*THE COMMAND OS JSR PC,@#MIDDLE
9752                                     ;*BAI IS SET
9753
9754                                     MIDDLE:
9755 045172 010046      MOV    R0,-(SP)      ;;PUSH R0 ON STACK
9756 045174 010146      MOV    R1,-(SP)      ;;PUSH R1 ON STACK
9757 045176 013777 002066 134434      MOV    @#WRIFOR,@RHCS1 ;WRITE HEADER AND DATA=62
9758                                     ;IN RHCS1
9759 045204 012777 177766 134420      MOV    #-10,@RHWC    ;10 WORDS
9760 045212 012777 002114 134414      MOV    #WRFROM,@RHBA ;BUS ADDRESS=WRFROM
9761 045220 012777 000010 134416      MOV    #10,@RHDST    ;DESIRED TRACK=0 SECTOR=10
9762 045226 052777 000010 134402      BIS    #BAI,@RHCS2   ;BUS ADDRESS INCREMENT INHIBIT
9763 045234 012777 010000 134406      MOV    #FMT22,@RHOF ;FORMAT 16 BIT WORDS
  
```

```

9764 045242 005077 134404          CLR    @RHCA          ;CYLINDER=0
9765 045246 012737 000001 045274  MOV    #1,@#MID      ;SECTOR IS SET TO 1 SO THAT
9766                                     ;WE CAN GET OUT AT THE
9767                                     ;MIDDLE OF AN OPERATION
9768                                     ;LOOKING FOR SECTOR 10
9769 045254 012777 000001 134376  MOV    #DMD,@RHMR    ;SET DIAGNOSTIC MODE
9770 045262 052777 000001 134350  BIS    #GO,@RHCS1    ;GO TO RHCS1 WITH 62
9771 045270 004137 053664          JSR    R1,@#SEARCH
9772 045274 000000          MID:   .WORD    0          ;SECTOR
9773 045276 012601          MOV    (SP)+,R1      ;:POP STACK INTO R1
9774 045300 012600          MOV    (SP)+,R0      ;:POP STACK INTO R0
9775 045302 000207          RTS    PC
  
```

.SBTTL JAM CURRENT CYLINDER ROUTINE

```

;*THIS SUBROUTINE WILL CHANGE THE CURRENT CYLINDER REGISTER
;*THIS IS DONE BY GIVING A SEEK COMMAND THEN AN INIT
;*WHICH WILL LOAD THE CURRENT CYLINDER WITH THE DESIRED CYLINDER VALUE
;*
;*CALL IS
;*JSR RO,@#MAKECYL
;*XC          ;DESIRED VALUE OF CURRENT CYLINDER
  
```

MAKECYL:

```

9791 045304          MOV    R5,-(SP)      ;:PUSH R5 ON STACK
9792 045304 010546          MOV    R0,@#PCJSR   ;PC OF JSR+4
9793 045306 010037 002014          SUB    #4,@#PCJSR   ;SAVE PC OF JSR
9794 045312 162737 000004 002014  MOV    (R0)+,R5      ;GETTING READY TO FILL DESIRED CYLINDER
9795 045320 012005          MOV    R5,@RHCA     ;FILL DESIRED CYLINDER REGISTER
9796 045322 010577 134324          CLR    @RHDST       ;MAKE SURE DESIRED SECTOR TRACK IS NOT ILLEGAL
9797 045326 005077 134312          MOV    @#SEECOM,@RHCS1 ;FILL SEEK COMMAND
9798 045332 013777 002074 134300  MOV    #DMD,@RHMR    ;SET DIAGNOSTIC MODE
9799 045340 012777 000001 134312  BIS    #GO,@RHCS1    ;GO TO SEEK
9800 045346 052777 000001 134264  NOP                                     ;ALLOW TIME FOR SEEK TO HANG UP
9801 045354 000240          NOP                                     ;ALLOW TIME FOR SEEK TO HANG UP
9802 045356 000240          NOP                                     ;ALLOW TIME FOR SEEK TO HANG UP
9803 045360 000240          NOP                                     ;ALLOW TIME FOR SEEK TO HANG UP
9804 045362 000240          JSR    PC,@#CLDISK  ;GIVE INIT
9805 045364 004737 042732          MOV    @RHCC,@#SBDDAT ;TEST DATA
9806 045370 017737 134302 001126  CMP    R5,@#SBDDAT   ;COMPARE CURRENT CYLINDER
9807 045376 020537 001126          BEQ    1$           ;BRANCH IF GOOD
9808 045402 001406          MOV    R5,@#SGDDAT  ;GOOD VALUE OF RHCC
9809 045404 010537 001124          MOV    @#RHCC,@#REGADR ;FAILING REGISTER ADDRESS
9810 045410 013737 001676 042402  ERROR  30           ;CURRENT CYLINDER DOES NOT MATCH DESIRED CYLINDER
9811 045416 104030          ;REGISTER AFTER A SEEK AND AN INIT
9812
9813 045420          1$:
9814 045420 012605          MOV    (SP)+,R5      ;:POP STACK INTO R5
9815 045422 000200          RTS    R0
  
```

.SBTTL ECC GENERATION AND COMPARISON ROUTINE

9816
9817
9818
9819

9820
9821
9822
9823
9824
9825
9826
9827
9828
9829
9830
9831
9832
9833
9834
9835
9836
9837
9838
9839
9840
9841
9842
9843
9844
9845
9846
9847
9848
9849
9850
9851
9852
9853
9854
9855
9856
9857
9858
9859
9860
9861
9862
9863
9864
9865
9866
9867
9868
9869
9870
9871
9872
9873
9874
9875

;*THIS SUBROUTINE GENERATES AND TESTS ECC
;*CALL JSR PC,ECTEST

100000
040000
020000
010000
004000
002000
001000
000400
000200
000100
000040
000020
000010
000004
000002
000001
100000
040000
020000
010000
004000
002000
001000
000400
000200
000100
000040
000020
000010
000004
000002
000001

PIE1 =100000
PIE2 =40000
PIE3 =20000
PIE4 =10000
PIE5 =4000
PIE6 =2000
PIE7 =1000
PIE8 =400
PIE9 =200
PIE10 =100
PIE11 =40
PIE12 =20
PIE13 =10
PIE14 =4
PIE15 =2
PIE16 =1
PIE17 =100000
PIE18 =40000
PIE19 =20000
PIE20 =10000
PIE21 =4000
PIE22 =2000
PIE23 =1000
PIE24 =400
PIE25 =200
PIE26 =100
PIE27 =40
PIE28 =20
PIE29 =10
PIE30 =4
PIE31 =2
PIE32 =1

045424 000000
045426 000000
045430 000000
045432 000000

ECDATA: 0
GECC1: 0
GECC2: 0
TSECCG: 0

;DATA BIT FOR ECC
;IF ALL ONES THEN CURRENT BIT IS A ONE
;IF ZERO THEN CURRENT BIT IS A ZERO
;LOW ORDER ECC WORD TO BE GENERATED HERE
;=R1
;HIGH ORDER ECC WORD TO BE GENERATED HERE
;=R2
;IF =177777 GENERATE AND TEST ECC FOR THIS BIT
;IF =0 DO NOT GENERATE AND TEST ECC FOR THIS BIT

9876 045434 113713
9877 045436 000000
9878 045440 000000
9879 045442 010041

NCODE: 38859.
NCOUNT: 0
POSIT1: 0
HARDER: 4129.

:N-CODE WORD
:TEMPORARY N CODE
:POSITION REGISTER
:HARD ERROR COUNT
:TRUE COUNT IS 4128 BUT AS COMPARES ARE
:DONE ONE STAGE LATER SO 4129
:DATA ENVELOPE FOR TYPE OUT
:MAX FOR WRITE IS 4096
:MAX FOR READ IS 4128
:LEADING ZEROS ENVELOPE FOR TYPE OUT
:THIS IS SHUT OFF WHEN POSITION COUNTER
:IN ENABLED
:MAX COUNT IS 38859

9880
9881
9882 045444 000000
9883

DATENV: 0

9884
9885 045446 000000
9886

ZCODE: 0

9887
9888
9889

9890
9891

9892 045450 000000
9893 045452 000000
9894 045454 000000

HADTMP: 0

:TEMPORARY HARD ERROR COUNT

9895 045456 000000
9896 045460 000000
9897

P3: 0

P12: 0

P22: 0

P24: 0

9898
9899
9900

9901
9902 045462
9903 045462 010046

ECTEST:

MOV R0,-(SP) ;;PUSH R0 ON STACK
MOV R1,-(SP) ;;PUSH R1 ON STACK
MOV R2,-(SP) ;;PUSH R2 ON STACK
MOV R3,-(SP) ;;PUSH R3 ON STACK
MOV R4,-(SP) ;;PUSH R4 ON STACK
MOV R5,-(SP) ;;PUSH R5 ON STACK

9904 045464 010146
9905 045466 010246
9906 045470 010346

9907 045472 010446
9908 045474 010546

MOV @#GECC1,R1 ;ECC1 WORD
MOV @#GECC2,R2 ;ECC2 WORD

9909 045476 013701 045426
9910 045502 013702 045430
9911 045506 005737 045424

TST @#ECDATA ;IS CURRENT BIT A ONE
BEQ 2\$;BRANCH IF CURRENT DATA D=0

9912 045512 001406
9913
9914

;*IF CARRY IS NOT ZERO THEN D=1
;*INVERT X32 TO GIVE R0

9915
9916
9917 045514 010103
9918 045516 052703 177776

1\$: MOV R1,R3
BIS #^CPIE32,R3
COM R3
MOV R3,R0
BR 3\$

9919 045522 005103
9920 045524 010300
9921 045526 000404

9922
9923
9924

;*IF CARRY IS ZERO THEN D=0
;*X32 BECOMES R0

9925 045530 010103
9926 045532 042703 177776
9927 045536 010300

2\$: MOV R1,R3
BIC #^CPIE32,R3
MOV R3,R0

9928
9929 045540 000241
9930 045542 006000
9931 045544 006000

3\$: CLC
ROR R0
ROR R0

```
9932 045546 005700          TST      R0
9933 045550 001462          BEQ      10$          ;BRANCH IF R0=0
9934                                ;*INVERT X2
9935
9936 045552 010203          MOV      R2,R3
9937 045554 052703 137777    BIS      #^CPIE2,R3
9938 045560 005103          COM      R3
9939 045562 010337 045452    MOV      R3,@#P3
9940 045566 006237 045452    ASR      @#P3
9941
9942                                ;*INVERT X11
9943
9944
9945 045572 010203          MOV      R2,R3
9946 045574 052703 177737    BIS      #^CPIE11,R3
9947 045600 005103          COM      R3
9948 045602 010337 045454    MOV      R3,@#P12
9949 045606 006237 045454    ASR      @#P12
9950
9951                                ;*INVERT X21
9952
9953 045612 010103          MOV      R1,R3
9954 045614 052703 173777    BIS      #^CPIE21,R3
9955 045620 005103          COM      R3
9956 045622 010337 045456    MOV      R3,@#P22
9957 045626 006237 045456    ASR      @#P22
9958
9959                                ;*INVERT X23
9960
9961 045632 010103          MOV      R1,R3
9962 045634 052703 176777    BIS      #^CPIE23,R3
9963 045640 005103          COM      R3
9964 045642 010337 045460    MOV      R3,@#P24
9965 045646 006237 045460    ASR      @#P24
9966
9967                                ;*NOW THAT R0 FOR POSITION 1
9968                                ;*      P3 FOR POSITION 3
9969                                ;*      P12 FOR POSITION 12
9970                                ;*      P22 FOR POSITION 22
9971                                ;*      P24 FOR POSITION 24
9972                                ;*ARE KNOWN THE ROTATE WILL BE DONE AND
9973                                ;*THESE BITS JAMED IN
9974
9975 045652 006002          ROR      R2
9976 045654 006001          ROR      R1
9977 045656 053700 045452    BIS      @#P3,R0
9978 045662 053700 045454    BIS      @#P12,R0
9979 045666 042702 120020    BIC      #PIE1!PIE3!PIE12,R2
9980 045672 050002          BIS      R0,R2
9981
9982 045674 005000          CLR      R0
9983 045676 053700 045456    BIS      @#P22,R0
9984 045702 053700 045460    BIS      @#P24,R0
9985 045706 042701 002400    BIC      #PIE22!PIE24,R1
9986 045712 050001          BIS      R0,R1
9987 045714 000404          BR      12$
```

```
9988
9989
9990          ;*THE PROGRAM COMES HERE IF R0=0
9991          ;*SO AFTER ROTATE R0 GETS PUT INTO POSITION 1
9992 045716 006002      10$: ROR    R2
9993 045720 006001      ROR    R1
9994 045722 042702 100000  BIC    #PIE1,R2
9995 045726 010137 045426 12$: MOV    R1,@#GECC1      ;SAVE ECC1
9996 045732 010237 045430  MOV    R2,@#GECC2      ;SAVE ECC2
9997 045736 005737 045432  TST    @#TSECCG      ;IS HARDWARE TO BE CHECKED
9998                                     ;IF =1777777 TEST HARDWARE
9999                                     ;IF = 0 DO NOT TEST HARDWARE
10000 045742 001432      BEQ    14$      ;BRANCH IF HARDWARE NOT TO BE CHECKED
10001
10002
10003          ;*CHECK HARDWARE
10004 045744 032777 000400 133166  BIT    #SW8,@SWR      ;IS SWITCH 8 SET
10005 045752 001005      BNE    15$      ;BRANCH IF SW8 IS SET
10006 045754 032777 000100 133156  BIT    #SW6,@SWR      ;IS SWITCH 6 SET
10007 045762 001401      BEQ    15$      ;BRANCH IF SW6 IS NOT SET
10008 045764 000421      BR     14$      ;IF SWITCH 8 IS NOT SET AND
10009                                     ;SWITCH 6 IS SET THEN
10010                                     ;DO NOT DO COMPARES
10011 045766 010146      15$: MOV    R1,-(SP)      ;GOOD PATTERN REGISTER
10012 045770 042716 174000  BIC    #174000,(SP)   ;GET ONLY PATTERN BITS
10013 045774 022677 133672  CMP    (SP)+,@RHEC2   ;COMPARE PATTERN REGISTER
10014 046000 001404      BEQ    13$      ;BRANCH IF GOOD
10015          ;*TO SAVE TIME
10016 046002 004737 042336  JSR    PC,@#PUTREG    ;SAVE REGISTERS
10017 046006 104035      ERROR 35      ;PATTERN REGISTER IN 11 BITS IN ERROR
10018 046010 000407      BR     14$      ;BRANCH OUT
10019 046012 023777 045440 133650 13$: CMP    @#POSIT1,@RHEC1 ;COMPARE POSITION REGISTER
10020 046020 001403      BEQ    14$      ;BRANCH IF GOOD
10021          ;*TO SAVE TIME
10022 046022 004737 042336  JSR    PC,@#PUTREG    ;SAVE REGISTERS
10023 046026 104035      ERROR 35      ;POSITION REGISTER IN ERROR
10024                                     ;"DATA ENVLOP" GIVES NUMBER OF CLOCK
10025                                     ;PULSES FROM BEGINING OF COMMAND
10026                                     ;THAT IS THE CLOCKS IN THE R/W DATA FIELD ENVELOPE
10027
10028                                     ;
10029                                     ;!N A WRITE THERE ARE 10000 OCTAL CLOCKS
10030                                     ;IN A READ THERE ARE 10040 OCTAL CLOCKS
10031
10032                                     ;
10033                                     ;"N-CODE ZEROS" GIVE THE NUMBER OF CLOCKS
10034                                     ;GIVEN FOR THE LEADING ZEROS FIELD
10035                                     ;MAX COUNT IS 113713 OCTAL
10036
10037                                     ;
10038                                     ;"GOOD POSITION" GIVES NUMBER OF CLOCKS
10039                                     ;GIVEN AFTER LEADING ZEROS WHICH IS FOR THE DATA
10040                                     ;FIELD
10041                                     ;MAX COUNT IS 10040 OR 10041 OCTAL
10042
10043 046030 012605      14$: MOV    (SP)+,R5      ;;POP STACK INTO R5
```

CZRJGDO,RP04/5/6 DSKLS CTRLR1
CZRJGD.P11 10-SEP-79 11:00

MACY11 30A(1052) 10-SEP-79 11:11 H 2 PAGE 228
ECC GENERATION AND COMPARISON ROUTINE

SEQ 0227

| | | | | | |
|-------|--------|--------|-----|----------|---------------------|
| 10044 | 046032 | 012604 | MOV | (SP)+,R4 | ::POP STACK INTO R4 |
| 10045 | 046034 | 012603 | MOV | (SP)+,R3 | ::POP STACK INTO R3 |
| 10046 | 046036 | 012602 | MOV | (SP)+,R2 | ::POP STACK INTO R2 |
| 10047 | 046040 | 012601 | MOV | (SP)+,R1 | ::POP STACK INTO R1 |
| 10048 | 046042 | 012600 | MOV | (SP)+,R0 | ::POP STACK INTO R0 |
| 10049 | 046044 | 000207 | RTS | PC | |

```
10050
10051
10052          ;*THIS SUBROUTINE WILL CONTROL THE ECC GENERATION ROUTINE
10053          ;*FOR ERROR CORRECTION PROCESS
10054          ;*CALL JSR, PC,@#ECORR
10055          ;* XP ;EXPECTED POSITION REGISTER WHEN CORRECTION IS COMPLETE
10056
10057
10058
10059 046046 000000          ERPOS: 0          ;POSITION REG. WHEN CORRECTION IS COMPLETE
10060
10061
10062
10063 046050 010037 002014          ECORR: MOV R0,@#PCJSR          ;SAVE PC OF JSR + 4
10064 046054 162737 000004 002014          SUB #4,@#PCJSR          ;SAVE PC OF JSR
10065 046062 012037 046046          MOV (R0)+,@#ERPOS          ;GET POSITION REG. WHEN CORRECTION IS COMPLETE
10066 046066 010146          MOV R1,-(SP)          ;PUSH R1 ON STACK
10067 046070 013701 001660          MOV @#RHMR,R1          ;MAINTENANCE REGISTER
10068 046074 012711 000001          MOV #DMD,@R1          ;SET DIAGNOSTIC MODE BIT
10069 046100 005037 045424          CLR @#ECDATA          ;ECC DATA IS ZERO
10070
10071
10072
10073 046104 005737 045440          1$: TST @#POSITI          ;IS SOFTWARE POSITION NON ZERO
10074 046110 001007          BNE 2$          ;BRANCH IF N-CODE S COMPLETE
10075 046112 005337 045436          DEC @#NCOUNT          ;DECREMENT N-CODE
10076 046116 001001          BNE 6$          ;BRANCH IF N-CODE IS NOT COMPLETE
10077 046120 000403          BR 2$          ;BRANCH AS N-CODE IS COMPLETE
10078 046122 005237 045446          6$: INC @#ZCODE          ;INCREMENT CLOCKS GIVEN FOR LEADING ZEROS
10079 046126 000420          BR 3$          ;BRANCH AS N-CODE IS NOT COMPLETE
10080          ;GO TO GIVE CLOCK AND TEST ECC
10081 046130 005237 045440          2$: INC @#POSITI          ;INCREMENT SOFTWARE POSITION
10082 046134 023737 046046 045440          CMP @#ERPOS,@#POSITI          ;HAVE ENOUGH CLOCKS BEEN GIVEN TO DETECT ERROR
10083 046142 103012          BHIS 3$          ;BRANCH IF MORE CLOCKS TO BE GIVEN
10084 046144 023737 045450 045440          CMP @#HADTMP,@#POSITI          ;HAVE ENOUGH CLOCKS BEEN GIVEN FOR HARD ERROR
10085          ;THAT IS HAVE 4128 MORE CLOCKS BEEN GIVEN
10086 046152 001415          BEQ 5$          ;BRANCH IF YES
10087 046154 032711 000400          BIT #ZER,@R1          ;CHECK ZERO DETECT BIT IN RHMR
10088 046160 001016          BNE 4$          ;BRANCH IS ZER SET
10089          ;*TO SAVE TIME
10090 046162 004737 042336          JSR PC,@#PUTREG          ;SAVE REGISTERS
10091 046166 104034          ERROR 34          ;ZERO DETECT BIT NOT HIGH
10092          ;WHEN 21 BITS IN ECC 32 BIT REGISTER IS 0
10093
10094
10095 046170 052711 000002          3$: BIS #MCLK,@R1          ;SET CLOCK
10096 046174 042711 000002          BIC #MCLK,@R1          ;CLEAR CLOCK
10097 046200 004737 045462          JSR PC,@#ECTEST          ;GO TO GENERATE AND TEST ECC
10098 046204 000737          BR 1$          ;CONTINUE
10099
10100          ;*THIS EXTRA CLOCK IS TO BRING ECH HIGH
10101          ;*AFTER THIS CLOCK POSITION REGISTER MAY BE 10040 OR 10041 OCTAL
10102
10103 046206 052711 000002          5$: BIS #MCLK,@R1          ;SET CLOCK
10104 046212 042711 000002          BIC #MCLK,@R1          ;CLEAR CLOCK
10105
```

10106 046216
10107 046216 012601
10108 046220 000200

4\$:
MOV (SP)+,R1 ;;POP STACK INTO R1
RTS R0

10109
10110
10111
10112
10113
10114
10115
10116
10117
10118
10119

;*THIS SUBROUTINE GENERATES THE ECC FOR WHAT IS ON DISK AND INSERTS THEM
;*ON LOCATIONS "DISK+1000" AND "DISK+1002"

10120 046222

FILLEC:

10121 046222 010046
10122 046224 010146
10123 046226 010246
10124 046230 010346
10125 046232 010446
10126 046234 010546
10127 046236 005037 045440
10128 046242 005037 045426
10129 046246 005037 045430
10130 046252 012701 051534
10131 046256 012702 000400
10132 046262 012703 000020
10133 046266 012104
10134 046270 006004
10135 046272 103004
10136 046274 012737 177777 045424
10137 046302 000402
10138 046304 005037 045424
10139 046310 004737 045462
10140 046314 005303
10141 046316 001364
10142 046320 005302
10143 046322 001357
10144 046324 013737 045426 052534
10145 046332 013737 045430 052536
10146 046340 012605
10147 046342 012604
10148 046344 012603
10149 046346 012602
10150 046350 012601
10151 046352 012600
10152 046354 000207

MOV R0,-(SP) ;;PUSH R0 ON STACK
MOV R1,-(SP) ;;PUSH R1 ON STACK
MOV R2,-(SP) ;;PUSH R2 ON STACK
MOV R3,-(SP) ;;PUSH R3 ON STACK
MOV R4,-(SP) ;;PUSH R4 ON STACK
MOV R5,-(SP) ;;PUSH R5 ON STACK
CLR @#POSITI ;CLEAR POSITION
CLR @#GECC1 ;CLEAR GECC1
CLR @#GECC2 ;CLEAR
MOV #DISK,R1 ;POINTER TO DATA FOR ECC GENERATION
MOV #256.,R2 ;COUNTER FOR NUMBER OF DATA WORDS
9\$: MOV #16.,R3 ;COUNTER FOR NUMBER OF BITS PER WORD
MOV (R1)+,R4 ;DATA IN R4
10\$: ROR R4 ;GET ONE DATA BIT IN CARRY
BCC 11\$;BRANCH IF DATA BIT IS ZERO
MOV #-1,@#ECDATA ;ECC DATA BIT IS A ONE
BR 12\$;BRANCH TO GENERATE ECC
11\$: CLR @#ECDATA ;ECC DATA BIT IS A ZERO
12\$: JSR PC,@#ECTEST ;GO TO GENERATE ECC
DEC R3 ;DECREMENT BIT COUNT
BNE 10\$;BRANCH IF 16 BITS NOT DONE
DEC R2 ;DECREMENT WORD COUNT
BNE 9\$;BRANCH IF 256 WORDS NOT DONE
MOV @#GECC1,@#DISK+<256.*2>;INSERT ECC1 ON DISK
MOV @#GECC2,@#DISK+<257.*2>;INSERT ECC2 ON DISK
MOV (SP)+,R5 ;;POP STACK INTO R5
MOV (SP)+,R4 ;;POP STACK INTO R4
MOV (SP)+,R3 ;;POP STACK INTO R3
MOV (SP)+,R2 ;;POP STACK INTO R2
MOV (SP)+,R1 ;;POP STACK INTO R1
MOV (SP)+,R0 ;;POP STACK INTO R0
RTS PC

10153
10154
10155
10156
10157
10158
10159
10160
10161

.SBTTL RH BASE ADDRESS CHANGE ROUTINE

;; THIS ROUTINE WILL ALLOW THE CHANGE OF THE BASE
;; ADDRESS FROM 176700 TO ANY TYPED VALUE

| | | | | | |
|-------|--------|--------|--------|--------|--|
| 10162 | | | | | |
| 10163 | 046356 | | | | |
| 10164 | 046356 | 104401 | 046364 | | |
| 10165 | 046362 | 000425 | | | |
| 10166 | | | | | |
| 10167 | 046436 | | | | |
| 10168 | 046436 | 013746 | 001640 | | |
| 10169 | 046442 | 104402 | | | |
| 10170 | 046444 | 104401 | 046452 | | |
| 10171 | 046450 | 000425 | | | |
| 10172 | | | | | |
| 10173 | 046524 | | | | |
| 10174 | 046524 | 004737 | 055310 | | |
| 10175 | 046530 | 104412 | | | |
| 10176 | 046532 | 012700 | 001630 | | |
| 10177 | 046536 | 012701 | 000026 | | |
| 10178 | 046542 | 012737 | 047342 | 000004 | |
| 10179 | 046550 | 021637 | 001640 | | |
| 10180 | 046554 | 001407 | | | |
| 10181 | 046556 | 005776 | 000000 | | |
| 10182 | 046562 | 163716 | 001640 | | |
| 10183 | 046566 | 061620 | | | |
| 10184 | 046570 | 005301 | | | |
| 10185 | 046572 | 001375 | | | |
| 10186 | 046574 | | | | |
| 10187 | 046574 | 104401 | 046602 | | |
| 10188 | 046600 | 000417 | | | |
| 10189 | | | | | |
| 10190 | 046640 | | | | |
| 10191 | 046640 | 013746 | 001626 | | |
| 10192 | 046644 | 104402 | | | |
| 10193 | 046646 | 104401 | 046654 | | |
| 10194 | 046652 | 000437 | | | |
| 10195 | | | | | |
| 10196 | 046752 | | | | |
| 10197 | 046752 | 104412 | | | |
| 10198 | 046754 | 012637 | 001626 | | |
| 10199 | 046760 | 104401 | 046766 | | |
| 10200 | 046764 | 000416 | | | |
| 10201 | | | | | |
| 10202 | 047022 | | | | |
| 10203 | 047022 | 013746 | 001640 | | |
| 10204 | 047026 | 104402 | | | |
| 10205 | 047030 | 104401 | 047036 | | |
| 10206 | 047034 | 000416 | | | |
| 10207 | | | | | |
| 10208 | 047072 | | | | |
| 10209 | 047072 | 013746 | 001626 | | |
| 10210 | 047076 | 104402 | | | |
| 10211 | 047100 | 104401 | 047106 | | |
| 10212 | 047104 | 000417 | | | |
| 10213 | | | | | |
| 10214 | 047144 | | | | |
| 10215 | 047144 | 104401 | 047152 | | |
| 10216 | 047150 | 000402 | | | |
| 10217 | | | | | |

```
BASECH:
TYPE ,65$           ;;TYPE ASCIZ STRING
BR 64$            ;;GET OVER THE ASCIZ
;;65$: .ASCIZ <15><12>/PRESENT BASE ADDRESS OF REGISTERS IS /
64$:
MOV @#RHCS1,-(SP) ;GET READY TO TYPE OLD BASE
TYPOC
TYPE ,67$         ;;TYPE ASCIZ STRING
BR 66$           ;;GET OVER THE ASCIZ
;;67$: .ASCIZ <15><12>/TYPE NEW BASE ADDRESS FOLLOWED BY 'CR' /
66$:
JSR PC,@#$TKINT  ;INITIALIZE THE TTY KEYBOARD
RDOCT
MOV #RHDB,R0     ;GET STARTING ADDRESS OF REGISTERS
MOV #22.,R1      ;NUMBER OF REGISTERS
MOV #ADTIMO,@#4  ;SET UP TO CHECK NEW ADDRESS
CMP @SP,@#RHCS1 ;NEW CSR?
BEQ 1$          ;NO-SKIP NEXT
TST @0(SP)       ;ACCESS THE NEW ADDRESS
SUB @#RHCS1,@SP  ;GET THE ADDRESS OFFSET
2$: ADD @SP,(R0)+ ;AND PLUG IT IN
DEC R1           ;DONE ALL OF THEM YET?
BNE 2$          ;NOT YET, SO DO MORE
1$:
TYPE ,69$         ;;TYPE ASCIZ STRING
BR 68$           ;;GET OVER THE ASCIZ
;;69$: .ASCIZ <15><12>/PRESENT VECTOR ADDRESS IS /
68$:
MOV @#RPVEC,-(SP) ;GET READY TO TYPE OLD VECTOR ADDRESS
TYPOC
TYPE ,71$         ;;TYPE ASCIZ STRING
BR 70$           ;;GET OVER THE ASCIZ
;;71$: .ASCIZ <15><12>/TYPE NEW VECTOR ADDRESS OR RETYPE OLD ONE FOLLOWED BY "CR" /
70$:
RDOCT
MOV (SP)+,@#RPVEC ;SETUP VECTOR ADDRESS
TYPE ,73$         ;;TYPE ASCIZ STRING
BR 72$           ;;GET OVER THE ASCIZ
;;73$: .ASCIZ <15><12>/NEW BASE WILL REMAIN - /
72$:
MOV @#RHCS1,-(SP)
TYPOC
TYPE ,75$         ;;TYPE ASCIZ STRING
BR 74$           ;;GET OVER THE ASCIZ
;;75$: .ASCIZ <15><12>/NEW VECTOR WILL REMAIN - /
74$:
MOV @#RPVEC,-(SP)
TYPOC
TYPE ,77$         ;;TYPE ASCIZ STRING
BR 76$           ;;GET OVER THE ASCIZ
;;77$: .ASCIZ <15><12>/UNTIL PROGRAM IS RELOADED./
76$:
TYPE ,79$         ;;TYPE ASCIZ STRING
BR 78$           ;;GET OVER THE ASCIZ
;;79$: .ASCIZ <15><12>/ /
```


10218 047156
10219 047156 104401 047164
10220 047162 000424
10221
10222 047234
10223 047234 104401 047242
10224 047240 000426
10225
10226 047316
10227 047316 012746 000200
10228 047322 104402
10229 047324 104401 047332
10230 047330 000402
10231
10232 047336
10233 047336 000137 004244
10234 047342
10235 047342 104401 047350
10236 047346 000424
10237
10238 047420
10239 047420 022626
10240 047422 000137 046356
10241
10242
10243
10244
10245
10246
10247
10248
10249
10250 047426 000000
10251 047430 004737 042732
10252 047434 013712 047426
10253 047440 005714
10254 047442 032712 010000
10255 047446 001401
10256 047450 000773
10257 047452 000772

78\$:
TYPE ,81\$;:TYPE ASCIZ STRING
BR 80\$;:GET OVER THE ASCIZ
;:81\$: .ASCIZ <15><12>/UNLESS HALTED AND MANUALLY RESTARTED,/
80\$:
TYPE ,83\$;:TYPE ASCIZ STRING
BR 82\$;:GET OVER THE ASCIZ
;:83\$: .ASCIZ <15><12>/PROGRAM WILL AUTOMATICALLY RESTART FROM /
82\$:
MOV #RA,-(SP)
TYPOC
TYPE ,85\$;:TYPE ASCIZ STRING
BR 84\$;:GET OVER THE ASCIZ
;:85\$: .ASCIZ <15><12>/ /
84\$:
JMP @#BEGIN ;OK, NOW START OVER WITH NEW ADDRESS
ADTIMO:
TYPE ,65\$;:TYPE ASCIZ STRING
BR 64\$;:GET OVER THE ASCIZ
;:65\$: .ASCIZ <15><12><377>/SPECIFIED ADDRESS DID NOT RESPOND. /
64\$:
CMP (SP)+,(SP)+ ;RESTORE THE STACK
JMP @#BASECH ;AND DO IT AGAIN.

;*THIS IS A LITTLE ROUTINE THAT TESTS NED BIT 11 IN RHCS2
;*THIS LOOPS HERE FOR EVER
;*TO BE USED ONLY IF DRIVES PRESENT LOOKING AT NED DOES NOT AGREE
;*WITH WHAT IS REALY THERE
ERUNIT: 0 ;UNIT UNDER MANUAL TEST
ERSTART:JSR PC,@#CLDISK ;SET GENERAL REG.
MOV @#ERUNIT,@R2 ;SELECT UNIT
1\$: TST @R4 ;TEST RHER1
BIT #NED,@R2 ;TEST NED
BEQ 2\$;BRANCH IF GOOD
BR 1\$;NED NOT SET
2\$: BR 1\$;NED SET

10258
10259
10260
10261
10262
10263
10264
10265
10266
10267
10268
10269
10270
10271
10272
10273
10274
10275
10276
10277
10278
10279
10280
10281
10282
10283
10284
10285
10286
10287
10288
10289
10290
10291
10292
10293
10294
10295
10296
10297
10298
10299
10300
10301
10302
10303
10304
10305
10306
10307
10308
10309
10310
10311
10312
10313

```
.SBTTL DISK SIMULATION
*****
*IN A WRITE HEADER AND DATA COMMAND FILL THE FOLLOWING
*WCLY=WITH CYLINDER TO BE ON DISK
*WSECTR=WITH SECTOR AND TRACK TO BE ON DISK
*WKEY1= WITH KEY1 TO BE ON DISK
*WKEY2= WITH KEY2 TO BE ON DISK
*FNWORD= NO OF DATA WORDS TO BE WRITTEN ON DISK
*THE COMMAND THEN IS JSR PC,COMWHD
*
*
*
*IN A WRITE DATA COMMAND FILL THE FOLLOWING
*CYL=WITH CYLINDER TO BE FOUND ON DISK
*SECOTR= WITH SECTOR AND TRACK TO BE FOUND ON DISK
*KEY1= WITH KEY1 TO BE FOUND ON DISK
*KEY2= WITH KEY2 TO BE FOUND ON DISK
*X= 1 MUST BE ONE
*NOWORD= WITH NUMBER OF DATA WORDS TO BE WRITTEN
*THE COMMAND THEN IS JSR PC,COMHD
*
*
*
*IN A READ HEADER AND DATA COMMAND FILL THE FOLLOWING
*CYL= WITH CYLINDER TO BE FOUND ON DISK
*SECOTR= WITH SECTOR AND TRACK TO BE FOUND ON DISK
*KEY1= WITH KEY1 TO BE FOUND ON DISK
*KEY2=WITH KEY2 TO BE FOUND ON DISK
*DAWORD= WITH NUMBER OF WORDS TO BE FOUND ON DISK
*;*X=0 MUST BE ZERO
*THE COMMAND THEN IS JSR PC,COMHD
*
*
*
*IN A READ DATA COMMAND FILL THE FOLLOWING
*CYL= WITH CYLINDER TO BE FOUND ON DISK
*SECOTR= WITH SECTOR AND TRACK TO BE FOUND ON DISK
*KEY1= WITH KEY1 TO BE FOUND ON DISK
*KEY2=WITH KEY2 TO BE FOUND ON DISK
*DAWORD= WITH NUMBER OF WORDS TO BE FOUND ON DISK
*;*X=0 MUST BE ZERO
*THE COMMAND THEN IS JSR PC,COMHD
*
```

10314
10315
10316
10317
10318
10319
10320
10321
10322
10323
10324
10325
10326
10327
10328
10329
10330
10331
10332
10333
10334
10335
10336
10337
10338
10339
10340
10341
10342
10343
10344
10345
10346

```

*****
:WRITE DATA COMMAND
:OR READ COMMAND I.E DATA ONLY OR HEADER AND DATA
*****

```

```

:**THIS SUBROUTINE IS THE FIRST IN A SERIES OF NESTED SUBROUTINES
:**IT ISSURE DIAGNOSTIC MODE, AN EXTRA DIAGNOSTIC INDEX, AND THE
:**'GO' BIT
:**IT THEN CALL THREE OTHER SUBROUTINES, WHICH IN TURN CALL OTHER
:**SUBROUTINES. THESE ARE:

```

```

:** SEARCH
:** RDHEAD
:** WRDATA
:** REDATA

```

```

10347 047454 000000          RUNCTR: .WORD 0
10348 047456 011637 002014  COMHD: MOV (SP),@#PCJSR ;SAVE PC OF JSR + 4
10349 047462 162737 000004 002014  SUB #4,@#PCJSR ;SAVE PC OF JSR
10350 047470 010046          MOV R0,-(SP) ;:PUSH R0 ON STACK
10351 047472 010146          MOV R1,-(SP) ;:PUSH R1 ON STACK
10352 047474 010246          MOV R2,-(SP) ;:PUSH R2 ON STACK
10353 047476 010346          MOV R3,-(SP) ;:PUSH R3 ON STACK
10354 047500 010446          MOV R4,-(SP) ;:PUSH R4 ON STACK
10355 047502 010546          MOV R5,-(SP) ;:PUSH R5 ON STACK
10356
10357 047504 012777 000001 132146  MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE
10358 047512 052777 000004 132140  BIS #MINX,@RHMR ;SET DIAGNOSTIC INDEX
10359 047520 042777 000004 132132  BIC #MINX,@RHMR ;CLEAR DIAGNOSTIC INDEX
10360 047526 052777 000001 132104  BIS #GO,@RHCS1 ;ISSUE 'GO' BIT & STALL 'TILL 'RUN'
10361                                     ;(FUNCTION CODE IS ISSUED BY THE TEST)
10362 047534 012737 000113 047454  RUNWAT: MOV #75.,@#RUNCTR ;LOAD STALL COUNT = APPROX. 450US
10363                                     ;FOR 11/50 CPU WITH CORE MEMORY
10364 047542 005337 047454 1$: DEC @#RUNCTR ;COUNT DOWN ONE
10365 047546 001375          BNE 1$ ;CONTINUE UNTIL = 0
10366
10367 047550 013746 047620          MOV SECOTR,-(SP) ;GET DESIRED SECTOR/TRACK
10368 047554 042716 177740          BIC #177740,(SP) ;MAKE ONLY SECTOR
10369 047560 012637 047570          MOV (SP)+,@#TRK ;SAVE SECTOR

```



```
10426                                     ;IF NOSYNC=0 THEN CONTINUE
10427 047656 00101i                       BNE   OUT   ;EXIT IF SET ----->
10428
10429 047660 004137 051204                JSR   R1,@#WRDATA ;WRITE DATA <----->
10430
10431 047664 000000                        NOWORD: .WORD 0   ;NO OF WORDS TO BE WRITTEN
10432 047666 000000                        Y:      .WORD 0   ;
10433 047670 000404                        BR     OUT   ;EXIT ----->
10434
10435 047672 004137 054140                DAREAD: JSR   R1,@#REDATA ;READ DATA <----->
10436 047676 000000                        DAWORD: .WORD 0   ;NO OF WORDS TO BE READ
10437 047700 000000                        .WORD 0
10438 047702
10439 047702 012605                        OUT:   MOV   (SP)+,R5 ;;POP STACK INTO R5
10440 047704 012604                        MOV   (SP)+,R4 ;;POP STACK INTO R4
10441 047706 012603                        MOV   (SP)+,R3 ;;POP STACK INTO R3
10442 047710 012602                        MOV   (SP)+,R2 ;;POP STACK INTO R2
10443 047712 012601                        MOV   (SP)+,R1 ;;POP STACK INTO R1
10444 047714 012600                        MOV   (SP)+,R0 ;;POP STACK INTO R0
10445 047716 000207                        RTS   PC      ;EXIT ROUTINE
```

10446
10447
10448
10449
10450
10451
10452
10453 047720 014400
10454 047722 000000
10455 047724 000000
10456 047726 000000
10457 047730 000000
10458
10459
10460
10461
10462
10463
10464
10465
10466
10467
10468
10469
10470
10471

```
*****  
;*THE DISK SECTOR IS DEVIDED AS FOLLOWS  
;*19 WORDS OF 0, ONE WORD 144000  
;*THESE MAKE 39 BYTES FOR SECTOR GAP AND ONE SYNC. BYTE  
  
RSYNC: 14400  
RCYL: 0  
RSETR: 0  
RKEY1: 0  
RKEY2: 0  
  
;*5 WORDS OF 0 ONE WORD 144000  
;*THESE MAKE 11 BYTES FOR HEADER GAP AND ONE SYNC. BYTE  
;*THESE ARE DCL GENERATED  
  
;*THERE ARE 256 WORDS OF DATA  
;*THERE ARE 2 WORDS FOR ECC GENERATED BY DCL  
;*15 WORDS OF 0 FOR DATA GAP AND TOLERANCE GAP  
*****
```

10472
10473
10474
10475
10476
10477
10478
10479
10480
10481
10482
10483
10484
10485
10486
10487
10488
10489
10490
10491
10492
10493
10494
10495
10496
10497
10498
10499
10500
10501
10502
10503
10504
10505
10506
10507
10508
10509
10510
10511
10512
10513
10514
10515
10516
10517
10518
10519
10520
10521
10522
10523
10524
10525
10526
10527

047732 000000
047734 000000
047736 000000
047740 012137 047722
047744 012137 047724
047750 012137 047726
047754 012137 047730
047760 012137 050530
047764 010146
047766 013700 001660
047772 012705 000002
047776 012710 000001
050002 052710 000010
050006 052710 000002
050012 042710 000012
050016 000404
050020 012710 000013
050024 042710 000012
050030 012702 000007
050034 052710 000002
050040 042710 000002
050044 005302
050046 001372
050050 005305
050052 001362
050054 012702 000022
050060 005037 050526
050064 004737 050532
050070 005302
050072 001372
050074 013737 047720 050526
050102 004737 050532
050106 032710 001000
050112 001012
050114 012737 000001 047736
050122 013737 047720 001124
050130 012737 104002 047630
050136 000571
050140 013737 047722 050526

: *READ DISK HEADER
: *****

NOSYNC: 0 ;FORCED HEADER ERROR = -1
TY: 0 ;NORMAL = 0
ERWORD: 0 ;ERROR TYPE NO.
;ERROR WORD NO.

RDHEAD: MOV (R1)+, @#RCYL ;STORE CYLINDER ADDRESS
MOV (R1)+, @#RSETR ;STORE SECTOR AND TRACK ADDRESS
MOV (R1)+, @#RKEY1 ;STORE KEY1
MOV (R1)+, @#RKEY2 ;STORE KEY2
MOV (R1)+, @#COMPA ;STORE COMPARE OR NOT
MOV R1, -(SP) ;PUSH R1 ON STACK
MOV @#RHMR, R0 ;R0 CONTAINS MAINTANENCE REG.
MOV #2, R5 ;R5 IS A COUNTER FOR WORDS
MOV #DMD, @R0 ;DIAG. MODE
BIS #MSTCK, @R0 ;SET SECTOR FOR FIRST WORD
BIS #MCLK, @R0 ;SET CLOCK FOR FIRST WORD
BIC #MSTCK!MCLK, @R0 ;RESET SECTOR AND CLOCK
BR 2\$;BRANCH OVER GIVING SECTOR FOR FIRST TIME
1\$: MOV #MSTCK!MCLK!DMD, @R0 ;SET SECTOR, CLOCK, DIAG. MODE, RESET INDEX
BIC #MSTCK!MCLK, @R0 ;RESET SECTOR, CLOCK
2\$: MOV #7, R2 ;R2 IS A COUNTER FOR BYTES
3\$: BIS #MCLK, @R0 ;SET CLOCK
BIC #MCLK, @R0 ;RESET CLOCK
DEC R2 ;BYTE COUNTER
BNE 3\$;BRANCH IF BYTE NOT COMPLETE
DEC R5 ;WORD COUNTER
BNE 1\$;BRANCH IF WORD NOT COMPLETE
MOV #18., R2 ;NO OF WORDS OF ZEROS
4\$: CLR @#WORD ;READ 0
JSR PC, @#READ ;GO TO READ
DEC R2 ;COUNT
BNE 4\$
MOV @#RSYNC, @#WORD ;SYNC. WORD
JSR PC, @#READ
BIT #DTSY, @R0 ;SYNC. BYTE DETECTED?
BNE 5\$;BRANCH IF SYNC DETECTED
MOV #1, @#ERWORD ;ERROR WORD NO
MOV @#RSYNC, @#SGDDAT ;SYNC WORD
MOV #104002, @#SSYN ;INSERT "ERROR 2" IN SSYN
BR 13\$;BRANCH OUT
5\$: MOV @#RCYL, @#WORD ;SETUP CYLINDER

```
10528 050146 004737 050532 JSR PC, @#READ :READ
10529 050152 013737 047724 050526 MOV @#RSETR,@#WORD :SETUP SECTOR/TRACK
10530 050160 004737 050532 JSR PC,@#READ :READ
10531 050164 013737 047726 050526 MOV @#RKEY1,@#WORD :SETUP KEY1
10532 050172 004737 050532 JSR PC,@#READ :READ
10533 050176 013737 047730 050526 MOV @#RKEY2,@#WORD :SETUP KEY2
10534 050204 004737 050532 JSR PC,@#READ :READ
10535 050210 013737 051516 050526 MOV @#WCRC,@#WORD :SETUP CRC
10536 050216 004737 050532 JSR PC,@#READ :READ
10537 050222 005737 002026 TST @#TESDTE :IS THIS A DRIVE TIMING ERROR
10538 050226 001135 BNE 13$ :BRANCH OUT IF YES
10539 050230 005737 050530 TST @#COMPA :IS THIS A READ OR WRITE COMMAND
10540 050234 001472 BEQ 11$
10541 050236 012705 051520 MOV #HEGAP, R5 :POINTER FOR HEADER GAP
10542 050242 012702 000005 MOV #5, R2 :NO OF WORDS OF ZEROS
10543 050246 012737 000006 047736 6$: MOV #6,@#ERWORD :ERROR WORD NO SET
10544 050254 004737 050764 JSR PC,@#WRITE :FOR HEADER GAP
10545 050260 005737 050762 TST @#WORD :TEST WRITTEN WORD
10546 050264 001413 BEQ 7$ :BRANCH IF GOOD THAT IS 0
10547 050266 160237 047736 SUB R2,@#ERWORD :WORD NO IN ERROR
10548 050272 005037 001124 CLR @#$GDDAT :GOOD WORD SHOULD BE 0
10549 050276 013737 050762 001126 MOV @#WORD, $BDDAT :BAD DATA
10550 050304 012737 104003 047632 MOV #104003,@#HEDGAP :"ERROR 2" GOES IN HEDGAP
10551 050312 000503 BR 13$ :BRANCH OUT
10552 050314 013725 050762 7$: MOV @#WORD,(R5)+ :SAVE HEADER GAP
10553 050320 005302 DEC R2
10554 050322 001351 BNE 6$
10555 050324 004737 050764 JSR PC, @#WRITE :WRITE HEADER (DATA) GAP SYNC
10556 050330 023737 047720 050762 CMP @#RSYNC,@#WORD
10557 050336 001426 BEQ 10$
10558 050340 005737 047732 TST @#NOSYNC :IS THIS FORCED HEADER ERROR COMMAND
10559 :IF YES NOSYNC=-1 THEN WRITE OR READ
10560 :IS SHUT OFF SO BRANCH OUT
10561 :IF NO NOSYNC=0 THEN CONTINUE
10562 050344 001406 BEQ 14$ :BRANCH IF TRUE ERROR
10563 050346 005737 050762 TST @#WORD
10564 050352 001420 BEQ 10$ :BRANCH IF GOOD
10565 050354 005037 001124 CLR @#$GDDAT :IT SHOULD BE ZERO
10566 050360 000403 BR 15$ :BRANCH TO TYPE ERROR
10567 050362 013737 047720 001124 14$: MOV @#RSYNC,@#$GDDAT :GOOD DATA
10568 050370 013737 050762 001126 15$: MOV @#WORD,@#$BDDAT :BAD DATA
10569 050376 012737 000006 047736 MOV #6, @#ERWORD
10570 050404 012737 104003 047634 MOV #104003,@#HEDSYN
10571 050412 000443 BR 13$ :BRANCH OUT
10572 050414 013725 050762 10$: MOV @#WORD,(R5)+ :SAVE DATA SYNC.
10573 050420 000440 BR 13$
10574 :*READ COMMAND START FROM HERE
10575 050422 012702 000005 11$: MOV #5, R2
10576 050426 005037 050526 12$: CLR WORD
10577 050432 004737 050532 JSR PC, READ :READ HEADER GAP
10578 050436 005302 DEC R2 :IS 5 HEADER GAP ZEROS COMPLETE
10579 050440 001372 BNE 12$ :IF NOT BRANCH
10580 050442 013737 047720 050526 MOV @#RSYNC,@#WORD :SYNC WORD
10581 050450 004737 050532 JSR PC, READ :READ HEADER (DATA) SYNC)
10582 050454 005737 047732 TST @#NOSYNC
10583 050460 001404 BEQ 16$ :IF NOT ERROR COMMAND BRANCH
```


10597
10598
10599
10600
10601
10602
10603
10604
10605
10606
10607
10608
10609
10610 050526 000000
10611 050530 000000
10612
10613
10614
10615
10616 050532
10617 050532 010246
10618 050534 012705 000002
10619 050540 012710 000001
10620 050544 006037 050526
10621 050550 103002
10622 050552 052710 000020
10623 050556 012702 000007
10624 050562 052710 000012
10625 050566 005737 045432
10626 050572 001411
10627 050574 032710 000020
10628 050600 001404
10629 050602 012737 177777 045424
10630 050610 000402
10631 050612 005037 045424
10632 050616 012746 000001
10633 050622 006037 050526
10634 050626 103002
10635 050630 012716 000021
10636 050634 012610
10637 050636 005737 045432
10638 050642 001404
10639 050644 005237 045444
10640 050650 004737 045462
10641 050654 052710 000002
10642 050660 005737 045432
10643 050664 001411
10644 050666 032710 000020
10645 050672 001404
10646 050674 012737 177777 045424
10647 050702 000402
10648 050704 005037 045424
10649 050710 012746 000001
10650 050714 006037 050526
10651 050720 103002
10652 050722 012716 000021

: *READ ONE WORD IN 'WORD'
: *****

WORD: 0
COMPA: 0

READ:

```
MOV R2, -(SP) ;: PUSH R2 ON STACK
MOV #2, R5 ;: WORD COUNTER
MOV #DMD, @RO ;: SET DIAG. MODE
ROR @#WORD ;: CHECKING IF THERE IS A ONE
BCC 1$ ;: IF NO ONE BRANCH
BIS #MRD, @RO ;: SET BIT 4 IF DATA HAS ONE
1$: MOV #7, R2 ;: BYTE COUNTER
BIS #MSTCK!MCLK, @RO ;: SET CLOCK, DATA IF ANY, SECTOR
TST @#TSECCG ;: IS THIS BIT TO GENERATE AND TEST ECC
BEQ 6$ ;: BRANCH IF NO
BIT #MRD, @RO ;: IS DATA BIT A ONE
BEQ 5$ ;: BRANCH IF DATA BIT IS 0
MOV #-1, @#ECDATA ;: ECC DATA BIT IS A ONE
BR 6$ ;: BRANCH
5$: CLR @#ECDATA ;: ECC DATA BIT IS A 0
6$: MOV #DMD, -(SP) ;: KEEP ONLY DIAG. MODE
ROR @#WORD ;: CHECKING IF THERE IS A ONE
BCC 2$ ;: IF NO ONE BRANCH
MOV #MRD!DMD, (SP) ;: KEEP DATA AND DIAG. MODE
2$: MOV (SP)+, @RO ;: PUT IN DATA, RESET CLOCK, SECTOR
TST @#TSECCG ;: IS ECC TO BE GENERATED FOR THIS BIT
BEQ 3$ ;: BRANCH IF NO
INC @#DATENV ;: NUMBER OF CLOCKS GIVEN FOR DATA ENVELOPE
JSR PC, @#ECTEST ;: GO TO GENERATE AND TEST ECC
3$: BIS #MCLK, @RO ;: SET CLOCK
TST @#TSECCG ;: IS THIS BIT TO GENERATE ECC
BEQ 8$ ;: BRANCH IF NO
BIT #MRD, @RO ;: IS DATA BIT A ONE
BEQ 7$ ;: BRANCH IF DATA BIT IS = 0
MOV #-1, @#ECDATA ;: ECC DATA BIT IS A ONE
BR 8$ ;: BRANCH
7$: CLR @#ECDATA ;: ECC DATA BIT IS = 0
8$: MOV #DMD, -(SP) ;: KEEP DIAG. MODE
ROR @#WORD ;: CHECKING IF THERE IS A ONE
BCC 4$ ;: BRANCH IF NO ONE
MOV #MRD!DMD, (SP) ;: KEEP DIAG. MODE AND DATA
```

| | | | | | | | |
|-------|--------|--------|--------|------|-----|-------------|---|
| 10653 | 050726 | 012610 | | 4\$: | MOV | (SP)+, @R0 | ;SET DATA, DIAG. MODE, CLEAR CLOCK |
| 10654 | 050730 | 005737 | 045432 | | TST | @#TSECCG | ;IS THIS BIT TO GENERATE ECC |
| 10655 | 050734 | 001404 | | | BEQ | 4\$ | ;BRANCH IF NO |
| 10656 | 050736 | 005237 | 045444 | | INC | @#DATENV | ;NUMBER OF CLOCKS GIVEN FOR DATA ENVELOPE |
| 10657 | 050742 | 004737 | 045462 | | JSR | PC,@#ECTEST | ;GO TO GENERATE AND TEST ECC |
| 10658 | 050746 | 005302 | | 9\$: | DEC | R2 | ;BYTE COUNTER |
| 10659 | 050750 | 001341 | | | BNE | 3\$ | ;BRANCH IF ONE BYTE NOT COMPLETE |
| 10660 | 050752 | 005305 | | | DEC | R5 | ;WORD COUNTER |
| 10661 | 050754 | 001300 | | | BNE | 1\$ | ;BRANCH IF ONE WORD NOT COMPLETE |
| 10662 | 050756 | 012602 | | | MOV | (SP)+,R2 | ;POP STACK INTO R2 |
| 10663 | 050760 | 000207 | | | RTS | PC | |
| 10664 | | | | | | | |
| 10665 | | | | | | | |

10666
10667
10668
10669
10670
10671
10672
10673
10674
10675
10676
10677
10678
10679 050762 000000
10680
10681
10682
10683
10684 050764
10685 050764 010046
10686 050766 010246
10687 050770 010346
10688 050772 010546
10689 050774 012705 000002
10690 051000 012710 000001
10691
10692 051004 012702 000007
10693 051010 012710 000013
10694 051014 032710 000040
10695 051020 001406
10696 051022 012737 177777 045424
10697 051030 000261
10698 051032 006003
10699 051034 000404
10700 051036 005037 045424
10701 051042 000241
10702 051044 006003
10703 051046 012710 000001
10704 051052 005737 045432
10705 051056 001404
10706 051060 005237 045444
10707 051064 004737 045462
10708 051070 052710 000002
10709 051074 032710 000040
10710 051100 001406
10711 051102 012737 177777 045424
10712 051110 000261
10713 051112 006003
10714 051114 000404
10715 051116 005037 045424
10716 051122 000241
10717 051124 006003
10718 051126 012710 000001
10719 051132 005737 045432
10720 051136 001404
10721 051140 005237 045444

: *WRITE ONE WORD WHICH COMES BACK IN 'WWORD'

WWORD: 0

WRITE:

MOV R0,-(SP) ;:PUSH R0 ON STACK
MOV R2,-(SP) ;:PUSH R2 ON STACK
MOV R3,-(SP) ;:PUSH R3 ON STACK
MOV R5,-(SP) ;:PUSH R5 ON STACK
MOV #2,R5 ;:WORD COUNTER
MOV #1,@R0 ;:SET DIAG. MODE
;:R0 HAS RHMR ADDRESS IN IT
1\$: MOV #7,R2 ;:BYTE COUNTER
MOV #MSTCK!MCLK!DMD,@R0 ;:SET SECTOR AND CLOCK
BIT #MWR,@R0 ;:CHECK WRITE BIT IN MAINT. REG.
BEQ 2\$;:BRANCH IF ZERO
MOV #-1,@#ECDATA ;:ECC DATA BIT IS A ONE
SEC ;:SET CARRY
ROR R3 ;:MOVE 1 FORWARD
BR 3\$
2\$: CLR @#ECDATA ;:ECC DATA BIT IS = 0
CLC ;:CLEAR CARRY
ROR R3 ;:MOVE 0 FOR WWORD
3\$: MOV #DMD,@R0 ;:CLEAR SECTOR AND CLOCK
TST @#TSECCG ;:IS THIS BIT TO GENERATE ECC
BEQ 4\$;:BRANCH IF NO
INC @#DATENV ;:NUMBER OF CLOCKS GIVEN FOR DATA ENVELOPE
JSR PC,@#ECTEST ;:GO TO GENERATE AND TEST ECC
4\$: BIS #MCLK,@R0 ;:SET CLOCK IN RHMR
BIT #MWR,@R0 ;:CHECK WRITE BIT IN RHMR
BEQ 5\$;:BRANCH IF ZERO
MOV #-1,@#ECDATA ;:ECC DATA BIT IS A ONE
SEC ;:SET CARRY
ROR R3 ;:MOVE 1 FOR WWORD
BR 6\$
5\$: CLR @#ECDATA ;:ECC DATA BIT IS ZERO
CLC ;:CLEAR CARRY
ROR R3 ;:MOVE 0 FOR WWORD
6\$: MOV #DMD,@R0 ;:CLEAR CLOCK
TST @#TSECCG ;:IS THIS BIT TO GENERATE ECC
BEQ 7\$;:BRANCH IF NO
INC @#DATENV ;:NUMBER OF CLOCKS GIVEN FOR DATA ENVELOPE

```
10722 051144 004737 045462      JSR    PC,@#ECTEST    ;GO TO GENERATE AND TEST ECC
10723 051150 005302      7$:   DEC    R2        ;COUNT FOR BYTE END
10724 051152 001346      BNE    4$            ;IF NOT BYTE END BRANCH
10725 051154 005305      DEC    R5            ;COUNT FOR WORD END
10726 051156 001312      BNE    1$            ;IF NOT WORD END BRANCH
10727
10728 051160 010337 050762      MOV    R3,@#WORD     ;STORE THE WORD
10729
10730 051164 012605      MOV    (SP)+,R5     ;;POP STACK INTO R5
10731 051166 012603      MOV    (SP)+,R3     ;;POP STACK INTO R3
10732 051170 012602      MOV    (SP)+,R2     ;;POP STACK INTO R2
10733 051172 012600      MOV    (SP)+,R0     ;;POP STACK INTO R0
10734 051174 000207      RTS    PC
10735
10736
```

10737
10738
10739
10740
10741
10742
10743
10744
10745
10746
10747
10748
10749
10750
10751
10752 051176 000000
10753 051200 000400
10754 051202 000000
10755 051204
10756 051204 011137 051176
10757 051210 012102
10758 051212 012137 050530
10759
10760 051216 010046
10761 051220 010146
10762 051222 010246
10763 051224 010346
10764 051226 010446
10765
10766 051230 012701 000016
10767 051234 012703 052542
10768 051240 012723 177777
10769 051244 005301
10770 051246 001374
10771 051250 013700 001660
10772 051254 013746 051200
10773 051260 163716 051176
10774 051264 011637 051202
10775 051270 012604
10776 051272 005737 002024
10777 051276 001403
10778 051300 012737 177777 045432
10779 051306 012703 051534
10780 051312 004737 050764
10781 051316 013723 050762
10782 051322 005302
10783 051324 001372
10784 051326 005704
10785 051330 001406
10786 051332 004737 050764
10787 051336 013723 050762
10788 051342 005304
10789 051344 001372
10790 051346 005037 045432
10791 051352 012701 000002
10792 051356 004737 050764

:WRITE DATA - PUT DATA INTO 'DISK' AREA FROM 'WORD'
:ONE WORD AT A TIME

COUNTD: 0
FORMAT: 256.
ZWORDS: 0
WRDATA:
MOV (R1),@#COUNTD ;STORE NO. OF WORDS TO BE WRITTEN
MOV (R1)+,R2 ;SAME IN R2
MOV (R1)+,@#COMPA ;COMPARE OR NOT
MOV R0,-(SP) ;;PUSH R0 ON STACK
MOV R1,-(SP) ;;PUSH R1 ON STACK
MOV R2,-(SP) ;;PUSH R2 ON STACK
MOV R3,-(SP) ;;PUSH R3 ON STACK
MOV R4,-(SP) ;;PUSH R4 ON STACK
MOV #14.,R1 ;NO. OF TOLERANCE GAP WORDS
MOV #TOLGAP,R3 ;START OF TOLERANCE GAP TABLE
1\$: MOV #-1,(R3)+ ;MAKE IT 177777
DEC R1 ;IS 14 COMPLETED
BNE 1\$;IF NO BRANCH
MOV @#RHMR,R0 ;R0 CONTAINS MAINTANENCE REG.
MOV @#FORMAT,-(SP)
SUB @#COUNTD,(SP)
MOV (SP),@#ZWORDS ;NO. OF ZERO WORDS TO BE WRITTEN
MOV (SP)+,R4
TST @#TSECC ;IS THIS AN ECC TEST ?
BEQ 7\$;BRANCH IF NO
MOV #-1,@#TSECCG ;THESE BITS ARE TO GENERATE ECC
7\$: MOV #DISK,R3 ;ADDRESS THE 'DISK' AREA
2\$: JSR PC,@#WRITE ;WRITE INTO 'WORD'
MOV @#WORD,(R3)+ ;STORE ON SIMULATED DISK
DEC R2 ;COUNT DOWN
BNE 2\$;CONTINUE IF ALL WORDS NOT WRITTEN
TST R4 ;ANY ZEROS TO BE WRITTEN ?
BEQ 4\$;BRANCH IF NONE TO BE WRITTEN
3\$: JSR PC,@#WRITE ;WRITE ZEROS INTO 'WORD'
MOV @#WORD,(R3)+ ;STORE INTO 'DISK'
DEC R4
BNE 3\$
4\$: CLR @#TSECCG ;NO MORE ECC TO BE GENERATED
MOV #2,R1
5\$: JSR PC,WRITE ;WRITE ECC1 AND ECC2 ON SIMULATED DISK

```
10793 051362 013723 050762      MOV    @#WWORD,(R3)+    ;STORE ON WEEC1 AND WEEC2
10794 051366 005301              DEC    R1
10795 051370 001372              BNE   5$
10796 051372 004737 050764      JSR   PC,WRITE          ;WRITE DATA GAP INTO 'WWORD'
10797 051376 013723 050762      MOV    @#WWORD,(R3)+    ;STORE INTO 'DISK'
10798 051402 012701 000016      MOV    #14.,R1
10799 051406 004737 050764      6$:   JSR   PC,@#WRITE      ;WRITE TOLERANCE GAP ZEROS
10800 051412 013723 050762      MOV    @#WWORD,(R3)+    ;STORE INTO 'DISK'
10801 051416 005301              DEC    R1
10802 051420 001372              BNE   6$
10803
10804 051422 012604              MOV    (SP)+,R4          ;;POP STACK INTO R4
10805 051424 012603              MOV    (SP)+,R3          ;;POP STACK INTO R3
10806 051426 012602              MOV    (SP)+,R2          ;;POP STACK INTO R2
10807 051430 012601              MOV    (SP)+,R1          ;;POP STACK INTO R1
10808 051432 012600              MOV    (SP)+,R0          ;;POP STACK INTO R0
10809
10810 051434 000201              RTS   R1
10811
10812
```

10813
10814
10815
10816
10817
10818
10819
10820
10821
10822
10823
10824
10825
10826
10827
10828
10829
10830
10831
10832
10833
10834
10835
10836
10837
10838
10839
10840
10841
10842
10843
10844
10845
10846

```
*****
: *WRITE HEADER AND DATA
: *
: *
: *THIS IS THE SIMULATED DISK
: *ONLY ONE SECTOR OF SPACE IS ALLOWED
: *****
```

051436 000023
051504 000001
051506 000004
051516 000001
051520 000005
051532 000001
051534
051534 000400
052534 000001
052536 000001
052540 000001
052542 000016

```
SECGAP: .BLKW 19.          :SECTOR GAP 38 BYTES OF 0
WSSYNC: .BLKW 1           :SECTOR GAP 1 BYTE OF 0 ONE SYNC BYTE
HEADER: .BLKW 4           :HEADER = CYL, SECTOR/TRACK, KEY1, KEY2
WCRC: .BLKW 1            :CRC
HEGAP: .BLKW 5           :HEADER GAP 10 BYTES OF 0
HDWSYN: .BLKW 1          :HEADER GAP 1 BYTE OF 0 ONE SYNC. BYTE
SILOTB:                   ;USED IN SILO TEST AS SILO TABLE
DISK: .BLKW 256.         :DATA SPACE
WECC1: .BLKW 1           :ECC1
WECC2: .BLKW 1           :ECC2
DTAGAP: .BLKW 1          :DATA GAP 2 BYTES OF 0
TOLGAP: .BLKW 14.        :TOLERANCE GAP 28 BYTES OF 0
```


10847
10848
10849
10850
10851
10852
10853
10854
10855
10856
10857
10858
10859
10860
10861
10862
10863
10864
10865
10866
10867
10868
10869
10870
10871
10872
10873
10874
10875
10876
10877
10878
10879
10880
10881
10882
10883
10884
10885
10886
10887
10888
10889
10890
10891
10892
10893
10894
10895
10896
10897
10898
10899
10900
10901
10902

;WRITE HEADER AND DATA

;**THIS SUBROUTINE IS THE FIRST IN A SERIES OF NESTED SUBROUTINES
;**IT ISSUES DIAGNOSTIC MODE, AN EXTRA DIAGNOSTIC INDEX, AND THE
;**'GO' BIT

;**IT THEN CALL THREE OTHER SUBROUTINES, WHICH IN TURN CALL OTHER
;**SUBROUTINES. THESE ARE:

;** SEARCH
;** WRHEAD
;** WRDATA

RNCTR1: .WORD 0 ;'RUN' LINE STALL COUNTER
COMWHD: MOV (SP),@#PCJSR ;SAVE PC OF JSR + 4
SUB #4,@#PCJSR ;SAVE PC OF JSR
MOV R0,-(SP) ;:PUSH R0 ON STACK
MOV R1,-(SP) ;:PUSH R1 ON STACK
MOV R2,-(SP) ;:PUSH R2 ON STACK
MOV R3,-(SP) ;:PUSH R3 ON STACK
MOV R4,-(SP) ;:PUSH R4 ON STACK
MOV R5,-(SP) ;:PUSH R5 ON STACK
10886 052626 012777 000001 127024 MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE
10887 052634 052777 000004 127016 BIS #MINX,@RHMR ;SET DIAGNOSTIC INDEX
10888 052642 042777 000004 127010 BIC #MINX,@RHMR ;CLEAR IT
10889 052650 052777 000001 126762 BIS #GO,@RHCS1 ;SET 'GO' BIT & STALL 'TILL 'RUN'
10891 052656 012737 000113 052576 RNWAT1: MOV #75.,@#RNCTR1 ;LOAD STALL COUNTER = APPROX. 450US
10892 ;FOR 11/50 CPU WITH CORE MEMORY
10893 052664 005337 052576 1\$: DEC @#RNCTR1 ;COUNT DOWN 1 TIME
10894 052670 001375 BNE 1\$;CONTINUE UNTIL 0
10896 052672 013746 052756 MOV @#WSECTR,-(SP) ;GET DESIRED SECTOR/TRACK
10897 052676 042716 177740 BIC #177740,(SP) ;MAKE ONLY SECTOR
10898 052702 012637 052712 MOV (SP)+,@#WTRK ;SAVE SECTOR
10900 052706 004137 053664 WTRK: JSR R1,@#SEARCH ;ISSUE SECTOR CLOCKS <----->
10901 052712 000000 .WORD 0 ;SECTOR NO.

```
10903 052714 012701 000240      MOV      #+NOP,R1      ;GOING TO MOVE NOPS
10904 052720 010137 052766      MOV      R1,@#SEGPEN  ;NOP INTO SEGAP
10905 052724 010137 052770      MOV      R1,@#FSYNER  ;NOP INTP FSYNER
10906 052730 010137 052772      MOV      R1,@#ERHEAD  ;NOP INTO ERHEAD
10907 052734 010137 052774      MOV      R1,@#ERCRC   ;NOP INTO ERCRC
10908 052740 010137 052776      MOV      R1,@#ERHDGP  ;NOP INTO ERHDGAP
10909 052744 010137 053000      MOV      R1,@#HDESYN  ;NOP INTO HDESYN
10910
10911 052750 004137 053050      JSR      R1,@#WRHEAD  ;WRITE THE HEADER <----->
10912 052754 000000      WCYL:    0            ;CYLINDER
10913 052756 000000      WSECTR:  0            ;SECTOR AND TRACK
10914 052760 000000      WKEY1:   0            ;KEY1
10915 052762 000000      WKEY2:   0            ;KEY2
10916 052764 000000      GCRC:    0            ;GOOD CRC
10917
10918      ;*DUMMY ERROR CALL LOCATIONS FOR THE WRITE HEADER OPERATION
10919
10920
10921 052766 000240      SEGPEN:  NOP          ;IF "ERROR 6" INSERTED BY
10922      ;WRHEAD SUBROUTINE THEN
10923      ;SECTOR GAP GOING ON DISK
10924      ;IS NOT RIGHT.
10925
10926      ;WORD NO. CONTAINS WHICH
10927      ;WORD IS WRONG, THAT IS
10928      ;FIRST OF TENTH OR WHAT EVER NO.
10929      ;BAD WORD IS GOING ON DISK
10930
10931 052770 000240      FSYNER:  NOP          ;IF "ERROR 6" INSERTED BY
10932      ;WRHEAD SUBROUTINE THEN
10933      ;THE LAST 0 BYTE OF SECTOR
10934      ;GAP, OR FIRST SYNC. BYTE
10935      ;AFTER SECTOR GAP IS IN
10936      ;ERROR.
10937
10938      ;WORD NO. CONTAINS 20
10939      ;RIGHT BYTE IS SECTOR GAP
10940      ;LEFT BYTE IS SYNC. BYTE
10941      ;BAD WORD IS WHAT IS GOING ON
10942      ;DISK.
10943
10944 052772 000240      ERHEAD:  NOP          ;IF "ERROR 6" INSERTED BY
10945      ;WRHEAD SUBROUTINE THEN
10946      ;HEADER GOING ON DISK
10947      ;IS WRONG.
10948
10949      ;WORD NO 1 = CYLINDER NO
10950      ;WORD NO 2 = SECTOR/TRACK
10951      ;WORD NO 3 = KEY1
10952      ;WORD NO 4 = KEY2
10953      ;BAD WORD IS WHAT IS GOING ON
10954      ;DISK
10955
10956 052774 000240      ERCRC:  NOP          ;IF "ERROR 6" INSERTED BY
10957      ;WRHEAD SUBROUTINE THEN CRC WRITTEN
10958
```


11006
11007
11008
11009
11010
11011
11012
11013
11014
11015
11016
11017
11018
11019
11020
11021
11022
11023
11024
11025
11026
11027
11028
11029
11030
11031
11032
11033
11034
11035
11036
11037
11038
11039
11040
11041
11042
11043
11044
11045
11046
11047
11048
11049
11050
11051
11052
11053
11054
11055
11056
11057
11058
11059
11060
11061

053036 000000
053040 000000
053042 000000
053044 000000
053046 000000

053050 012137 053036
053054 012137 053040
053060 012137 053042
053064 012137 053044
053070 012137 053046
053074 010146
053076 012701 051436
053102 013700 001660
053106 012710 000001
053112 012705 000002
053116 052710 000010
053122 012710 000013
053126 032710 000040
053132 001403
053134 000261
053136 006003
053140 000402
053142 000241
053144 006003
053146 012710 000001
053152 012702 000007
053156 052710 000002
053162 032710 000040
053166 001403
053170 000261
053172 006003
053174 000402
053176 000241
053200 006003
053202 012710 000001
053206 005302
053210 001362

:WRITE HEADER

;*R0 = MAINT.REG.
;*R1 = SIMULATED DISK
;*R2 = BYTE COUNT
;*R3 = WRITE WORD
;*R5 = WORD COUNT

SCYL: 0
SSECTR: 0
SKEY1: 0
SKEY2: 0
SCRC: 0

WRHEAD: MOV (R1)+,@#SCYL
MOV (R1)+,@#SSECTR
MOV (R1)+,@#SKEY1
MOV (R1)+,@#SKEY2
MOV (R1)+,@#SCRC
MOV R1,-(SP) ;:PUSH R1 ON STACK
MOV #SECGAP,R1 ;:SIMULATED DISK INDICATOR
MOV @#RHMR,R0 ;:R0 NOW HAS MAINT. REG. ADDR.
MOV #DMD,@RO ;:SET DIAG. MODE IN RHMR
MOV #2,R5 ;:WORD COUNTER
BIS #MSTCK,@RO ;:SET SECTOR FOR FIRST BYTE
1\$: MOV #MSTCK!MCLK!DMD,@RO ;:SET SECTOR, CLOCK, DIAG. MODE, RESET INDEX
BIT #MWR,@RO ;:CHECK WRITE BIT IN MAINT. REG.
BEQ 2\$
SEC ;:SET CARRY
ROR R3 ;:MOVE ONE FORWARD
BR 3\$
2\$: CLC ;:CLEAR CARRY
ROR R3 ;:MOVE ZERO FORWARD
3\$: MOV #DMD,@RO ;:CLEAR CLOCK, SECTOR
MOV #7,R2 ;:BYTE COUNTER
4\$: BIS #MCLK,@RO ;:SET CLOCK
BIT #MWR,@RO ;:CHECK WRITE BIT IN MAINT.REG.
BEQ 5\$;:BRANCH IF ZERO
SEC ;:SET CARRY
ROR R3 ;:MOVE ONE FORWARD
BR 6\$
5\$: CLC
ROR R3
6\$: MOV #DMD,@RO ;:SET DIAG. MODE AGAIN IN RHMR
DEC R2
BNE 4\$

```
11062 053212 005305          DEC      R5
11063 053214 001342          BNE      1$          ;CONTINUE
11064
11065 053216 010321          MOV      R3,(R1)+
11066 053220 005703          TST      R3
11067 053222 001414          BEQ      7$
11068 053224 012737 000001 047736  MOV      #1,@#ERWORD
11069 053232 005037 001124          CLR      @#$GDDAT
11070 053236 010337 001126          MOV      R3,@#$BDDAT
11071 053242 012737 104006 052766  MOV      #104006,@#SEGP
11072 053250 000137 053656          JMP      @#17$          ;BRANCH OUT ----->
11073
11074 053254 012702 000022          MOV      #18.,R2          ;COUNT NO. OF SECTOR GAP
11075 053260 012737 000024 047736 10$:  MOV      #20.,@#ERWORD    ;COUNT TO GIVE ERROR WORD
11076 053266 004737 050764          JSR      PC,@#WRITE      ;WRITE SECTOR GAP
11077 053272 013721 050762          MOV      @#WORD,(R1)+    ;STORE SECTOR GAP WORD
11078 053276 001413          BEQ      11$
11079 053300 160237 047736          SUB      R2,@#ERWORD     ;IF NOT GET ERROR WORD NO.
11080 053304 005037 001124          CLR      @#$GDDAT       ;GOOD WORD
11081 053310 013737 050762 001126  MOV      @#WORD,@#$BDDAT ;BAD WORD
11082 053316 012737 104006 052766  MOV      #104006,@#SEGP ;STORE 'ERROR 6' IN SEGP
11083 053324 000554          BR       17$          ;BRANCH OUT ----->
11084
11085 053326 005302          11$:  DEC      R2          ;HAVE 18 WORDS OF ZEROS BEEN WRITTEN ?
11086 053330 001353          BNE      10$          ;IF NOT DO SO
11087
11088          ;*AT THIS POINT THE SECTOR FOUND FLOP SHOULD
11089          ;*BE HIGH. SO THAT THE HEADER SYNC BYTE CAN BE GIVEN
11090
11091          ;*HOWEVER IN THE DRIVE TIMING ERROR TEST THE REST OF THE ROUTINE
11092          ;*IS ABORTED - HEADER SYNC BYTE IS NOT GIVEN
11093
11094 053332 005737 002026          TST      @#TESDTE        ;IS THIS A DRIVE TIMING ERROR
11095 053336 001147          BNE      17$          ;BRANCH OUT IF YES
11096 053340 004737 050764          JSR      PC,@#WRITE      ;WRITE ONE SECTOR GAP 0 BYTE
11097          ;AND ONE SYNC. BYTE = 230
11098 053344 013711 050762          MOV      @#WORD,(R1)     ;SAVE 0 BYTE AND SYNC BYTE
11099 053350 023721 047720          CMP      @#RSYNC,(R1)+   ;IF SYNC. BYTE RIGHT
11100 053354 001414          BEQ      12$          ;IF YES BRANCH
11101 053356 012737 000024 047736  MOV      #20.,@#ERWORD    ;IF NOT GET READY FOR ERROR
11102 053364 013737 047720 001124  MOV      @#RSYNC,@#$GDDAT ;GOOD WORD
11103 053372 014137 001126          MOV      -(R1),@#$BDDAT  ;BAD WORD
11104 053376 012737 104006 052770  MOV      #104006,@#FSYNER ;INSERT 'ERROR 6' IN FSYNER
11105 053404 000524          BR       17$          ;BRANCH OUT ----->
11106
11107 053406 012702 000004          12$:  MOV      #4,R2          ;FOUR HEADER WORDS
11108 053412 012703 053036          MOV      #SCYL,R3        ;POINTER FOR HEADER TABLE
11109 053416 012737 000005 047736 13$:  MOV      #5,@#ERWORD     ;ERROR WORD NO SET
11110 053424 004737 050764          JSR      PC,@#WRITE      ;WRITE 4 HEADER WORDS
11111 053430 013711 050762          MOV      @#WORD,(R1)     ;STORE WRITTEN WORD
11112 053434 022321          CMP      (R3)+,(R1)+     ;IS IT RIGHT?
11113 053436 001412          BEQ      14$          ;IF GOOD CONTINUE
11114          ;IF NOT GET READY FOR PRINT
11115 053440 160237 047736          SUB      R2,@#ERWORD     ;WORD NO
11116 053444 014337 001124          MOV      -(R3),@#$GDDAT  ;GOOD DATA
11117 053450 014137 001126          MOV      -(R1),@#$BDDAT  ;BAD DATA
```

```

11118 053454 012737 104006 052772      MOV      #104006,@#ERHEAD;INSERT "ERROR 6"
11119 053462 000475                    BR       17$                ;BRANCH OUT ----->
11120
11121 053464 005302                    14$:   DEC      R2                ;ARE 4 HEADER WORDS DONE?
11122 053466 001353                    BNE     13$                ;IF NOT DO THEM
11123 053470 004737 050764                    JSR     PC,@#WRITE         ;WRITE CRC
11124 053474 013711 050762                    MOV     @#WORD,(R1)        ;STORE CRC
11125 053500 022137 052764                    CMP     (R1)+,@#GCRC       ;COMPARE GOOD CRC
11126 053504 001414                    BEQ     20$                ;BRANCH IF GOOD
11127 053506 014137 001126                    MOV     -(R1),@#SBDDATA    ;BAD CRC WRITTEN
11128 053512 013737 052764 001124                    MOV     @#GCRC,@#SGDDAT   ;GOOD CRC
11129 053520 012737 000005 047736                    MOV     #5,@#ERWORD       ;ERROR WORD NO
11130 053526 012737 104006 052774                    MOV     #104006,@#ERCRC   ;INSERT ERROR 6
11131 053534 000450                    BR       17$                ;EXIT ----->
11132
11133 053536 012702 000005 20$:   MOV     #5,R2                ;NO OF HEADER GAP
11134 053542 012737 000006 047736 15$:   MOV     #6,@#ERWORD       ;ERROR WORD NO SET
11135 053550 004737 050764                    JSR     PC,@#WRITE         ;WRITE HEADER GAP
11136 053554 013721 050762                    MOV     @#WORD,(R1)+      ;STORE
11137 053560 001412                    BEQ     16$                ;IF GOOD BRANCH
11138 053562 160237 047736                    SUB     R2,@#ERWORD       ;ERROR WORD NO
11139 053566 005037 001124                    CLR     @#SGDDAT          ;GOOD DATA
11140 053572 014137 001126                    MOV     -(R1),@#SBDDAT    ;BAD DATA
11141 053576 012737 104006 052776                    MOV     #104006,@#ERHDGP ;STORE "ERROR 6"
11142 053604 000424                    BR       17$                ;BRANCH OUT ----->
11143
11144 053606 005302                    16$:   DEC      R2                ;ARE 5 HEADER GAP ZEROS DONE
11145 053610 001354                    BNE     15$                ;IF NOT BRANCH
11146 053612 004 37 050764                    JSR     PC,@#WRITE         ;WRITE CRC
11147 053616 013,11 050762                    MOV     @#WORD,(R1)        ;STORE CRC
11148 053622 023721 047720                    CMP     @#RSYNC,(R1)+     ;COMPARE GOOD CRC
11149 053626 001413                    BEQ     17$                ;BRANCH IF GOOD
11150 053630 012737 000005 047736                    MOV     #5,@#ERWORD       ;ERROR WORD NO
11151 053636 014137 001126                    MOV     -(R1),@#SBDDAT    ;BAD DATA
11152 053642 013737 047720 001124                    MOV     @#RSYNC,@#SGDDAT  ;GOOD DATA
11153 053650 012737 104006 053000                    MOV     #104006,@#HDESYN ;STORE "ERROR 6"
11154
11155 053656                    17$:
11156 053656 012601                    MOV     (SP)+,R1          ;:POP STACK INTO R1
11157
11158 053660 000201                    RTS      R1
11159
11160

```

11161
11162
11163
11164
11165
11166
11167
11168
11169
11170
11171
11172
11173
11174
11175
11176
11177
11178
11179
11180
11181
11182
11183
11184
11185
11186
11187
11188
11189
11190
11191
11192
11193
11194
11195
11196
11197
11198
11199
11200
11201
11202
11203
11204
11205
11206
11207
11208
11209
11210
11211
11212
11213
11214
11215
11216

: *SEARCH SECTOR

: * R0=RHMR ADDRESS
: * R1=PASSED ARGUMENT (SECTOR SEARCHED FOR)
: * R2=CLOCK COUNT (PER BYTE)
: * R3=SECTOR COUNTER FROM R1
: * R5=BYTES PER WORD COUNT
: * BEBORE INDEX IS GIVEN TWO SECTOR CLOCKS ARE GIVEN TO RESET
: * SECTOR PULSE IN CASE IT IS SET
: * AT BEGINNING OF EACH SECTOR ONE SECTOR CLOCK HAS TO RISE
: * BEFORE CLOCK THEN EVERY EIGHT CLOCKS ONE SECTOR CLOCK IS
: * IDENTICAL WITH CLOCK
: * NUMBERING THE SECTOR CLOCKS AS FOLLOWS
: * THE SECTOR CLOCK UNDER INDEX - 0
: * THE NEXT - 1
: * THE NEXT - 2
: * ETC.
: * THEN THE LAST SECTOR CLOCK IN ONE SECTOR HAS NUMBER - 608
: * THE NEXT SECTOR THEN HAS 608 SECTOR CLOCKS
: * THE NEXT SECTOR THEN HAS ANOTHER 608 SECTOR CLOCKS
: * AND SO ON

053662 000000
053664 012137 053662
053670 010046
053672 010146
053674 010246
053676 010346
053700 010446
053702 010546
053704 013700 001660
053710 013703 053662
053714 012710 000001
053720 052710 000010
053724 042710 000010
053730 052710 000010
053734 042710 000010

SECTR: 0 ;SECTOR SEARCHED FOR
SEARCH: MOV (R1)+, @#SECTR ;SAVE SECTOR SEARCHED FOR
MOV R0,-(SP) ;:PUSH R0 ON STACK
MOV R1,-(SP) ;:PUSH R1 ON STACK
MOV R2,-(SP) ;:PUSH R2 ON STACK
MOV R3,-(SP) ;:PUSH R3 ON STACK
MOV R4,-(SP) ;:PUSH R4 ON STACK
MOV R5,-(SP) ;:PUSH R5 ON STACK
MOV @#RHMR, R0 ;NOW R0 HAS MAINTENANCE REG. ADR.
MOV @#SECTR, R3 ;SECTOR COUNTER
MOV #DMD, @R0 ;SET DIAGNOSTIC MODE
BIS #MSTCK, @R0 ;SET SECTOR CLOCK
BIC #MSTCK, @R0 ;CLEAR SECTOR CLOCK
BIS #MSTCK, @R0 ;SET SECTOR CLOCK
BIC #MSTCK, @R0 ;CLEAR SECTOR CLOCK
;THE ABOVE TWO SECTOR CLOCKS ARE GIVEN FOR
;RESETTING SECTOR PULSE
;IN CASE IT STARTS SET

11217 053740 052710 000014
11218 053744 012710 000001
11219 053750 005703
11220 053752 001461

BIS #MINX!MSTCK,@R0 ;SET INDEX AND SECTOR CLOCK
MOV #DMD, @R0 ;RESET INDEX AND SECTOR CLOCK
TST R3 ;IF SECTOR REQUIRED JUMP OUT
BEQ 7\$;BRANCH OF SECTOR ZERO REQUIRED
;*NOW THE 304 WORDS WILL START

11221
11222
11223
11224
11225
11226
11227
11228
11229

;*FOR FIRST BYTE SECTOR CLOCK WILL GO HIGH THEN CLOCK WILL GO HIGH
;*BOTH WILL COME DOWN TOGETHER THEN SEVEN CLOCKS WILL BE GIVEN
;*FOR SECOND BYTE AND ALL OTHER BYTES TILL NEXT SECTOR SECTOR CLOCK
;*WILL BE IDENTICAL WITH ONE CLOCK

11230
11231
11232

;*ONE WORD ONLY

11233 053754 012702 000010
11234 053760 012705 000002
11235 053764 052710 000010
11236 053770 052710 000002
11237 053774 000402
11238 053776 052710 000012
11239 054002 042710 000012
11240 054006 052710 000002
11241 054012 042710 000002
11242 054016 005302
11243 054020 001372
11244 054022 012702 000007
11245 054026 005305
11246 054030 001362

1\$: MOV #8., R2 ;BYTE COUNTER
MOV #2, R5 ;BYTES PER WORD
BIS #MSTCK,@R0 ;SET SECTOR CLOCK
BIS #MCLK,@R0 ;SET CLOCK
BR 3\$;BRANCH TO CLEAR SECTOR AND CLOCK
2\$: BIS #MSTCK!MCLK,@R0 ;SET SECTOR AND CLOCK
3\$: BIC #MSTCK!MCLK,@R0 ;CLEAR SECTOR AND CLOCK
8\$: BIS #MCLK, @R0 ;SET CLOCK
BIC #MCLK, @R0 ;CLEAR CLOCK
DEC R2 ;BYTE COUNTER
BNE 8\$;BRANCH IF BYTE NOT COMPLETE
MOV #7, R2 ;SETUP FOR SECOND BYTE
DEC R5 ;IS WORD COMPLETE?
BNE 2\$;BRANCH IF NOT COMPLETE
;TO GIVE SECTOR CLOCK AND CLOCK

11247
11248
11249
11250

;*NOW 303 WORDS ARE LEFT AND ALL ARE IDENTICAL

11251
11252 054032 012701 000457
11253 054036 012705 000002
11254 054042 012702 000007
11255 054046 052710 000012
11256 054052 042710 000012
11257 054056 052710 000002
11258 054062 042710 000002
11259 054066 005302
11260 054070 001372
11261 054072 005305
11262 054074 001362
11263 054076 005301
11264 054100 001356
11265 054102 052710 000010
11266 054106 042710 000010
11267 054112 005303
11268 054114 001317

4\$: MOV #303., R1 ;WORDS PER SECTOR COUNTER
5\$: MOV #2, R5 ;BYTES PER WORD COUNTER
MOV #7, R2 ;BYTE COUNTER (CLOCK COUNTER)
6\$: BIS #MSTCK!MCLK,@R0 ;SET SECTOR CLOCK AND CLOCK
BIC #MSTCK!MCLK,@R0 ;CLEAR SECTOR CLOCK AND CLOCK
BIS #MCLK, @R0 ;SET CLOCK
BIC #MCLK, @R0 ;RESET CLOCK
DEC R2 ;IS BYTE COMPLETE?
BNE 6\$;BRANCH IF NOT COMPLETE
DEC R5 ;IS WORD COMPLETE?
BNE 5\$;BRANCH IF NOT
DEC R1 ;IS SECTOR COMPLETE
BNE 4\$;BRANCH IF NOT
BIS #MSTCK,@R0 ;SET SECTOR
BIC #MSTCK,@R0 ;CLEAR SECTOR
DEC R3 ;IS REQUIRED NO OF SECTORS COMPLETE
BNE 1\$;BRANCH IF NOT

11269
11270 054116
11271 054116 012605
11272 054120 012604

7\$: MOV (SP)+,R5 ;;POP STACK INTO R5
MOV (SP)+,R4 ;;POP STACK INTO R4


```
11273 054122 012603      MOV      (SP)+,R3      ;;POP STACK INTO R3
11274 054124 012602      MOV      (SP)+,R2      ;;POP STACK INTO R2
11275 054126 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
11276 054130 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
11277 054132 000201      RTS      R1
11278
11279
11280      ;*****
11281      ;*READ ONE SECTOR OF DATA
11282      ;*****
11283
11284 054134 000000      RNO:      0      ;NO. OF WORDS READ
11285 054136 000000      RCOM:     0      ;EXTRA STORAGE
11286
11287
11288
11289 054140 012137 054134      REDATA: MOV      (R1)+,@#RNO      ;SAVE NO. OF WORDS ONLY FOR INFORMATION
11290 054144 012137 054136      MOV      (R1)+,@#RCOM     ;EXTRA WORD ONLY FOR INFORMATION
11291 054150 010146      MOV      R1,-(SP)        ;;PUSH R1 ON STACK
11292 054152 005737 002024      TST      @#TSECC         ;IS THIS AN ECC TEST
11293 054156 001403      BEQ      1$              ;BRANCH IF NO
11294 054160 012737 177777 045432      MOV      #-1,@#TSECCG    ;THESE BITS ARE TO GENERATE ECC
11295 054166 012702 000402      1$:      MOV      #258.,R2       ;256 WORDS PER SECTOR
11296      ;PLUS 2 ECC WORDS
11297 054172 012703 051534      MOV      #DISK,R3        ;POINTE TO DISK SIMULATION
11298 054176 012337 050526      2$:      MOV      (R3)+,@#WORD    ;READY TO READ CONTENTS
11299 054202 004737 050532      JSR      PC,@#READ       ;READ
11300 054206 005302      DEC      R2              ;IS 256 WORDS DONE?
11301 054210 001372      BNE      2$              ;IF NOT BRANCH
11302 054212 005737 002024      TST      @#TSECC         ;IS THIS AN ECC TEST
11303 054216 001012      BNE      4$              ;BRANCH OUT IF YES
11304 054220 005037 045432      CLR      @#TSECCG        ;NO MORE ECC BITS ARE TO BE GENERATED
11305 054224 012702 000017      MOV      #15.,R2        ;ONE DATA GAP, 14 TOLERANCE GAP
11306 054230 012337 050526      3$:      MOV      (R3)+,@#WORD    ;READY TO READ CONTENTS OF WORD
11307 054234 004737 050532      JSR      PC,@#READ       ;READ
11308 054240 005302      DEC      R2              ;COUNT
11309 054242 001372      BNE      3$              ;BRANCH IF 14 NOT DONE
11310      4$:
11311 054244 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
11312 054246 000201      RTS      R1              ;RETURN
11313
11314
```

| | | | | | | | | | |
|-------|--------|--------|--------|---------|--------|------------------|--------------------|---------------------------|--|
| 11315 | | | | | | | | | |
| 11316 | | | | | | | | | |
| 11317 | 054250 | | | | | | | | |
| 11318 | 054250 | 104401 | 054256 | RPVECT: | TYPE | ,65\$ | | ::TYPE ASCIZ STRING | |
| 11319 | 054254 | 000421 | | | BR | 64\$ | | ::GET OVER THE ASCIZ | |
| 11320 | | | | ::65\$: | .ASCIZ | /UNEXPECTED RP04 | INTERRUPT @ PC = / | | |
| 11321 | 054320 | | | 64\$: | TYPOC | | | :TYPE FROM PC | |
| 11322 | 054320 | 104402 | | | MOV | #RPVECT,@RPVEC | | :RESTORE TRAP RP04 VECTOR | |
| 11323 | 054322 | 012777 | 054250 | 125276 | HALT | | | :CHANGE TO CONTINUE | |
| 11324 | 054330 | 000000 | | | | | | | |

```
11325 .SBTTL SYSMAC LIBRARY ROUTINES
11326
11327 .SBTTL SCOPE HANDLER ROUTINE
11328
11329
11330 ;:*****
11331 ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
11332 ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
11333 ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
11334 ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
11335 ;*SW14=1 LOOP ON TEST
11336 ;*SW11=1 INHIBIT ITERATIONS
11337 ;*SW09=1 LOOP ON ERROR
11338 ;*SW08=1 LOOP ON TEST IN SWR<7:0>
11339 ;*CALL
11340 ;* SCOPE ;:SCOPE=IOT
11341 $SCOPE:
11342 054332 104407 CKSWR ;:TEST FOR CHANGE IN SOFT-SWR
11343 054334 005037 047732 CLR @#NOSYNC ;:CLEAR FLAG FOR HEADER ERROR COMMANDS
11344 054340 005037 002024 CLR @#TSECC ;:CLEAR FLAG FOR ECC TEST
11345 ;:WHEN =177777 IT IS AN ECC TEST
11346 ;:WHEN =0 IT IS NOT AN ECC TEST
11347
11348 054344 005037 045432 CLR @#TSECCG ;:EVEN IN AN ECC TEST EVERY CLOCK
11349 ;:IS NOT TO GENERATE ECC
11350 ;:IF =177777 GENERATE ECC
11351 ;:IF =0 DO NOT GENERATE ECC
11352 054350 005037 002026 CLR @#TESDTE ;:DRIVE TIMING ERROR TEST
11353 054354
11354 054354 032777 040000 124556 1$: BIT #BIT14,@SWR ;:LOOP ON PRESENT TEST?
11355 054362 001111 BNE $OVER ;:YES IF SW14=1
11356 ;:#####START OF CODE FOR THE XOR TESTER#####
11357 054364 000416 $XTSTR: BR 6$ ;:IF RUNNING ON THE "XOR" TESTER CHANGE
11358 ;:THIS INSTRUCTION TO A "NOP" (NOP=240)
11359 054366 013746 000004 MOV @#ERRVEC,-(SP) ;:SAVE THE CONTENTS OF THE ERROR VECTOR
11360 054372 012737 054412 000004 MOV #5$,@#ERRVEC ;:SET FOR TIMEOUT
11361 054400 005737 177060 TST @#177060 ;:TIME OUT ON XOR?
11362 054404 012637 000004 MOV (SP)+,@#ERRVEC ;:RESTORE THE ERROR VECTOR
11363 054410 000463 BR $SVLAD ;:GO TO THE NEXT TEST
11364 054412 022626 5$: CMP (SP)+,(SP)+ ;:CLEAR THE STACK AFTER A TIME OUT
11365 054414 012637 000004 MOV (SP)+,@#ERRVEC ;:RESTORE THE ERROR VECTOR
11366 054420 000423 BR 7$ ;:LOOP ON THE PRESENT TEST
11367 054422 6$;#####END OF CODE FOR THE XOR TESTER#####
11368 054422 032777 000400 124510 BIT #BIT08,@SWR ;:LOOP ON SPEC. TEST?
11369 054430 001404 BEQ 2$ ;:BR IF NO
11370 054432 127737 124502 001102 CMPB @SWR,$TSTNM ;:ON THE RIGHT TEST? SWR<7:0>
11371 054440 001462 BEQ $OVER ;:BR IF YES
11372 054442 105737 001103 2$: TSTB $ERFLG ;:HAS AN ERROR OCCURRED?
11373 054446 001421 BEQ 3$ ;:BR IF NO
11374 054450 123737 001115 001103 CMPB $ERMAX,$ERFLG ;:MAX. ERRORS FOR THIS TEST OCCURRED?
11375 054456 101015 BHI 3$ ;:BR IF NO
11376 054460 032777 001000 124452 BIT #BIT09,@SWR ;:LOOP ON ERROR?
11377 054466 001404 BEQ 4$ ;:BR IF NO
11378 054470 013737 001110 001106 7$: MOV $LPERR,$LPADR ;:SET LOOP ADDRESS TO LAST SCOPE
11379 054476 000443 BR $OVER
11380 054500 105037 001103 4$: CLRB $ERFLG ;:ZERO THE ERROR FLAG
```

```
11381 054504 005037 001212          CLR    $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
11382 054510 000415                   BR     1$              ;;ESCAPE TO THE NEXT TEST
11383 054512 032777 004000 124420 3$: BIT    #BIT11,@SWR     ;;INHIBIT ITERATIONS?
11384 054520 001011                   BNE    1$              ;;BR IF YES
11385 054522 005737 001100          TST    $PASS          ;;IF FIRST PASS OF PROGRAM
11386 054526 001406                   BEQ    1$              ;;      INHIBIT ITERATIONS
11387 054530 005237 001104          INC    $ICNT          ;;INCREMENT ITERATION COUNT
11388 054534 023737 001212 001104    CMP    $TIMES,$ICNT   ;;CHECK THE NUMBER OF ITERATIONS MADE
11389 054542 002021                   BGE    $OVER          ;;BR IF MORE ITERATION REQUIRED
11390 054544 012737 000001 001104 1$: MOV    #1,$ICNT      ;;REINITIALIZE THE ITERATION COUNTER
11391 054552 013737 054622 001212    MOV    $MXCNT,$TIMES ;;SET NUMBER OF ITERATIONS TO DO
11392 054560 105237 001102          $SVLAD: INCB          $STNM                ;;COUNT TEST NUMBERS
11393 054564 011637 001106          MOV    (SP),$LPADR    ;;SAVE SCOPE LOOP ADDRESS
11394 054570 011637 001110          MOV    (SP),$LPERR    ;;SAVE ERROR LOOP ADDRESS
11395 054574 005037 001214          CLR    $ESCAPE        ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
11396 054600 112737 000001 001115    MOVB   #1,$ERMAX      ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
11397 054606 013777 001102 124326 $OVER: MOV   $STNM,@DISPLAY ;;DISPLAY TEST NUMBER
11398 054614 013716 001106          MOV    $LPADR,(SP)    ;;FUDGE RETURN ADDRESS
11399 054620 000002                   RTI                    ;;FIXES PS
11400 054622 000004          $MXCNT: 4            ;;MAX. NUMBER OF ITERATIONS
```

```
11401 .SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
11402
11403 *****
11404 *THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
11405 *SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
11406 *NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
11407 *BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
11408 *REPLACED WITH SPACES.
11409 *CALL:
11410 *      MOV      NUM,-(SP)      ;;PUT THE BINARY NUMBER ON THE STACK
11411 *      TYPDS                    ;;GO TO THE ROUTINE
11412
11413 $TYPDS:
11414 054624      010046      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
11415 054626      010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
11416 054630      010246      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
11417 054632      010346      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
11418 054634      010546      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
11419 054636      012746      020200      MOV      #20200,-(SP)  ;;SET BLANK SWITCH AND SIGN
11420 054642      016605      000020      MOV      20(SP),R5    ;;GET THE INPUT NUMBER
11421 054646      100004      BPL      1$           ;;BR IF INPUT IS POS.
11422 054650      005405      NEG      R5           ;;MAKE THE BINARY NUMBER POS.
11423 054652      112766      000055      000001      MOVB     #'-,1(SP)    ;;MAKE THE ASCII NUMBER NEG.
11424 054660      005000      1$:      CLR      R0           ;;ZERO THE CONSTANTS INDEX
11425 054662      012703      055040      MOV      #$DBLK,R3    ;;SETUP THE OUTPUT POINTER
11426 054666      112723      000040      MOVB     #' ,(R3)+    ;;SET THE FIRST CHARACTER TO A BLANK
11427 054672      005002      2$:      CLR      R2           ;;CLEAR THE BCD NUMBER
11428 054674      016001      055030      MOV      $DTBL(R0),R1 ;;GET THE CONSTANT
11429 054700      160105      3$:      SUB      R1,R5        ;;FORM THIS BCD DIGIT
11430 054702      002402      BLT      4$           ;;BR IF DONE
11431 054704      005202      INC      R2           ;;INCREASE THE BCD DIGIT BY 1
11432 054706      000774      BR       3$
11433 054710      060105      4$:      ADD      R1,R5        ;;ADD BACK THE CONSTANT
11434 054712      005702      TST      R2           ;;CHECK IF BCD DIGIT=0
11435 054714      001002      BNE      5$           ;;FALL THROUGH IF 0
11436 054716      105716      TSTB     (SP)         ;;STILL DOING LEADING 0'S?
11437 054720      100407      BMI      7$           ;;BR IF YES
11438 054722      106316      5$:      ASLB     (SP)         ;;MSD?
11439 054724      103003      BCC      6$           ;;BR IF NO
11440 054726      116663      000001      177777      MOVB     1(SP),-1(R3)  ;;YES--SET THE SIGN
11441 054734      052702      000060      6$:      BIS      #'0,R2       ;;MAKE THE BCD DIGIT ASCII
11442 054740      052702      000040      7$:      BIS      #' ,R2       ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
11443 054744      110223      MOVB     R2,(R3)+    ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
11444 054746      005720      TST      (R0)+        ;;JUST INCREMENTING
11445 054750      020027      000010      CMP      R0,#10      ;;CHECK THE TABLE INDEX
11446 054754      002746      BLT      2$           ;;GO DO THE NEXT DIGIT
11447 054756      003002      BGT      8$           ;;GO TO EXIT
11448 054760      010502      MOV      R5,R2        ;;GET THE LSD
11449 054762      000764      BR       6$           ;;GO CHANGE TO ASCII
11450 054764      105726      8$:      TSTB     (SP)+        ;;WAS THE LSD THE FIRST NON-ZERO?
11451 054766      100003      BPL      9$           ;;BR IF NO
11452 054770      116663      177777      177776      MOVB     -1(SP),-2(R3) ;;YES--SET THE SIGN FOR TYPING
11453 054776      105013      9$:      CLRB     (R3)         ;;SET THE TERMINATOR
11454 055000      012605      MOV      (SP)+,R5     ;;POP STACK INTO R5
11455 055002      012603      MOV      (SP)+,R3     ;;POP STACK INTO R3
11456 055004      012602      MOV      (SP)+,R2     ;;POP STACK INTO R2
```

CZRJGDO,RP04/5/6 DSKLS CTRLR1
CZRJGD.P11 10-SEP-79 11:00

MACY11 30A(1052) 10-SEP-79 11:11 PAGE 261
CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

SEQ 0260

| | | | | | | | |
|-------|--------|--------|--------|--------|---------|-------------|-----------------------|
| 11457 | 055006 | 012601 | | | MOV | (SP)+,R1 | ::POP STACK INTO R1 |
| 11458 | 055010 | 012600 | | | MOV | (SP)+,R0 | ::POP STACK INTO R0 |
| 11459 | 055012 | 104401 | 055040 | | TYPE | ,SDBLK | ::NOW TYPE THE NUMBER |
| 11460 | 055016 | 016666 | 000002 | 000004 | MOV | 2(SP),4(SP) | ::ADJUST THE STACK |
| 11461 | 055024 | 012616 | | | MOV | (SP)+,(SP) | |
| 11462 | 055026 | 000002 | | | RTI | | ::RETURN TO USER |
| 11463 | 055030 | 023420 | | | \$DTBL: | 10000. | |
| 11464 | 055032 | 001750 | | | | 1000. | |
| 11465 | 055034 | 000144 | | | | 100. | |
| 11466 | 055036 | 000012 | | | | 10. | |
| 11467 | 055040 | 000004 | | | \$DBLK: | .BLKW 4 | |

```
11468 .SBTTL TYPE ROUTINE
11469
11470
11471
11472
11473
11474
11475
11476
11477
11478
11479
11480
11481
11482
11483
11484
11485 055050 105737 001157
11486 055054 100002
11487 055056 000000
11488 055060 000407
11489 055062 010046
11490 055064 017600 000002
11491 055070 112046
11492 055072 001005
11493 055074 005726
11494 055076 012600
11495 055100 062716 000002
11496 055104 000002
11497 055106 122716 000011
11498 055112 001430
11499 055114 122716 000200
11500 055120 001006
11501 055122 005726
11502 055124 104401
11503 055126 001223
11504 055130 105037 055264
11505 055134 000755
11506 055136 004737 055220
11507 055142 123726 001156
11508 055146 001350
11509 055150 013746 001154
11510
11511 055154 105366 000001
11512 055160 002770
11513 055162 004737 055220
11514 055166 105337 055264
11515 055172 000770
11516
11517
11518
11519 055174 112716 000040
11520 055200 004737 055220
11521 055204 132737 000007 055264
11522 055212 001372
11523 055214 005726

*****
*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
*
$TYPE: TSTB $TPFLG ;;IS THERE A TERMINAL?
BPL 1$ ;;BR IF YES
HALT ;;HALT HERE IF NO TERMINAL
BR 3$ ;;LEAVE
1$: MOV RO,-(SP) ;;SAVE RO
MOV @2(SP),RO ;;GET ADDRESS OF ASCIZ STRING
2$: MOVB (RO)+,-(SP) ;;PUSH CHARACTER TO BE TYPED ONTO STACK
BNE 4$ ;;BR IF IT ISN'T THE TERMINATOR
TST (SP)+ ;;IF TERMINATOR POP IT OFF THE STACK
60$: MOV (SP)+,RO ;;RESTORE RO
3$: ADD #2,(SP) ;;ADJUST RETURN PC
RTI ;;RETURN
4$: CMPB #HT,(SP) ;;BRANCH IF <HT>
BEQ 8$
CMPB #CRLF,(SP) ;;BRANCH IF NOT <CRLF>
BNE 5$
TST (SP)+ ;;POP <CR><LF> EQUIV
TYPE ;;TYPE A CR AND LF
$CRLF
CLRB $CHARCNT ;;CLEAR CHARACTER COUNT
BR 2$ ;;GET NEXT CHARACTER
5$: JSR PC,$TYPEC ;;GO TYPE THIS CHARACTER
6$: CMPB $FILLC,(SP)+ ;;IS IT TIME FOR FILLER CHARS.?
BNE 2$ ;;IF NO GO GET NEXT CHAR.
MOV $NULL,-(SP) ;;GET # OF FILLER CHARS. NEEDED
;;AND THE NULL CHAR.
7$: DECB 1(SP) ;;DOES A NULL NEED TO BE TYPED?
BLT 6$ ;;BR IF NO--GO POP THE NULL OFF OF STACK
JSR PC,$TYPEC ;;GO TYPE A NULL
DECB $CHARCNT ;;DO NOT COUNT AS A COUNT
BR 7$ ;;LOOP
;HORIZONTAL TAB PROCESSOR
8$: MOVB #' ,(SP) ;;REPLACE TAB WITH SPACE
9$: JSR PC,$TYPEC ;;TYPE A SPACE
BITB #7,$CHARCNT ;;BRANCH IF NOT AT
BNE 9$ ;;TAB STOP
TST (SP)+ ;;POP SPACE OFF STACK
```

| | | | | | | | | |
|-------|--------|--------|--------|--------|------|------------------|--------------|--|
| 11524 | 055216 | 000724 | | | | BR | 2\$ | ::GET NEXT CHARACTER |
| 11525 | 055220 | 105777 | 123724 | | | \$TYPEC: TSTB | @\$TPS | ::WAIT UNTIL PRINTER IS READY |
| 11526 | 055224 | 100375 | | | | BPL | \$TYPEC | |
| 11527 | 055226 | 116677 | 000002 | 123716 | | MOVB | 2(SP),@\$TPB | ::LOAD CHAR TO BE TYPED INTO DATA REG. |
| 11528 | 055234 | 122766 | 000015 | 000002 | | CMPB | #CR,2(SP) | ::IS CHARACTER A CARRIAGE RETURN? |
| 11529 | 055242 | 001003 | | | | BNE | 1\$ | ::BRANCH IF NO |
| 11530 | 055244 | 105037 | 055264 | | | CLRB | \$CHARCNT | ::YES--CLEAR CHARACTER COUNT |
| 11531 | 055250 | 000406 | | | | BR | \$TYPEX | ::EXIT |
| 11532 | 055252 | 122766 | 000012 | 000002 | 1\$: | CMPB | #LF,2(SP) | ::IS CHARACTER A LINE FEED? |
| 11533 | 055260 | 001402 | | | | BEQ | \$TYPEX | ::BRANCH IF YES |
| 11534 | 055262 | 105227 | | | | INCB | (PC)+ | ::COUNT THE CHARACTER |
| 11535 | 055264 | 000000 | | | | \$CHARCNT: .WORD | 0 | ::CHARACTER COUNT STORAGE |
| 11536 | 055266 | 000207 | | | | \$TYPEX: RTS | PC | |
| 11537 | | | | | | | | |


```
11538 .SBTTL TTY INPUT ROUTINE
11539
11540 ;:*****
11541 .ENABL LSB
11542 055270 000000 $TKCNT: .WORD 0 ;:NUMBER OF ITEMS IN QUEUE
11543 055272 000000 $TKQIN: .WORD 0 ;:INPUT POINTER
11544 055274 000000 $TKQOUT: .WORD 0 ;:OUTPUT POINTER
11545 055276 000011 $TKQSRV: .BLKB 9. ;:TTY KEYBOARD QUEUE
11546 055307
11547 055310
11548
11549 ;*TK INITIALIZE ROUTINE
11550 ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
11551 ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
11552
11553 ;*CALL:
11554 ;* JSR PC,$TKINT
11555 ;* RETURN
11556
11557 055310 005037 055270 $TKINT: CLR $TKCNT ;:CLEAR COUNT OF ITEMS IN QUEUE
11558 055314 012737 055276 055272 MOV #TKQSRV,$TKQIN ;:MOVE THE STARTING ADDRESS OF THE
11559 055322 013737 055272 055274 MOV $TKQIN,$TKQOUT ;:QUEUE INTO THE INPUT & OUTPUT POINTERS.
11560 055330 012737 055360 000060 MOV #TKSRV,@TKVEC ;:INITIALIZE THE KEYBOARD VECTOR
11561 055336 012737 000200 000062 MOV #200,@TKVEC+2 ;:"BR" LEVEL 4
11562 055344 005777 123576 TST @TKB ;:CLEAR DONE FLAG
11563 055350 012777 000100 123566 MOV #100,@TKS ;:ENABLE TTY KEYBOARD INTERRUPT
11564 055356 000207 RTS PC ;:RETURN TO CALLER
11565
11566 ;*TK SERVICE ROUTINE
11567 ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
11568 ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
11569 ;*IT IN THE QUEUE.
11570 ;*IF THE CHARACTER IS A "CONTROL-C" (^C) $TKINT IS CALLED AND
11571 ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (OPERSEL)
11572
11573 055360 117746 123562 $TKSRV: MOVB @TKB,-(SP) ;:PICKUP THE CHARACTER
11574 055364 042716 177600 BIC #^C177,(SP) ;:STRIP THE JUNK
11575 055370 021627 000003 CMP (SP),#3 ;:IS IT A CONTROL C?
11576 055374 001007 BNE 1$ ;:BRANCH IF NO
11577 055376 104401 056347 TYPE ,SCNTLC ;:TYPE A CONTROL-C (^C)
11578 055402 004737 055310 JSR PC,$TKINT ;:INIT THE KEYBOARD
11579 055406 005726 TST (SP)+ ;:CLEAN UP STACK
11580 055410 000137 041556 JMP OPERSEL ;:CONTROL C RESTART
11581 055414 021627 000007 1$: CMP (SP),#7 ;:IS IT A CONTROL G?
11582 055420 001004 BNE 2$ ;:BRANCH IF NO
11583 055422 022737 000176 001140 CMP #SWREG,SWR ;:IS SOFT-SWR SELECTED?
11584 055430 001500 BEQ 6$ ;:GO TO SWR CHANGE
11585
11586 055432 2$:
11587 055432 022737 000011 055270 CMP #9,$TKCNT ;:IS THE QUEUE FULL?
11588 055440 001004 BNE 3$ ;:BRANCH IF NO
11589 055442 104401 001216 TYPE ,SBELL ;:RING THE TTY BELL
11590 055446 005726 TST (SP)+ ;:CLEAN CHARACTER OFF OF STACK
11591 055450 000451 BR 5$ ;:EXIT
11592 055452 021627 000023 3$: CMP (SP),#23 ;:IS IT A CONTROL-S?
11593 055456 001021 BNE 32$ ;:BRANCH IF NO
```

```
11594 055460 005077 123460          CLR    @STKS          ;;DISABLE TTY KEYBOARD INTERRUPTS
11595 055464 005726                   TST    (SP)+          ;;CLEAN CHAR OFF STACK
11596 055466 105777 123452          31$:  TSTB   @STKS          ;;WAIT FOR A CHAR
11597 055472 100375                   BPL    31$            ;;LOOP UNTIL ITS THERE
11598 055474 117746 123446          MOVVB  @STKB,-(SP)    ;;GET THE CHARACTER
11599 055500 042716 177600          BIC    #^C177,(SP)   ;;MAKE IT 7-BIT ASCII
11600 055504 022627 000021          CMP    (SP)+,#21     ;;IS IT A CONTROL-Q?
11601 055510 001366                   BNE    31$            ;;BRANCH IF NO
11602 055512 012777 000100 123424  MOV    #100,@STKS    ;;REENABLE TTY KEYBOARD INTERRUPTS
11603 055520 000002                   RTI                      ;;RETURN
11604 055522 005237 055270          32$:  INC    $TKCNT     ;;COUNT THIS CHARACTER
11605 055526 021627 000140          CMP    (SP),#140     ;;IS IT UPPER CASE?
11606 055532 002405                   BLT    4$             ;;BRANCH IF YES
11607 055534 021627 000175          CMP    (SP),#175     ;;IS IT A SPECIAL CHAR?
11608 055540 003002                   BGT    4$             ;;BRANCH IF YES
11609 055542 042716 000040          BIC    #40,(SP)      ;;MAKE IT UPPER CASE
11610 055546 112677 177520          4$:  MOVVB  (SP)+,@$TKQIN ;;AND PUT IT IN QUEUE
11611 055552 005237 055272          INC    $TKQIN        ;;UPDATE THE POINTER
11612 055556 023727 055272 055307  CMP    $TKQIN,$$TKQEND ;;GO OFF THE END?
11613 055564 001003                   BNE    5$             ;;BRANCH IF NO
11614 055566 012737 055276 055272  MOV    $$TKQSRT,$$TKQIN ;;RESET THE POINTER
11615 055574 000002          5$:  RTI                      ;;RETURN
11616
11617          ;;*****
11618          ;;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
11619          ;;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
11620          ;;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
11621          ;;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
11622 055576 022737 000176 001140  $CKSWR: CMP    #SWREG,SWR    ;;IS THE SOFT-SWR SELECTED
11623 055604 001124                   BNE    15$            ;;EXIT IF NOT
11624 055606 105777 123332          TSTB   @STKS          ;;IS A CHAR WAITING?
11625 055612 100121                   BPL    15$            ;;IF NOT, EXIT
11626 055614 117746 123326          MOVVB  @STKB,-(SP)    ;;YES
11627 055620 042716 177600          BIC    #^C177,(SP)   ;;MAKE IT 7-BIT ASCII
11628 055624 021627 000007          CMP    (SP),#7       ;;IS IT A CONTROL-G?
11629 055630 001300                   BNE    2$             ;;IF NOT, PUT IT IN THE TTY QUEUE
11630
11631
11632          ;;*****
11633          ;;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
11634          ;;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
11635          ;;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
11636 055632 123727 001134 000001  6$:  CMPB   $AUTOB,#1     ;;ARE WE RUNNING IN AUTO-MODE?
11637 055640 001674                   BEQ    2$             ;;BRANCH IF YES
11638 055642 005726                   TST    (SP)+          ;;CLEAR CONTROL-G OFF STACK
11639 055644 004737 055310          JSR    PC,$TKINT     ;;FLUSH THE TTY INPUT QUEUE
11640 055650 005077 123270          CLR    @STKS          ;;DISABLE TTY KEYBOARD INTERRUPTS
11641 055654 112737 000001 001135  MOVVB  #1,$INTAG     ;;SET INTERRUPT MODE INDICATOR
11642
11643 055662 104401 056361          TYPE   ,$CNTLG       ;;ECHO THE CONTROL-G (^G)
11644 055666 104401 056366          $GTSWR: TYPE   ,$MSWR    ;;TYPE CURRENT CONTENTS
11645 055672 013746 000176          MOV    SWREG,-(SP)   ;;SAVE SWREG FOR TYPEOUT
11646 055676 104402          TYPOC          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
11647 055700 104401 056377          TYPE   ,$MNEW        ;;PROMPT FOR NEW SWR
11648 055704 005046          19$:  CLR    -(SP)        ;;CLEAR COUNTER
11649 055706 005046          CLR    -(SP)        ;;THE NEW SWR
```

```
11650 055710 105777 123230      7$:  TSTB  @$TKS      ;;CHAR THERE?  
11651 055714 100375              BPL  7$           ;;IF NOT TRY AGAIN  
11652  
11653 055716 117746 123224      MOVB  @$TKB,-(SP)  ;;PICK UP CHAR  
11654 055722 042716 177600      BIC  #^C177,(SP)  ;;MAKE IT 7-BIT ASCII  
11655  
11656 055726 021627 000003      CMP   (SP),#3     ;;IS IT A CONTROL-C?  
11657 055732 001015      BNE  9$           ;;BRANCH IF NOT  
11658 055734 104401 056347      TYPE ,%CNTLC     ;;YES, ECHO CONTROL-C (^C)  
11659 055740 062706 000006      ADD  #6,SP        ;;CLEAN UP STACK  
11660 055744 123727 001135 000001  CMPB  $INTAG,#1   ;;REENABLE TTY KEYBOARD INTERRUPTS?  
11661 055752 001003      BNE  8$           ;;BRANCH IF NO  
11662 055754 012777 000100 123162  MOV  #100,@$TKS  ;;ALLOW TTY KEYBOARD INTERRUPTS  
11663 055762 000137 041556      8$:  JMP   OPERSEL     ;;CONTROL-C RESTART  
11664  
11665  
11666 055766 021627 000025      9$:  CMP   (SP),#25  ;;IS IT A CONTROL-U?  
11667 055772 001005      BNE  10$          ;;BRANCH IF NOT  
11668 055774 104401 056354      TYPE ,%CNTLU     ;;YES, ECHO CONTROL-U (^U)  
11669 056000 062706 000006      20$: ADD  #6,SP      ;;IGNORE PREVIOUS INPUT  
11670 056004 000737      BR   19$         ;;LET'S TRY IT AGAIN  
11671  
11672  
11673 056006 021627 000015      10$: CMP   (SP),#15  ;;IS IT A <CR>?  
11674 056012 001022      BNE  16$          ;;BRANCH IF NO  
11675 056014 005766 000004      TST  4(SP)        ;;YES, IS IT THE FIRST CHAR?  
11676 056020 001403      BEQ  11$          ;;BRANCH IF YES  
11677 056022 016677 000002 123110  MOV  2(SP),@SWR   ;;SAVE NEW SWR  
11678 056030 062706 000006      11$: ADD  #6,SP      ;;CLEAR UP STACK  
11679 056034 104401 001223      14$: TYPE ,%CRLF   ;;ECHO <CR> AND <LF>  
11680 056040 123727 001135 000001  CMPB  $INTAG,#1   ;;RE-ENABLE TTY KBD INTERRUPTS?  
11681 056046 001003      BNE  15$          ;;BRANCH IF NOT  
11682 056050 012777 000100 123066  MOV  #100,@$TKS  ;;RE-ENABLE TTY KBD INTERRUPTS  
11683 056056 000002      15$: RTI          ;;RETURN  
11684 056060 004737 055220      16$: JSR   PC,$TYPEC  ;;ECHO CHAR  
11685 056064 021627 000060      CMP   (SP),#60   ;;CHAR < 0?  
11686 056070 002420      BLT  18$          ;;BRANCH IF YES  
11687 056072 021627 000067      CMP   (SP),#67   ;;CHAR > 7?  
11688 056076 003015      BGT  18$          ;;BRANCH IF YES  
11689 056100 042726 000060      BIC  #60,(SP)+   ;;STRIP-OFF ASCII  
11690 056104 005766 000002      TST  2(SP)        ;;IS THIS THE FIRST CHAR  
11691 056110 001403      BEQ  17$          ;;BRANCH IF YES  
11692 056112 006316      ASL  (SP)         ;;NO, SHIFT PRESENT  
11693 056114 006316      ASL  (SP)         ;; CHAR OVER TO MAKE  
11694 056116 006316      ASL  (SP)         ;; ROOM FOR NEW ONE.  
11695 056120 005266 000002 17$: INC  2(SP)      ;;KEEP COUNT OF CHAR  
11696 056124 056616 177776      BIS  -2(SP),(SP) ;;SET IN NEW CHAR  
11697 056130 000667      BR   7$          ;;GET THE NEXT ONE  
11698 056132 104401 001222      18$: TYPE ,%QUES   ;;TYPE ?<CR><LF>  
11699 056136 000720      BR   20$        ;;SIMULATE CONTROL-U  
11700  
11701  
11702  
11703  
11704  
11705
```

*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
*CALL:

```
11706      ;*      RDCHR      ;:GET A CHARACTER FROM THE QUEUE
11707      ;*      RETURN HERE ;:CHARACTER IS ON THE STACK
11708      ;*      ;:WITH PARITY BIT STRIPPED OFF
11709      ;*
11710
11711 056140 011646 $RDCHR: MOV (SP),-(SP) ;:PUSH DOWN THE PC AND
11712 056142 016666 000004 000002 MOV 4(SP),2(SP) ;:THE PS
11713 056150 005066 000004 CLR 4(SP) ;:GET READY FOR A CHARACTER
11714 056154 005046 CLR -(SP) ;:PUT NEW PS ON STACK
11715 056156 012746 056164 MOV #64$,-(SP) ;:PUT NEW PC ON STACK
11716 056162 000002 RTI ;:POP NEW PC AND PS
11717 056164
11718 056164 005737 055270 64$: TST $TKCNT ;:WAIT ON A CHARACTER
11719 056170 001775 1$: BEQ 1$
11720 056172 005337 055270 DEC $TKCNT ;:DECREMENT THE COUNTER
11721 056176 117766 177072 000004 MOVB @TKQOUT,4(SP) ;:GET ONE CHARACTER
11722 056204 005237 055274 INC $TKQOUT ;:UPDATE THE POINTER
11723 056210 023727 055274 055307 CMP $TKQOUT,#TKQEND ;:DID IT GO OFF OF THE END?
11724 056216 001003 BNE 2$ ;:BRANCH IF NO
11725 056220 012737 055276 055274 MOV #TKQSRRT,$TKQOUT ;:RESET THE POINTER
11726 056226 000002 2$: RTI ;:RETURN
11727
11728 ;:*****
11729 ;:THIS ROUTINE WILL INPUT A STRING FROM THE TTY
11730 ;:CALL:
11731 ;*      RDLIN      ;:INPUT A STRING FROM THE TTY
11732 ;*      RETURN HERE ;:ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
11733 ;*      ;:TERMINATOR WILL BE A BYTE OF ALL 0'S
11734 056230 010346 $RDLIN: MOV R3, -(SP) ;:SAVE R3
11735 056232 012703 056336 1$: MOV #TTYIN,R3 ;:GET ADDRESS
11736 056236 022703 056347 2$: CMP #TTYIN+9.,R3 ;:BUFFER FULL?
11737 056242 101405 BLOS 4$ ;:BR IF YES
11738 056244 104410 RDCHR ;:GO READ ONE CHARACTER FROM THE TTY
11739 056246 112613 MOV (SP)+,(R3) ;:GET CHARACTER
11740 056250 122713 000177 10$: CMPB #177,(R3) ;:IS IT A RUBOUT
11741 056254 001003 BNE 3$ ;:SKIP IF NOT
11742 056256 104401 001222 4$: TYPE ,QUES ;:TYPE A '?'
11743 056262 000763 BR 1$ ;:CLEAR THE BUFFER AND LOOP
11744 056264 111337 056334 3$: MOV (R3),9$ ;:ECHO THE CHARACTER
11745 056270 104401 056334 TYPE ,9$
11746 056274 122723 000015 CMPB #15,(R3)+ ;:CHECK FOR RETURN
11747 056300 001356 BNE 2$ ;:LOOP IF NOT RETURN
11748 056302 105063 177777 CLR -1(R3) ;:CLEAR RETURN (THE 15)
11749 056306 104401 001224 TYPE ,LF ;:TYPE A LINE FEED
11750 056312 012603 MOV (SP)+,R3 ;:RESTORE R3
11751 056314 011646 MOV (SP),-(SP) ;:ADJUST THE STACK AND PUT ADDRESS OF THE
11752 056316 016666 000004 000002 MOV 4(SP),2(SP) ;:FIRST ASCII CHARACTER ON IT
11753 056324 012766 056336 000004 MOV #TTYIN,4(SP)
11754 056332 000002 RTI ;:RETURN
11755 056334 000 9$: .BYTE 0 ;:STORAGE FOR ASCII CHAR. TO TYPE
11756 056335 000 .BYTE 0 ;:TERMINATOR
11757 056336 000011 $TTYIN: .BLKB 9. ;:RESERVE 9. BYTES FOR TTY INPUT
11758 056347 136 006503 000012 $CNTLC: .ASCII /^C/<15><12> ;:CONTROL 'C'
11759 056354 052536 005015 000 $CNTLU: .ASCII /^U/<15><12> ;:CONTROL 'U'
11760 056361 136 006507 000012 $CNTLG: .ASCII /^G/<15><12> ;:CONTROL 'G'
11761 056366 005015 053523 020122 $MSWR: .ASCII <15><12>/SWR = /
```

| | | | | |
|-------|--------|--------|--------|---------------------------------|
| 11762 | 056374 | 020075 | 000 | |
| 11763 | 056377 | 040 | 047040 | 053505 \$MNEW: .ASCIZ / NEW = / |
| 11764 | 056404 | 036440 | 000040 | |
| 11765 | | | | |

;FROM THE TTY

```

11766 .SBTTL READ AN OCTAL NUMBER FROM THE TTY
11767
11768 ;*****
11769 ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
11770 ;*CHANGE IT TO BINARY.
11771 ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
11772 ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
11773 ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
11774 ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
11775 ;*CALL:
11776 ;* RDOCT ;:READ AN OCTAL NUMBER
11777 ;* RETURN HERE ;:LOW ORDER BITS ARE ON TOP OF THE STACK
11778 ;* ;:HIGH ORDER BITS ARE IN $HIOCT
11779
11780 056410 011646 $RDOCT: MOV (SP),-(SP) ;:PROVIDE SPACE FOR THE
11781 056412 016666 000004 000002 MOV 4(SP),2(SP) ;:INPUT NUMBER
11782 056420 010046 MOV R0,-(SP) ;:PUSH R0 ON STACK
11783 056422 010146 MOV R1,-(SP) ;:PUSH R1 ON STACK
11784 056424 010246 MOV R2,-(SP) ;:PUSH R2 ON STACK
11785 056426 104411 1$: RDLIN ;:READ AN ASCII LINE
11786 056430 012600 MOV (SP)+,R0 ;:GET ADDRESS OF 1ST CHARACTER
11787 056432 010037 056536 MOV R0,5$ ;:AND SAVE IT
11788 056436 005001 CLR R1 ;:CLEAR DATA WORD
11789 056440 005002 CLR R2
11790 056442 112046 2$: MOV (R0)+,-(SP) ;:PICKUP THIS CHARACTER
11791 056444 001420 BEQ 3$ ;:IF ZERO GET OUT
11792 056446 122716 000060 CMPB #'0,(SP) ;:MAKE SURE THIS CHARACTER
11793 056452 003026 BGT 4$ ;:IS AN OCTAL DIGIT
11794 056454 122716 000067 CMPB #'7,(SP)
11795 056460 002423 BLT 4$
11796 056462 006301 ASL R1 ;:*2
11797 056464 006102 ROL R2
11798 056466 006301 ASL R1 ;:*4
11799 056470 006102 ROL R2
11800 056472 006301 ASL R1 ;:*8
11801 056474 006102 ROL R2
11802 056476 042716 177770 BIC #'C7,(SP) ;:STRIP THE ASCII JUNK
11803 056502 062601 ADD (SP)+,R1 ;:ADD IN THIS DIGIT
11804 056504 000756 BR 2$ ;:LOOP
11805 056506 005726 3$: TST (SP)+ ;:CLEAN TERMINATOR FROM STACK
11806 056510 010166 000012 MOV R1,12(SP) ;:SAVE THE RESULT
11807 056514 010237 056546 MOV R2,$HIOCT
11808 056520 012602 MOV (SP)+,R2 ;:POP STACK INTO R2
11809 056522 012601 MOV (SP)+,R1 ;:POP STACK INTO R1
11810 056524 012600 MOV (SP)+,R0 ;:POP STACK INTO R0
11811 056526 000002 RTI ;:RETURN
11812 056530 005726 4$: TST (SP)+ ;:CLEAN PARTIAL FROM STACK
11813 056532 105010 CLRB (R0) ;:SET A TERMINATOR
11814 056534 104401 TYPE ;:TYPE UP THRU THE BAD CHAR.
11815 056536 000000 5$: .WORD 0
11816 056540 104401 001222 TYPE ,SQUES ;: "?" "CR" & "LF"
11817 056544 000730 BR 1$ ;:TRY AGAIN
11818 056546 000000 $HIOCT: .WORD 0 ;:HIGH ORDER BITS GO HERE

```

```
11819      .SBTTL  ERROR HANDLER ROUTINE
11820
11821      ;:*****
11822      ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
11823      ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
11824      ;*AND GO TO $ERRTYP ON ERROR
11825      ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
11826      ;*SW15=1      HALT ON ERROR
11827      ;*SW13=1      INHIBIT ERROR TYPEOUTS
11828      ;*SW10=1      BELL ON ERROR
11829      ;*SW09=1      LOOP ON ERROR
11830      ;*CALL
11831      ;*      ERROR      N      ;;ERRGR=EMT AND N=ERROR ITEM NUMBER
11832
11833      $ERROR:
11834      056550      104407      CKSWR      ;;TEST FOR CHANGE IN SOFT-SWR
11835      056552      012737      177777      002006      MOV      #-1,@#ERFLG$      ;;SET ERROR FLAG
11836      056560      REGSA1:
11837      056560      105237      001103      7$:      INCB      $ERFLG      ;;SET THE ERROR FLAG
11838      056564      001775      BEQ      7$      ;;DON'T LET THE FLAG GO TO ZERO
11839      056566      013777      001102      122346      MOV      $TSTNM,@DISPLAY      ;;DISPLAY TEST NUMBER AND ERROR FLAG
11840      056574      032777      002000      122336      BIT      #BIT10,@SWR      ;;BELL ON ERROR?
11841      056602      001402      BEQ      1$      ;;NO - SKIP
11842      056604      104401      001216      TYPE      ,SBELL      ;;RING BELL
11843      056610      005237      001112      1$:      INC      $ERTTL      ;;COUNT THE NUMBER OF ERRORS
11844      056614      011637      001116      MOV      (SP),$ERRPC      ;;GET ADDRESS OF ERROR INSTRUCTION
11845      056620      162737      000002      001116      SUB      #2,$ERRPC
11846      056626      117737      122264      001114      MOV      @ERRPC,$ITEMB      ;;STRIP AND SAVE THE ERROR ITEM CODE
11847      056634      032777      020000      122276      BIT      #BIT13,@SWR      ;;SKIP TYPEOUT IF SET
11848      056642      001004      BNE      20$      ;;SKIP TYPEOUTS
11849      056644      004737      056716      JSR      PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
11850      056650      104401      001223      TYPE      ,SCRLF
11851      056654      20$:
11852      056654      005777      122260      2$:      TST      @SWR      ;;HALT ON ERROR
11853      056660      100002      BPL      3$      ;;SKIP IF CONTINUE
11854      056662      000000      HALT      ;;HALT ON ERROR!
11855      056664      104407      CKSWR      ;;TEST FOR CHANGE IN SOFT-SWR
11856      056666      032777      001000      122244      3$:      BIT      #BIT09,@SWR      ;;LOOP ON ERROR SWITCH SET?
11857      056674      001402      BEQ      4$      ;;BR IF NO
11858      056676      013716      001110      MOV      $LPERR,(SP)      ;;FUDGE RETURN FOR LOOPING
11859      056702      005737      001214      4$:      TST      $ESCAPE      ;;CHECK FOR AN ESCAPE ADDRESS
11860      056706      001402      BEQ      5$      ;;BR IF NONE
11861      056710      013716      001214      MOV      $ESCAPE,(SP)      ;;FUDGE RETURN ADDRESS FOR ESCAPE
11862      056714      5$:
11863      056714      000002      RTI      ;;RETURN
```

11864
11865
11866
11867
11868
11869
11870
11871 056716
11872 056716 104401 001223
11873 056722 010046
11874 056724 005000
11875 056726 153700 001114
11876 056732 001004
11877
11878 056734 013746 001116
11879
11880 056740 104402
11881 056742 000445
11882 056744 005300
11883 056746 006300
11884 056750 006300
11885 056752 006300
11886 056754 062700 001226
11887 056760 012037 056770
11888 056764 001404
11889 056766 104401
11890 056770 000000
11891 056772 104401 001223
11892 056776 012037 057006
11893 057002 001404
11894 057004 104401
11895 057006 000000
11896 057010 104401 001223
11897 057014 010146
11898 057016 012001
11899 057020 001415
11900 057022 012000
11901 057024 105720
11902 057026 001003
11903 057030 013146
11904 057032 104402
11905 057034 000402
11906 057036
11907 057036 013146
11908 057040 104405
11909 057042 005711
11910 057044 001403
11911 057046 104401 057066
11912 057052 000764
11913
11914 057054 012601
11915 057056 012600
11916 057060 104401 001223
11917 057064 000207
11918 057066 020040 000
11919 057072

.SBTTL ERROR MESSAGE TYPEOUT ROUTINE

*THIS ROUTINE USES THE "ITEM CONTROL BYTE" (\$ITEMB) TO DETERMINE WHICH
*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" (\$ERRTB),
*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.

\$ERRTYP:
TYPE , \$CRLF ;:"CARRIAGE RETURN" & "LINE FEED"
MOV RO,-(SP) ;:SAVE RO
CLR RO ;:PICKUP THE ITEM INDEX
BISB @#\$ITEMB,RO
BNE 1\$;:IF ITEM NUMBER IS ZERO, JUST
 ;:TYPE THE PC OF THE ERROR
MOV \$ERRPC,-(SP) ;:SAVE \$ERRPC FOR TYPEOUT
 ;:ERROR ADDRESS
TYPDC ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
BR 10\$;:GET OUT
1\$: DEC RO ;:ADJUST THE INDEX SO THAT IT WILL
 ASL RO ;: WORK FOR THE ERROR TABLE
 ASL RO
 ASL RO
ADD # \$ERRTB,RO ;:FORM TABLE POINTER
MOV (RO)+,2\$;:PICKUP "ERROR MESSAGE" POINTER
BEQ 3\$;:SKIP TYPEOUT IF NO POINTER
TYPE ;:TYPE THE "ERROR MESSAGE"
 ;:"ERROR MESSAGE" POINTER GOES HERE
2\$: .WORD 0
TYPE , \$CRLF ;:"CARRIAGE RETURN" & "LINE FEED"
3\$: MOV (RO)+,4\$;:PICKUP "DATA HEADER" POINTER
BEQ 5\$;:SKIP TYPEOUT IF 0
TYPE ;:TYPE THE "DATA HEADER"
 ;:"DATA HEADER" POINTER GOES HERE
4\$: .WORD 0
TYPE , \$CRLF ;:"CARRIAGE RETURN" & "LINE FEED"
5\$: MOV R1,-(SP) ;:SAVE R1
MOV (R0)+,R1 ;:PICKUP "DATA TABLE" POINTER
BEQ 9\$;:BR IF NO DATA TO BE TYPED
MOV (R0)+,RO ;:PICKUP "DATA FORMAT" POINTER
6\$: TSTB (R0)+ ;:"OCTAL" OR "DECIMAL"
BNE 7\$;:BR IF DECIMAL
MOV @ (R1)+,-(SP) ;:SAVE @ (R1)+ FOR TYPEOUT
TYPDC ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
BR 8\$
7\$: MOV @ (R1)+,-(SP) ;:SAVE @ (R1)+ FOR TYPEOUT
TYPDS ;:GO TYPE--DECIMAL ASCII WITH SIGN
8\$: TST (R1) ;:IS THERE ANOTHER NUMBER?
BEQ 9\$;:BR IF NO
TYPE ,11\$;:TYPE TWO(2) SPACES
BR 6\$;:LOOP
9\$: MOV (SP)+,R1 ;:RESTORE R1
10\$: MOV (SP)+,RO ;:RESTORE RO
TYPE , \$CRLF ;:"CARRIAGE RETURN" & "LINE FEED"
RTS PC ;:RETURN
11\$: .ASCIZ / / ;:TWO(2) SPACES
.EVEN

11920
11921
11922
11923
11924
11925
11926
11927
11928
11929
11930
11931
11932
11933
11934
11935
11936
11937
11938
11939
11940
11941
11942
11943
11944
11945 057072 017646 000000
11946 057076 116637 000001 057315
11947 057104 112637 057317
11948 057110 062716 000002
11949 057114 000406
11950 057116 112737 000001 057315
11951 057124 112737 000006 057317
11952 057132 112737 000005 057314
11953 057140 010346
11954 057142 010446
11955 057144 010546
11956 057146 113704 057317
11957 057152 005404
11958 057154 062704 000006
11959 057160 110437 057316
11960 057164 113704 057315
11961 057170 016605 000012
11962 057174 005003
11963 057176 006105
11964 057200 000404
11965 057202 006105
11966 057204 006105
11967 057206 006105
11968 057210 010503
11969 057212 006103
11970 057214 105337 057316
11971 057220 100016
11972 057222 042703 177770
11973 057226 001002
11974 057230 005704
11975 057232 001403

```
.SBTTL BINARY TO OCTAL (ASCII) AND TYPE  
*****  
*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT  
*OCTAL (ASCII) NUMBER AND TYPE IT.  
*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE  
*CALL:  
*   MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED  
*   TYPOS   ;;CALL FOR TYPEOUT  
*   .BYTE  N              ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE  
*   .BYTE  M              ;;M=1 OR 0  
*                               ;;1=TYPE LEADING ZEROS  
*                               ;;0=SUPPRESS LEADING ZEROS  
*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST  
*$TYPOS OR $TYPOC  
*CALL:  
*   MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED  
*   TYPON   ;;CALL FOR TYPEOUT  
*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER  
*CALL:  
*   MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED  
*   TYPOC   ;;CALL FOR TYPEOUT!  
$TYPOS: MOV     @ (SP),-(SP)  ;;PICKUP THE MODE  
        MOV     1(SP), $OFILL ;;LOAD ZERO FILL SWITCH  
        MOV     (SP)+, $OMODE+1 ;;NUMBER OF DIGITS TO TYPE  
        ADD     #2, (SP)      ;;ADJUST RETURN ADDRESS  
        BR     $TYPON  
$TYPOC: MOV     #1, $OFILL   ;;SET THE ZERO FILL SWITCH  
        MOV     #6, $OMODE+1 ;;SET FOR SIX(6) DIGITS  
$TYPON: MOV     #5, $OCNT    ;;SET THE ITERATION COUNT  
        MOV     R3, -(SP)    ;;SAVE R3  
        MOV     R4, -(SP)    ;;SAVE R4  
        MOV     R5, -(SP)    ;;SAVE R5  
        MOV     $OMODE+1, R4 ;;GET THE NUMBER OF DIGITS TO TYPE  
        NEG     R4  
        ADD     #6, R4      ;;SUBTRACT IT FOR MAX. ALLOWED  
        MOV     R4, $OMODE  ;;SAVE IT FOR USE  
        MOV     $OFILL, R4  ;;GET THE ZERO FILL SWITCH  
        MOV     12(SP), R5  ;;PICKUP THE INPUT NUMBER  
        CLR     R3          ;;CLEAR THE OUTPUT WORD  
1$:    ROL     R5          ;;ROTATE MSB INTO 'C'  
        BR     3$          ;;GO DO MSB  
2$:    ROL     R5          ;;FORM THIS DIGIT  
        ROL     R5  
        ROL     R5  
        MOV     R5, R3  
3$:    ROL     R3          ;;GET LSB OF THIS DIGIT  
        DECB   $OMODE     ;;TYPE THIS DIGIT?  
        BPL   7$          ;;BR IF NO  
        BIC   #177770, R3 ;;GET RID OF JUNK  
        BNE   4$          ;;TEST FOR 0  
        TST  R4          ;;SUPPRESS THIS 0?  
        BEQ  5$          ;;BR IF YES
```

| | | | | | | | |
|-------|--------|--------|---------------|------|----------|-------------|-----------------------------------|
| 11976 | 057234 | 005204 | | 4\$: | INC | R4 | ::DON'T SUPPRESS ANYMORE 0'S |
| 11977 | 057236 | 052703 | 000060 | | BIS | #'0,R3 | ::MAKE THIS DIGIT ASCII |
| 11978 | 057242 | 052703 | 000040 | 5\$: | BIS | #',R3 | ::MAKE ASCII IF NOT ALREADY |
| 11979 | 057246 | 110337 | 057312 | | MOVB | R3,8\$ | ::SAVE FOR TYPING |
| 11980 | 057252 | 104401 | 057312 | | TYPE | ,8\$ | ::GO TYPE THIS DIGIT |
| 11981 | 057256 | 105337 | 057314 | 7\$: | DECB | \$OCNT | ::COUNT BY 1 |
| 11982 | 057262 | 003347 | | | BGT | 2\$ | ::BR IF MORE TO DO |
| 11983 | 057264 | 002402 | | | BLT | 6\$ | ::BR IF DONE |
| 11984 | 057266 | 005204 | | | INC | R4 | ::INSURE LAST DIGIT ISN'T A BLANK |
| 11985 | 057270 | 000744 | | | BR | 2\$ | ::GO DO THE LAST DIGIT |
| 11986 | 057272 | 012605 | | 6\$: | MOV | (SP)+,R5 | ::RESTORE R5 |
| 11987 | 057274 | 012604 | | | MOV | (SP)+,R4 | ::RESTORE R4 |
| 11988 | 057276 | 012603 | | | MOV | (SP)+,R3 | ::RESTORE R3 |
| 11989 | 057300 | 016666 | 000002 000004 | | MOV | 2(SP),4(SP) | ::SET THE STACK FOR RETURNING |
| 11990 | 057306 | 012616 | | | MOV | (SP)+,(SP) | |
| 11991 | 057310 | 000002 | | | RTI | | ::RETURN |
| 11992 | 057312 | 000 | | 8\$: | .BYTE | 0 | ::STORAGE FOR ASCII DIGIT |
| 11993 | 057313 | 000 | | | .BYTE | 0 | ::TERMINATOR FOR TYPE ROUTINE |
| 11994 | 057314 | 000 | | | \$OCNT: | .BYTE 0 | ::OCTAL DIGIT COUNTER |
| 11995 | 057315 | 060 | | | \$OFILL: | .BYTE 0 | ::ZERO FILL SWITCH |
| 11996 | 057316 | 000000 | | | \$OMODE: | .WORD 0 | ::NUMBER OF DIGITS TO TYPE |

11997
11998
11999
12000
12001
12002
12003
12004
12005 057320 010046
12006 057322 016600 000002
12007 057326 005740
12008 057330 111000
12009 057332 006300
12010 057334 016000 057354
12011 057340 000200
12012
12013
12014
12015
12016 057342 011646
12017 057344 016666 000004 000002
12018 057352 000002
12019
12020
12021
12022
12023
12024
12025
12026
12027 057354 057342
12028 057356 055050
12029 057360 057116
12030 057362 057072
12031 057364 057132
12032 057366 054624
12033
12034 057370 055666
12035
12036 057372 055576
12037 057374 056140
12038 057376 056230
12039 057400 056410
12040 057402 042714
12041 057404 042766
12042 057406 043304

.SBTTL TRAP DECODER

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

\$TRAP: MOV R0,-(SP) ;;SAVE R0
MOV 2(SP),R0 ;;GET TRAP ADDRESS
TST -(R0) ;;BACKUP BY 2
MOVB (R0),R0 ;;GET RIGHT BYTE OF TRAP
ASL R0 ;;POSITION FOR INDEXING
MOV \$TRPAD(R0),R0 ;;INDEX TO TABLE
RTS R0 ;;GO TO ROUTINE

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

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

.SBTTL TRAP TABLE

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

ROUTINE

\$TRPAD: .WORD \$TRAP2
\$TYPE ;;CALL=TYPE TRAP+1(104401) TTY TYPEOUT ROUTINE
\$TYPOC ;;CALL=TYPOC TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
\$TYPOS ;;CALL=TYPOS TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
\$TYPON ;;CALL=TYPON TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
\$TYPDS ;;CALL=TYPDS TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
\$GTSWR ;;CALL=GTSWR TRAP+6(104406) GET SOFT-SWR SETTING
\$CKSWR ;;CALL=CKSWR TRAP+7(104407) TEST FOR CHANGE IN SOFT-SWR
\$RDCHR ;;CALL=RDCHR TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
\$RDLIN ;;CALL=RDLIN TRAP+11(104411) TTY TYPEIN STRING ROUTINE
\$RDOCT ;;CALL=RDOCT TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
T.SCOPI ;;CALL=SCOPI TRAP+13(104413) MY LOCAL SCOPES
CHECKD ;;CALL=CHECKD TRAP+14(104414) CHECK DVA,RDY,DPR,DRY
WAIT.T ;;CALL=WAT TRAP+15(104415) WAIT LOOP

```
12043 .SBTTL POWER DOWN AND UP ROUTINES
12044
12045 ::*****
12046 ::POWER DOWN ROUTINE
12047 057410 012737 057554 000024 $PWRDN: MOV $ILLUP,@#PWRVEC ;;SET FOR FAST UP
12048 057416 012737 000340 000026 MOV #340,@#PWRVEC+2 ;;PRIO:7
12049 057424 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK
12050 057426 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
12051 057430 010246 MOV R2,-(SP) ;;PUSH R2 ON STACK
12052 057432 010346 MOV R3,-(SP) ;;PUSH R3 ON STACK
12053 057434 010446 MOV R4,-(SP) ;;PUSH R4 ON STACK
12054 057436 010546 MOV R5,-(SP) ;;PUSH R5 ON STACK
12055 057440 017746 121474 MOV @SWR,-(SP) ;;PUSH @SWR ON STACK
12056 057444 010637 057560 MOV SP,$SAVR6 ;;SAVE SP
12057 057450 012737 057462 000024 MOV #PWRUP,@#PWRVEC ;;SET UP VECTOR
12058 057456 000000 HALT
12059 057460 000776 BR .-2 ;;HANG UP
12060
12061 ::*****
12062 ::POWER UP ROUTINE
12063 057462 012737 057554 000024 $PWRUP: MOV $ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
12064 057470 013706 057560 MOV $SAVR6,SP ;;GET SP
12065 057474 005037 057560 CLR $SAVR6 ;;WAIT LOOP FOR THE TTY
12066 057500 005237 057560 1$: INC $SAVR6 ;;WAIT FOR THE INC
12067 057504 001375 BNE 1$ ;;OF WORD
12068 057506 012677 121426 MOV (SP)+,@SWR ;;POP STACK INTO @SWR
12069 057512 012605 MOV (SP)+,R5 ;;POP STACK INTO R5
12070 057514 012604 MOV (SP)+,R4 ;;POP STACK INTO R4
12071 057516 012603 MOV (SP)+,R3 ;;POP STACK INTO R3
12072 057520 012602 MOV (SP)+,R2 ;;POP STACK INTO R2
12073 057522 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
12074 057524 012600 MOV (SP)+,R0 ;;POP STACK INTO R0
12075 057526 012737 057410 000024 MOV #PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
12076 057534 012737 000340 000026 MOV #340,@#PWRVEC+2 ;;PRIO:7
12077 057542 104401 TYPE ;;REPORT THE POWER FAILURE
12078 057544 057562 $PWRMG: .WORD $POWER ;;POWER FAIL MESSAGE POINTER
12079 057546 012716 MOV (PC)+,(SP) ;;RESTART AT BEGIN
12080 057550 004244 $PWRAD: .WORD BEGIN ;;RESTART ADDRESS
12081 057552 000002 RTI
12082 057554 000000 $ILLUP: HALT ;;THE POWER UP SEQUENCE WAS STARTED
12083 057556 000776 BR .-2 ;; BEFORE THE POWER DOWN WAS COMPLETE
12084 057560 000000 $SAVR6: 0 ;;PUT THE SP HERE
12085 057562 005015 047520 042527 $POWER: .ASCIZ <15><12>'POWER'
12086 057570 000122
12087 .EVEN
```

```
12088 :*****
12089 :*
12090 :*ERROR AND MESSAGE TABLE CONDIMENTS
12091 :*
12092 :*****
12093
12094
12095
12096
12097 057572 051127 047117 020107 EM1: .ASCIZ /WRONG DATA IN READING OR WRITING HARDWARE REGISTER/
12098 057600 040504 040524 044440
12099 057606 020116 042522 042101
12100 057614 047111 020107 051117
12101 057622 053440 044522 044524
12102 057630 043516 044040 051101
12103 057636 053504 051101 020105
12104 057644 042522 044507 052123
12105 057652 051105 000
12106 057655 105 051122 051117 EM2: .ASCIZ /ERROR ON DATA COMMAND/
12107 057662 047440 020116 042040
12108 057670 052101 020101 047503
12109 057676 046515 047101 000104
12110 057704 051105 047522 020122 EM6: .ASCIZ /ERROR ON WRITE HEADER AND DATA/
12111 057712 047117 053440 044522
12112 057720 042524 044040 040505
12113 057726 042504 020122 047101
12114 057734 020104 040504 040524
12115 057742 000
12116 057743 103 047117 051124 EM11: .ASCIZ /CONTROLLER OR DRIVE STATUS/
12117 057750 046117 042514 020122
12118 057756 051117 042040 044522
12119 057764 042526 051440 040524
12120 057772 052524 000123
12121 057776 042522 044507 052123 EM14: .ASCIZ /REGISTER FAILED/
12122 060004 051105 043040 044501
12123 060012 042514 000104
12124 060016 047516 020116 054105 EM15: .ASCIZ /NON EXISTENT REGISTER, PROGRAM ABORTED./
12125 060024 051511 042524 052116
12126 060032 051040 043505 051511
12127 060040 042524 026122 020040
12128 060046 051120 043517 040522
12129 060054 020115 041101 051117
12130 060062 042524 027104 000
12131 060067 127 044501 020124 EM16: .ASCIZ /WAIT LOOP FAILED/
12132 060074 047514 050117 043040
12133 060102 044501 042514 000104
12134 060110 051127 052111 020105 EM17: .ASCIZ /WRITE CHECK FAILING/
12135 060116 044103 041505 020113
12136 060124 040506 046111 047111
12137 060132 000107
12138 060134 042522 044507 052123 EM20: .ASCIZ /REGISTER FAILING/
12139 060142 051105 043040 044501
12140 060150 044514 043516 000
12141 060155 111 052116 051105 EM21: .ASCIZ /INTERRUPT FAILING/
12142 060162 052522 052120 043040
12143 060170 044501 044514 043516
```

| | | | | | | |
|-------|--------|--------|--------|--------|--------|---|
| 12144 | 060176 | 000 | | | | |
| 12145 | 060177 | 105 | 051122 | 051117 | EM22: | .ASCII /ERROR ON DRIVES PRESENT -/<15><12> |
| 12146 | 060204 | 047440 | 020116 | 051104 | | |
| 12147 | 060212 | 053111 | 051505 | 050040 | | |
| 12148 | 060220 | 042522 | 042523 | 052116 | | |
| 12149 | 060226 | 026440 | 005015 | | | |
| 12150 | 060232 | 044124 | 020105 | 047125 | .ASCII | /THE UNIT NO'S FOUND BY SETTING RHAS USING RHER1/<15><12> |
| 12151 | 060240 | 052111 | 047040 | 023517 | | |
| 12152 | 060246 | 020123 | 047506 | 047125 | | |
| 12153 | 060254 | 020104 | 054502 | 051440 | | |
| 12154 | 060262 | 052105 | 044524 | 043516 | | |
| 12155 | 060270 | 051040 | 040510 | 020123 | | |
| 12156 | 060276 | 051525 | 047111 | 020107 | | |
| 12157 | 060304 | 044122 | 051105 | 006461 | | |
| 12158 | 060312 | 012 | | | | |
| 12159 | 060313 | 050 | 032124 | 020051 | .ASCII | /(T4) DO NOT AGREE WITH THE UNIT NO'S FOUND/<15><12> |
| 12160 | 060320 | 047504 | 047040 | 052117 | | |
| 12161 | 060326 | 040440 | 051107 | 042505 | | |
| 12162 | 060334 | 053440 | 052111 | 020110 | | |
| 12163 | 060342 | 044124 | 020105 | 047125 | | |
| 12164 | 060350 | 052111 | 047040 | 023517 | | |
| 12165 | 060356 | 020123 | 047506 | 047125 | | |
| 12166 | 060364 | 006504 | 012 | | | |
| 12167 | 060367 | 102 | 020131 | 047514 | .ASCII | /BY LOOKING FOR 'NED' = 0 IN RHCS2 (BIT #12)/<15><12><15><12> |
| 12168 | 060374 | 045517 | 047111 | 020107 | | |
| 12169 | 060402 | 047506 | 020122 | 047047 | | |
| 12170 | 060410 | 042105 | 020047 | 020075 | | |
| 12171 | 060416 | 020060 | 047111 | 051040 | | |
| 12172 | 060424 | 041510 | 031123 | 024040 | | |
| 12173 | 060432 | 044502 | 020124 | 030443 | | |
| 12174 | 060440 | 024462 | 005015 | 005015 | | |
| 12175 | 060446 | 047516 | 042524 | 020072 | .ASCII | /NOTE: ON DUAL PORT SYSTEM, A DRIVE ON OTHER PORT WILL /<15><12> |
| 12176 | 060454 | 047117 | 042040 | 040525 | | |
| 12177 | 060462 | 020114 | 047520 | 052122 | | |
| 12178 | 060470 | 051440 | 051531 | 042524 | | |
| 12179 | 060476 | 026115 | 040440 | 042040 | | |
| 12180 | 060504 | 044522 | 042526 | 047440 | | |
| 12181 | 060512 | 020116 | 052117 | 042510 | | |
| 12182 | 060520 | 020122 | 047520 | 052122 | | |
| 12183 | 060526 | 053440 | 046111 | 020114 | | |
| 12184 | 060534 | 005015 | | | | |
| 12185 | 060536 | 047516 | 020124 | 044507 | .ASCII | /NOT GIVE 'NED', BUT WILL GIVE RHAS RESPONSES/<15><12> |
| 12186 | 060544 | 042526 | 023440 | 042516 | | |
| 12187 | 060552 | 023504 | 020054 | 052502 | | |
| 12188 | 060560 | 020124 | 044527 | 046114 | | |
| 12189 | 060566 | 043440 | 053111 | 020105 | | |
| 12190 | 060574 | 044122 | 051501 | 051040 | | |
| 12191 | 060602 | 051505 | 047520 | 051516 | | |
| 12192 | 060610 | 051505 | 005015 | | | |
| 12193 | 060614 | 042510 | 041516 | 020105 | .ASCII | /HENCE THERE WILL BE AN EXTRA DRIVE/ |
| 12194 | 060622 | 044124 | 051105 | 020105 | | |
| 12195 | 060630 | 044527 | 046114 | 041040 | | |
| 12196 | 060636 | 020105 | 047101 | 042440 | | |
| 12197 | 060644 | 052130 | 040522 | 042040 | | |
| 12198 | 060652 | 044522 | 042526 | 000 | | |
| 12199 | 060657 | 114 | 047517 | 020113 | EM24: | .ASCII /LOOK AHEAD REGISTER AT THE BEGINNING OF SECTOR IS IN ERROR/ |

| | | | | | |
|-------|--------|--------|--------|--------|--|
| 12200 | 060664 | 044101 | 040505 | 020104 | |
| 12201 | 060672 | 042522 | 044507 | 052123 | |
| 12202 | 060700 | 051105 | 040440 | 020124 | |
| 12203 | 060706 | 044124 | 020105 | 042502 | |
| 12204 | 060714 | 044507 | 047116 | 047111 | |
| 12205 | 060722 | 020107 | 043117 | 051440 | |
| 12206 | 060730 | 041505 | 047524 | 020122 | |
| 12207 | 060736 | 051511 | 044440 | 020116 | |
| 12208 | 060744 | 051105 | 047522 | 000122 | |
| 12209 | 060752 | 047514 | 045517 | 040440 | EM25: .ASCIZ /LOOK AHEAD REGISTER IS IN ERROR/ |
| 12210 | 060760 | 042510 | 042101 | 051040 | |
| 12211 | 060766 | 043505 | 051511 | 042524 | |
| 12212 | 060774 | 020122 | 051511 | 044440 | |
| 12213 | 061002 | 020116 | 051105 | 047522 | |
| 12214 | 061010 | 000122 | | | |
| 12215 | 061012 | 052503 | 051122 | 047105 | EM30: .ASCII /CURRENT CYLINDER DOES NOT MATCH DESIRED CYLINDER REGISTER/<15><12> |
| 12216 | 061020 | 020124 | 054503 | 044514 | |
| 12217 | 061026 | 042116 | 051105 | 042040 | |
| 12218 | 061034 | 042517 | 020123 | 047516 | |
| 12219 | 061042 | 020124 | 040515 | 041524 | |
| 12220 | 061050 | 020110 | 042504 | 044523 | |
| 12221 | 061056 | 042522 | 020104 | 054503 | |
| 12222 | 061064 | 044514 | 042116 | 051105 | |
| 12223 | 061072 | 051040 | 043505 | 051511 | |
| 12224 | 061100 | 042524 | 006522 | 012 | |
| 12225 | 061105 | 101 | 052106 | 051105 | .ASCIZ /AFTER A SEEK AND INIT/ |
| 12226 | 061112 | 040440 | 051440 | 042505 | |
| 12227 | 061120 | 020113 | 047101 | 020104 | |
| 12228 | 061126 | 047111 | 052111 | 000 | |
| 12229 | 061133 | 105 | 041503 | 043440 | EM31: .ASCII /ECC GENERATED IS INCORRECT/<15><12> |
| 12230 | 061140 | 047105 | 051105 | 052101 | |
| 12231 | 061146 | 042105 | 044440 | 020123 | |
| 12232 | 061154 | 047111 | 047503 | 051122 | |
| 12233 | 061162 | 041505 | 006524 | 012 | |
| 12234 | 061167 | 105 | 042526 | 054522 | .ASCIZ /EVERY WORD ON THIS SECTOR IS THAT GIVEN IN "DATA USED"/ |
| 12235 | 061174 | 053440 | 051117 | 020104 | |
| 12236 | 061202 | 047117 | 052040 | 044510 | |
| 12237 | 061210 | 020123 | 042523 | 052103 | |
| 12238 | 061216 | 051117 | 044440 | 020123 | |
| 12239 | 061224 | 044124 | 052101 | 043440 | |
| 12240 | 061232 | 053111 | 047105 | 044440 | |
| 12241 | 061240 | 020116 | 042042 | 052101 | |
| 12242 | 061246 | 020101 | 051525 | 042105 | |
| 12243 | 061254 | 000042 | | | |
| 12244 | 061256 | 047117 | 051040 | 040505 | EM32: .ASCII /ON READ COMMAND, AFTER DATA AND ECC HAVE BEEN READ,/<15><12> |
| 12245 | 061264 | 020104 | 047503 | 046515 | |
| 12246 | 061272 | 047101 | 026104 | 040440 | |
| 12247 | 061300 | 052106 | 051105 | 042040 | |
| 12248 | 061306 | 052101 | 020101 | 047101 | |
| 12249 | 061314 | 020104 | 041505 | 020103 | |
| 12250 | 061322 | 040510 | 042526 | 041040 | |
| 12251 | 061330 | 042505 | 020116 | 042522 | |
| 12252 | 061336 | 042101 | 006454 | 012 | |
| 12253 | 061343 | 105 | 041503 | 051040 | .ASCII /ECC REGISTERS OR RHER1 ARE IN ERROR/<15><12> |
| 12254 | 061350 | 043505 | 051511 | 042524 | |
| 12255 | 061356 | 051522 | 047440 | 020122 | |

| | | | | |
|-------|--------|--------|--------|--------|
| 12256 | 061364 | 044122 | 051105 | 020061 |
| 12257 | 061372 | 051101 | 020105 | 047111 |
| 12258 | 061400 | 042440 | 051122 | 051117 |
| 12259 | 061406 | 005015 | | |
| 12260 | 061410 | 047117 | 054514 | 046040 |
| 12261 | 061416 | 053517 | 051105 | 030440 |
| 12262 | 061424 | 020061 | 044502 | 051524 |
| 12263 | 061432 | 047440 | 020106 | 040520 |
| 12264 | 061440 | 052124 | 051105 | 020116 |
| 12265 | 061446 | 042522 | 027107 | 041440 |
| 12266 | 061454 | 047101 | 041040 | 020105 |
| 12267 | 061462 | 042522 | 042101 | 005015 |
| 12268 | 061470 | 044124 | 051511 | 051440 |
| 12269 | 061476 | 047510 | 046125 | 020104 |
| 12270 | 061504 | 040515 | 041524 | 020110 |
| 12271 | 061512 | 047514 | 042527 | 020122 |
| 12272 | 061520 | 030461 | 041040 | 052111 |
| 12273 | 061526 | 020123 | 043117 | 043440 |
| 12274 | 061534 | 047517 | 020104 | 041505 |
| 12275 | 061542 | 030503 | 000 | |
| 12276 | 061545 | 110 | 043511 | 020110 |
| 12277 | 061552 | 047503 | 047125 | 020124 |
| 12278 | 061560 | 044502 | 020124 | 047516 |
| 12279 | 061566 | 020124 | 042523 | 020124 |
| 12280 | 061574 | 043101 | 042524 | 020122 |
| 12281 | 061602 | 034063 | 032470 | 020071 |
| 12282 | 061610 | 046103 | 041517 | 051513 |
| 12283 | 061616 | 000 | | |
| 12284 | 061617 | 132 | 051105 | 020117 |
| 12285 | 061624 | 042504 | 042524 | 052103 |
| 12286 | 061632 | 041040 | 052111 | 047040 |
| 12287 | 061640 | 052117 | 044040 | 043511 |
| 12288 | 061646 | 020110 | 044127 | 047105 |
| 12289 | 061654 | 031440 | 020062 | 044502 |
| 12290 | 061662 | 020124 | 041505 | 020103 |
| 12291 | 061670 | 042522 | 027107 | 044040 |
| 12292 | 061676 | 051501 | 031040 | 020061 |
| 12293 | 061704 | 042532 | 047522 | 000123 |
| 12294 | 061712 | 047520 | 044523 | 044524 |
| 12295 | 061720 | 047117 | 051040 | 043505 |
| 12296 | 061726 | 051511 | 042524 | 020122 |
| 12297 | 061734 | 051117 | 030440 | 020061 |
| 12298 | 061742 | 044502 | 051524 | 047440 |
| 12299 | 061750 | 020106 | 040520 | 052124 |
| 12300 | 061756 | 051105 | 020116 | 042522 |
| 12301 | 061764 | 044507 | 052123 | 051105 |
| 12302 | 061772 | 044440 | 041516 | 051117 |
| 12303 | 062000 | 042522 | 052103 | 005015 |
| 12304 | 062006 | 047514 | 042527 | 020122 |
| 12305 | 062014 | 030461 | 041040 | 052111 |
| 12306 | 062022 | 020123 | 043117 | 050040 |
| 12307 | 062030 | 052101 | 042524 | 047122 |
| 12308 | 062036 | 051040 | 043505 | 051511 |
| 12309 | 062044 | 042524 | 020122 | 044123 |
| 12310 | 062052 | 052517 | 042114 | 046440 |
| 12311 | 062060 | 052101 | 044103 | 046040 |

.ASCII /ONLY LOWER 11 BITS OF PATTERN REG. CAN BE READ/<15><12>

.ASCIZ /THIS SHOULD MATCH LOWER 11 BITS OF GOOD ECC1/

EM33: .ASCIZ /HIGH COUNT BIT NOT SET AFTER 38859 CLOCKS/

EM34: .ASCIZ /ZERO DETECT BIT NOT HIGH WHEN 32 BIT ECC REG. HAS 21 ZEROS/

EM35: .ASCII /POSITION REGISTER OR 11 BITS OF PATTERN REGISTER INCORRECT/<15><12>

.ASCII /LOWER 11 BITS OF PATTERN REGISTER SHOULD MATCH LOWER/<15><12>

| | | | | | |
|-------|--------|--------|--------|--------|---|
| 12312 | 062066 | 053517 | 051105 | 005015 | |
| 12313 | 062074 | 030461 | 041040 | 052111 | .ASCII /11 BITS OF GOOD ECC1/<15><12> |
| 12314 | 062102 | 020123 | 043117 | 043440 | |
| 12315 | 062110 | 047517 | 020104 | 041505 | |
| 12316 | 062116 | 030503 | 005015 | | |
| 12317 | 062122 | 040504 | 020124 | 047105 | .ASCIZ /DAT ENVLOP GOOD POSITION AND N-CODE ZEROS ARE IN OCTAL/ |
| 12318 | 062130 | 046126 | 050117 | 043440 | |
| 12319 | 062136 | 047517 | 020104 | 047520 | |
| 12320 | 062144 | 044523 | 044524 | 047117 | |
| 12321 | 062152 | 040440 | 042116 | 047040 | |
| 12322 | 062160 | 041455 | 042117 | 020105 | |
| 12323 | 062166 | 042532 | 047522 | 020123 | |
| 12324 | 062174 | 051101 | 020105 | 047111 | |
| 12325 | 062202 | 047440 | 052103 | 046101 | |
| 12326 | 062210 | 000 | | | |
| 12327 | 062211 | 117 | 020116 | 042522 | EM36: .ASCIZ /ON READ COMMAND WITH NON-CORRECTABLE ERROR 'DCK' AND 'ECH' SHOULD BE SE |
| 12328 | 062216 | 042101 | 041440 | 046517 | |
| 12329 | 062224 | 040515 | 042116 | 053440 | |
| 12330 | 062232 | 052111 | 020110 | 047516 | |
| 12331 | 062240 | 026516 | 047503 | 051122 | |
| 12332 | 062246 | 041505 | 040524 | 046102 | |
| 12333 | 062254 | 020105 | 051105 | 047522 | |
| 12334 | 062262 | 020122 | 042047 | 045503 | |
| 12335 | 062270 | 020047 | 047101 | 020104 | |
| 12336 | 062276 | 042447 | 044103 | 020047 | |
| 12337 | 062304 | 044123 | 052517 | 042114 | |
| 12338 | 062312 | 041040 | 020105 | 042523 | |
| 12339 | 062320 | 000124 | | | |
| 12340 | 062322 | 051120 | 043517 | 040522 | EM37: .ASCII /PROGRAM ERROR BIT #10 IN RHCS2 DID NOT SET/<15><12> |
| 12341 | 062330 | 020115 | 051105 | 047522 | |
| 12342 | 062336 | 020122 | 044502 | 020124 | |
| 12343 | 062344 | 030443 | 020060 | 047111 | |
| 12344 | 062352 | 051040 | 041510 | 031123 | |
| 12345 | 062360 | 042040 | 042111 | 047040 | |
| 12346 | 062366 | 052117 | 051440 | 052105 | |
| 12347 | 062374 | 005015 | | | |
| 12348 | 062376 | 043111 | 050040 | 051517 | .ASCIZ /IF POSITION REGISTER =10040 OR 10041, IT IS GOOD/ |
| 12349 | 062404 | 052111 | 047511 | 020116 | |
| 12350 | 062412 | 042522 | 044507 | 052123 | |
| 12351 | 062420 | 051105 | 036440 | 030061 | |
| 12352 | 062426 | 032060 | 020060 | 051117 | |
| 12353 | 062434 | 030440 | 030060 | 030464 | |
| 12354 | 062442 | 020054 | 052111 | 044440 | |
| 12355 | 062450 | 020123 | 047507 | 042117 | |
| 12356 | 062456 | 000 | | | |
| 12357 | | | | | |
| 12358 | 062457 | 122 | 053510 | 020103 | EM40: .ASCII /RHWC DID NOT = 0 UPON COMPLETION OF READ/<15><12> |
| 12359 | 062464 | 044504 | 020104 | 047516 | |
| 12360 | 062472 | 020124 | 020075 | 020060 | |
| 12361 | 062500 | 050125 | 047117 | 041440 | |
| 12362 | 062506 | 046517 | 046120 | 052105 | |
| 12363 | 062514 | 047511 | 020116 | 043117 | |
| 12364 | 062522 | 051040 | 040505 | 006504 | |
| 12365 | 062530 | 012 | | | |
| 12366 | 062531 | 117 | 020122 | 051127 | .ASCIZ /OR WRITE HEADER AND DATA/ |
| 12367 | 062536 | 052111 | 020105 | 042510 | |

| | | | | |
|-------|--------|--------|--------|--------|
| 12368 | 062544 | 042101 | 051105 | 040440 |
| 12369 | 062552 | 042116 | 042040 | 052101 |
| 12370 | 062560 | 000101 | | |

| | | | | | |
|-------|--------|--------|--------|--------|--|
| 12371 | | | | | |
| 12372 | 062562 | 040506 | 040524 | 020114 | CPHALT: .ASCII /FATAL ERROR - SEE DOCUMENT LISTING/<15><12> |
| 12373 | 062570 | 051105 | 047522 | 020122 | |
| 12374 | 062576 | 020055 | 042523 | 020105 | |
| 12375 | 062604 | 047504 | 052503 | 042515 | |
| 12376 | 062612 | 052116 | 046040 | 051511 | |
| 12377 | 062620 | 044524 | 043516 | 005015 | |
| 12378 | 062626 | 006440 | 103412 | 177777 | .ASCII / /<15><12><207><377><377><207><377><377><207><377><377> |
| 12379 | 062634 | 177607 | 103777 | 177777 | |
| 12380 | 062642 | 044124 | 020105 | 047503 | .ASCII /THE CONTROLLER OR DEVICE HAS GONE OFFLINE, LOST/<15><12> |
| 12381 | 062650 | 052116 | 047522 | 046114 | |
| 12382 | 062656 | 051105 | 047440 | 020122 | |
| 12383 | 062664 | 042504 | 044526 | 042503 | |
| 12384 | 062672 | 044040 | 051501 | 043440 | |
| 12385 | 062700 | 047117 | 020105 | 043117 | |
| 12386 | 062706 | 046106 | 047111 | 026105 | |
| 12387 | 062714 | 046040 | 051517 | 006524 | |
| 12388 | 062722 | 012 | | | |
| 12389 | 062723 | 047 | 042522 | 042101 | .ASCII /'READY', BECOME UNAVAILABLE, OR HAS STATUS BITS/<15><12> |
| 12390 | 062730 | 023531 | 020054 | 042502 | |
| 12391 | 062736 | 047503 | 042515 | 052440 | |
| 12392 | 062744 | 040516 | 040526 | 046111 | |
| 12393 | 062752 | 041101 | 042514 | 020054 | |
| 12394 | 062760 | 051117 | 044040 | 051501 | |
| 12395 | 062766 | 051440 | 040524 | 052524 | |
| 12396 | 062774 | 020123 | 044502 | 051524 | |
| 12397 | 063002 | 005015 | | | |
| 12398 | 063004 | 044127 | 041511 | 020110 | .ASCIZ /WHICH CANNOT BE CLEARED/ |
| 12399 | 063012 | 040503 | 047116 | 052117 | |
| 12400 | 063020 | 041040 | 020105 | 046103 | |
| 12401 | 063026 | 040505 | 042522 | 000104 | |
| 12402 | | | | | |
| 12403 | 063034 | 020040 | 020040 | 020040 | SPACE8: .ASCII / / / |
| 12404 | 063042 | 020040 | 000 | | SPACE2: .ASCIZ / / / |
| 12405 | | | | | |
| 12406 | | | | | |
| 12407 | 063045 | 120 | 020103 | 020040 | DH1: .ASCII /PC TEST REG. GOOD RECEIVED/<15><12> |
| 12408 | 063052 | 020040 | 052040 | 051505 | |
| 12409 | 063060 | 020124 | 020040 | 051040 | |
| 12410 | 063066 | 043505 | 020056 | 020040 | |
| 12411 | 063074 | 043440 | 047517 | 020104 | |
| 12412 | 063102 | 020040 | 051040 | 041505 | |
| 12413 | 063110 | 044505 | 042526 | 006504 | |
| 12414 | 063116 | 012 | | | |
| 12415 | 063117 | 040 | 020040 | 020040 | .ASCIZ / NO ADDR. DATA DATA / |
| 12416 | 063124 | 020040 | 047040 | 020117 | |
| 12417 | 063132 | 020040 | 020040 | 040440 | |
| 12418 | 063140 | 042104 | 027122 | 020040 | |
| 12419 | 063146 | 042040 | 052101 | 020101 | |
| 12420 | 063154 | 020040 | 042040 | 052101 | |
| 12421 | 063162 | 020101 | 020040 | 000040 | |
| 12422 | 063170 | 041520 | 020040 | 020040 | DH2: .ASCII /PC TEST WORD GOOD BAD /<15><12> |
| 12423 | 063176 | 020040 | 042524 | 052123 | |
| 12424 | 063204 | 020040 | 020040 | 047527 | |
| 12425 | 063212 | 042122 | 020040 | 020040 | |
| 12426 | 063220 | 047507 | 042117 | 020040 | |

| | | | | |
|-------|--------|--------|--------|--------|
| 12595 | 065106 | 027124 | 020040 | 041440 |
| 12596 | 065114 | 047117 | 027124 | 020040 |
| 12597 | 065122 | 041440 | 047117 | 027124 |
| 12598 | 065130 | 020040 | 041440 | 047117 |
| 12599 | 065136 | 027124 | 020040 | 006440 |
| 12600 | 065144 | 012 | | |
| 12601 | 065145 | 040 | 020040 | 020040 |
| 12602 | 065152 | 020040 | 047040 | 020117 |
| 12603 | 065160 | 020040 | 020040 | 045040 |
| 12604 | 065166 | 051123 | 020040 | 020040 |
| 12605 | 065174 | 051040 | 043505 | 040440 |
| 12606 | 065202 | 042104 | 051040 | 041510 |
| 12607 | 065210 | 030523 | 020040 | 051040 |
| 12608 | 065216 | 041510 | 031123 | 020040 |
| 12609 | 065224 | 051040 | 042110 | 030523 |
| 12610 | 065232 | 020040 | 051040 | 042510 |
| 12611 | 065240 | 030522 | 020040 | 000040 |
| 12612 | 065246 | 041520 | 020040 | 020040 |
| 12613 | 065254 | 020040 | 042524 | 052123 |
| 12614 | 065262 | 020040 | 020040 | 041520 |
| 12615 | 065270 | 047440 | 020106 | 020040 |
| 12616 | 065276 | 040506 | 046111 | 047111 |
| 12617 | 065304 | 020107 | 047507 | 042117 |
| 12618 | 065312 | 020040 | 020040 | 042522 |
| 12619 | 065320 | 042503 | 053111 | 042105 |
| 12620 | 065326 | 005015 | | |
| 12621 | 065330 | 020040 | 020040 | 020040 |
| 12622 | 065336 | 020040 | 047516 | 020040 |
| 12623 | 065344 | 020040 | 020040 | 051512 |
| 12624 | 065352 | 020122 | 020040 | 020040 |
| 12625 | 065360 | 042522 | 020107 | 020040 |
| 12626 | 065366 | 020040 | 040504 | 040524 |
| 12627 | 065374 | 020040 | 020040 | 040504 |
| 12628 | 065402 | 040524 | 020040 | 020040 |
| 12629 | 065410 | 000 | | |
| 12630 | 065411 | 120 | 020103 | 020040 |
| 12631 | 065416 | 020040 | 052040 | 051505 |
| 12632 | 065424 | 020124 | 020040 | 050040 |
| 12633 | 065432 | 020103 | 043117 | 020040 |
| 12634 | 065440 | 051040 | 043505 | 020056 |
| 12635 | 065446 | 020040 | 043440 | 047517 |
| 12636 | 065454 | 020104 | 020040 | 041040 |
| 12637 | 065462 | 042101 | 020040 | 020040 |
| 12638 | 065470 | 006440 | 012 | |
| 12639 | 065473 | 040 | 020040 | 020040 |
| 12640 | 065500 | 020040 | 047040 | 020117 |
| 12641 | 065506 | 020040 | 020040 | 045040 |
| 12642 | 065514 | 051123 | 020040 | 020040 |
| 12643 | 065522 | 040440 | 042104 | 042522 |
| 12644 | 065530 | 051523 | 042040 | 052101 |
| 12645 | 065536 | 020101 | 020040 | 042040 |
| 12646 | 065544 | 052101 | 020101 | 020040 |
| 12647 | 065552 | 000040 | | |
| 12648 | 065554 | 041520 | 020040 | 020040 |
| 12649 | 065562 | 020040 | 042524 | 052123 |
| 12650 | 065570 | 020040 | 020040 | 047527 |

.ASCIZ / NO JSR REG ADD RHCS1 RHCS2 RHDS1 RHER1 /

DH27: .ASCII /PC TEST PC OF FAILING GOOD RECEIVED/<15><12>

.ASCIZ / NO JSR REG DATA DATA /

DH30: .ASCII /PC TEST PC OF REG. GOOD BAD /<15><12>

.ASCIZ / NO JSR ADDRESS DATA DATA /

DH31: .ASCII /PC TEST WORD GOOD BAD CONT. CONT. CONT /<15><1

| | | | | | | | | | |
|-------|--------|--------|--------|--------|-------|-------|--|--|--|
| 12819 | | | | | | | | | |
| 12820 | 067466 | 001116 | 002032 | 001122 | DT11: | .WORD | \$ERRPC,TSTNM,\$BDADR,CS1,CS2,DS1,ER1,0 | | |
| 12821 | 067474 | 001714 | 001712 | 001736 | | | | | |
| 12822 | 067502 | 001716 | 000000 | | | | | | |
| 12823 | 067506 | 001116 | 002032 | 001122 | DT14: | .WORD | \$ERRPC,TSTNM,\$BDADR,\$BDDAT,CS1,CS2,DS1,ER1,0 | | |
| 12824 | 067514 | 001126 | 001714 | 001712 | | | | | |
| 12825 | 067522 | 001736 | 001716 | 000000 | | | | | |
| 12826 | 067530 | 001116 | 002032 | 001200 | DT15: | .WORD | \$ERRPC,TSTNM,\$TMP1,0 | | |
| 12827 | 067536 | 000000 | | | | | | | |
| 12828 | 067540 | 001116 | 002032 | 001204 | DT16: | .WORD | \$ERRPC,TSTNM,\$TMP3,\$TMP1,\$TMP0,\$BDDAT,0 | | |
| 12829 | 067546 | 001200 | 001176 | 001126 | | | | | |
| 12830 | 067554 | 000000 | | | | | | | |
| 12831 | 067556 | 001116 | 002032 | 001710 | DT17: | .WORD | \$ERRPC,TSTNM,BA,DB,WC,CS1,CS2,0 | | |
| 12832 | 067564 | 001704 | 001706 | 001714 | | | | | |
| 12833 | 067572 | 001712 | 000000 | | | | | | |
| 12834 | | | | | | | | | |
| 12835 | 067576 | 001116 | 002032 | 001716 | DT20: | .WORD | \$ERRPC,TSTNM,ER1,ER2,ER3,AS,DS1,0 | | |
| 12836 | 067604 | 001722 | 001730 | 001732 | | | | | |
| 12837 | 067612 | 001736 | 000000 | | | | | | |
| 12838 | 067616 | 001116 | 002032 | 001714 | DT21: | .WORD | \$ERRPC,TSTNM,CS1,AS,DS1,0 | | |
| 12839 | 067624 | 001732 | 001736 | 000000 | | | | | |
| 12840 | 067632 | 001116 | 002032 | 000000 | DT22: | .WORD | \$ERRPC,TSTNM,0 | | |
| 12841 | 067640 | 001116 | 002032 | 001720 | DT24: | .WORD | \$ERRPC,TSTNM,DST,\$BDDAT,\$TMP1,\$TMP2,\$TMP3,0 | | |
| 12842 | 067646 | 001126 | 001200 | 001202 | | | | | |
| 12843 | 067654 | 001204 | 000000 | | | | | | |
| 12844 | 067660 | 001116 | 002032 | 002014 | DT26: | .WORD | \$ERRPC,TSTNM,PCJSR,\$BDADR,CS1,CS2,DS1,ER1,0 | | |
| 12845 | 067666 | 001122 | 001714 | 001712 | | | | | |
| 12846 | 067674 | 001736 | 001716 | 000000 | | | | | |
| 12847 | 067702 | 001116 | 002032 | 002014 | DT27: | .WORD | \$ERRPC,TSTNM,PCJSR,REGADR,\$GDDAT,\$BDDAT,0 | | |
| 12848 | 067710 | 042402 | 001124 | 001126 | | | | | |
| 12849 | 067716 | 000000 | | | | | | | |
| 12850 | | | | | | | | | |
| 12851 | 067720 | 001116 | 002032 | 002014 | DT30: | .WORD | \$ERRPC,TSTNM,PCJSR,REGADR,\$GDDAT,\$BDDAT,0 | | |
| 12852 | 067726 | 042402 | 001124 | 001126 | | | | | |
| 12853 | 067734 | 000000 | | | | | | | |
| 12854 | 067736 | 001116 | 002032 | 047736 | DT31: | .WORD | \$ERRPC,TSTNM,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1,0 | | |
| 12855 | 067744 | 001124 | 001126 | 001714 | | | | | |
| 12856 | 067752 | 001736 | 001716 | 000000 | | | | | |
| 12857 | 067760 | 001116 | 002032 | 002014 | DT32: | .WORD | \$ERRPC,TSTNM,PCJSR,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1,0 | | |
| 12858 | 067766 | 047736 | 001124 | 001126 | | | | | |
| 12859 | 067774 | 001714 | 001736 | 001716 | | | | | |
| 12860 | 070002 | 000000 | | | | | | | |
| 12861 | 070004 | 001116 | 002032 | 002014 | DT33: | .WORD | \$ERRPC,TSTNM,PCJSR,ERWORD,\$GDDAT,CS1,DS1,ER1,0 | | |
| 12862 | 070012 | 047736 | 001124 | 001714 | | | | | |
| 12863 | 070020 | 001736 | 001716 | 000000 | | | | | |
| 12864 | 070026 | 001116 | 002032 | 045426 | DT34: | .WORD | \$ERRPC,TSTNM,GECC1,GECC2,WECC1,WECC2,DISK,0 | | |
| 12865 | 070034 | 045430 | 052534 | 052536 | | | | | |
| 12866 | 070042 | 051534 | 000000 | | | | | | |
| 12867 | 070046 | 001116 | 002032 | 045426 | DT35: | .WORD | \$ERRPC,TSTNM,GECC1,GECC2,EC2,EC1,POSIT1,ER1,0 | | |
| 12868 | 070054 | 045430 | 001746 | 001744 | | | | | |
| 12869 | 070062 | 045440 | 001716 | 000000 | | | | | |
| 12870 | 070070 | 001116 | 002032 | 002014 | DT36: | .WORD | \$ERRPC,TSTNM,PCJSR,MR,EC1,EC2,0 | | |
| 12871 | 070076 | 001734 | 001744 | 001746 | | | | | |
| 12872 | 070104 | 000000 | | | | | | | |
| 12873 | 070106 | 001116 | 002032 | 001744 | DT37: | .WORD | \$ERRPC,TSTNM,EC1,POSIT1,GECC1,GECC2,EC2,DATENV,ZCODE,0 | | |
| 12874 | 070114 | 045440 | 045426 | 045430 | | | | | |

| | | | | | | | |
|-------|--------|--------|--------|--------|-------|-------|-------------------------|
| 12875 | 070122 | 001746 | 045444 | 045446 | | | |
| 12876 | 070130 | 000000 | | | | | |
| 12877 | | | | | | | |
| 12878 | 070132 | 001116 | 002032 | 001126 | DT40: | .WORD | \$ERRPC,TSTNM,\$BDDAT,0 |
| 12879 | 070140 | 000000 | | | | | |
| 12880 | | | | | | | |
| 12881 | 070142 | 000 | 000 | 000 | DF1: | .BYTE | 0,0,0,0,0 |
| 12882 | 070145 | 000 | 000 | | | | |
| 12883 | 070147 | 000 | 000 | 001 | DF2: | .BYTE | 0,0,1,0 |
| 12884 | 070152 | 000 | | | | | |
| 12885 | 070153 | 000 | 000 | 001 | DF3: | .BYTE | 0,0,1,0,0 |
| 12886 | 070156 | 000 | 000 | | | | |
| 12887 | | | | | | | |
| 12888 | 070160 | 000 | 000 | 000 | DF11: | .BYTE | 0,0,0,0,0,0,0 |
| 12889 | 070163 | 000 | 000 | 000 | | | |
| 12890 | 070166 | 000 | | | | | |
| 12891 | 070167 | 000 | 000 | 000 | DF14: | .BYTE | 0,0,0,0,0,0,0,0 |
| 12892 | 070172 | 000 | 000 | 000 | | | |
| 12893 | 070175 | 000 | 000 | | | | |
| 12894 | 070177 | 000 | 000 | 000 | DF15: | .BYTE | 0,0,0 |
| 12895 | 070202 | 000 | 000 | 000 | DF16: | .BYTE | 0,0,0,0,0 |
| 12896 | 070205 | 000 | 000 | | | | |
| 12897 | 070207 | 000 | 000 | 000 | DF17: | .BYTE | 0,0,0,0,0,0,0 |
| 12898 | 070212 | 000 | 000 | 000 | | | |
| 12899 | 070215 | 000 | | | | | |
| 12900 | | | | | | | |
| 12901 | 070216 | 000 | 000 | 000 | DF20: | .BYTE | 0,0,0,0,0,0,0 |
| 12902 | 070221 | 000 | 000 | 000 | | | |
| 12903 | 070224 | 000 | | | | | |
| 12904 | 070225 | 000 | 000 | 000 | DF21: | .BYTE | 0,0,0,0,0 |
| 12905 | 070230 | 000 | 000 | | | | |
| 12906 | 070232 | 000 | 000 | 000 | DF22: | .BYTE | 0,0,0,0 |
| 12907 | 070235 | 000 | | | | | |
| 12908 | 070236 | 000 | 000 | 000 | DF24: | .BYTE | 0,0,0,0,0,0,0 |
| 12909 | 070241 | 000 | 000 | 000 | | | |
| 12910 | 070244 | 000 | | | | | |
| 12911 | 070245 | 000 | 000 | 000 | DF26: | .BYTE | 0,0,0,0,0,0,0,0 |
| 12912 | 070250 | 000 | 000 | 000 | | | |
| 12913 | 070253 | 000 | 000 | | | | |
| 12914 | 070255 | 000 | 000 | 000 | DF27: | .BYTE | 0,0,0,0,0,0 |
| 12915 | 070260 | 000 | 000 | 000 | | | |
| 12916 | | | | | | | |
| 12917 | 070263 | 000 | 000 | 000 | DF30: | .BYTE | 0,0,0,0,0,0 |
| 12918 | 070266 | 000 | 000 | 000 | | | |
| 12919 | 070271 | 000 | 000 | 001 | DF31: | .BYTE | 0,0,1,0,0,0,0,0 |
| 12920 | 070274 | 000 | 000 | 000 | | | |
| 12921 | 070277 | 000 | 000 | | | | |
| 12922 | 070301 | 000 | 000 | 000 | DF32: | .BYTE | 0,0,0,1,0,0,0,0,0 |
| 12923 | 070304 | 001 | 000 | 000 | | | |
| 12924 | 070307 | 000 | 000 | 000 | | | |
| 12925 | 070312 | 000 | 000 | 000 | DF33: | .BYTE | 0,0,0,1,0,0,0,0 |
| 12926 | 070315 | 001 | 000 | 000 | | | |
| 12927 | 070320 | 000 | 000 | | | | |
| 12928 | 070322 | 000 | 000 | 000 | DF34: | .BYTE | 0,0,0,0,0,0,0 |
| 12929 | 070325 | 000 | 000 | 000 | | | |
| 12930 | 070330 | 000 | | | | | |

CZRJGDO,RP04/5/6 DSKLS CTRLR1
CZRJGD.P11 10-SEP-79 11:00

MACY11 30A(1052) 10-SEP-79 11:11 PAGE 292
POWER DOWN AND UP ROUTINES

6 7

SEQ 0291

| | | | | | | | |
|-------|--------|--------|-----|-----|-------|-------|-------------------|
| 12931 | 070331 | 000 | 000 | 000 | DF35: | .BYTE | 0,0,0,0,0,0,0,0 |
| 12932 | 070334 | 000 | 000 | 000 | | | |
| 12933 | 070337 | 000 | 000 | | | | |
| 12934 | 070341 | 000 | 000 | 000 | DF36: | .BYTE | 0,0,0,0,0,0 |
| 12935 | 070344 | 000 | 000 | 000 | | | |
| 12936 | 070347 | 000 | 000 | 000 | DF37: | .BYTE | 0,0,0,0,0,0,0,0,0 |
| 12937 | 070352 | 000 | 000 | 000 | | | |
| 12938 | 070355 | 000 | 000 | 000 | | | |
| 12939 | | | | | | | |
| 12940 | 070360 | 000 | 000 | 000 | DF40: | .BYTE | 0,0,0 |
| 12941 | | 070364 | | | | .EVEN | |
| 12942 | | | | | | | |
| 12943 | | 000001 | | | .END | | |

| | | | | | | | | | | | | | | |
|----------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| EM16 | 060067 | 945 | 12131# | | | | | | | | | | | |
| EM17 | 060110 | 956 | 12134# | | | | | | | | | | | |
| EM2 | 057655 | 800 | 815 | 832 | 12106# | | | | | | | | | |
| EM20 | 060134 | 969 | 12138# | | | | | | | | | | | |
| EM21 | 060155 | 982 | 12141# | | | | | | | | | | | |
| EM22 | 060177 | 993 | 12145# | | | | | | | | | | | |
| EM24 | 060657 | 1015 | 12199# | | | | | | | | | | | |
| EM25 | 060752 | 1028 | 12209# | | | | | | | | | | | |
| EM30 | 061012 | 1071 | 12215# | | | | | | | | | | | |
| EM31 | 061133 | 1084 | 12229# | | | | | | | | | | | |
| EM32 | 061256 | 1101 | 12244# | | | | | | | | | | | |
| EM33 | 061545 | 1121 | 12276# | | | | | | | | | | | |
| EM34 | 061617 | 1134 | 12284# | | | | | | | | | | | |
| EM35 | 061712 | 1153 | 12294# | | | | | | | | | | | |
| EM36 | 062211 | 1176 | 12327# | | | | | | | | | | | |
| EM37 | 062322 | 1191 | 12340# | | | | | | | | | | | |
| EM40 | 062457 | 1203 | 12358# | | | | | | | | | | | |
| EM6 | 057704 | 856 | 874 | 12110# | | | | | | | | | | |
| ERCLFC | 020702 | 4107 | 4118 | 4129 | 4143 | 4159 | 4169 | 4179 | 4189 | 4199 | 4209 | 4219 | 4229 | 4238 |
| | | 4249 | 4258 | 4269 | 4279 | 4289 | 4306# | | | | | | | |
| ERCRC | 052774 | 10907* | 10956# | 11130* | | | | | | | | | | |
| ERCS2C | 013546 | 2812 | 2822 | 2834 | 2845 | 2855 | 2865 | 2875 | 2885 | 2895 | 2905 | 2914 | 2924 | 2935 |
| | | 2945 | 2957 | 2968 | 2979 | 2997# | | | | | | | | |
| ERFLG\$ | 002006 | 1559# | 2599* | 2768* | 3341* | 3355 | 5496* | 5517 | 5531* | 5609* | 5627 | 5646* | 5731* | 5751 |
| | | 5769* | 5850* | 5956* | 5979 | 6083* | 6106 | 6209* | 6232 | 6325* | 6335 | 6347 | 6418* | 6429 |
| | | 6452 | 6591* | 6697* | 6934* | 6945* | 6965 | 7053* | 7159* | 7217* | 7329* | 7352 | 7456* | 7464 |
| | | 7475 | 8136* | 8146 | 8158 | 8266* | 8276 | 8288 | 8397* | 8476* | 8491 | 8567* | 8672* | 8777* |
| | | 9397 | 9637* | 9648 | 10419 | 10990 | 11835* | | | | | | | |
| ERHDGP | 052776 | 10908* | 10964# | 11141* | | | | | | | | | | |
| ERHEAD | 052772 | 10906* | 10944# | 11118* | | | | | | | | | | |
| ERPOS | 046046 | 10059# | 10065* | 10082 | | | | | | | | | | |
| ERR = | 040000 | 1286# | 3790 | 6803 | 6823 | 6859 | 7566 | 8442 | 8589 | 8698 | 8811 | | | |
| ERRVEC = | 000004 | 622# | 1674 | 1675* | 1686* | 1778* | 1785* | 1820* | 11359 | 11360* | 11362* | 11365* | | |
| ERSTAR | 047430 | 10251# | | | | | | | | | | | | |
| ERUNIT | 047426 | 10250# | 10252 | | | | | | | | | | | |
| ERWORD | 047736 | 3061 | 3156 | 3899 | 3967 | 4035 | 4410 | 4464 | 4525 | 4575 | 4656 | 4702 | 4804 | 4850 |
| | | 4934 | 4980 | 5066 | 5111 | 5182 | 5414 | 6341* | 6346* | 6446* | 6451* | 6959* | 6964* | 7469* |
| | | 7474* | 7593 | 8152* | 8157* | 8282* | 8287* | 8469 | 8485* | 8490* | 8613 | 8722 | 8835 | 9390* |
| | | 9396* | 10487# | 10523* | 10543* | 10547* | 10569* | 10589* | 11068* | 11075* | 11079* | 11101* | 11109* | 11115* |
| | | 11129* | 11134* | 11138* | 11150* | 12815 | 12817 | 12854 | 12857 | 12861 | | | | |
| ER1 | 001716 | 1532# | 3769 | 3780 | 6867 | 7391 | 7494 | 12820 | 12823 | 12835 | 12844 | 12854 | 12857 | 12861 |
| | | 12867 | | | | | | | | | | | | |
| ER2 | 001722 | 1534# | 12835 | | | | | | | | | | | |
| ER3 | 001730 | 1537# | 12835 | | | | | | | | | | | |
| EXT1 = | 000001 | 1350# | | | | | | | | | | | | |
| EXT10 = | 000010 | 1353# | | | | | | | | | | | | |
| EXT2 = | 000002 | 1351# | | | | | | | | | | | | |
| EXT20 = | 000020 | 1354# | | | | | | | | | | | | |
| EXT4 = | 000004 | 1352# | | | | | | | | | | | | |
| EXT40 = | 000040 | 1355# | | | | | | | | | | | | |
| FEN = | 000200 | 1376# | | | | | | | | | | | | |
| FER = | 000020 | 1294# | 7391 | 7494 | | | | | | | | | | |
| FILLEC | 046222 | 10120# | | | | | | | | | | | | |
| FIRST | 002034 | 1588# | 1694 | 1735* | | | | | | | | | | |
| FMT22 = | 010000 | 1413# | 3870 | 5229 | 5465 | 5481 | 5497 | 5575 | 5591 | 5610 | 5696 | 5713 | 5732 | 5820 |
| | | 5836 | 5851 | 5924 | 5944 | 5985 | 6051 | 6071 | 6112 | 6177 | 6197 | 6238 | 6294 | 6318 |

| | | | | | | | | | | | | | | | | | | |
|---------|----------|--------|--------|-------|--------|--------|--------|--------|--------|--------|--------|-------|-------|-------|--|--|--|--|
| PRER2 | 011730 | 2487* | 2488# | | | | | | | | | | | | | | | |
| PRER3 | 012104 | 2558* | 2559# | | | | | | | | | | | | | | | |
| PROF | 011774 | 2512* | 2513# | | | | | | | | | | | | | | | |
| PROG = | 001000 | 1281# | 2932 | 3129 | 3789 | 4246 | 6802 | 6822 | 6858 | 8174 | 8304 | 9191 | | | | | | |
| PRWC | 011322 | 2331* | 2332# | | | | | | | | | | | | | | | |
| PRO | = 000000 | 556# | | | | | | | | | | | | | | | | |
| PR1 | = 000040 | 557# | | | | | | | | | | | | | | | | |
| PR2 | = 000100 | 558# | | | | | | | | | | | | | | | | |
| PR3 | = 000140 | 559# | | | | | | | | | | | | | | | | |
| PR4 | = 000200 | 560# | | | | | | | | | | | | | | | | |
| PR5 | = 000240 | 561# | | | | | | | | | | | | | | | | |
| PR6 | = 000300 | 562# | | | | | | | | | | | | | | | | |
| PR7 | = 000340 | 563# | | | | | | | | | | | | | | | | |
| PS | = 177776 | 536# | 537 | 1691* | 8856* | 8963* | | | | | | | | | | | | |
| PSEL | = 002000 | 1264# | | | | | | | | | | | | | | | | |
| PSU | = 000001 | 1443# | | | | | | | | | | | | | | | | |
| PSW | = 177776 | 537# | | | | | | | | | | | | | | | | |
| PUTREG | 042336 | 3195 | 3215 | 3260 | 3275 | 3336 | 3403 | 3442 | 3462 | 3768 | 3779 | 3816 | 5858 | 6334 | | | | |
| | | 6549 | 6680 | 6792 | 6812 | 6850 | 7170 | 7389 | 7463 | 8145 | 8275 | 8481 | 9018# | 9168 | | | | |
| | | 9220 | 9647 | 10016 | 10022 | 10090 | | | | | | | | | | | | |
| PWRVEC= | 000024 | 628# | 1665* | 1666* | 12047* | 12048* | 12057* | 12063* | 12075* | 12076* | | | | | | | | |
| P12 | 045454 | 9894# | 9948* | 9949* | 9978 | | | | | | | | | | | | | |
| P22 | 045456 | 9895# | 9956* | 9957* | 9983 | | | | | | | | | | | | | |
| P24 | 045460 | 9896# | 9964* | 9965* | 9984 | | | | | | | | | | | | | |
| P3 | 045452 | 9893# | 9939* | 9940* | 9977 | | | | | | | | | | | | | |
| RA | 000200 | 659# | 10227 | | | | | | | | | | | | | | | |
| RAW | = 000020 | 1391# | | | | | | | | | | | | | | | | |
| RCOM | 054136 | 11285# | 11290* | | | | | | | | | | | | | | | |
| RCYL | 047722 | 10454# | 10492* | 10527 | | | | | | | | | | | | | | |
| RDCHR = | 104410 | 11738 | 12037# | | | | | | | | | | | | | | | |
| RDHEAD | 047740 | 10377 | 10492# | | | | | | | | | | | | | | | |
| RDLIN = | 104411 | 11785 | 12038# | | | | | | | | | | | | | | | |
| RDOCT = | 104412 | 1757 | 8989 | 9000 | 10175 | 10197 | 12039# | | | | | | | | | | | |
| RDY | = 000200 | 1261# | 2288 | 2291 | 3119 | 3204 | 3216 | 3531 | 3533 | 3541 | 3543 | 3571 | 3573 | 3581 | | | | |
| | | 3583 | 3940 | 4008 | 7060 | 7062 | 7568 | 8188 | 8318 | 9084 | 9169 | 9179 | 9221 | | | | | |
| READ | 050532 | 10516 | 10520 | 10528 | 10530 | 10532 | 10534 | 10536 | 10577 | 10581 | 10616# | 11299 | 11307 | | | | | |
| READAT | 002070 | 1612# | 6404 | 6919 | 7037 | | | | | | | | | | | | | |
| READIN | 002104 | 1618# | 3855 | | | | | | | | | | | | | | | |
| RECALI | 002050 | 1604# | 5017 | | | | | | | | | | | | | | | |
| REDATA | 054140 | 10435 | 11289# | | | | | | | | | | | | | | | |
| REFOR | 002072 | 1613# | 5955 | 6082 | 6208 | 7328 | 8565 | | | | | | | | | | | |
| REGADR | 042402 | 1888* | 2292* | 2303* | 2312* | 2412* | 2434* | 2793* | 2997* | 3063* | 3158* | 3289* | 3296* | 3354* | | | | |
| | | 3413* | 3478* | 3486* | 3499* | 3534* | 3544* | 3574* | 3584* | 3622* | 3637* | 3664* | 3672* | 3697* | | | | |
| | | 3705* | 3901* | 3969* | 4037* | 4306* | 4412* | 4466* | 4527* | 4577* | 4658* | 4704* | 4806* | 4852* | | | | |
| | | 4936* | 4982* | 5068* | 5113* | 5184* | 5416* | 6815* | 7063* | 7073* | 7081* | 7595* | 8177* | 8307* | | | | |
| | | 8471* | 8615* | 8724* | 8837* | 9045# | 9049* | 9707* | 9718* | 9728* | 9739* | 9810* | 12813 | 12847 | | | | |
| | | 12851 | | | | | | | | | | | | | | | | |
| REGSA1 | 056560 | 1776 | 11836# | | | | | | | | | | | | | | | |
| REINTO | 003160 | 1626# | 3026 | 3033* | 3047 | 3054 | 3106 | 3115* | 3118* | 3119* | 3120* | 3121* | 3122* | 3123* | | | | |
| | | 3124* | 3125* | 3128 | 3132* | 3133* | 3134* | 3145 | 3149 | 3862 | 3869* | 3870* | 3872* | 3873* | | | | |
| | | 3886 | 3892 | 4371 | 4379* | 4380* | 4381* | 4397 | 4403 | 4430* | 4431* | 4432* | 4433* | 4434* | | | | |
| | | 4450 | 4456 | 4487 | 4495* | 4496* | 4497* | 4512 | 4519 | 4543* | 4544* | 4545* | 4546* | 4547* | | | | |
| | | 4561 | 4567 | 4607 | 4614* | 4615* | 4616* | 4643 | 4650 | 4673* | 4674* | 4675* | 4676* | 4690 | | | | |
| | | 4694 | 4767 | 4774* | 4775* | 4776* | 4790 | 4797 | 4821* | 4822* | 4823* | 4824* | 4825* | 4838 | | | | |
| | | 4842 | 4890 | 4906* | 4907* | 4908* | 4921 | 4927 | 4951* | 4952* | 4953* | 4954* | 4966 | 4971 | | | | |
| | | 5022 | 5038* | 5039* | 5040* | 5053 | 5059 | 5083* | 5084* | 5085* | 5086* | 5098 | 5103 | 5146 | | | | |

| | | | | | | | | | | | | | | |
|---------|--------|-------|-------|-------|-------|--------|--------|--------|-------|-------|--------|--------|--------|-------|
| | | 5155* | 5156* | 5169 | 5175 | 5236 | 5382* | 5383* | 5385* | 5386* | 5387* | 5401 | 5407 | 5528 |
| | | 5529 | 5536 | 5638 | 5639 | 5642 | 5643 | 5651 | 5762 | 5763 | 5766 | 5767 | 5774 | 5940 |
| | | 6001 | 6067 | 6128 | 6193 | 6254 | 6384 | 6385 | 6406 | 6444 | 6500 | 6506 | 6507 | 6511 |
| | | 6518 | 6519 | 6595 | 6635 | 6640 | 6647 | 6648 | 6720 | 6771 | 6785 | 6899 | 6900 | 6921 |
| | | 6957 | 7120 | 7125 | 7132 | 7133 | 7194 | 7208 | 7240 | 7313 | 7374 | 7551 | 7563* | 7564* |
| | | 7565* | 7566* | 7567* | 7568* | 7581 | 7587 | 7650 | 7651 | 7680 | 7731 | 7732 | 7761 | 7822 |
| | | 7823 | 7899 | 7900 | 7951 | 7952 | 7981 | 8029 | 8030 | 8403 | 8415* | 8416* | 8420* | 8421* |
| | | 8424* | 8425* | 8426* | 8427* | 8428* | 8436* | 8438* | 8440* | 8441* | 8442* | 8456 | 8462 | 8546 |
| | | 8574 | 8584* | 8585* | 8586* | 8587* | 8588* | 8589* | 8600 | 8606 | 8684 | 8694* | 8695* | 8696* |
| | | 8697* | 8698* | 8709 | 8715 | 8797 | 8807* | 8808* | 8809* | 8810* | 8811* | 8822 | 8828 | 9343 |
| | | 9443 | 9661 | 9686 | | | | | | | | | | |
| | | 1606# | 5141 | | | | | | | | | | | |
| RELEAS | 002054 | | | | | | | | | | | | | |
| RESVEC= | 000010 | | | | | | | | | | | | | |
| RETCL | 002100 | 1616# | 4885 | | | | | | | | | | | |
| RHAS | 001656 | 1504# | 1882 | 1922 | 6780 | 6849* | | | | | | | | |
| RHBA | 001634 | 1487# | 2355 | 2774* | 3091* | 3688 | 3850* | 4064* | 5479* | 5589* | 5711* | 5834* | 5940* | 6067* |
| | | 6193* | 6314* | 6406* | 6921* | 7039* | 7076 | 7081 | 7313* | 7447* | 7541* | 8125* | 8255* | 8376* |
| | | 8547* | 8652* | 8762* | 9343* | 9443* | 9632* | 9760* | | | | | | |
| RHBAE | 001700 | 1516# | 1806 | | | | | | | | | | | |
| RHCA | 001652 | 1502# | 2537 | 2781* | 3098* | 3853* | 4071* | 4356* | 4365* | 4482* | 4744* | 5228* | 5498* | 5611* |
| | | 5733* | 5852* | 5948* | 6075* | 6201* | 6319* | 6413* | 6928* | 7047* | 7321* | 7450* | 7531 | 7545* |
| | | 8130* | 8260* | 8387* | 8389* | 8569* | 8674* | 8783* | 8786* | 9350* | 9450* | 9624* | 9764* | 9796* |
| RHCC | 001676 | 1512# | 4341 | 4744 | 9024 | 9806 | 9810 | | | | | | | |
| RHCS1 | 001640 | 1497# | 2255 | 2622 | 2776* | 3021* | 3030* | 3093* | 3856* | 3866* | 4066* | 4340* | 4375* | 4481* |
| | | 4491* | 4601* | 4611* | 4761* | 4771* | 4885* | 4894* | 5017* | 5026* | 5141* | 5150* | 5255* | 5854* |
| | | 5856* | 6811* | 7538* | 7550 | 7560* | 7574 | 7595 | 8185 | 8315 | 8581* | 8691* | 8804* | 9082 |
| | | 9148 | 9757* | 9770* | 9798* | 9800* | 10168 | 10179 | 10182 | 10203 | 10360* | 10889* | | |
| RHCS2 | 001636 | 1488# | 1828 | 1862 | 1923 | 1982* | 2094* | 2230 | 2231* | 2256* | 2280* | 2332* | 2356* | 2380* |
| | | 2462* | 2488* | 2513* | 2538* | 2559* | 2775* | 3092* | 3112* | 3269 | 3455 | 3988* | 4065* | 5861 |
| | | 6789* | 6790* | 6933* | 7052* | 7533* | 7534* | 7543* | 9067* | 9086 | 9149 | 9762* | | |
| RHCS3 | 001702 | 1517# | | | | | | | | | | | | |
| RHDB | 001630 | 1485# | 1780 | 1826* | 2772* | 2788 | 3192 | 3267* | 3268* | 3282 | 3283 | 3289 | 3296 | 3330* |
| | | 3344 | 3354 | 3399* | 3409 | 3413 | 3448 | 3468 | 4062* | 4097 | 10176 | | | |
| RHDST | 001644 | 1499# | 2461 | 2778* | 3095* | 3851* | 4068* | 4339* | 5226* | 5251 | 5271 | 5342 | 5486* | 5600* |
| | | 5722* | 5841* | 5943* | 6070* | 6196* | 6317* | 6409* | 6924* | 7043* | 7316* | 7448* | 7542* | 8128* |
| | | 8258* | 8379* | 8428 | 8558* | 8663* | 8768* | 9346* | 9446* | 9629* | 9761* | 9797* | | |
| RHDS1 | 001662 | 1506# | 3797 | 3931 | 3999 | 8177 | 8307 | 9150 | | | | | | |
| RHDT | 001664 | 1507# | 1983 | 1985 | 1991 | 1993 | 1996 | 1998 | 2018 | 2096 | 2098 | 2124 | 2128 | 2130 |
| | | 2133 | 2135 | 2138 | 2140 | 2168 | 2170 | 2634 | 2636 | 2640 | 2642 | 2645 | 2647 | |
| RHEC1 | 001670 | 1509# | 10019 | | | | | | | | | | | |
| RHEC2 | 001672 | 1510# | 3927 | 3986 | 3995 | 10013 | | | | | | | | |
| RHER1 | 001642 | 1498# | 1926 | 2282* | 2379 | 2777* | 3094* | 4067* | 6773 | 8683 | 8702 | 8724 | 8796 | 8815 |
| | | 8837 | 9151 | 9703 | 9707 | 9713 | 9718 | 9725 | 9728 | 9733 | 9739 | | | |
| RHER2 | 001646 | 1500# | 2487 | 2779* | 3096* | 4069* | 6775 | | | | | | | |
| RHER3 | 001654 | 1503# | 2558 | 2782* | 3099* | 4072* | 6777 | | | | | | | |
| RHLA | 001674 | 1511# | 4433 | 4546 | 4676 | 4825 | 4954 | 5086 | 5156 | 5270 | 5340 | | | |
| RHMR | 001660 | 1505# | 2401 | 2783* | 2784* | 3020* | 3100* | 3101* | 3748* | 3765* | 3857* | 3923* | 3980* | 4073* |
| | | 4074* | 4094* | 4336* | 4424* | 4477* | 4537* | 4597* | 4668* | 4733* | 4738* | 4739* | 4750* | 4751* |
| | | 4757* | 4816* | 4876* | 4881* | 4882* | 4899* | 4900* | 4946* | 5007* | 5014* | 5015* | 5031* | 5032* |
| | | 5078* | 5137* | 5151* | 5201* | 5250 | 5429* | 5853* | 7559* | 7614* | 7695* | 7776* | 7853* | 8071* |
| | | 8203* | 8514* | 8538* | 8539* | 8540* | 8579 | 8644* | 8645* | 8646* | 8689 | 8754* | 8755* | 8756* |
| | | 8802 | 9769* | 9799* | 10067 | 10357* | 10358* | 10359* | 10498 | 10771 | 10886* | 10887* | 10888* | 11037 |
| | | 11207 | | | | | | | | | | | | |
| RHOF | 001650 | 1501# | 2512 | 2780* | 3097* | 3852* | 4070* | 4762* | 5229* | 5497* | 5610* | 5732* | 5851* | 5944* |
| | | 6071* | 6197* | 6318* | 6410* | 6925* | 7044* | 7317* | 7449* | 7544* | 8129* | 8259* | 8380* | 8568* |
| | | 8673* | 8778* | 9347* | 9447* | 9635* | 9763* | | | | | | | |

| | | | | | | | | | | | | | | | | | | | | |
|--------|--------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|--|--|--|--|--|--|
| TST17 | 011706 | 2481# | | | | | | | | | | | | | | | | | | |
| TST2 | 006152 | 1848# | | | | | | | | | | | | | | | | | | |
| TST20 | 011752 | 2507# | | | | | | | | | | | | | | | | | | |
| TST21 | 012016 | 2532# | | | | | | | | | | | | | | | | | | |
| TST22 | 012062 | 2553# | | | | | | | | | | | | | | | | | | |
| TST23 | 012126 | 2589# | | | | | | | | | | | | | | | | | | |
| TST24 | 012534 | 2765# | | | | | | | | | | | | | | | | | | |
| TST25 | 013560 | 2994 | 3014# | | | | | | | | | | | | | | | | | |
| TST26 | 013722 | 3082# | | | | | | | | | | | | | | | | | | |
| TST27 | 014272 | 3179# | | | | | | | | | | | | | | | | | | |
| TST3 | 006236 | 1857 | 1877# | | | | | | | | | | | | | | | | | |
| TST30 | 014450 | 3187 | 3226 | 3245# | | | | | | | | | | | | | | | | |
| TST31 | 014676 | 3253 | 3293 | 3311# | | | | | | | | | | | | | | | | |
| TST32 | 015136 | 3318 | 3385# | | | | | | | | | | | | | | | | | |
| TST33 | 015262 | 3392 | 3412 | 3427# | | | | | | | | | | | | | | | | |
| TST34 | 015562 | 3435 | 3494 | 3512# | | | | | | | | | | | | | | | | |
| TST35 | 015720 | 3520 | 3559# | | | | | | | | | | | | | | | | | |
| TST36 | 016040 | 3542 | 3549 | 3582 | 3597# | | | | | | | | | | | | | | | |
| TST37 | 016216 | 3605 | 3636 | 3650# | | | | | | | | | | | | | | | | |
| TST4 | 006310 | 1886 | 1896# | | | | | | | | | | | | | | | | | |
| TST40 | 016342 | 3670 | 3683# | | | | | | | | | | | | | | | | | |
| TST41 | 016466 | 3703 | 3743# | | | | | | | | | | | | | | | | | |
| TST42 | 016762 | 3841# | | | | | | | | | | | | | | | | | | |
| TST43 | 017206 | 3918# | | | | | | | | | | | | | | | | | | |
| TST44 | 017576 | 4055# | | | | | | | | | | | | | | | | | | |
| TST45 | 020730 | 4304 | 4331# | | | | | | | | | | | | | | | | | |
| TST46 | 021646 | 4592# | | | | | | | | | | | | | | | | | | |
| TST47 | 022322 | 4728# | | | | | | | | | | | | | | | | | | |
| TST5 | 007656 | 2042 | 2059# | 8880 | 8890 | | | | | | | | | | | | | | | |
| TST50 | 022730 | 4871# | | | | | | | | | | | | | | | | | | |
| TST51 | 023274 | 4714 | 5002# | | | | | | | | | | | | | | | | | |
| TST52 | 023640 | 5132# | | | | | | | | | | | | | | | | | | |
| TST53 | 024016 | 5197# | | | | | | | | | | | | | | | | | | |
| TST54 | 024052 | 5220# | | | | | | | | | | | | | | | | | | |
| TST55 | 024740 | 5425# | | | | | | | | | | | | | | | | | | |
| TST56 | 024774 | 5451# | | | | | | | | | | | | | | | | | | |
| TST57 | 025304 | 5518 | 5541 | 5562# | | | | | | | | | | | | | | | | |
| TST6 | 010430 | 2182# | | | | | | | | | | | | | | | | | | |
| TST60 | 025642 | 5628 | 5656 | 5682# | | | | | | | | | | | | | | | | |
| TST61 | 026202 | 5752 | 5779 | 5806# | | | | | | | | | | | | | | | | |
| TST62 | 026446 | 5899# | | | | | | | | | | | | | | | | | | |
| TST63 | 027032 | 5980 | 6005 | 6026# | | | | | | | | | | | | | | | | |
| TST64 | 027416 | 6107 | 6132 | 6152# | | | | | | | | | | | | | | | | |
| TST65 | 030002 | 6233 | 6258 | 6279# | | | | | | | | | | | | | | | | |
| TST66 | 030342 | 6336 | 6357 | 6374# | | | | | | | | | | | | | | | | |
| TST67 | 030706 | 6430 | 6471 | 6489# | | | | | | | | | | | | | | | | |
| TST7 | 010774 | 2186 | 2214# | | | | | | | | | | | | | | | | | |
| TST70 | 031236 | 6599 | 6621# | | | | | | | | | | | | | | | | | |
| TST71 | 031544 | 6725 | 6759# | | | | | | | | | | | | | | | | | |
| TST72 | 032134 | 6869 | 6889# | | | | | | | | | | | | | | | | | |
| TST73 | 032500 | 6984 | 7003# | | | | | | | | | | | | | | | | | |
| TST74 | 033044 | 7011 | 7105# | | | | | | | | | | | | | | | | | |
| TST75 | 033452 | 7245 | 7270# | | | | | | | | | | | | | | | | | |
| TST76 | 034054 | 7353 | 7392 | 7413# | | | | | | | | | | | | | | | | |
| TST77 | 034404 | 7495 | 7522# | | | | | | | | | | | | | | | | | |
| TINO = | 000122 | 1774# | 1851# | 1880# | 1900# | 2063# | 2183# | 2215# | 2228# | 2253# | 2275# | 2329# | 2353# | 2377# | | | | | | |

| | | | | | | | | | | | | | | |
|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| WKEY2 | 052762 | 5469* | 5579* | 5700* | 5824* | 10915# | | | | | | | | |
| WLE = | 004000 | 1301# | | | | | | | | | | | | |
| WORD | 050526 | 10515* | 10519* | 10527* | 10529* | 10531* | 10533* | 10535* | 10576* | 10580* | 10610# | 10620* | 10633* | 10650* |
| WRCHDA | 043770 | 11298* | 11306* | | | | | | | | | | | |
| WRCHDT | 002062 | 6671 | 7160 | 9425# | | | | | | | | | | |
| WRCHEK | 002060 | 1609# | 9355 | | | | | | | | | | | |
| WRCHHD | 043456 | 1608# | 9455 | | | | | | | | | | | |
| WRDATA | 051204 | 6541 | 9327# | | | | | | | | | | | |
| WRFROM | 002114 | 10429 | 10755# | 10993 | | | | | | | | | | |
| | | 1625# | 1833 | 3041 | 3047* | 3055 | 3139 | 3145* | 3150 | 3880 | 3886* | 3893 | 3947 | 3953* |
| | | 3961 | 4015 | 4021* | 4029 | 4390 | 4397* | 4404 | 4442 | 4450* | 4457 | 4505 | 4512* | 4520 |
| | | 4555 | 4561* | 4568 | 4637 | 4643* | 4651 | 4683 | 4690* | 4695 | 4784 | 4790* | 4798 | 4832 |
| | | 4838* | 4843 | 4915 | 4921* | 4928 | 4960 | 4966* | 4972 | 5047 | 5053* | 5060 | 5092 | 5098* |
| | | 5104 | 5163 | 5169* | 5176 | 5395 | 5401* | 5409 | 5478 | 5588 | 5709 | 5833 | 5984 | 6000 |
| | | 6111 | 6127 | 6237 | 6253 | 6309 | 6310 | 6314 | 6432 | 6433 | 6437 | 6438 | 6443 | 6567 |
| | | 6574 | 6575 | 6579 | 6586 | 6587 | 6594 | 6699 | 6705 | 6712 | 6713 | 6719 | 6947 | 6948 |
| | | 6952* | 6956 | 7219 | 7225 | 7232 | 7233 | 7239 | 7357 | 7373 | 7442 | 7443 | 7447 | 7541 |
| | | 7575 | 7581* | 7588 | 7656 | 7665 | 7666 | 7737 | 7746 | 7747 | 7813 | 7814 | 7837 | 7890 |
| | | 7891 | 7914 | 7957 | 7966 | 7967 | 8020 | 8021 | 8044 | 8120 | 8121 | 8125 | 8250 | 8251 |
| | | 8255 | 8371 | 8372 | 8376 | 8416 | 8421 | 8449 | 8456* | 8463 | 8594 | 8600* | 8607 | 8651 |
| | | 8703 | 8709* | 8716 | 8761 | 8816 | 8822* | 8829 | 9660 | 9760 | | | | |
| WRHEAD | 053050 | 10911 | 11030# | | | | | | | | | | | |
| WRIDAT | 002064 | 1610# | 6324 | 7455 | 8135 | 8265 | 8396 | 8775 | | | | | | |
| WRIFOR | 002066 | 1611# | 5494 | 5607 | 5729 | 5848 | 7538 | 8670 | 9757 | | | | | |
| WRITE | 050764 | 10544 | 10555 | 10684# | 10780 | 10786 | 10792 | 10796 | 10799 | 11076 | 11096 | 11110 | 11123 | 11135 |
| | | 11146 | | | | | | | | | | | | |
| WRL = | 004000 | 1283# | | | | | | | | | | | | |
| WRU = | 000400 | 1377# | 1395# | | | | | | | | | | | |
| WSECTR | 052756 | 5467* | 5577* | 5698* | 5822* | 10896 | 10913# | | | | | | | |
| WSSYNC | 051504 | 10834# | | | | | | | | | | | | |
| WSU = | 000004 | 1371# | 1389# | | | | | | | | | | | |
| WTRK | 052712 | 10898* | 10901# | | | | | | | | | | | |
| WWORD | 050762 | 10545 | 10549 | 10552 | 10556 | 10563 | 10568 | 10572 | 10679# | 10728* | 10781 | 10787 | 10793 | 10797 |
| | | 10800 | 11077 | 11081 | 11098 | 11111 | 11124 | 11136 | 11147 | | | | | |
| X | 047626 | 5931* | 6058* | 6184* | 6301* | 6396* | 6911* | 7029* | 7304* | 7434* | 8112* | 8242* | 8363* | 9333* |
| | | 9431* | 9633* | 9650 | 10386# | 10421 | | | | | | | | |
| | | 1937 | 1966# | | | | | | | | | | | |
| XE2 | 007304 | 10432# | | | | | | | | | | | | |
| Y | 047666 | 9885# | 10078* | 12873 | | | | | | | | | | |
| ZCODE | 045446 | 1316# | 10087 | | | | | | | | | | | |
| ZER = | 000400 | 10754# | 10774* | | | | | | | | | | | |
| ZWORDS | 051202 | 734# | 1744* | 11636 | 11765 | | | | | | | | | |
| %AUTOB | 001134 | 729# | 3198* | 3207* | 3219* | 3227* | 3263* | 3278* | 3339* | 3407* | 3445* | 3465* | 3771* | 3782* |
| %BDADR | 001122 | 3797* | 3821* | 5861* | 9175* | 9181* | 9185* | 9197* | 9202* | 9205* | 9224* | 9231* | 9237* | 9242* |
| | | 12820 | 12823 | 12844 | | | | | | | | | | |
| %BDAT | 001126 | 731# | 1884* | 1885 | 2287* | 2288 | 2300* | 2301 | 2307* | 2308 | 2411* | 2433* | 2684* | 2704 |
| | | 2706 | 2794* | 2795 | 2809* | 2810 | 2819* | 2820 | 2831* | 2832 | 2842* | 2843 | 2852* | 2853 |
| | | 2862* | 2863 | 2872* | 2873 | 2882* | 2883 | 2892* | 2893 | 2902* | 2903 | 2910* | 2912 | 2921* |
| | | 2922 | 2931* | 2942* | 2943 | 2954* | 2955 | 2965* | 2966 | 2976* | 2977 | 3288* | 3295* | 3352* |
| | | 3409* | 3411 | 3480* | 3488* | 3498* | 3530* | 3531 | 3540* | 3541 | 3569* | 3571 | 3580* | 3581 |
| | | 3616* | 3617 | 3633* | 3634 | 3660* | 3661 | 3668* | 3669 | 3693* | 3694 | 3701* | 3702 | 4104* |
| | | 4105 | 4115* | 4116 | 4126* | 4127 | 4140* | 4141 | 4155* | 4157 | 4166* | 4167 | 4176* | 4177 |
| | | 4186* | 4187 | 4196* | 4197 | 4206* | 4207 | 4216* | 4217 | 4226* | 4227 | 4235* | 4236 | 4245* |
| | | 4255* | 4256 | 4266* | 4267 | 4276* | 4277 | 4286* | 4287 | 5270* | 5272 | 5340* | 5343 | 5512* |
| | | 5622* | 5744* | 5969* | 6096* | 6222* | 6345* | 6450* | 6817* | 6963* | 7058* | 7060 | 7070* | 7071 |
| | | 7076* | 7078 | 7342* | 7473* | 8156* | 8179* | 8286* | 8309* | 8489* | 9075* | 9077* | 9079 | 9081* |

| | | | | | | | | | | | | | | |
|-----------|---------|--------|--------|--------|--------|--------|--------|--------|-------|-------|--------|-------|-------|-------|
| SLF | 001224 | 766# | 11538 | 11749 | 11758 | 11819 | 11864 | | | | | | | |
| SLPADR | 001106 | 722# | 1670* | 2061* | 8991* | 9002 | 11378* | 11393* | 11398 | 11400 | | | | |
| SLPERR | 001110 | 723# | 1671* | 3767* | 3978* | 6782* | 6847* | 8974 | 9001* | 11378 | 11394* | 11400 | 11858 | |
| \$MAIL = | ***** U | 1688 | 1740 | 11393 | 11491 | 11852 | | | | | | | | |
| \$MNEW | 056377 | 11647 | 11763# | | | | | | | | | | | |
| \$MSWR | 056366 | 11644 | 11761# | | | | | | | | | | | |
| \$MXCNT | 054622 | 11391 | 11400# | | | | | | | | | | | |
| \$NULL | 001154 | 743# | 11509 | 11538 | | | | | | | | | | |
| \$NWTST = | 000001 | 1765# | 1767 | 1841# | 1843 | 1874# | 1893# | 2047# | 2049 | 2174# | 2176 | 2208# | 2210 | 2244# |
| | | 2246 | 2266# | 2268 | 2320# | 2322 | 2344# | 2346 | 2368# | 2370 | 2391# | 2393 | 2449# | 2451 |
| | | 2475# | 2477 | 2501# | 2503 | 2526# | 2528 | 2547# | 2549 | 2581# | 2583 | 2754# | 2756 | 3005# |
| | | 3007 | 3074# | 3076 | 3169# | 3171 | 3231# | 3233 | 3303# | 3305 | 3379# | 3381 | 3419# | 3421 |
| | | 3506# | 3508 | 3553# | 3555 | 3591# | 3593 | 3644# | 3646 | 3677# | 3679 | 3736# | 3738 | 3832# |
| | | 3834 | 3910# | 3912 | 4045# | 4047 | 4322# | 4324 | 4584# | 4586 | 4719# | 4721 | 4862# | 4864 |
| | | 4993# | 4995 | 5123# | 5125 | 5194# | 5209# | 5211 | 5422# | 5441# | 5443 | 5552# | 5554 | 5673# |
| | | 5675 | 5797# | 5799 | 5891# | 5893 | 6018# | 6020 | 6144# | 6146 | 6270# | 6272 | 6365# | 6367 |
| | | 6480# | 6482 | 6611# | 6613 | 6745# | 6747 | 6879# | 6881 | 6993# | 6995 | 7093# | 7095 | 7261# |
| | | 7263 | 7404# | 7406 | 7504# | 7506 | 7607# | 7622# | 7624 | 7688# | 7703# | 7705 | 7769# | 7784# |
| | | 7786 | 7846# | 7861# | 7863 | 7922# | 7924 | 7990# | 7992 | 8064# | 8082# | 8084 | 8196# | 8212# |
| | | 8214 | 8322# | 8324 | 8507# | 8522# | 8524 | 8625# | 8627 | 8735# | 8737 | 8845# | 8847 | |
| \$OCNT | 057314 | 11952* | 11981* | 11994# | | | | | | | | | | |
| \$OMODE | 057316 | 11947* | 11951* | 11956 | 11959* | 11970* | 11996# | | | | | | | |
| \$OVER | 054606 | 11355 | 11371 | 11379 | 11389 | 11397# | | | | | | | | |
| \$PASS | 001100 | 718# | 4618 | 8875* | 8877 | 8910* | 8911* | 8919 | 8932 | 11385 | 11401 | | | |
| \$POWER | 057562 | 12078 | 12085# | | | | | | | | | | | |
| \$PWDRAD | 057550 | 12080# | | | | | | | | | | | | |
| \$PWDRN | 057410 | 1665 | 12047# | 12075 | | | | | | | | | | |
| \$PWDRMG | 057544 | 12078# | | | | | | | | | | | | |
| \$PWDRUP | 057462 | 12057 | 12063# | | | | | | | | | | | |
| \$QUES | 001222 | 764# | 11538 | 11698 | 11742 | 11758 | 11816 | 11819 | 11864 | | | | | |
| \$RDCHR | 056140 | 11711# | 12037 | | | | | | | | | | | |
| \$RDDEC = | ***** U | 12040 | | | | | | | | | | | | |
| \$RDLIN | 056230 | 11734# | 12038 | | | | | | | | | | | |
| \$RDOCT | 056410 | 11780# | 12039 | | | | | | | | | | | |
| \$RDSZ = | 000011 | 11727# | | | | | | | | | | | | |
| \$REGAD | 001160 | 747# | | | | | | | | | | | | |
| \$REG0 | 001162 | 749# | | | | | | | | | | | | |
| \$REG1 | 001164 | 750# | | | | | | | | | | | | |
| \$REG2 | 001166 | 751# | | | | | | | | | | | | |
| \$REG3 | 001170 | 752# | | | | | | | | | | | | |
| \$REG4 | 001172 | 753# | | | | | | | | | | | | |
| \$REG5 | 001174 | 754# | | | | | | | | | | | | |
| \$RTNAD | 041532 | 8931# | | | | | | | | | | | | |
| \$R2A = | ***** U | 12040 | | | | | | | | | | | | |
| \$SAVRE = | ***** U | 12040 | | | | | | | | | | | | |
| \$SAVR6 | 057560 | 12056* | 12064 | 12065* | 12066* | 12084# | | | | | | | | |
| \$SCOPE | 054332 | 1659 | 11341# | | | | | | | | | | | |
| \$SETUP = | 000117 | 1650# | 1658 | 1659 | 1661 | 1663 | 1665 | 1667 | 1668 | 1670 | 1737 | 8908 | 11342 | 11581 |
| | | 11586 | 11587 | 11617 | 11765 | 11834 | 11855 | 11863 | | | | | | |
| \$SS1 = | 000000 | 1689# | | | | | | | | | | | | |
| \$STUP = | 177777 | 1650# | | | | | | | | | | | | |
| \$SVLAD | 054560 | 11363 | 11392# | | | | | | | | | | | |
| \$SVPC = | 000200 | 649# | 654 | | | | | | | | | | | |
| \$SWR = | 167770 | 474# | 484 | 514 | 515 | 516 | 517 | 518 | 519 | 520 | 521 | 761 | 762 | 763 |
| | | 1667 | 1668 | 1670 | 1671 | 1772 | 1849 | 1878 | 1897 | 2060 | 2183 | 2215 | 2251 | 2273 |
| | | 2327 | 2351 | 2375 | 2398 | 2456 | 2482 | 2508 | 2533 | 2554 | 2590 | 2766 | 3015 | 3083 |

| | | | | | | | | | | | | | | |
|----------|--------|--------|--------|--------|--------|--------|--------|--------|--------|-------|--------|--------|--------|--------|
| | 3180 | 3246 | 3312 | 3386 | 3428 | 3513 | 3560 | 3598 | 3651 | 3684 | 3744 | 3842 | 3919 | |
| | 4056 | 4332 | 4593 | 4729 | 4872 | 5003 | 5133 | 5198 | 5221 | 5426 | 5452 | 5563 | 5683 | |
| | 5807 | 5900 | 6027 | 6153 | 6280 | 6375 | 6490 | 6622 | 6760 | 6890 | 7004 | 7106 | 7271 | |
| | 7414 | 7523 | 7611 | 7636 | 7692 | 7717 | 7773 | 7798 | 7850 | 7875 | 7936 | 8004 | 8068 | |
| | 8093 | 8200 | 8223 | 8338 | 8511 | 8533 | 8639 | 8749 | 8855 | 8903 | 8909 | 8924 | 8930 | |
| | 8932 | 11333 | 11334 | 11335 | 11336 | 11337 | 11354 | 11366 | 11368 | 11369 | 11372 | 11373 | 11374 | |
| | 11381 | 11382 | 11383 | 11394 | 11397 | 11400 | 11825 | 11826 | 11827 | 11828 | 11829 | 11840 | 11847 | |
| | 11852 | 11856 | 11864 | 12081 | | | | | | | | | | |
| \$SWRMK= | 000000 | 521 | 522 | 11337 | 11338 | 11370 | | | | | | | | |
| \$TIMES | 001212 | 761# | 1667* | 1772* | 1849* | 1878* | 1897* | 2060* | 8855* | 8909* | 11381* | 11388 | 11391* | 11400 |
| \$TKB | 001146 | 740# | 11541 | 11562 | 11573 | 11598 | 11626 | 11653 | | | | | | |
| \$TKCNT | 055270 | 11542# | 11557* | 11587 | 11604* | 11718 | 11720* | | | | | | | |
| \$TKINT | 055310 | 1693 | 1901 | 3111 | 9068 | 10174 | 11557# | 11578 | 11639 | | | | | |
| \$TKQEN= | 055307 | 11546# | 11612 | 11723 | | | | | | | | | | |
| \$TKQIN | 055272 | 11543# | 11558* | 11559 | 11610* | 11611* | 11612 | 11614* | | | | | | |
| \$TKQOU | 055274 | 11544# | 11559* | 11721 | 11722* | 11723 | 11725* | | | | | | | |
| \$TKQSR | 055276 | 11545# | 11558 | 11614 | 11725 | | | | | | | | | |
| \$TKS | 001144 | 739# | 11541 | 11563* | 11594* | 11596 | 11602* | 11624 | 11640* | 11650 | 11662* | 11682* | | |
| \$TKSRV | 055360 | 11560 | 11573# | | | | | | | | | | | |
| \$TMP0 | 001176 | 755# | 1978* | 1979* | 9261* | 9266 | 9271 | 9275 | 9385* | 9399 | 9498* | 9544* | 12828 | |
| \$TMP1 | 001200 | 756# | 1787* | 5247* | 5272 | 5339* | 5343 | 5357* | 5387 | 6784* | 6839* | 9262* | 9266 | 9271 |
| | | 9386* | 9401 | 12826 | 12828 | 12841 | | | | | | | | |
| \$TMP2 | 001202 | 757# | 5275* | 5276* | 5345* | 5346* | 9265* | 9268* | 9270* | 9273* | 9500* | 9542* | 12841 | |
| \$TMP3 | 001204 | 758# | 5279* | 5349* | 9259* | 9260* | 9499* | 9501 | 9502* | 12828 | 12841 | | | |
| \$TMP4 | 001206 | 759# | 5269* | 5278 | 5300* | 5329* | 5348 | 9501* | 9525* | | | | | |
| \$TMP5 | 001210 | 760# | 4353* | 4355* | 4356 | 4362* | 4364* | 4365 | 4434 | 9497* | 9503* | 9509 | 9622* | 9701 |
| | | 9723 | | | | | | | | | | | | |
| \$TN = | 000124 | 474# | 484 | 1752 | 1765 | 1772# | 1774 | 1841 | 1849# | 1851 | 1857 | 1874 | 1878# | 1880 |
| | | 1886 | 1893 | 1897# | 1900 | 2042 | 2047 | 2060# | 2063 | 2174 | 2183# | 2186 | 2208 | 2215# |
| | | 2221 | 2228 | 2244 | 2251# | 2253 | 2266 | 2273# | 2275 | 2320 | 2327# | 2329 | 2344 | 2351# |
| | | 2353 | 2368 | 2375# | 2377 | 2391 | 2398# | 2399 | 2449 | 2456# | 2459 | 2475 | 2482# | 2485 |
| | | 2501 | 2508# | 2510 | 2526 | 2533# | 2535 | 2547 | 2554# | 2556 | 2581 | 2590# | 2591 | 2754 |
| | | 2766# | 2993 | 3005 | 3015# | 3016 | 3074 | 3083# | 3084 | 3169 | 3180# | 3181 | 3187 | 3226 |
| | | 3231 | 3246# | 3247 | 3253 | 3293 | 3303 | 3312# | 3318 | 3379 | 3386# | 3392 | 3412 | 3419 |
| | | 3428# | 3429 | 3435 | 3494 | 3506 | 3513# | 3514 | 3520 | 3542 | 3549 | 3553 | 3560# | 3561 |
| | | 3582 | 3591 | 3598# | 3599 | 3605 | 3636 | 3644 | 3651# | 3652 | 3670 | 3677 | 3684# | 3686 |
| | | 3703 | 3736 | 3744# | 3746 | 3832 | 3842# | 3843 | 3910 | 3919# | 4045 | 4056# | 4057 | 4303 |
| | | 4322 | 4332# | 4333 | 4584 | 4593# | 4594 | 4714 | 4719 | 4729# | 4730 | 4862 | 4872# | 4873 |
| | | 4993 | 5003# | 5004 | 5123 | 5133# | 5134 | 5194 | 5198# | 5199 | 5209 | 5221# | 5222 | 5422 |
| | | 5426# | 5427 | 5441 | 5452# | 5453 | 5518 | 5541 | 5552 | 5563# | 5564 | 5628 | 5656 | 5673 |
| | | 5683# | 5684 | 5752 | 5779 | 5797 | 5807# | 5808 | 5891 | 5900# | 5901 | 5980 | 6005 | 6018 |
| | | 6027# | 6028 | 6107 | 6132 | 6144 | 6153# | 6154 | 6233 | 6258 | 6270 | 6280# | 6281 | 6285 |
| | | 6336 | 6357 | 6365 | 6375# | 6377 | 6430 | 6471 | 6480 | 6490# | 6497 | 6599 | 6611 | 6622# |
| | | 6630 | 6725 | 6745 | 6760# | 6762 | 6869 | 6879 | 6890# | 6892 | 6984 | 6993 | 7004# | 7005 |
| | | 7011 | 7093 | 7106# | 7114 | 7245 | 7261 | 7271# | 7275 | 7353 | 7392 | 7404 | 7414# | 7419 |
| | | 7495 | 7504 | 7523# | 7525 | 7607 | 7611# | 7612 | 7622 | 7636# | 7639 | 7688 | 7692# | 7693 |
| | | 7703 | 7717# | 7720 | 7769 | 7773# | 7774 | 7784 | 7798# | 7801 | 7846 | 7850# | 7851 | 7861 |
| | | 7875# | 7878 | 7922 | 7936# | 7939 | 7990 | 8004# | 8007 | 8064 | 8068# | 8069 | 8082 | 8093# |
| | | 8096 | 8196 | 8200# | 8201 | 8212 | 8223# | 8226 | 8322 | 8338# | 8339 | 8507 | 8511# | 8512 |
| | | 8522 | 8533# | 8534 | 8625 | 8639# | 8640 | 8735 | 8749# | 8750 | 8845 | 8855# | | |
| \$TPB | 001152 | 742# | 11527* | 11538 | | | | | | | | | | |
| \$TPFLG | 001157 | 746# | 11485 | 11538 | | | | | | | | | | |
| \$TPS | 001150 | 741# | 11525 | 11538 | | | | | | | | | | |
| \$TRAP | 057320 | 1663 | 12005# | | | | | | | | | | | |
| \$TRAP2 | 057342 | 12016# | 12027 | | | | | | | | | | | |
| \$TRP = | 000016 | 12020# | 12029# | 12030# | 12031# | 12032# | 12033# | 12034 | 12035# | 12036 | 12037# | 12038# | 12039# | 12040# |

CZRJGDO,RP04/5/6 DSKLS CTRLR1
CZRJGD.P11 10-SEP-79 11:00

MACY11 30A(1052) 10-SEP-79 11:11 PAGE 315
CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0312

| | | | | | | | | | | | | | | | |
|--------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| CHECKA | 509# | 5519 | 5629 | 5753 | 5981 | 6108 | 6234 | 7354 | | | | | | | |
| CHECKB | 509# | 4314 | 5239 | 5489 | 5602 | 5724 | 5843 | 5950 | 6077 | 6203 | 6320 | 6414 | 6764 | 6929 | 7048 |
| | 7323 | 7451 | 7527 | 8131 | 8261 | 8392 | 8560 | 8665 | 8770 | 9351 | 9451 | 9618 | | | |
| COMMEN | 1# | 485 | 634# | 2721 | | | | | | | | | | | |
| ENDCOM | 1# | 492 | 634# | 2746 | | | | | | | | | | | |
| ERROR | 528# | 1788 | 1866 | 1889 | 2235 | 2260 | 2293 | 2304 | 2313 | 2336 | 2360 | 2384 | 2413 | 2435 | 2466 |
| | 2492 | 2517 | 2542 | 2563 | 2678 | 2797 | 2998 | 3065 | 3159 | 3199 | 3208 | 3220 | 3228 | 3264 | 3279 |
| | 3290 | 3297 | 3340 | 3357 | 3359 | 3408 | 3414 | 3446 | 3466 | 3481 | 3489 | 3500 | 3535 | 3545 | 3575 |
| | 3585 | 3623 | 3638 | 3665 | 3673 | 3698 | 3706 | 3772 | 3783 | 3798 | 3822 | 3902 | 3970 | 4038 | 4307 |
| | 4414 | 4467 | 4528 | 4578 | 4660 | 4705 | 4807 | 4853 | 4937 | 4983 | 5069 | 5114 | 5185 | 5280 | 5350 |
| | 5417 | 5514 | 5542 | 5544 | 5624 | 5657 | 5659 | 5746 | 5780 | 5782 | 5862 | 5971 | 6006 | 6008 | 6098 |
| | 6133 | 6135 | 6224 | 6259 | 6261 | 6349 | 6352 | 6454 | 6456 | 6554 | 6560 | 6600 | 6602 | 6685 | 6690 |
| | 6727 | 6729 | 6796 | 6805 | 6818 | 6825 | 6853 | 6863 | 6870 | 6967 | 6969 | 7064 | 7074 | 7082 | 7175 |
| | 7180 | 7190 | 7197 | 7204 | 7211 | 7247 | 7249 | 7344 | 7379 | 7381 | 7393 | 7477 | 7483 | 7496 | 7596 |
| | 8160 | 8163 | 8180 | 8290 | 8293 | 8310 | 8472 | 8493 | 8495 | 8616 | 8725 | 8838 | 9176 | 9182 | 9186 |
| | 9198 | 9203 | 9206 | 9225 | 9232 | 9238 | 9243 | 9276 | 9666 | 9668 | 9692 | 9694 | 9709 | 9720 | 9730 |
| | 9740 | 9811 | 10017 | 10023 | 10091 | | | | | | | | | | |
| ESCAPE | 1# | 634# | | | | | | | | | | | | | |
| GETPRI | 1# | 634# | | | | | | | | | | | | | |
| GETSWR | 1# | 474# | 634# | 1737 | | | | | | | | | | | |
| HCOMPR | 509# | 7637 | 7718 | 7937 | | | | | | | | | | | |
| HCOMPW | 509# | 7799 | 7876 | 8005 | | | | | | | | | | | |
| MAKECL | 509# | 5194 | 5422 | 7607 | 7688 | 7769 | 7846 | 8064 | 8195 | 8507 | | | | | |
| MSG | 1764# | 1767 | 1840# | 1843 | 2046# | 2049 | 2174# | 2176 | 2208# | 2210 | 2243# | 2246 | 2265# | 2268 | 2319# |
| | 2322 | 2343# | 2346 | 2367# | 2370 | 2390# | 2393 | 2448# | 2451 | 2474# | 2477 | 2500# | 2503 | 2525# | 2528 |
| | 2546# | 2549 | 2580# | 2583 | 2752# | 2756 | 3004# | 3007 | 3073# | 3076 | 3168# | 3171 | 3230# | 3233 | 3302# |
| | 3305 | 3378# | 3381 | 3418# | 3421 | 3505# | 3508 | 3552# | 3555 | 3590# | 3593 | 3643# | 3646 | 3676# | 3679 |
| | 3735# | 3738 | 3831# | 3834 | 3909# | 3912 | 4044# | 4047 | 4321# | 4324 | 4583# | 4586 | 4718# | 4721 | 4861# |
| | 4864 | 4992# | 4995 | 5122# | 5125 | 5208# | 5211 | 5440# | 5443 | 5551# | 5554 | 5672# | 5675 | 5796# | 5799 |
| | 5890# | 5893 | 6017# | 6020 | 6143# | 6146 | 6269# | 6272 | 6364# | 6367 | 6477# | 6482 | 6610# | 6613 | 6743# |
| | 6747 | 6878# | 6881 | 6992# | 6995 | 7092# | 7095 | 7260# | 7263 | 7403# | 7406 | 7503# | 7506 | 8081# | 8084 |
| | 8211# | 8214 | 8320# | 8324 | 8521# | 8524 | 8624# | 8627 | 8734# | 8737 | 8845# | 8847 | | | |
| MSGA | 7622# | 7624 | 7705 | 7924 | | | | | | | | | | | |
| MSGB | 7783# | 7786 | 7863 | 7992 | | | | | | | | | | | |
| MULT | 1# | 634# | | | | | | | | | | | | | |
| NEWTST | 1# | 634# | 1765 | 1841 | 1874 | 1893 | 2047 | 2174 | 2208 | 2244 | 2266 | 2320 | 2344 | 2368 | 2391 |
| | 2449 | 2475 | 2501 | 2526 | 2547 | 2581 | 2754 | 3005 | 3074 | 3169 | 3231 | 3303 | 3379 | 3419 | 3506 |
| | 3553 | 3591 | 3644 | 3677 | 3736 | 3832 | 3910 | 4045 | 4322 | 4584 | 4719 | 4862 | 4993 | 5123 | 5194 |
| | 5209 | 5422 | 5441 | 5552 | 5673 | 5797 | 5891 | 6018 | 6144 | 6270 | 6365 | 6480 | 6611 | 6745 | 6879 |
| | 6993 | 7093 | 7261 | 7404 | 7504 | 7607 | 7622 | 7688 | 7703 | 7769 | 7784 | 7846 | 7861 | 7922 | 7990 |
| | 8064 | 8082 | 8196 | 8212 | 8321 | 8507 | 8522 | 8625 | 8735 | 8845 | | | | | |
| POP | 1# | 634# | 9028 | 9125 | 9313 | 9409 | 9547 | 9595 | 9773 | 9813 | 10042 | 10106 | 10146 | 10438 | 10592 |
| | 10662 | 10730 | 10804 | 10997 | 11155 | 11270 | 11310 | 11454 | 11808 | 12068 | 12069 | | | | |
| PUSH | 1# | 634# | 9018 | 9112 | 9303 | 9376 | 9489 | 9492 | 9567 | 9754 | 9791 | 9902 | 10066 | 10120 | 10350 |
| | 10497 | 10616 | 10684 | 10760 | 10879 | 11035 | 11201 | 11291 | 11413 | 11782 | 12049 | 12055 | | | |
| REPORT | 1# | 634# | | | | | | | | | | | | | |
| RFORGC | 509# | 5541 | 5656 | 5779 | 6005 | 6132 | 6258 | 6599 | 6725 | 7245 | | | | | |
| RH7OCK | 509# | 1852 | 2216 | 3182 | 3248 | 3313 | 3387 | 3430 | 3515 | 3600 | 7006 | | | | |
| SAVE | 509# | 11835 | | | | | | | | | | | | | |
| SAVTST | 509# | 1774 | 1851 | 1880 | 1900 | 2063 | 2183 | 2215 | 2228 | 2253 | 2275 | 2329 | 2353 | 2377 | 2399 |
| | 2459 | 2485 | 2510 | 2535 | 2556 | 2591 | 3016 | 3084 | 3181 | 3247 | 3312 | 3386 | 3429 | 3514 | 3561 |
| | 3599 | 3652 | 3686 | 3746 | 3843 | 3919 | 4057 | 4333 | 4594 | 4730 | 4873 | 5004 | 5134 | 5199 | 5222 |
| | 5427 | 5453 | 5564 | 5684 | 5808 | 5901 | 6028 | 6154 | 6281 | 6285 | 6377 | 6497 | 6630 | 6762 | 6892 |
| | 7005 | 7114 | 7275 | 7419 | 7525 | 7612 | 7639 | 7693 | 7720 | 7774 | 7801 | 7851 | 7878 | 7939 | 8007 |
| | 8069 | 8096 | 8201 | 8226 | 8339 | 8512 | 8534 | 8640 | 8750 | | | | | | |
| SCOPE | 529# | 1771 | 1848 | 1877 | 1896 | 2059 | 2182 | 2214 | 2250 | 2272 | 2326 | 2350 | 2374 | 2397 | 2455 |

| CZRJGDO,RP04/5/6 DSKLS CTRLR1 | | MACY11 30A(1052) 10-SEP-79 11:11 | | PAGE 316 | | | | | | | | | | | |
|-------------------------------|--------|--------------------------------------|-------|----------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| CZRJGD.P11 10-SEP-79 11:00 | | CROSS REFERENCE TABLE -- MACRO NAMES | | | | | | | | | | | | | |
| | | SEQ 0313 | | | | | | | | | | | | | |
| | 2481 | 2507 | 2532 | 2553 | 2589 | 2765 | 3014 | 3082 | 3179 | 3245 | 3311 | 3385 | 3427 | 3512 | 3559 |
| | 3597 | 3650 | 3683 | 3743 | 3841 | 3918 | 4055 | 4331 | 4592 | 4728 | 4871 | 5002 | 5132 | 5197 | 5220 |
| | 5425 | 5451 | 5562 | 5682 | 5806 | 5899 | 6026 | 6152 | 6279 | 6374 | 6489 | 6621 | 6759 | 6889 | 7003 |
| | 7105 | 7270 | 7413 | 7522 | 7610 | 7635 | 7691 | 7716 | 7772 | 7797 | 7849 | 7874 | 7935 | 8003 | 8067 |
| | 8092 | 8199 | 8222 | 8337 | 8510 | 8532 | 8638 | 8748 | 8854 | 8907 | | | | | |
| SETPRI | 1# | 634# | 11714 | | | | | | | | | | | | |
| SETTRA | 12020# | 12029 | 12030 | 12031 | 12032 | 12034 | 12036 | 12037 | 12038 | 12039 | 12040 | 12041 | 12042 | | |
| SETUP | 1# | 634# | 1651 | | | | | | | | | | | | |
| SKIP | 1# | 509# | 634# | 1752 | 1886 | 2042 | 2186 | 2993 | 3226 | 3293 | 3412 | 3494 | 3542 | 3582 | 3636 |
| | 3670 | 3703 | 4303 | 5518 | 5628 | 5752 | 5980 | 6107 | 6233 | 6336 | 6357 | 6430 | 6471 | 6869 | 6984 |
| | 7353 | 7392 | 7495 | | | | | | | | | | | | |
| SLASH | 1# | 634# | | | | | | | | | | | | | |
| SMORE | 509# | 11343 | | | | | | | | | | | | | |
| SPACE | 634# | | | | | | | | | | | | | | |
| STARS | 1# | 500 | 507 | 634# | 647 | 707 | 712 | 767 | 1211 | 1213 | 1251 | 1253 | 1765 | 1770 | 1841 |
| | 1847 | 1874 | 1876 | 1893 | 1895 | 1990 | 2000 | 2047 | 2058 | 2093 | 2104 | 2127 | 2165 | 2174 | 2181 |
| | 2208 | 2213 | 2244 | 2249 | 2266 | 2271 | 2320 | 2325 | 2344 | 2349 | 2368 | 2373 | 2391 | 2396 | 2449 |
| | 2454 | 2475 | 2480 | 2501 | 2506 | 2526 | 2531 | 2547 | 2552 | 2571 | 2577 | 2581 | 2588 | 2639 | 2649 |
| | 2733 | 2736 | 2754 | 2764 | 3005 | 3013 | 3074 | 3081 | 3169 | 3178 | 3231 | 3244 | 3303 | 3310 | 3379 |
| | 3384 | 3419 | 3426 | 3506 | 3511 | 3553 | 3558 | 3591 | 3596 | 3644 | 3649 | 3677 | 3682 | 3711 | 3721 |
| | 3728 | 3732 | 3736 | 3742 | 3832 | 3840 | 3910 | 3917 | 4045 | 4054 | 4322 | 4330 | 4584 | 4591 | 4719 |
| | 4727 | 4862 | 4870 | 4993 | 5001 | 5123 | 5131 | 5194 | 5196 | 5209 | 5219 | 5422 | 5424 | 5441 | 5450 |
| | 5552 | 5561 | 5673 | 5681 | 5797 | 5805 | 5866 | 5886 | 5891 | 5898 | 6018 | 6025 | 6144 | 6151 | 6270 |
| | 6278 | 6365 | 6373 | 6480 | 6488 | 6611 | 6620 | 6745 | 6758 | 6879 | 6888 | 6993 | 7002 | 7093 | 7104 |
| | 7261 | 7269 | 7404 | 7412 | 7504 | 7521 | 7607 | 7609 | 7622 | 7634 | 7688 | 7690 | 7703 | 7715 | 7769 |
| | 7771 | 7784 | 7796 | 7846 | 7848 | 7861 | 7873 | 7922 | 7934 | 7990 | 8002 | 8054 | 8060 | 8064 | 8066 |
| | 8082 | 8091 | 8196 | 8198 | 8212 | 8221 | 8322 | 8336 | 8507 | 8509 | 8522 | 8531 | 8625 | 8637 | 8735 |
| | 8747 | 8845 | 8853 | 8900 | 8938 | 8957 | 10260 | 10261 | 10320 | 10323 | 10447 | 10469 | 10476 | 10478 | 10603 |
| | 10605 | 10670 | 10672 | 10741 | 10744 | 10819 | 10825 | 10851 | 10853 | 11007 | 11009 | 11167 | 11169 | 11280 | 11282 |
| | 11329 | 11403 | 11470 | 11540 | 11617 | 11632 | 11703 | 11727 | 11768 | 11821 | 11866 | 11922 | 11999 | 12045 | 12061 |
| SWRSU | 1# | 634# | 1672# | | | | | | | | | | | | |
| TJUMP | 509# | 1857 | 2221 | 3187 | 3253 | 3318 | 3392 | 3435 | 3520 | 3549 | 3605 | 4714 | 7011 | | |
| TRMTRP | 12020# | | | | | | | | | | | | | | |
| TSCLR | 509# | 2807 | 2817 | 2840 | 2850 | 2860 | 2870 | 2880 | 2890 | 2900 | 2919 | 2963 | 2974 | | |
| TSCLR1 | 509# | 2940 | 2952 | | | | | | | | | | | | |
| TSCLR2 | 509# | 4102 | 4113 | 4124 | 4164 | 4174 | 4184 | 4194 | 4204 | 4214 | 4233 | 4274 | 4284 | | |
| TSCLR3 | 509# | 4253 | 4264 | | | | | | | | | | | | |
| TSCLR4 | 509# | 2827 | | | | | | | | | | | | | |
| TSCLR5 | 509# | 4136 | | | | | | | | | | | | | |
| TYPBIN | 1# | 634# | | | | | | | | | | | | | |
| TYPDEC | 1# | 634# | 8919 | 11906 | | | | | | | | | | | |
| TYPNAM | 1# | 634# | | | | | | | | | | | | | |
| TYPNUM | 1# | 634# | | | | | | | | | | | | | |
| TYPOCS | 1# | 634# | 2693 | 2706 | | | | | | | | | | | |
| TYPOCT | 1# | 634# | 2118 | 2124 | 11645 | 11878 | 11903 | | | | | | | | |
| TYPTXT | 1# | 634# | 1699 | 1705 | 1710 | 1714 | 1718 | 1722 | 1726 | 1730 | 1753 | 1792 | 1808 | 1814 | 1905 |
| | 1909 | 1913 | 1917 | 1944 | 1949 | 1953 | 1957 | 2007 | 2014 | 2020 | 2108 | 2114 | 2120 | 2148 | 2154 |
| | 2160 | 2187 | 2191 | 2195 | 2201 | 4623 | 8857 | 8863 | 8964 | 8970 | 8977 | 8981 | 8985 | 8992 | 8996 |
| | 10163 | 10170 | 10186 | 10193 | 10199 | 10205 | 10211 | 10215 | 10219 | 10223 | 10229 | 10234 | 11317 | | |
| WDATAR | 509# | 6283 | 8094 | 8224 | | | | | | | | | | | |
| XREAD | 509# | 5903 | 6030 | 6156 | 7276 | | | | | | | | | | |
| SSCMRE | 710# | 749 | 750 | 751 | 752 | 753 | 754 | | | | | | | | |
| SSCMTM | 710# | 755 | 756 | 757 | 758 | 759 | 760 | | | | | | | | |
| SSESCA | 1# | 634# | | | | | | | | | | | | | |
| SSNEWT | 1# | 634# | 1765 | 1841 | 1874 | 1893 | 2047 | 2174 | 2208 | 2244 | 2266 | 2320 | 2344 | 2368 | 2391 |
| | 2449 | 2475 | 2501 | 2526 | 2547 | 2581 | 2754 | 3005 | 3074 | 3169 | 3231 | 3303 | 3379 | 3419 | 3506 |

| | | | | | | | | | | | | | | | |
|----------|--------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|------|------|
| | 3553 | 3591 | 3644 | 3677 | 3736 | 3832 | 3910 | 4045 | 4322 | 4584 | 4719 | 4862 | 4993 | 5123 | 5194 |
| | 5209 | 5422 | 5441 | 5552 | 5673 | 5797 | 5891 | 6018 | 6144 | 6270 | 6365 | 6480 | 6611 | 6745 | 6879 |
| | 6993 | 7093 | 7261 | 7404 | 7504 | 7607 | 7622 | 7688 | 7703 | 7769 | 7784 | 7846 | 7861 | 7922 | 7990 |
| | 8064 | 8082 | 8196 | 8212 | 8322 | 8507 | 8522 | 8625 | 8735 | 8845 | | | | | |
| \$\$SET | 12020# | 12029 | 12030 | 12031 | 12032 | 12034 | 12036 | 12037 | 12038 | 12039 | 12040 | 12041 | 12042 | | |
| \$\$SKIP | 1# | 634# | | | | | | | | | | | | | |
| .EQUAT | 1# | 474# | 524 | | | | | | | | | | | | |
| .FLOAT | 509# | 2226 | 2251 | 2327 | 2351 | 2375 | 2457 | 2483 | 2508 | 2533 | 2554 | | | | |
| .HEADE | 1# | 474# | | | | | | | | | | | | | |
| .KT11 | 1# | 474# | 672 | | | | | | | | | | | | |
| .SETUP | 1# | 474# | 1649 | | | | | | | | | | | | |
| .SWRHI | 1# | 474# | 510 | | | | | | | | | | | | |
| .SWRLO | 474# | 522# | 523 | | | | | | | | | | | | |
| .\$ACT1 | 1# | 474# | 645 | | | | | | | | | | | | |
| .\$APT8 | 1# | | | | | | | | | | | | | | |
| .\$APTH | 1# | | | | | | | | | | | | | | |
| .\$APTY | 1# | | | | | | | | | | | | | | |
| .\$ASTA | 1# | | | | | | | | | | | | | | |
| .\$CATC | 1# | 474# | 635 | | | | | | | | | | | | |
| .\$CMTA | 1# | 474# | 710 | | | | | | | | | | | | |
| .\$DB2D | 1# | | | | | | | | | | | | | | |
| .\$DB2O | 1# | | | | | | | | | | | | | | |
| .\$DIV | 1# | | | | | | | | | | | | | | |
| .\$EOP | 1# | 474# | 8898 | | | | | | | | | | | | |
| .\$ERRO | 1# | 474# | 11819 | | | | | | | | | | | | |
| .\$ERRT | 1# | 474# | 11864 | | | | | | | | | | | | |
| .\$MULT | 1# | | | | | | | | | | | | | | |
| .\$POWE | 1# | 474# | 12043 | | | | | | | | | | | | |
| .\$RAND | 1# | | | | | | | | | | | | | | |
| .\$RDDE | 1# | | | | | | | | | | | | | | |
| .\$RDOC | 1# | 474# | 11766 | | | | | | | | | | | | |
| .\$READ | 1# | 474# | 11538 | | | | | | | | | | | | |
| .\$R2AZ | 1# | | | | | | | | | | | | | | |
| .\$SAVE | 1# | | | | | | | | | | | | | | |
| .\$SB2D | 1# | | | | | | | | | | | | | | |
| .\$SB2O | 1# | | | | | | | | | | | | | | |
| .\$SCOP | 1# | 474# | 11327 | | | | | | | | | | | | |
| .\$SIZE | 1# | | | | | | | | | | | | | | |
| .\$SUPR | 1# | | | | | | | | | | | | | | |
| .\$STRAP | 1# | 474# | 11997 | | | | | | | | | | | | |
| .\$TYPB | 1# | | | | | | | | | | | | | | |
| .\$TYPD | 1# | 474# | 11401 | | | | | | | | | | | | |
| .\$TYPE | 1# | 474# | 11468 | | | | | | | | | | | | |
| .\$TYPO | 1# | 474# | 11920 | | | | | | | | | | | | |
| .\$4OCA | 1# | | | | | | | | | | | | | | |
| .1170 | 1# | | | | | | | | | | | | | | |

. ABS. 070364 000

ERRORS DETECTED: 0

CZRJGD.BIN,CZRJGD.LST/CRF/SOL/NL:TOC:MC:MD:CND/LI:ME=CZRJGD.SML,CZRJGD.P11
 RUN-TIME: 86 126 9 SECONDS
 RUN-TIME RATIO: 416/223=1.8

CZRJGDO,RP04/5/6 DSKLS CTRLR1
CZRJGD.P11 10-SEP-79 11:00

MACY11 30A(1052) 10-SEP-79 11:11 PAGE 318
CROSS REFERENCE TABLE -- MACRO NAMES

E 9

SEQ 0315

CORE USED: 38K (75 PAGES)