

RP04/5/6

RP04 5/6 DSKLS 2
CZRJHD0

AH 9215D MC

COPYRIGHT 76 79

FICHE 1 OF 2

NOV 1979

digital

MADE IN USA

RP04/5/6

RP04/5/6 DSKLS 2
CZRJHD0

AH 9215D MC

NOV 1979

COPYRIGHT 76 79

digital

FICHE 2 OF 2

MADE IN USA

CZRJHDO,RP04/5/6 DSKLS CTRLR2
CZRJHD.P'1 28-MAR-79 09:16

MACV11 30A(1052) 24-MAY-79 15:07 PAGE 2

8 1

SEQ 0001

.REM 2

IDENTIFICATION

PRODUCT CODE: AC-9213D-MC
PRODUCT NAME: CZRJHDO RP04/5/6 DISKLESS CONTROLLER TEST-PART II
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
DEC

PDP
DECUS

UNIBUS
DECTAPE

MASSBUS

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. PROGRAM DESCRIPTION

1.0 ABSTRACT

THIS DIAGNOSTIC TESTS THE RH11 AND DCL OF AN RP04/5/6 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, THAT 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 TO TEST RP04/5/6 DISK DRIVES CONNECTED TO THAT TYPE OF CONTROLLER. IT IS ASSUMED THAT THE RH70 SPECIFIC CONTROLLER DIAGNOSTICS HAVE BEEN SUCCESSFULLY RUN TO COMPLETION BEFORE THIS PROGRAM IS RUN.

2.0 REQUIREMENTS

2.1 EQUIPMENT

PDP-11 COMPUTER WITH CONSOLE TELETYPE, AND AN 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 PROGRAM ASSUMES THAT MAINDEC-11-DZRJG- (LATEST REV) HAS BEEN RUN WITHOUT ERRORS

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 FRONT PANEL, OR IN THE SOFTWARE SWITCH REGISTER IF THE OPERATOR SO DESIRES. SEE 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 PARAMETERS
START AT ADDRESS 210---FOR UNIT SELECTION

200 START

ALL SWITCHES MUST BE DOWN FOR WORST CASE RUN. WITH THIS STARTING ADDRESS ALL THE RP04/5/6'S 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 200 START WITH FOLLOWING EXCEPTIONS: PROGRAM WILL QUERY OPERATOR FOR THE CORRECT CSR AND VECTOR ADDRESS OF THE RHXX CONTROLLER. WHEN THIS ACTION HAS BEEN COMPLETED, THE PROGRAM WILL AUTOMATICALLY RESTART FROM ADDRESS 200, WITH THE SAME CONVENTIONS AS DESCRIBED FOR A 200 START.

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 PASSED 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) THE PROGRAM WILL DETERMINE THAT THE HARDWARE SWITCH

REGISTER IS NOT PRESENT AND WILL USE AN '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 THE 'CONTROL G' AT ANY TIME EXCEPT WHEN THE PROGRAM IS AT A HIGHER PRIORITY PROCESSING AN RP04/5/6 INTERRUPT. THE 'SOFTWARE' SWITCH VALUES ARE ENTERED AS AN OCTAL NUMBER IN RESPONSE TO THE 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 REGISTER, THE 'SOFTWARE' SWITCH REGISTER MAY BE USED. IF THE PROGRAM FINDS ALL 16 SWITCHES IN THE 'UP' POSITION, 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 DESCRIPTIONS 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 CONTROLLER SELECT
THIS SWITCH 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 MEANING 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 SUB-SEQUENT 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 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, THERE ARE TWO OPTIONS FOR THE OPERATOR:

1. 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.

2. GO BACK AND RERUN DZRPS AS IT IS QUITE POSSIBLE THAT A HARD FAILURE HAS OCCURRED IN ONE OF THE HARDWARE REGISTERS.

IT IS ALSO POSSIBLE TO CONTINUE FROM THE HALT POINT, BUT THIS 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 THIS FACT, THE PROGRAM CANNOT BE RUN IN CHAIN MODE WHEN USING THE SOFTWARE SWITCH REGISTER AS THE ROUTINE WHICH ASKS FOR THE SWITCH REGISTER SETTINGS IS NOT OPERABLE WHEN IN CHAIN MODE.

8.0 MISCELLANEOUS

8.1 EXECUTION TIME

THE FIRST PASS OF THE PROGRAM WILL TAKE 1.75 MINUTES PER DRIVE. SUBSEQUENT PASSES WILL TAKE 7 MINUTE.

8.2 STACK POINTER

THE STACK IS INITIALLY SET TO 1000

8.3 OPERATOR SELECTABLE SCOPE LOOPS

HERE IS A DETAILED EXPLANATION OF HOW THE LOOP ON ERROR WORKS. FOR INSTRUCTIONS REGARDING USAGE OF THESE LOOPS, HIT CONTROL C ANY TIME WHILE THE PROGRAM IS RUNNING. ON HITTING AN ERROR IF THE LOOP ON ERROR SWITCH IS SET, THE PROGRAM GOES BACK - USUALLY BACK TO THE BEGINNING OF THE TEST.

WHEN THIS OPERATOR SELECTABLE SCOPE LOOP IS USED THEN THE POINT
THE PROGRAM GOES BACK TO CAN BE CHANGED.
THE RESTRICTIONS TO THE POINT WHERE THE PROGRAM CAN GO ARE: -
1. IT MUST BE WITHIN THE TEST UNDER CONSIDERATION
2. LOOP ON ERROR SWITCH MUST BE SET
3. THE ERROR MUST OCCUR WITHIN THE TEST UNDER CONSIDERATION
IF THE ERROR DOES NOT OCCUR WITHIN THE TEST UNDER CONSIDERATION
THE PROGRAM WILL REVERT TO NORMAL OPERATION. HOWEVER, IF LOOP ON
TEST SWITCH IS SET AND THIS OPERATOR SELECTABLE SCOPE LOOP IS USED
THEN THE PROGRAM WILL LOOP BACK TO THE SELECTED POINT WHEN IT
COMES TO THE END OF THE TEST UNDER CONSIDERATION.

AFTER LOOPING FOR SOME TIME IF THE LOOP SWITCH IS PUT DOWN THEN
NORMAL OPERATION WILL CONTINUE.

8.4 PROGRAM REVISION HISTORY

9.0 PROGRAM DESCRIPTION

THE FOLLOWING SECTIONS DESCRIBE EACH TEST AND SUBROUTINES
IN DETAIL AND CAN ALSO BE USED AS AN INDEX TO THE LISTING.
THE LEFT MOST COLUMN IS THE LINE NUMBER WITHIN THE LISTING
WHERE THAT ITEM WILL BE FOUND.

@

474
475
476
477
478
479
480
481
482
483
484
485

; /★\:/★\:/★\:/★\:/★\:/★\:/★\:/★\:/★\:/★\:/★\:/★\:/★\:/★\:/★\:/★\:
;/★\:/★\:/★\:/★\:/★\:/★\:/★\:/★\:/★\:/★\:/★\:/★\:/★\:/★\:/★\:/★\:
:/★\:/★\:/★\:/★\:/★\:/★\:/★\:/★\:/★\:/★\:/★\:/★\:/★\:/★\:/★\:

;

```

:DRIVE MUST BE LOCKED ON PORT A OR PORT B

```

[illegible]

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
524
525
526
527

```
;INTERNAL PROGRAM MACROS BEGIN HERE
:*****
```

```

;*****
;
; *NOTE: ALL MACRO CALLS BEGINNING WITH '$' ARE SUPPLIED FROM AN
; *      EXTERNAL SYSMAC.SML 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
```

```

528      .SBTTL  BASIC DEFINITIONS
529
530      ;*INITIAL ADDRESS OF THE STACK POINTER *** 1000 ***
531      001000  STACK= 1000
532      .EQUIV  EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
533      .EQUIV  IOT,SCOPE      ;;BASIC DEFINITION OF SCOPE CALL
534
535      ;*MISCELLANEOUS DEFINITIONS
536      000011  HT= 11          ;;CODE FOR HORIZONTAL TAB
537      000012  LF= 12          ;;CODE FOR LINE FEED
538      000015  CR= 15          ;;CODE FOR CARRIAGE RETURN
539      000200  CRLF= 200       ;;CODE FOR CARRIAGE RETURN-LINE FEED
540      177776  PS= 177776      ;;PROCESSOR STATUS WORD
541      .EQUIV  PS,PSW
542      177774  STKLM= 177774   ;;STACK LIMIT REGISTER
543      177772  PIRQ= 177772    ;;PROGRAM INTERRUPT REQUEST REGISTER
544      177570  DSWR= 177570    ;;HARDWARE SWITCH REGISTER
545      177570  DDISP= 177570   ;;HARDWARE DISPLAY REGISTER
546
547      ;*GENERAL PURPOSE REGISTER DEFINITIONS
548      000000  R0= %0          ;;GENERAL REGISTER
549      000001  R1= %1          ;;GENERAL REGISTER
550      000002  R2= %2          ;;GENERAL REGISTER
551      000003  R3= %3          ;;GENERAL REGISTER
552      000004  R4= %4          ;;GENERAL REGISTER
553      000005  R5= %5          ;;GENERAL REGISTER
554      000006  R6= %6          ;;GENERAL REGISTER
555      000007  R7= %7          ;;GENERAL REGISTER
556      000006  SP= %6          ;;STACK POINTER
557      000007  PC= %7          ;;PROGRAM COUNTER
558
559      ;*PRIORITY LEVEL DEFINITIONS
560      000000  PR0= 0          ;;PRIORITY LEVEL 0
561      000040  PR1= 40         ;;PRIORITY LEVEL 1
562      000100  PR2= 100        ;;PRIORITY LEVEL 2
563      000140  PR3= 140        ;;PRIORITY LEVEL 3
564      000200  PR4= 200        ;;PRIORITY LEVEL 4
565      000240  PR5= 240        ;;PRIORITY LEVEL 5
566      000300  PR6= 300        ;;PRIORITY LEVEL 6
567      000340  PR7= 340        ;;PRIORITY LEVEL 7
568
569      ;*'SWITCH REGISTER' SWITCH DEFINITIONS
570      100000  SW15= 100000
571      040000  SW14= 40000
572      020000  SW13= 20000
573      010000  SW12= 10000
574      004000  SW11= 4000
575      002000  SW10= 2000
576      001000  SW09= 1000
577      000400  SW08= 400
578      000200  SW07= 200
579      000100  SW06= 100
580      000040  SW05= 40
581      000020  SW04= 20
582      000010  SW03= 10
583      000004  SW02= 4

```

```
584      000002      SW01= 2
585      000001      SW00= 1
586      .EQUIV      SW09,SW9
587      .EQUIV      SW08,SW8
588      .EQUIV      SW07,SW7
589      .EQUIV      SW06,SW6
590      .EQUIV      SW05,SW5
591      .EQUIV      SW04,SW4
592      .EQUIV      SW03,SW3
593      .EQUIV      SW02,SW2
594      .EQUIV      SW01,SW1
595      .EQUIV      SW00,SW0
596
597      ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
598      100000      BIT15= 100000
599      040000      BIT14= 40000
600      020000      BIT13= 20000
601      010000      BIT12= 10000
602      004000      BIT11= 4000
603      002000      BIT10= 2000
604      001000      BIT09= 1000
605      000400      BIT08= 400
606      000200      BIT07= 200
607      000100      BIT06= 100
608      000040      BIT05= 40
609      000020      BIT04= 20
610      000010      BIT03= 10
611      000004      BIT02= 4
612      000002      BIT01= 2
613      000001      BIT00= 1
614      .EQUIV      BIT09,BIT9
615      .EQUIV      BIT08,BIT8
616      .EQUIV      BIT07,BIT7
617      .EQUIV      BIT06,BIT6
618      .EQUIV      BIT05,BIT5
619      .EQUIV      BIT04,BIT4
620      .EQUIV      BIT03,BIT3
621      .EQUIV      BIT02,BIT2
622      .EQUIV      BIT01,BIT1
623      .EQUIV      BIT00,BIT0
624
625      ;*BASIC "CPU" TRAP VECTOR ADDRESSES
626      000004      ERRVEC= 4      ;;TIME OUT AND OTHER ERRORS
627      000010      RESVEC= 10     ;;RESERVED AND ILLEGAL INSTRUCTIONS
628      000014      TBITVEC=14     ;;"T" BIT
629      000014      TRTVEC= 14     ;;TRACE TRAP
630      000014      BPTVEC= 14     ;;BREAKPOINT TRAP (BPT)
631      000020      IOTVEC= 20     ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
632      000024      PWRVEC= 24     ;;POWER FAIL
633      0 0030      EMTVEC= 30     ;;EMULATOR TRAP (EMT) **ERROR**
634      000034      TRAPVEC=34     ;;"TRAP" TRAP
635      000060      TKVEC= 60      ;;TTY KEYBOARD VECTOR
636      000064      TPVEC= 64     ;;TTY PRINTER VECTOR
637      000240      PIQVEC=240     ;;PROGRAM INTERRUPT REQUEST VECTOR
638
```

```
639 .SBTTL TRAP CATCHER
640
641      000000      . = 0
642      ; *ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
643      ; *SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
644      ; *LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
645      . = 174
646 000174 000000  DISPREG: .WORD 0      ;; SOFTWARE DISPLAY REGISTER
647 000176 000000  SWREG:   .WORD 0      ;; SOFTWARE SWITCH REGISTER
648
649 .SBTTL ACT11 HOOKS
650
651      ; *****
652      ; HOOKS REQUIRED BY ACT11
653      $SVPC = .      ; SAVE PC
654      . = 46
655 000046 045000  $ENDAD      ;; 1) SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
656      . = 52
657 000052 020000  .WORD 20000      ;; 2) SET LOC.52 TO 20000
658      . = $SVPC      ;; RESTORE PC
659
660 .SBTTL STARTING ADDRESSES
661
662      . = 200
663 000200 000137 017360  RA:      JMP      @#BEGIN      ; NORMAL START
664
665 000204 000137 051550  ADDMOD: JMP      @#BASECH      ; MODIFY DEVICE PARAMETERS
666
667 000210 000137 017350      JMP      @#BEGIN2      ; SELECT DRIVE START
668
669
670
671      ; *STARTING ADDRESS 200 FOR NORMAL STARTS
672      ; *THIS WILL TEST ALL RP04'S ON THE SYSTEM A SINGLE DRIVE AT A TIME
673
674      ; *STARTING ADDRESS 204 WILL LOAD NON-DEFAULT PARAMETERS
675      ; *AND AUTO RESTART AT 200 AFTER LOADING PARAMETERS.
676
677      ; *STARTING ADDRESS 210 WILL TEST ONLY ONE SPECIFIED DRIVE
678
```

```
679      .SBTTL  MEMORY MANAGEMENT DEFINITIONS
680
681      ;*KT11 VECTOR ADDRESS
682
683      000250      MMVEC= 250
684
685      ;*KT11 STATUS REGISTER ADDRESSES
686
687      177572      SR0= 177572
688      177574      SR1= 177574
689      177576      SR2= 177576
690      172516      SR3= 172516
691
692      ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS
693
694      172300      KIPDR0= 172300
695      172302      KIPDR1= 172302
696      172304      KIPDR2= 172304
697      172306      KIPDR3= 172306
698      172310      KIPDR4= 172310
699      172312      KIPDR5= 172312
700      172314      KIPDR6= 172314
701      172316      KIPDR7= 172316
702
703      ;*KERNEL "I" PAGE ADDRESS REGISTERS
704
705      172340      KIPAR0= 172340
706      172342      KIPAR1= 172342
707      172344      KIPAR2= 172344
708      172346      KIPAR3= 172346
709      172350      KIPAR4= 172350
710      172352      KIPAR5= 172352
711      172354      KIPAR6= 172354
712      172356      KIPAR7= 172356
713
714
715      001110      . =1110      ;
```

```
716 .SBTTL COMMON TAGS
717
718 ;*****
719 ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
720 ;*USED IN THE PROGRAM.
721
722 001100 .=1100
723 $CMTAG: .WORD 0 ;:START OF COMMON TAGS
724 001100 000000 $PASS: .WORD 0 ;:CONTAINS PASS COUNT
725 001102 000 $TSTNM: .BYTE 0 ;:CONTAINS THE TEST NUMBER
726 001103 000 $ERFLG: .BYTE 0 ;:CONTAINS ERROR FLAG
727 001104 000000 $ICNT: .WORD 0 ;:CONTAINS SUBTEST ITERATION COUNT
728 001106 000000 $LPADR: .WORD 0 ;:CONTAINS SCOPE LOOP ADDRESS
729 001110 000000 $LPERR: .WORD 0 ;:CONTAINS SCOPE RETURN FOR ERRORS
730 001112 000000 $ERTTL: .WORD 0 ;:CONTAINS TOTAL ERRORS DETECTED
731 001114 000 $ITEMB: .BYTE 0 ;:CONTAINS ITEM CONTROL BYTE
732 001115 001 $ERMAX: .BYTE 1 ;:CONTAINS MAX. ERRORS PER TEST
733 001116 000000 $ERRPC: .WORD 0 ;:CONTAINS PC OF LAST ERROR INSTRUCTION
734 001120 000000 $GDADR: .WORD 0 ;:CONTAINS ADDRESS OF 'GOOD' DATA
735 001122 000000 $BDADR: .WORD 0 ;:CONTAINS ADDRESS OF 'BAD' DATA
736 001124 000000 $GDDAT: .WORD 0 ;:CONTAINS 'GOOD' DATA
737 001126 000000 $BDDAT: .WORD 0 ;:CONTAINS 'BAD' DATA
738 001130 000000 .WORD 0 ;:RESERVED--NOT TO BE USED
739 001132 000000 .WORD 0
740 001134 000 $AUTOB: .BYTE 0 ;:AUTOMATIC MODE INDICATOR
741 001135 000 $INTAG: .BYTE 0 ;:INTERRUPT MODE INDICATOR
742 001136 000000 .WORD 0
743 001140 177570 SWR: .WORD DSWR ;:ADDRESS OF SWITCH REGISTER
744 001142 177570 DISPLAY: .WORD DDISP ;:ADDRESS OF DISPLAY REGISTER
745 001144 177560 $TKS: 177560 ;:TTY KBD STATUS
746 001146 177562 $TKB: 177562 ;:TTY KBD BUFFER
747 001150 177564 $TPS: 177564 ;:TTY PRINTER STATUS REG. ADDRESS
748 001152 177566 $TPB: 177566 ;:TTY PRINTER BUFFER REG. ADDRESS
749 001154 000 $NULL: .BYTE 0 ;:CONTAINS NULL CHARACTER FOR FILLS
750 001155 002 $FILLS: .BYTE 2 ;:CONTAINS # OF FILLER CHARACTERS REQUIRED
751 001156 012 $FILLC: .BYTE 12 ;:INSERT FILL CHARS. AFTER A 'LINE FEED'
752 001157 000 $TPFLG: .BYTE 0 ;: 'TERMINAL AVAILABLE' FLAG (BIT<07>=0=YES)
753 001160 000000 $REGAD: .WORD 0 ;:CONTAINS THE ADDRESS FROM
754 ;:WHICH ($REG0) WAS OBTAINED
755 001162 000000 $REG0: .WORD 0 ;:CONTAINS (($REGAD)+0)
756 001164 000000 $REG1: .WORD 0 ;:CONTAINS (($REGAD)+2)
757 001166 000000 $REG2: .WORD 0 ;:CONTAINS (($REGAD)+4)
758 001170 000000 $REG3: .WORD 0 ;:CONTAINS (($REGAD)+6)
759 001172 000000 $REG4: .WORD 0 ;:CONTAINS (($REGAD)+10)
760 001174 000000 $REG5: .WORD 0 ;:CONTAINS (($REGAD)+12)
761 001176 000000 $TMP0: .WORD 0 ;:USER DEFINED
762 001200 000000 $TMP1: .WORD 0 ;:USER DEFINED
763 001202 000000 $TMP2: .WORD 0 ;:USER DEFINED
764 001204 000000 $TMP3: .WORD 0 ;:USER DEFINED
765 001206 000000 $TMP4: .WORD 0 ;:USER DEFINED
766 001210 000000 $TMP5: .WORD 0 ;:USER DEFINED
767 001212 000000 $TIMES: 0 ;:MAX. NUMBER OF ITERATIONS
768 001214 000000 $ESCAPE: 0 ;:ESCAPE ON ERROR ADDRESS
769 001216 177607 000377 $BELL: .ASCII <207><377><377> ;:CODE FOR BELL
770 001222 077 $QUES: .ASCII /?/ ;:QUESTION MARK
771 001223 015 $CRLF: .ASCII <15> ;:CARRIAGE RETURN
```

CZRJHDO,RP04/5/6 DSKLS CTRLR2 MACY11 30A(1052) 24-MAY-79 15:07 ^{D 2} PAGE 17
CZRJHD.P11 28-MAR-79 09:16 COMMON TAGS

SEQ 0016

772 001224 000012
773

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

```

774 .SBTTL ERROR POINTER TABLE
775
776 ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
777 ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
778 ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
779 ;*NOTE1: IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
780 ;*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
781
782 ;* EM ;:POINTS TO THE ERROR MESSAGE
783 ;* DH ;:POINTS TO THE DATA HEADER
784 ;* DT ;:POINTS TO THE DATA
785 ;* DF ;:POINTS TO THE DATA FORMAT
786
787
788 001226 $ERRTB:
789
790
791
792
793
794 ;ITEM1
795 001226 002136 EM1 ;WRONG DATA IN READING OR WRITING HARDWARE REGISTER
796 001230 005451 DH1 ;PC
797 ;REG. ADDR.
798 ;GOOD DATA
799 ;RECEIVED DATA
800 001232 011764 DT1 ;$ERRPC,$STNM,REGADR,$GDDAT,$BDDAT
801 001234 012516 DF1 ;0,0,0,0,0
802
803
804
805
806
807 ;ITEM2
808 001236 002221 EM2 ;ERROR ON DATA COMMAND
809
810 001240 010463 DH33 ;PC
811 ;PC OF JSR
812 ;TEST NO
813 ;WORD NO.
814 ;GOOD DATA
815 ;CONTENTS OF RHCS1
816 ;CONTENTS OF RHDS1
817 ;CONTENTS OF RHER1
818 001242 012350 DT33 ;$ERRPC,PCJSR,$STNM,ERWORD,$GDDAT,CS1,DS1,ER1
819 001244 012666 DF33 ;0,0,0,1,0,0,0,0
820
821
822 ;ITEM3
823 001246 002221 EM2 ;ERROR ON DATA COMMAND
824
825 001250 010246 DH32 ;PC
826 ;PC OF JSR
827 ;TEST NO
828 ;WORD NO.
829 ;GOOD DATA
  
```

830					:BAD DATA
831					:CONTENTS OF RHCS1
832					:CONTENTS OF RHDS1
833					:CONTENTS OF RHER1
834					
835	001252	012324	DT32		:SERRPC,PCJSR,\$TSTNM,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1
836	001254	012655	DF32		:0,0,0,1,0,0,0,0,0,
837					
838					
839				:ITEM4	
840	001256	002221	EM2		:ERROR ON DATA COMMAND
841					
842	001260	010050	DH31		:PC
843					:TEST NO
844					:WORD NO.
845					:GOOD DATA
846					:BAD DATA
847					:CONTENTS OF RHCS1
848					:CONTENTS OF RHDS1
849					:CONTENTS OF RHER1
850					
851	001262	012302	DT31		:SERRPC,\$TSTNM,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1
852	001264	012645	DF31		:0,0,1,0,0,0,0,0,
853					
854					
855					
856				:ITEM5	
857	001266	000000	0		:
858	001270	000000	0		:
859	001272	012302	DT31		:SERRPC,\$TSTNM,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1
860	001274	012645	DF31		:0,0,1,0,0,0,0,0,
861					
862					
863				:ITEM6	
864	001276	002250	EM6		:ERROR ON WRITE HEADER AND DATA
865					
866	001300	010246	DH32		:PC
867					:PC OF JSR
868					:TEST NO
869					:WORD NO.
870					:GOOD DATA
871					:BAD DATA
872					:CONTENTS OF RHCS1
873					:CONTENTS OF RHDS1
874					:CONTENTS OF RHER1
875					
876	001302	012324	DT32		:SERRPC,PCJSR,\$TSTNM,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1
877	001304	012655	DF32		:0,0,0,1,0,0,0,0,0,
878					
879					
880					
881				:ITEM7	
882	001306	002250	EM6		:ERROR ON WRITE HEADER AND DATA
883	001310	005567	DH2		:PC
884					:TEST NO
885					:WORD NO.

886				;GOOD DATA
887				;BAD DATA
888	001312	012012	DT3	;SERRPC,\$TSTNM,ERWORD,\$GDDAT,\$BDDAT
889	001314	012527	DF3	;0,0,1,0,0,
890				
891				
892			;ITEM10	
893	001316	000000	0	:
894	001320	000000	0	:
895	001322	012012	DT3	;SERRPC,\$TSTNM,ERWORD,\$GDDAT,\$BDDAT
896	001324	012527	DF3	;0,0,1,0,0,
897				
898				
899			;ITEM11	
900	001326	002307	EM11	;CONTROLLER OR DRIVE STATUS
901	001330	005701	DH11	;PC
902				;TEST NO
903				;FAILING REG. ADDR
904				;CONTENTS OF RHCS1
905				;CONTENTS OF RHCS2
906				;CONTENTS OF RHDS1
907				;CONTENTS OF RHER1
908	001332	012026	DT11	;SERRPC,\$TSTNM,\$BDADR,CS1,CS2,DS1,ER1
909	001334	012534	DF11	;0,0,0,0,0,0
910				
911				
912			;ITEM12	
913	001336	002307	EM11	;WRONG DATA FROM SILO
914				
915	001340	005451	DH1	;PC
916				;REG.ADDR
917				;GOOD DATA
918				;RECEIVED DATA
919	001342	011764	DT1	;SERRPC,REGADR,\$GDDAT,\$BDDAT
920	001344	012516	DF1	;0,0,0,0
921				
922				
923			;ITEM13	
924	001346	000000	0	
925	001350	000000	0	
926	001352	011764	DT1	;SERRPC,TSTNM,REGADR,\$GDDAT,\$BDDAT
927	001354	012516	DF1	;0,0,0,0,0
928				
929				
930			;ITEM14	
931	001356	002342	EM14	;REGISTER FAILED
932	001360	006056	DH14	;PC
933				;FAILING REG. ADDR
934				;CONTENTS OF FAILING REG.
935				;CONTENTS OF RHCS1
936				;CONTENTS OF RHCS2
937				;CONTENTS OF RHDS1
938				;CONTENTS OF RHER1
939	001362	012046	DT14	;SERRPC,\$BDADR,\$BDDAT,CS1,CS2,DS1,ER1
940	001364	012543	DF14	;0,0,0,0,0,0,0
941				

942				
943			;ITEM15	
944	001366	002362	EM15	;SPECIFIED REG. NON EXISTANT SO ABORT
945				;PROGRAM
946	001370	006255	DH15	;PC
947				;ADDR. OF REG
948	001372	012070	DT15	;SERRPC,TEMP1
949	001374	012553	DF15	;0,0
950				
951				
952			;ITEM16	
953	001376	002432	EM16	;WAIT LOOP FAILED
954	001400	006305	DH16	;PC
955				;WAT PC
956				;BIT WANTED
957				;REG. ADR.
958				;REG. CONT.
959	001402	012100	DT16	;SERRPC,\$TMP3,\$TMP1,\$TMP0,\$BDDAT
960	001404	012556	DF16	;0,0,0,0
961				
962				
963			;ITEM17	
964	001406	002453	EM17	;WRITE CHECK FAILING
965	001410	006443	DH17	;PC
966				;TEST NO
967				;CONTENTS OF RHBA
968				;CONTENTS OF RHDB
969				;CONTENTS OF RHWC
970				;CONTENTS OF RHCS1
971				;CONTENTS OF RHCS2
972	001412	012116	DT17	;SERRPC,\$TSTNM,\$BA,DB,WC,CS1,CS2
973	001414	012563	DF17	;0,0,0,0,0,0,0
974				
975				
976			;ITEM20	
977	001416	002477	EM20	;REGISTER FAILING
978	001420	006620	DH20	;PC
979				;TST NO
980				;CONTENTS OF RHER1
981				;CONTENTS OF RHER2
982				;CONTENTS OF RHER3
983				;CONTENTS OF RHAS
984				;CONTENTS OF RHDS1
985	001422	012136	DT20	;SERRPC,TSTNM,ER1,ER2,ER3,AS,DS1
986	001424	012572	DF20	;0,0,0,0,0,0,0
987				
988			;ITEM21	
989				
990	001426	002520	EM21	;INTERRUPT FAILING
991	001430	006774	DH21	;PC
992				;TEST NO
993				;CONTENTS OF RHCS1
994				;CONTENTS OF RHAS
995				;CONTENTS OF RHDS1
996	001432	012156	DT21	;SERRPC,TSTNM,CS1,AS,DS1
997	001434	012601	DF21	;0,0,0,0,0

```
998
999
1000      ;ITEM22
1001      001436 002542      EM22      ;MISMATCH IN DRIVE PRESENT
1002                                         ;LOOKING AT RHAS AND RHCS2-NED(BIT#12)
1003                                         ;DRIVE PRESENT DO NOT AGREE
1004                                         ;NOTE: ON DUAL PORT SYSTEM
1005                                         ;DRIVE ON OTHER PORT WILL NOT GIVE NED
1006                                         ;HENCE THERE WILL BE A MISMATCH
1007                                         ;177777-MEANS NOT PRESENT
1008      001440 007110      DH22      ;PC
1009                                         ;TEST NO
1010                                         ;RHAS UNIT
1011                                         ;RHCS2 UNIT
1012
1013      001442 012172      DT22      ;$ERRPC,$TSTNMS,$GDDAT,$BDDAT
1014      001444 012606      DF22      ;0,0,0,0
1015
1016
1017      ;ITEM23
1018      001446 000000      0          ;MISMATCH IN DRIVE PRESENT
1019                                         ;LOOKING AT RHAS AND RHCS2-NED(BIT#12)
1020                                         ;DRIVE PRESENT DO NOT AGREE
1021                                         ;177777-MEANS NOT PRESENT
1022      001450 000000      0          ;PC
1023                                         ;TEST NO
1024                                         ;RHAS UNIT
1025                                         ;RHCS2 UNIT
1026
1027      001452 012172      DT22      ;$ERRPC,$TSTNMS,$GDDAT,$BDDAT
1028      001454 012606      DF22      ;0,0,0,0
1029
1030
1031
1032      ;ITEM 24
1033      001456 003143      EM24      ;LOOK AHEAD REGISTER AT THE
1034                                         ;BEGINNING OF A SECTOR IS IN
1035                                         ;ERROR
1036      001460 007204      DH24      ;PC
1037                                         ;RHDST
1038                                         ;BAD RHLA
1039                                         ;GOOD RHLA
1040                                         ;SECTOR NO
1041                                         ;SECTOR CLOCK
1042      001462 012204      DT24      ;$ERRPC,$DST,$BDDAT,$TMP1,$TMP2,$TMP3
1043      001464 012612      DF24      ;0,0,0,0,0
1044
1045      ;ITEM 25
1046      001466 003236      EM25      ;LOOK AHEAD REGISTER IS
1047                                         ;IN ERROR
1048
1049      001470 007204      DH24      ;PC
1050                                         ;RHDST
1051                                         ;BAD RHLA
1052                                         ;GOOD RHLA
1053                                         ;SECTOR NO
```

Line	Item	PC	Register	Value
1054				
1055	001472	012204	DT24	
1056	001474	012612	DF24	
1057				
1058	001476	002307	EM11	
1059				
1060	001500	007362	DH26	
1061				
1062				
1063				
1064				
1065				
1066				
1067				
1068	001502	012224	DT26	
1069	001504	012621	DF26	
1070				
1071				
1072				
1073				
1074	001506	002136	EM1	
1075				
1076	001510	007560	DH27	
1077				
1078				
1079				
1080				
1081				
1082				
1083	001512	012246	DT27	
1084	001514	012631	DF27	
1085				
1086				
1087				
1088				
1089	001516	003276	EM30	
1090	001520	007716	DH30	
1091				
1092				
1093				
1094				
1095				
1096	001522	012264	DT30	
1097	001524	012637	DF30	
1098				
1099				
1100				
1101				
1102	001526	003420	EM31	
1103				
1104				
1105	001530	010662	DH34	
1106				
1107				
1108				
1109				

1110				;WRITTEN ECC2
1111				;DATA USED
1112				
1113	001532	012372	DT34	;\$ERRPC,TSTNM,GECC1,GECC2,WECC1,WECC2,DISK
1114				
1115	001534	012676	DF34	;0,0,0,0,0,0,0
1116				
1117				
1118			;ITEM32	
1119	001536	003543	EM32	;ON READ COMMAND AFTER DATA AND ECC HAVE BEEN READ
1120				;ECC REGISTER OR RHER1 IS IN ERROR
1121				;ONLY LOWER 11 BITS OF PATTERN REGISTER
1122				;CAN BE READ
1123				;THIS SHUOLD MATCH LOWER 11 BITS OF ECC1
1124				
1125	001540	011035	DH35	;PC
1126				;TEST NUMBER
1127				;GOOD ECC1
1128				;GOOD ECC2
1129				;PATTERN REGISTER
1130				;RHER1
1131				
1132	001542	012412	DT35	;\$ERRPC,TSTNM,GECC1,GECC2,EC2,ER1
1133				
1134	001544	012705	DF35	;0,0,0,0,0,0
1135				
1136				
1137				
1138			;ITEM33	
1139	001546	004032	EM33	;HIGH COUNT BIT NOT HIGH AFTER 38859 CLOCKS
1140	001550	011232	DH36	;PC
1141				;PC OF JSR
1142				;TEST NUMBER
1143				;RHR
1144				;POSITION REG.
1145				;PATTERN REGISTER
1146				
1147	001552	012434	DT36	;\$ERRPC,PCJSR,TSTNM,MR,EC1,EC2
1148				
1149	001554	012715	DF36	;0,0,0,0,0,0
1150				
1151			;ITEM34	
1152	001556	004104	EM34	;ZERO DETECT BIT NOT HIGH WHEN THE
1153				;32 BIT ECC REGISTER HAS ITS 21 BITS
1154				;OF ZEROS
1155				;ERROR PRINTOUT WILL CONTINUE TILL
1156				;ZERO DETECT BIT IS HIGH
1157	001560	011232	DH36	;PC
1158				;PC OF JSR
1159				;TEST NUMBER
1160				;RHR
1161				;POSITION REG.
1162				;PATTERN REGISTER
1163				
1164	001562	012434	DT36	;\$ERRPC,PCJSR,TSTNM,MR,EC1,EC2
1165				

1166	001564	012715	DF36	:0,0,0,0,0,0
1167				
1168				
1169				
1170			:ITEM35	
1171	001566	004177	EM35	: POSITION REGISTER OR 11 BITS OF
1172				: PATTERN REGISTER INCORRECT
1173				: LOWER 11 BITS OF PATTERN REGISTER
1174				: SHOULD MATCH LOWER 11 BITS OF GOOD ECC1
1175				: DATA ENVELOPE AND N-CODE ZEROS ARE IN DECIMAL
1176				
1177	001570	011370	DH37	: PC
1178				: TEST NUMBER
1179				: ECC POSITION
1180				: GOOD POSITION
1181				: GOOD ECC1
1182				: GOOD ECC2
1183				: ECC PATTERN
1184				: DATA ENVELOPE
1185				: N-CODE ZEROS
1186				
1187	001572	012452	DT37	: \$ERRPC, TSTNM, EC1, POSITI, GECC1, GECC2, EC2, DATENV, ZCODE
1188				
1189	001574	012723	DF37	: 0,0,0,0,0,0,0,0,0,0
1190				
1191				
1192				
1193			:ITEM36	
1194	001576	004476	EM36	: ON A READ COMMAND WITH NON CORRECTABLE
1195				: ERROR INSERTED DCK AND ECH SHOULD BE SET
1196	001600	011035	DH35	: PC
1197				: TEST NUMBER
1198				: GOOD ECC1
1199				: GOOD ECC2
1200				: PATTERN REGISTER
1201				: POSITION REGISTER
1202				: RHER1
1203				
1204	001602	012412	DT35	: \$ERRPC, TSTNM, GECC1, GECC2, EC2, EC1, ER1
1205				
1206	001604	012705	DF35	: 0,0,0,0,0,0,0,0
1207				
1208				
1209				
1210				
1211				
1212				
1213			:ITEM37	
1214	001606	004664	EM37	: ERROR ON DATA COMMAND
1215				: WITH A16 A17 USED
1216				
1217	001610	010050	DH31	: PC
1218				: TEST NO
1219				: WORD NO.
1220				: GOOD DATA
1221				: BAD DATA

1222				;CONTENTS OF RHCS1
1223				;CONTENTS OF RHDS1
1224				;CONTENTS OF RHER1
1225				
1226	001612	012302	DT31	;\$ERRPC,\$STSTNM,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1
1227	001614	012645	DF31	;0,0,1,0,0,0,0,0,
1228				
1229				
1230				
1231			;ITEM40	
1232	001616	000000	0	:
1233	001620	000000	0	:
1234	001622	012302	DT31	;\$ERRPC,\$STSTNM,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1
1235	001624	012645	DF31	;0,0,1,0,0,0,0,0,
1236				
1237				
1238				
1239				
1240			;ITEM41	
1241	001626	004752	EM40	;THERE WAS A READ/WRITE HEADER & DATA
1242				;ERROR DURING 'DTE' TEST SETUP - THE
1243				;TEST IS ABORTED AT THAT POINT
1244	001630	011607	DH40	;PC
1245				;TEST NO
1246				;FAILING REGISTER ADDRESS
1247				;CONTENTS OF RHCS1
1248				;CONTENTS OF RHCS2
1249				;CONTENTS OF RHDS1
1250				;CONTENTS OF RHER1
1251	001632	012476	DT40	;\$ERRPC,\$STSTNM,\$BDADR,CS1,CS2,DS1,ER1
1252	001634	012734	DF40	;0,0,0,0,0,0,0

1253				
1254			;ITEM 42	
1255	001636	012744	EM42	
1256	001640	014410	DH42	
1257	001642	014672	DT42	;ERROR PC,TEST NUMBER, REGISTER ADDRESS.
1258	001644	014730	DF42	
1259				
1260			;ITEM 43	
1261				
1262	001646	013027	EM43	
1263	001650	014410	DH42	
1264	001652	014672	DT42	;ERROR PC,TEST NUMBER, REGISTER ADDRESS.
1265	001654	014730	DF42	
1266				
1267			;ITEM 44	
1268				
1269	001656	013073	EM44	
1270	001660	014471	DH44	
1271	001662	014702	DT44	;ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
1272	001664	014733	DF44	
1273				
1274			;ITEM 45	
1275	001666	013130	EM45	
1276	001670	014471	DH44	
1277	001672	014702	DT44	;ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
1278	001674	014733	DF44	
1279				
1280			;ITEM 46	
1281				
1282	001676	013157	EM46	
1283	001700	014471	DH44	
1284	001702	014702	DT44	;ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
1285	001704	014733	DF44	
1286				
1287			;ITEM 47	
1288				
1289	001706	013206	EM47	
1290	001710	014410	DH42	
1291	001712	014672	DT42	;ERROR PC,TEST NUMBER, REGISTER ADDRESS.
1292	001714	014730	DF42	
1293				
1294			;ITEM 50	
1295				
1296	001716	013243	EM50	
1297	001720	014471	DH44	
1298	001722	014702	DT44	;ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
1299	001724	014733	DF44	
1300				
1301			;ITEM 51	
1302				
1303	001726	013301	EM51	
1304	001730	014471	DH44	
1305	001732	014702	DT44	;ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
1306	001734	014733	DF44	
1307				
1308			;ITEM 52	

1309				
1310	001736	013322	EM52	
1311	001740	014471	DH44	
1312	001742	014702	DT44	
1313	001744	014733	DF44	
1314				
1315				:ITEM 53
1316				
1317	001746	013362	EM53	
1318	001750	014471	DH44	
1319	001752	014702	DT44	:ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
1320	001754	014733	DF44	
1321				
1322				:ITEM 54
1323				
1324	001756	013422	EM54	
1325	001760	014603	DH54	
1326	001762	014716	DT54	:ERROR PC, TEST NUMBER.
1327	001764	014740	DF54	
1328				
1329				:ITEM 55
1330				
1331	001766	013461	EM55	
1332	001770	014471	DH44	
1333	001772	014702	DT44	:ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
1334	001774	014733	DF44	
1335				
1336				:ITEM 56
1337				
1338	001776	013474	EM56	
1339	002000	014471	DH44	
1340	002002	014702	DT44	:ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
1341	002004	014733	DF44	
1342				
1343				:ITEM 57
1344				
1345	002006	013511	EM57	
1346	002010	014471	DH44	
1347	002012	014702	DT44	:ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
1348	002014	014733	DF44	
1349				
1350				:ITEM 60
1351				
1352	002016	013540	EM60	
1353	002020	014471	DH44	
1354	002022	014702	DT44	:ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
1355	002024	014733	DF44	
1356				
1357				:ITEM 61
1358				
1359	002026	013627	EM61	
1360	002030	014471	DH44	
1361	002032	014702	DT44	:ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
1362	002034	014733	DF44	
1363				
1364				:ITEM 62

1365				
1366	002036	013677	EM62	
1367	002040	014471	DH44	
1368	002042	014702	DT44	;ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
1369	002044	014733	DF44	
1370				
1371				;ITEM 63
1372				
1373	002046	013754	EM63	
1374	002050	014471	DH44	
1375	002052	014702	DT44	;ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
1376	002054	014733	DF44	
1377				
1378				;ITEM 64
1379				
1380	002056	014032	EM64	
1381	002060	014471	DH44	
1382	002062	014702	DT44	;ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
1383	002064	014733	DF44	
1384				
1385				;ITEM 65
1386				
1387	002066	014066	EM65	
1388	002070	014471	DH44	
1389	002072	014702	DT44	;ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
1390	002074	014733	DF44	
1391				
1392				;ITEM 66
1393				
1394	002076	014150	EM66	
1395	002100	014471	DH44	
1396	002102	014702	DT44	;ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
1397	002104	014733	DF44	
1398				
1399				;ITEM 67
1400				
1401	002106	014222	EM67	
1402	002110	014603	DH54	
1403	002112	014716	DT54	;ERROR PC, TEST NUMBER.
1404	002114	014740	DF54	
1405				
1406				;ITEM 70
1407				
1408	002116	014307	EM70	
1409	002120	014603	DH54	
1410	002122	014716	DT54	;ERROR PC, TEST NUMBER.
1411	002124	014740	DF54	
1412				
1413				;ITEM 71
1414				
1415	002126	014361	EM71	
1416	002130	014471	DH44	
1417	002132	014702	DT44	;ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
1418	002134	014733	DF44	

```
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429 002136 051127 047117 020107 EM1: .ASCIZ /WRONG DATA IN READING OR WRITING HARDWARE REGISTER/
1430 002144 040504 040524 044440
1431 002152 020116 042522 042101
1432 002160 047111 020107 051117
1433 002166 053440 044522 044524
1434 002174 043516 044040 051101
1435 002202 053504 051101 020105
1436 002210 042522 044507 052123
1437 002216 051105 000
1438 002221 105 051122 051117 EM2: .ASCIZ /ERROR ON DATA COMMAND/
1439 002226 047440 020116 042040
1440 002234 052101 020101 047503
1441 002242 046515 047101 000104
1442 002250 051105 047522 020122 EM6: .ASCIZ /ERROR ON WRITE HEADER AND DATA/
1443 002256 047117 053440 044522
1444 002264 042524 044040 040505
1445 002272 042504 020122 047101
1446 002300 020104 040504 040524
1447 002306 000
1448 002307 103 047117 051124 EM11: .ASCIZ /CONTROLLER OR DRIVE STATUS/
1449 002314 046117 042514 020122
1450 002322 051117 042040 044522
1451 002330 042526 051440 040524
1452 002336 052524 000123
1453 002342 042522 044507 052123 EM14: .ASCIZ /REGISTER FAILED/
1454 002350 051105 043040 044501
1455 002356 042514 000104
1456 002362 047516 020116 054105 EM15: .ASCIZ /NON EXISTENT REGISTER, PROGRAM ABORTED./
1457 002370 051511 042524 052116
1458 002376 051040 043505 051511
1459 002404 042524 026122 050040
1460 002412 047522 051107 046501
1461 002420 040440 047502 052122
1462 002426 042105 000056
1463 002432 040527 052111 046040 EM16: .ASCIZ /WAIT LOOP FAILED/
1464 002440 047517 020120 040506
1465 002446 046111 042105 000
1466 002453 127 044522 042524 EM17: .ASCIZ /WRITE CHECK FAILING/
1467 002460 041440 042510 045503
1468 002466 043040 044501 044514
1469 002474 043516 000
1470 002477 122 043505 051511 EM20: .ASCIZ /REGISTER FAILING/
1471 002504 042524 020122 040506
1472 002512 046111 047111 000107
1473 002520 047111 042524 051122 EM21: .ASCIZ /INTERRUPT FAILING/
1474 002526 050125 020124 040506
```

1475	002534	046111	047111	000107	
1476	002542	051105	047522	020122	EM22: .ASCII /ERROR ON DRIVE PRESENT/<15><12>
1477	002550	047117	042040	044522	
1478	002556	042526	050040	042522	
1479	002564	042523	052116	005015	
1480	002572	044124	020105	047125	.ASCII /THE UNIT NO'S FOUND BY SETTING RHAS/<15><12>
1481	002600	052111	047040	023517	
1482	002606	020123	047506	047125	
1483	002614	020104	054502	051440	
1484	002622	052105	044524	043516	
1485	002630	051040	040510	006523	
1486	002636	012			
1487	002637	104	020117	047516	.ASCII /DO NOT AGREE WITH THE UNIT NO. FOUND FROM/<15><12>
1488	002644	020124	043501	042522	
1489	002652	020105	044527	044124	
1490	002660	052040	042510	052440	
1491	002666	044516	020124	047516	
1492	002674	020056	047506	047125	
1493	002702	020104	051106	046517	
1494	002710	005015			
1495	002712	044122	051503	026462	.ASCII /RHCS2-'NED' BIT #12/<15><'2>
1496	002720	047047	042105	020047	
1497	002726	044502	020124	030443	
1498	002734	006462	012		
1499	002737	061	033467	033467	.ASCII /177777-MEANS NO UNIT FOUND/<15><12>
1500	002744	026467	042515	047101	
1501	002752	020123	047516	052440	
1502	002760	044516	020124	047506	
1503	002766	047125	006504	012	
1504	002773	116	052117	035105	.ASCII /NOTE: ON DUAL PORT SYSTEM, DRIVE ON OTHER PORT WILL NOT GIVE/<15><12>
1505	003000	047440	020116	052504	
1506	003006	046101	050040	051117	
1507	003014	020124	054523	052123	
1508	003022	046505	020054	051104	
1509	003030	053111	020105	047117	
1510	003036	047440	044124	051105	
1511	003044	050040	051117	020124	
1512	003052	044527	046114	047040	
1513	003060	052117	043440	053111	
1514	003066	006505	012		
1515	003071	047	042516	023504	.ASCIIZ /'NED', HENCE THERE WILL BE AN EXTRA DRIVE/
1516	003076	020054	042510	041516	
1517	003104	020105	044124	051105	
1518	003112	020105	044527	046114	
1519	003120	041040	020105	047101	
1520	003126	042440	052130	040522	
1521	003134	042040	044522	042526	
1522	003142	000			
1523	003143	114	047517	020113	EM24: .ASCIIZ /LOOK AHEAD REGISTER AT THE BEGINNING OF SECTOR IS IN ERROR/
1524	003150	044101	040505	020104	
1525	003156	042522	044507	052123	
1526	003164	051105	040440	020124	
1527	003172	044124	020105	042502	
1528	003200	044507	047116	047111	
1529	003206	020107	043117	051440	
1530	003214	041505	047524	020122	

1531	003222	051511	044440	020116	
1532	003230	051105	047522	000122	
1533	003236	047514	045517	040440	EM25: .ASCIZ /LOOK AHEAD REGISTER IS IN ERROR/
1534	003244	042510	042101	051040	
1535	003252	043505	051511	042524	
1536	003260	020122	051511	044440	
1537	003266	020116	051105	047522	
1538	003274	000122			
1539	003276	052503	051122	047105	EM30: .ASCII /CURRENT CYLINDER DOES NOT MATCH DESIRED CYLINDER REGISTER/<15><12>
1540	003304	020124	054503	044514	
1541	003312	042116	051105	042040	
1542	003320	042517	020123	047516	
1543	003326	020124	040515	041524	
1544	003334	020110	042504	044523	
1545	003342	042522	020104	054503	
1546	003350	044514	042116	051105	
1547	003356	051040	043505	044111	
1548	003364	052123	051105	005015	
1549	003372	043101	042524	020122	.ASCIZ /AFTER A SEEK AND INIT/
1550	003400	020101	042523	045505	
1551	003406	040440	042116	044440	
1552	003414	044516	000124		
1553	003420	041505	020103	042507	EM31: .ASCII /ECC GENERATED IS INCORRECT/<15><12>
1554	003426	042516	040522	042524	
1555	003434	020104	051511	044440	
1556	003442	041516	051117	042522	
1557	003450	052103	005015		
1558	003454	053105	051105	020131	.ASCIZ /EVERY WORD ON THIS SECTOR IS THAT GIVEN IN 'DATA USED'/'
1559	003462	047527	042122	047440	
1560	003470	020116	044124	051511	
1561	003476	051440	041505	047524	
1562	003504	020122	051511	052040	
1563	003512	040510	020124	044507	
1564	003520	042526	020116	047111	
1565	003526	021040	040504	040524	
1566	003534	052440	042523	021104	
1567	003542	000			
1568	003543	117	020116	042522	EM32: .ASCII /ON READ COMMAND, AFTER DATA AND ECC HAVE BEEN READ,/<15><12>
1569	003550	042101	041440	046517	
1570	003556	040515	042116	020054	
1571	003564	043101	042524	020122	
1572	003572	040504	040524	040440	
1573	003600	042116	042440	041503	
1574	003606	044040	053101	020105	
1575	003614	042502	047105	051040	
1576	003622	040505	026104	005015	
1577	003630	041505	020103	042522	.ASCII /ECC REGISTERS OR RHER1 ARE IN ERROR/<15><12>
1578	003636	044507	052123	051105	
1579	003644	020123	051117	051040	
1580	003652	042510	030522	040440	
1581	003660	042522	044440	020116	
1582	003666	051105	047522	006522	
1583	003674	012			
1584	003675	117	046116	020131	.ASCII /ONLY LOWER 11 BITS OF PATTERN REG. CAN BE READ/<15><12>
1585	003702	047514	042527	020122	
1586	003710	030461	041040	052111	

1587	003716	020123	043117	050040	
1588	003724	052101	042524	047122	
1589	003732	051040	043505	020056	
1590	003740	040503	020116	042502	
1591	003746	051040	040505	006504	
1592	003754	012			
1593	003755	124	044510	020123	.ASCIZ /THIS SHOULD MATCH LOWER 11 BITS OF GOOD ECC1/
1594	003762	044123	052517	042114	
1595	003770	046440	052101	044103	
1596	003776	046040	053517	051105	
1597	004004	030440	020061	044502	
1598	004012	051524	047440	020106	
1599	004020	047507	042117	042440	
1600	004026	041503	000061		
1601	004032	044510	044107	041440	EM33: .ASCIZ /HIGH COUNT BIT NOT SET AFTER 38859 CLOCKS/
1602	004040	052517	052116	041040	
1603	004046	052111	047040	052117	
1604	004054	051440	052105	040440	
1605	004062	052106	051105	031440	
1606	004070	034070	034465	041440	
1607	004076	047514	045503	000123	
1608	004104	042532	047522	042040	EM34: .ASCIZ /ZERO DETECT BIT NOT HIGH WHEN 32 BIT ECC REG. HAS 21 ZEROS/
1609	004112	052105	041505	020124	
1610	004120	044502	020124	047516	
1611	004126	020124	044510	044107	
1612	004134	053440	042510	020116	
1613	004142	031063	041040	052111	
1614	004150	042440	041503	051040	
1615	004156	043505	020056	040510	
1616	004164	020123	030462	055040	
1617	004172	051105	051517	000	
1618	004177	120	051517	052111	EM35: .ASCII /POSITION REGISTER OR 11 BITS OF PATTERN REGISTER INCORRECT/<15><12>
1619	004204	047511	020116	042522	
1620	004212	044507	052123	051105	
1621	004220	047440	020122	030461	
1622	004226	041040	052111	020123	
1623	004234	043117	050040	052101	
1624	004242	042524	047122	051040	
1625	004250	043505	051511	042524	
1626	004256	020122	047111	047503	
1627	004264	051122	041505	006524	
1628	004272	012			
1629	004273	114	053517	051105	.ASCII /LOWER 11 BITS OF PATTERN REGISTER SHOULD MATCH LOWER/<15><12>
1630	004300	030440	020061	044502	
1631	004306	051524	047440	020106	
1632	004314	040520	052124	051105	
1633	004322	020116	042522	044507	
1634	004330	052123	051105	051440	
1635	004336	047510	046125	020104	
1636	004344	040515	041524	020110	
1637	004352	047514	042527	006522	
1638	004360	012			
1639	004361	061	020061	044502	.ASCII /11 BITS OF GOOD ECC1/<15><12>
1640	004366	051524	047440	020106	
1641	004374	047507	042117	042440	
1642	004402	041503	006461	012	

1643	004407	104	052101	042440		.ASCIIZ /DAT ENVLOP GOOD POSITION AND N-CODE ZEROS ARE IN OCTAL/
1644	004414	053116	047514	020120		
1645	004422	047507	042117	050040		
1646	004430	051517	052111	047511		
1647	004436	020116	047101	020104		
1648	004444	026516	047503	042504		
1649	004452	055040	051105	051517		
1650	004460	040440	042522	044440		
1651	004466	020116	041517	040524		
1652	004474	000114				
1653	004476	047117	051040	040505	EM36:	.ASCII /ON READ COMMAND WITH NON-CORRECTABLE ERROR DCK AND ECH SHOULD BE SET/<1
1654	004504	020104	047503	046515		
1655	004512	047101	020104	044527		
1656	004520	044124	047040	047117		
1657	004526	041455	051117	042522		
1658	004534	052103	041101	042514		
1659	004542	042440	051122	051117		
1660	004550	042040	045503	040440		
1661	004556	042116	042440	044103		
1662	004564	051440	047510	046125		
1663	004572	020104	042502	051440		
1664	004600	052105	005015			
1665	004604	043111	050040	051517		.ASCIIZ /IF POSITION REGISTER =10040 OR 10041 IT IS GOOD/
1666	004612	052111	047511	020116		
1667	004620	042522	044507	052123		
1668	004626	051105	036440	030061		
1669	004634	032060	020060	051117		
1670	004642	030440	030060	030464		
1671	004650	044440	020124	051511		
1672	004656	043440	047517	000104		
1673	004664	051127	052111	047111	EM37:	.ASCIIZ /WRITING WITH BUS ADDRESS HIGHER THAN 28K CAUSED ERROR/
1674	004672	020107	044527	044124		
1675	004700	041040	051525	040440		
1676	004706	042104	042522	051523		
1677	004714	044040	043511	042510		
1678	004722	020122	044124	047101		
1679	004730	031040	045470	041440		
1680	004736	052501	042523	020104		
1681	004744	051105	047522	000122		
1682	004752	044124	051105	020105	EM40:	.ASCII /THERE WAS A READ-WRITE HEADER & DATA ERROR DURING 'DTE'/'<15><12>
1683	004760	040527	020123	020101		
1684	004766	042522	042101	053455		
1685	004774	044522	042524	044040		
1686	005002	040505	042504	020122		
1687	005010	020046	040504	040524		
1688	005016	042440	051122	051117		
1689	005024	042040	051125	047111		
1690	005032	020107	042047	042524		
1691	005040	006447	012			
1692	005043	124	051505	020124		.ASCIIZ /TEST SETUP - THE TEST WAS ABORTED AT THAT POINT/
1693	005050	042523	052524	020120		
1694	005056	020055	044124	020105		
1695	005064	042524	052123	053440		
1696	005072	051501	040440	047502		
1697	005100	052122	042105	040440		
1698	005106	020124	044124	052101		

CZRJHDO,RP04/5/6 DSKLS CTRLR2 MACY11 30A(1052) 24-MAY-79 15:07 ¹ ³ PAGE 35
CZRJHD.P11 28-MAR-79 09:16 ERROR POINTER TABLE

SEQ 0034

1699 005114 050040 044517 052116
1700 005122 000

1701	005123	106	052101	046101	CPHALT: .ASCII /FATAL ERROR - SEE DOCUMENT FOR BEST COURSE OF ACTION/<15><12>
1702	005130	042440	051122	051117	
1703	005136	026440	051440	042505	
1704	005144	042040	041517	046525	
1705	005152	047105	020124	047506	
1706	005160	020122	042502	052123	
1707	005166	041440	052517	051522	
1708	005174	020105	043117	040440	
1709	005202	052103	047511	006516	
1710	005210	012			
1711	005211	040	005015	177607	.ASCII / /<15><12><207><377><377><207><377><377><207><377><377>
1712	005216	103777	177777	177607	
1713	005224	377			
1714	005225	124	042510	050040	.ASCII /THE PROGRAM HAS HALTED DURING A TEST BECAUSE CONTROLLER/<15><12>
1715	005232	047522	051107	046501	
1716	005240	044040	051501	044040	
1717	005246	046101	042524	020104	
1718	005254	052504	044522	043516	
1719	005262	040440	052040	051505	
1720	005270	020124	042502	040503	
1721	005276	051525	020105	047503	
1722	005304	052116	047522	046114	
1723	005312	051105	005015		
1724	005316	051117	042040	053105	.ASCII /OR DEVICE HAS LOST 'READY', BECOME UNAVAILABLE,/<15><12>
1725	005324	041511	020105	040510	
1726	005332	020123	047514	052123	
1727	005340	023440	042522	042101	
1728	005346	023531	020054	042502	
1729	005354	047503	042515	052440	
1730	005362	040516	040526	046111	
1731	005370	041101	042514	006454	
1732	005376	012			
1733	005377	107	047117	020105	.ASCIIZ /GONE OFFLINE, OR CANNOT CLEAR STATUS BITS/
1734	005404	043117	046106	047111	
1735	005412	026105	047440	020122	
1736	005420	040503	047116	052117	
1737	005426	041440	042514	051101	
1738	005434	051440	040524	052524	
1739	005442	020123	044502	051524	
1740	005450	000			
1741					
1742	005451	120	020103	020040	DH1: .ASCII /PC TEST REG. GOOD ACTUAL /<15><12>
1743	005456	020040	052040	051505	
1744	005464	020124	020040	051040	
1745	005472	043505	020056	020040	
1746	005500	043440	047517	020104	
1747	005506	020040	040440	052103	
1748	005514	040525	004514	005015	
1749	005522	020040	020040	020040	.ASCIIZ / NO ADDR. DATA DATA/
1750	005530	020040	047516	020040	
1751	005536	020040	020040	042101	
1752	005544	051104	020056	020040	
1753	005552	040504	040524	020040	
1754	005560	020040	040504	040524	
1755	005566	000			
1756	005567	120	020103	020040	DH2: .ASCII /PC TEST WORD GOOD BAD/<15><12>

[illegible]

1813	006270	020124	020040	051040
1814	006276	043505	040440	051104
1815	006304	000		
1816	006305	120	020103	020040
1817	006312	020040	052040	051505
1818	006320	020124	020040	053440
1819	006326	052101	020040	020040
1820	006334	041040	052111	020040
1821	006342	020040	051040	043505
1822	006350	020040	020040	051040
1823	006356	043505	005015	
1824	006362	020040	020040	020040
1825	006370	020040	047516	020040
1826	006376	020040	020040	041520
1827	006404	020040	020040	020040
1828	006412	040527	052116	042105
1829	006420	020040	042101	051104
1830	006426	051505	020123	047503
1831	006434	052116	047105	051524
1832	006442	000		
1833	006443	120	020103	020040
1834	006450	020040	052040	051505
1835	006456	020124	020040	041440
1836	006464	047117	027124	020040
1837	006472	041440	047117	027124
1838	006500	020040	041440	047117
1839	006506	027124	020040	041440
1840	006514	047117	027124	020040
1841	006522	041440	047117	027124
1842	006530	005015		
1843	006532	020040	020040	020040
1844	006540	020040	047516	020040
1845	006546	020040	020040	044122
1846	006554	040502	020040	020040
1847	006562	044122	041104	020040
1848	006570	020040	044122	041527
1849	006576	020040	020040	044122
1850	006604	051503	020061	020040
1851	006612	044122	051503	000062
1852	006620	041520	020040	020040
1853	006626	020040	042524	052123
1854	006634	020040	020040	047503
1855	006642	052116	020040	020040
1856	006650	047503	052116	020040
1857	006656	020040	047503	052116
1858	006664	020040	020040	047503
1859	006672	052116	020040	020040
1860	006700	047503	052116	005015
1861	006706	020040	020040	020040
1862	006714	020040	047516	020040
1863	006722	020040	020040	044122
1864	006730	051105	020061	020040
1865	006736	044122	051105	020062
1866	006744	020040	044122	051105
1867	006752	020063	020040	044122
1868	006760	051501	020040	020040

[illegible]

1925	007461	040	020040	020040		.ASCIZ	/	NO	JSR	REG	ADD	RHCS1	RHCS2	RHDS1	RHER1	/
1926	007466	020040	047040	020117												
1927	007474	020040	020040	045040												
1928	007502	051123	020040	020040												
1929	007510	051040	043505	040440												
1930	007516	042104	051040	041510												
1931	007524	030523	020040	051040												
1932	007532	041510	031123	020040												
1933	007540	051040	042110	030523												
1934	007546	020040	051040	042510												
1935	007554	030522	000011													
1936	007560	041520	020040	020040	DH27:	.ASCII	/PC	TEST	PC	OF	FAILING	GOOD	ACTUAL	/<15><12>		
1937	007566	020040	042524	052123												
1938	007574	020040	020040	041520												
1939	007602	047440	020106	020040												
1940	007610	040506	046111	047111												
1941	007616	020107	047507	042117												
1942	007624	020040	020040	041501												
1943	007632	052524	046101	006411												
1944	007640	012														
1945	007641	040	020040	020040		.ASCIZ	/	NO	JSR	REG.	DATA	DATA/				
1946	007646	020040	047040	020117												
1947	007654	020040	020040	045040												
1948	007662	051123	020040	020040												
1949	007670	051040	043505	020056												
1950	007676	020040	042040	052101												
1951	007704	020101	020040	042040												
1952	007712	052101	000101													
1953	007716	041520	020040	020040	DH30:	.ASCII	/PC	TEST	PC	OF	REG.	GOOD	BAD/	<15><12>		
1954	007724	020040	042524	052123												
1955	007732	020040	020040	041520												
1956	007740	047440	020106	020040												
1957	007746	042522	027107	020040												
1958	007754	020040	047507	042117												
1959	007762	020040	020040	040502												
1960	007770	006504	012													
1961	007773	040	020040	020040		.ASCIZ	/	NO	JSR	ADDR	DATA	DATA/				
1962	010000	020040	047040	020117												
1963	010006	020040	020040	045040												
1964	010014	051123	020040	020040												
1965	010022	040440	042104	020122												
1966	010030	020040	042040	052101												
1967	010036	020101	020040	042040												
1968	010044	052101	000101													
1969	010050	041520	020040	020040	DH31:	.ASCII	/PC	TEST	WORD	GOOD	BAD	CONT.	CONT.	CONT./	<15><12>	
1970	010056	020040	042524	052123												
1971	010064	020040	020040	047527												
1972	010072	042122	020040	020040												
1973	010100	047507	042117	020040												
1974	010106	020040	040502	020104												
1975	010114	020040	020040	047503												
1976	010122	052116	020056	020040												
1977	010130	047503	052116	020056												
1978	010136	020040	047503	052116												
1979	010144	006456	012													
1980	010147	040	020040	020040		.ASCIZ	/	NO	NO	DATA	DATA	RHCS1	RHDS1	RHER1	/	

SEQ 0040

1981	010154	020040	047040	020117
1982	010162	020040	020040	047040
1983	010170	020117	020040	020040
1984	010176	042040	052101	020101
1985	010204	020040	042040	052101
1986	010212	020101	020040	051040
1987	010220	041510	030523	020040
1988	010226	051040	042110	030523
1989	010234	020040	051040	042510
1990	010242	030522	000011	
1991	010246	041520	020040	020040
1992	010254	020040	042524	052123
1993	010262	020040	020040	041520
1994	010270	047440	020106	020040
1995	010276	047527	042122	020040
1996	010304	020040	047507	042117
1997	010312	020040	020040	040502
1998	010320	020104	020040	020040
1999	010326	047503	052116	020056
2000	010334	020040	047503	052116
2001	010342	004456	020040	047503
2002	010350	052116	006456	012
2003	010355	040	020040	020040
2004	010362	020040	047040	020117
2005	010370	020040	020040	045040
2006	010376	051123	020040	020040
2007	010404	047040	020117	020040
2008	010412	020040	042040	052101
2009	010420	020101	020040	042040
2010	010426	052101	020101	020040
2011	010434	051040	041510	030523
2012	010442	020040	051040	042110
2013	010450	030523	020040	051040
2014	010456	042510	030522	000
2015	010463	120	020103	020040
2016	010470	020040	052040	051505
2017	010476	020124	020040	050040
2018	010504	020103	043117	020040
2019	010512	053440	051117	020104
2020	010520	020040	043440	047517
2021	010526	020104	020040	041440
2022	010534	047117	027124	020040
2023	010542	041440	047117	027124
2024	010550	020040	041440	047117
2025	010556	027124	006411	012
2026	010563	040	020040	020040
2027	010570	020040	047040	020117
2028	010576	020040	020040	045040
2029	010604	051123	020040	020040
2030	010612	047040	020117	020040
2031	010620	020040	042040	052101
2032	010626	020101	020040	051040
2033	010634	041510	030523	020040
2034	010642	051040	042110	030523
2035	010650	020040	051040	042510
2036	010656	030522	000011	

2037	010662	041520	020040	020040	DH34:	.ASCII	/PC	TEST	GOOD	GOOD	WRITTEN WRITTEN DATA/<15><12>			
2038	010670	020040	042524	052123										
2039	010676	020040	020040	047507										
2040	010704	042117	020040	020040										
2041	010712	047507	042117	020040										
2042	010720	020040	051127	052111										
2043	010726	042524	020116	051127										
2044	010734	052111	042524	020116										
2045	010742	040504	040524	005015										
2046	010750	020040	020040	020040		.ASCIZ	/	NO	ECC1	ECC2	ECC1	ECC2	USED/	
2047	010756	020040	047516	020040										
2048	010764	020040	020040	041505										
2049	010772	030503	020040	020040										
2050	011000	041505	031103	020040										
2051	011006	020040	041505	030503										
2052	011014	020040	020040	041505										
2053	011022	031103	020040	020040										
2054	011030	051525	042105	000										
2055	011035	120	020103	020040	DH35:	.ASCII	/PC	TEST	GOOD	GOOD	PATTERN	POSITON	GOOD	RHER1 /<15><12
2056	011042	020040	052040	051505										
2057	011050	020124	020040	043440										
2058	011056	047517	020104	020040										
2059	011064	043440	047517	020104										
2060	011072	020040	050040	052101										
2061	011100	042524	047122	050040										
2062	011106	051517	052111	047117										
2063	011114	043440	047517	020104										
2064	011122	020040	051040	042510										
2065	011130	030522	006411	012										
2066	011135	040	020040	020040		.ASCIZ	/	NO	ECC1	ECC2	REG.	REG.	POSITON	REG./
2067	011142	020040	047040	020117										
2068	011150	020040	020040	042440										
2069	011156	041503	020061	020040										
2070	011164	042440	041503	020062										
2071	011172	020040	051040	043505										
2072	011200	020056	020040	051040										
2073	011206	043505	020056	020040										
2074	011214	050040	051517	052111										
2075	011222	047117	051040	043505										
2076	011230	000056												
2077	011232	041520	020040	020040	DH36:	.ASCII	/PC	TEST	PC OF	RHMR	POSITON PATTERN/<15><12>			
2078	01													

[illegible]

2149	012034	015032	015030	015054					
2150	012042	015034	000000						
2151	012046	001116	017330	001122	DT14:	.WORD	\$ERRPC,TSTNM,\$BDADR,\$BDDAT,CS1,CS2,DS1,ER1,0		
2152	012054	001126	015032	015030					
2153	012062	015054	015034	000000					
2154	012070	001116	017330	001200	DT15:	.WORD	\$ERRPC,TSTNM,\$TMP1,0		
2155	012076	000000							
2156	012100	001116	017330	001204	DT16:	.WORD	\$ERRPC,TSTNM,\$TMP3,\$TMP1,\$TMP0,\$BDDAT,0		
2157	012106	001200	001176	001126					
2158	012114	000000							
2159	012116	001116	017330	015026	DT17:	.WORD	\$ERRPC,TSTNM,BA,DB,WC,CS1,CS2,0		
2160	012124	015022	015024	015032					
2161	012132	015030	000000						
2162									
2163	012136	001116	017330	015034	DT20:	.WORD	\$ERRPC,TSTNM,ER1,ER2,ER3,AS,DS1,0		
2164	012144	015040	015046	015050					
2165	012152	015054	000000						
2166	012156	001116	017330	015032	DT21:	.WORD	\$ERRPC,TSTNM,CS1,AS,DS1,0		
2167	012164	015050	015054	000000					
2168	012172	001116	017330	001124	DT22:	.WORD	\$ERRPC,TSTNM,\$GDDAT,\$BDDAT,0		
2169	012200	001126	000000						
2170	012204	001116	017330	015036	DT24:	.WORD	\$ERRPC,TSTNM,DST,\$BDDAT,\$TMP1,\$TMP2,\$TMP3,0		
2171	012212	001126	001200	001202					
2172	012220	001204	000000						
2173	012224	001116	017330	015132	DT26:	.WORD	\$ERRPC,TSTNM,PCJSR,\$BDADR,CS1,CS2,DS1,ER1,0		
2174	012232	001122	015032	015030					
2175	012240	015054	015034	000000					
2176	012246	001116	017330	015132	DT27:	.WORD	\$ERRPC,TSTNM,PCJSR,REGADR,\$GDDAT,\$BDDAT,0		
2177	012254	045662	001124	001126					
2178	012262	000000							
2179	012264	001116	017330	015132	DT30:	.WORD	\$ERRPC,TSTNM,PCJSR,REGADR,\$GDDAT,\$BDDAT,0		
2180	012272	045662	001124	001126					
2181	012300	000000							
2182									
2183	012302	001116	017330	053130	DT31:	.WORD	\$ERRPC,TSTNM,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1,0		
2184	012310	001124	001126	015032					
2185	012316	015054	015034	000000					
2186	012324	001116	017330	015132	DT32:	.WORD	\$ERRPC,TSTNM,PCJSR,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1,0		
2187	012332	053130	001124	001126					
2188	012340	015032	015054	015034					
2189	012346	000000							
2190	012350	001116	017330	015132	DT33:	.WORD	\$ERRPC,TSTNM,PCJSR,ERWORD,\$GDDAT,CS1,DS1,ER1,0		
2191	012356	053130	001124	015032					
2192	012364	015054	015034	000000					
2193	012372	001116	017330	050620	DT34:	.WORD	\$ERRPC,TSTNM,GECC1,GECC2,WECC1,WECC2,DISK,0		
2194	012400	050622	055726	055730					
2195	012406	054726	000000						
2196	012412	001116	017330	050620	DT35:	.WORD	\$ERRPC,TSTNM,GECC1,GECC2,EC2,EC1,POSIT1,ER1,0		
2197	012420	050622	015064	015062					
2198	012426	050632	015034	000000					
2199	012434	001116	017330	015132	DT36:	.WORD	\$ERRPC,TSTNM,PCJSR,MR,EC1,EC2,0		
2200	012442	015052	015062	015064					
2201	012450	000000							
2202	012452	001116	017330	015062	DT37:	.WORD	\$ERRPC,TSTNM,EC1,POSIT1,GECC1,GECC2,EC2,DATENV,ZCODE,0		
2203	012460	050632	050620	050622					
2204	012466	015064	050636	050640					

2205	012474	000000					
2206	012476	001116	017330	001122	DT40:	.WORD	\$ERRPC,TSTNM,\$BDADR,CS1,CS2,DS1,ER1,0
2207	012504	015032	015030	015054			
2208	012512	015034	000000				
2209							
2210							
2211							
2212							
2213	012516	000	000	000	DF1:	.BYTE	0,0,0,0,0
2214	012521	000	000				
2215	012523	000	000	001	DF2:	.BYTE	0,0,1,0
2216	012526	000					
2217	012527	000	000	001	DF3:	.BYTE	0,0,1,0,0
2218	012532	000	000				
2219							
2220	012534	000	000	000	DF11:	.BYTE	0,0,0,0,0,0,0
2221	012537	000	000	000			
2222	012542	000					
2223	012543	000	000	000	DF14:	.BYTE	0,0,0,0,0,0,0,0
2224	012546	000	000	000			
2225	012551	000	000				
2226	012553	000	000	000	DF15:	.BYTE	0,0,0
2227	012556	000	000	000	DF16:	.BYTE	0,0,0,0,0
2228	012561	000	000				
2229	012563	000	000	000	DF17:	.BYTE	0,0,0,0,0,0,0
2230	012566	000	000	000			
2231	012571	000					
2232	012572	000	000	000	DF20:	.BYTE	0,0,0,0,0,0,0
2233	012575	000	000	000			
2234	012600	000					
2235							
2236	012601	000	000	000	DF21:	.BYTE	0,0,0,0,0
2237	012604	000	000				
2238	012606	000	000	000	DF22:	.BYTE	0,0,0,0
2239	012611	000					
2240	012612	000	000	000	DF24:	.BYTE	0,0,0,0,0,0,0
2241	012615	000	000	000			
2242	012620	000					
2243	012621	000	000	000	DF26:	.BYTE	0,0,0,0,0,0,0,0
2244	012624	000	000	000			
2245	012627	000	000				
2246	012631	000	000	000	DF27:	.BYTE	0,0,0,0,0,0
2247	012634	000	000	000			
2248	012637	000	000	000	DF30:	.BYTE	0,0,0,0,0,0
2249	012642	000	000	000			
2250							
2251	012645	000	000	001	DF31:	.BYTE	0,0,1,0,0,0,0,0
2252	012650	000	000	000			
2253	012653	000	000				
2254	012655	000	000	000	DF32:	.BYTE	0,0,0,1,0,0,0,0,0
2255	012660	001	000	000			
2256	012663	000	000	000			
2257	012666	000	000	000	DF33:	.BYTE	0,0,0,1,0,0,0,0
2258	012671	001	000	000			
2259	012674	000	000				
2260	012676	000	000	000	DF34:	.BYTE	0,0,0,0,0,0,0,0

2261	012701	000	000	000	
2262	012704	000			
2263	012705	000	000	000	DF35: .BYTE 0,0,0,0,0,0,0,0
2264	012710	000	000	000	
2265	012713	000	000		
2266	012715	000	000	000	DF36: .BYTE 0,0,0,0,0,0
2267	012720	000	000	000	
2268	012723	000	000	000	DF37: .BYTE 0,0,0,0,0,0,0,0
2269	012726	000	000	000	
2270	012731	000	000	000	
2271	012734	000	000	000	DF40: .BYTE 0,0,0,0,0,0,0
2272	012737	000	000	000	
2273	012742	000			
2274					
2275		012744			.EVEN
2276	012744	044122	030461	051040	EM42: .ASCIZ /RH11 REGISTER FAILED TO RESPOND TO A "TST" COMMAND/
2277	012752	043505	051511	042524	
2278	012760	020122	040506	046111	
2279	012766	042105	052040	020117	
2280	012774	042522	050123	047117	
2281	013002	020104	047524	040440	
2282	013010	021040	051524	021124	
2283	013016	041440	046517	040515	
2284	013024	042116	000		
2285					
2286	013027	122	030510	020061	EM43: .ASCIZ /RH11 ILLEGAL REGISTER RESPONSE TEST/
2287	013034	046111	042514	040507	
2288	013042	020114	042522	044507	
2289	013050	052123	051105	051040	
2290	013056	051505	047520	051516	
2291	013064	020105	042524	052123	
2292	013072	000			
2293					
2294	013073	115	053117	020105	EM44: .ASCIZ /MOVE BYTE TO WORD COUNT TEST/
2295	013100	054502	042524	052040	
2296	013106	020117	047527	042122	
2297	013114	041440	052517	052116	
2298	013122	052040	051505	000124	
2299					
2300	013130	044502	020123	054502	EM45: .ASCIZ /BIS BYTE TO WORD COUNT/
2301	013136	042524	052040	020117	
2302	013144	047527	042122	041440	
2303	013152	052517	052116	000	
2304					
2305	013157	102	041511	041040	EM46: .ASCIZ /BIC BYTE TO WORD COUNT/
2306	013164	052131	020105	047524	
2307	013172	053440	051117	020104	
2308	013200	047503	047125	000124	
2309					
2310	013206	042522	044507	052123	EM47: .ASCIZ /REGISTER ADDRESS SELECT TEST/
2311	013214	051105	040440	042104	
2312	013222	042522	051523	051440	
2313	013230	046105	041505	020124	
2314	013236	042524	052123	000	
2315					
2316	013243	116	047117	042440	EM50: .ASCIZ /NON EXISTANT DRIVE (NED) TEST/

2317	013250	044530	052123	047101	
2318	013256	020124	051104	053111	
2319	013264	020105	047050	042105	
2320	013272	020051	042524	052123	
2321	013300	000			
2322					
2323	013301	101	020123	042522	EM51: .ASCIZ /AS REGISTER TEST/
2324	013306	044507	052123	051105	
2325	013314	052040	051505	000124	
2326					
2327	013322	052502	051523	040440	EM52: .ASCIZ /BUSS ADDRESS REGISTER DATA TEST/
2328	013330	042104	042522	051523	
2329	013336	051040	043505	051511	
2330	013344	042524	020122	040504	
2331	013352	040524	052040	051505	
2332	013360	000124			
2333					
2334	013362	052502	051523	040440	EM53: .ASCIZ /BUSS ADDRESS REGISTER MOVE BYTE/
2335	013370	042104	042522	051523	
2336	013376	051040	043505	051511	
2337	013404	042524	020122	047515	
2338	013412	042526	041040	052131	
2339	013420	000105			
2340					
2341	013422	044122	030461	044440	EM54: .ASCIZ /RH11 INTERRUPT FAILED TO OCCUR/
2342	013430	052116	051105	052522	
2343	013436	052120	043040	044501	
2344	013444	042514	020104	047524	
2345	013452	047440	041503	051125	
2346	013460	000			
2347					
2348	013461	122	051505	052105	EM55: .ASCIZ /RESET TEST/
2349	013466	052040	051505	000124	
2350					
2351	013474	054115	020106	044502	EM56: .ASCIZ /MXF BIT TEST/
2352	013502	020124	042524	052123	
2353	013510	000			
2354					
2355	013511	125	044516	052502	EM57: .ASCIZ /UNIBUS PARITY BIT TEST/
2356	013516	020123	040520	044522	
2357	013524	054524	041040	052111	
2358	013532	052040	051505	000124	
2359					
2360	013540	047504	051505	051440	EM60: .ASCIZ /DOES SELECTING THE RH11 CLEAR THE UNIT SELECT REGISTER/
2361	013546	046105	041505	044524	
2362	013554	043516	052040	042510	
2363	013562	051040	030510	020061	
2364	013570	046103	040505	020122	
2365	013576	044124	020105	047125	
2366	013604	052111	051440	046105	
2367	013612	041505	020124	042522	
2368	013620	044507	052123	051105	
2369	013626	000			
2370					
2371	013627	104	042517	020123	EM61: .ASCIZ /DOES TRE GET SET BY UNIBUS PARITY ERROR/
2372	013634	051124	020105	042507	

2373	013642	020124	042523	020124	
2374	013650	054502	052440	044516	
2375	013656	052502	020123	040520	
2376	013664	044522	054524	042440	
2377	013672	051122	051117	000	
2378					
2379	013677	116	047117	042440	EM62: .ASCIZ /NON EXISTANT DRIVE (NED) FAILED TO SET (TRE)/
2380	013704	044530	052123	047101	
2381	013712	020124	051104	053111	
2382	013720	020105	047050	042105	
2383	013726	020051	040506	046111	
2384	013734	042105	052040	020117	
2385	013742	042523	020124	052050	
2386	013750	042522	000051		
2387					
2388	013754	047516	020116	054105	EM63: .ASCIZ /NON EXISTANT MEMORY (NEM) FAILED TO SET (TRE)/
2389	013762	051511	040524	052116	
2390	013770	046440	046505	051117	
2391	013776	020131	047050	046505	
2392	014004	020051	040506	046111	
2393	014012	042105	052040	020117	
2394	014020	042523	020124	052050	
2395	014026	042522	000051		
2396					
2397	014032	047520	052122	051440	EM64: .ASCIZ /PORT SELECT FAILED TO CLEAR/
2398	014040	046105	041505	020124	
2399	014046	040506	046111	042105	
2400	014054	052040	020117	046103	
2401	014062	040505	000122		
2402					
2403	014066	047503	052116	047522	EM65: .ASCIZ /CONTROLLER CLEAR FAILED TO CLEAR COMMAND REGISTER/
2404	014074	046114	051105	041440	
2405	014102	042514	051101	043040	
2406	014110	044501	042514	020104	
2407	014116	047524	041440	042514	
2408	014124	051101	041440	046517	
2409	014132	040515	042116	051040	
2410	014140	043505	051511	042524	
2411	014146	000122			
2412					
2413	014150	042522	042523	042514	EM66: .ASCIZ /RESELECT FAILED TO CLEAR COMMAND REGISTER/
2414	014156	052103	043040	044501	
2415	014164	042514	020104	047524	
2416	014172	041440	042514	051101	
2417	014200	041440	046517	040515	
2418	014206	042116	051040	043505	
2419	014214	051511	042524	000122	
2420					
2421	014222	047111	042524	051122	EM67: .ASCIZ /INTERRUPT CAUSED BY RDY.IE BITS SET FAILED TO OCCUR/
2422	014230	050125	020124	040503	
2423	014236	051525	042105	041040	
2424	014244	020131	051040	054504	
2425	014252	044441	020105	044502	
2426	014260	051524	051440	052105	
2427	014266	043040	044501	042514	
2428	014274	020104	047524	047440	

2429	014302	041503	051125	000				
2430								
2431	014307	111	020105	040506	EM70:	.ASCIZ	/IE FAILED TO CLEAR FOLLOWING AN INTERRUPT/	
2432	014314	046111	042105	052040				
2433	014322	020117	046103	040505				
2434	014330	020122	047506	046114				
2435	014336	053517	047111	020107				
2436	014344	047101	044440	052116				
2437	014352	051105	052522	052120				
2438	014360	000						
2439								
2440	014361	101	020123	042522	EM71:	.ASCIZ	/AS REGISTER WRITE TEST/	
2441	014366	044507	052123	051105				
2442	014374	053440	044522	042524				
2443	014402	052040	051505	000124				
2444								
2445								
2446								
2447	014410	041520	020040	020040	DH42:	.ASCII	/PC	TEST ILLEGAL/<15><12>
2448	014416	020040	042524	052123				
2449	014424	020040	020040	046111				
2450	014432	042514	040507	006514				
2451	014440	012						
2452	014441	040	020040	020040		.ASCIZ	/	NO ADDRESS/
2453	014446	020040	047040	020117				
2454	014454	020040	020040	040440				
2455	014462	042104	042522	051523				
2456	014470	000						
2457	014471	120	020103	020040	DH44:	.ASCII	/PC	TEST REGISTR CONT CONT/<15><12>
2458	014476	020040	052040	051505				
2459	014504	020124	020040	051040				
2460	014512	043505	051511	051124				
2461	014520	041440	047117	020124				
2462	014526	020040	041440	047117				
2463	014534	006524	012					
2464	014537	040	020040	020040		.ASCIZ	/	NO ADDRESS GOOD BAD/
2465	014544	020040	047040	020117				
2466	014552	020040	020040	040440				
2467	014560	042104	042522	051523				
2468	014566	043440	047517	020104				
2469	014574	020040	041040	042101				
2470	014602	000						
2471	014603	120	020103	020040	DH54:	.ASCII	/PC	TEST REGISTR CONT/<15><12>
2472	014610	020040	052040	051505				
2473	014616	020124	020040	051040				
2474	014624	043505	051511	051124				
2475	014632	041440	047117	006524				
2476	014640	012						
2477	014641	040	020040	020040		.ASCIZ	/	NO ADDRESS/
2478	014646	020040	047040	020117				
2479	014654	020040	020040	040440				
2480	014662	042104	042522	051523				
2481	014670	000						
2482								
2483		014672						
2484								

.EVEN

CZRJHDO,RP04/5/6 DSKLS CTRLR2 MACY11 30A(1052) 24-MAY-79 15:07 ^{K 4} PAGE 50
CZRJHD.P11 28-MAR-79 09:16 ERROR POINTER TABLE

SEQ 0049

2485	014672	001116	017330	045662	DT42:	.WORD	\$ERRPC,TSTNM,REGADR,0
2486	014700	000000					
2487	014702	001116	017330	045662	DT44:	.WORD	\$ERRPC,TSTNM,REGADR,\$GDDAT,\$BDDAT,0
2488	014710	001124	001126	000000			
2489	014716	001116	017330	045662	DT54:	.WORD	\$ERRPC,TSTNM,REGADR,\$GDDAT,0
2490	014724	001124	000000				

CZRJHDO,RP04/5/6 DSKLS CTRLR2 MACY11 30A(1052) 24-MAY-79 15:07 L 4
CZRJHD.P11 28-MAR-79 09:16 ERROR POINTER TABLE PAGE 51

SEQ 0050

2491	014730	000	000	000	DF42:	.BYTE	0,0,0
2492	014733	000	000	000	DF44:	.BYTE	0,0,0,0,0
2493	014736	000	000				
2494	014740	000	000	000	DF54:	.BYTE	0,0,0,0
2495	014743	000					

```

2496      .SBTTL  HARDWARE REGISTER BIT DEFINITIONS
2497
2498
2499      ;*****
2500      ;RH11 REGISTERS
2501      ;*****
2502
2503
2504      ;WORD COUNT REGISTER (RHWG)
2505      ;EACH BIT IS CALLED BY BIT NUMBER
2506
2507
2508
2509      ;BUS ADDRESS REGISTER (RHBA)
2510      ;EACH BIT IS CALLED BY BIT NUMBER
2511
2512
2513
2514      ;CONTROL AND STATUS REGISTER 2 (RHCS2)
2515
2516      000001      US1=      1      ;UNIT SELECT (BIT #0)
2517      000002      US2=      2      ;UNIT SELECT (BIT #1)
2518      000004      US4=      4      ;UNIT SELECT (BIT #2)
2519      000010      BAI=     10      ;BUS ADDRESS INCREMENT INHIBIT (BIT #3)
2520      000020      PAT=     20      ;INVERT PARITY ON MASS BUS TO EVEN (BIT #4)
2521      000040      CLR=     40      ;CLEAR (BIT #5)
2522      000100      IR=     100      ;INPUT READY (BIT #6)
2523      000200      OR=      200      ;OUTPUT READY (BIT #7)
2524      000400      MPE=     400      ;MASS BUS PARITY ERROR (BIT #8)
2525      001000      MXF=    1000      ;MISSED TRANSFER ERROR (BIT #9)
2526      002000      PGE=    2000      ;PROGRAM ERROR (BIT #10)
2527      004000      NEM=    4000      ;NON EXISTANT MEMORY (BIT #11)
2528      010000      NED=   10000      ;NON EXISTANT DRIVE (BIT #12)
2529      020000      UPE=   20000      ;UNIBUS PARITY ERROR (BIT #13)
2530      040000      WCE=   40000      ;WRITE CHECK ERROR (BIT #14)
2531      100000      DLT=  100000      ;DATA LATE (BIT #15)
2532
2533      ;DATA BUFFER REGISTER (RHDB)
2534      ;EACH BIT IS CALLED BY BIT NUMBER
  
```

```
2535 ::*****
2536 ::RP04 REGISTERS
2537 ::*****
2538
2539
2540
2541 ;CONTROL AND STATUS 1 REGISTER. (#00)
2542
2543 GO= 1 ;GO (BIT #0)
2544 IE= 100 ;INTERRUPT ENABLE (BIT #6)
2545 RDY= 200 ;READY (BIT #7)
2546 A16= 400 ;HIGH ORDER UNIBUS BITS (BIT #8)
2547 A17= 1000 ;HIGH ORDER UNIBUS BITS (BIT #9)
2548 PSEL= 2000 ;PORT SELECT (BIT #10)
2549 DVA= 4000 ;DEVICE AVAILABLE (BIT #11)
2550 MCPE= 20000 ;MASSBUSS PARITY ERROR (BIT #13)
2551 TRE= 40000 ;TRANSFER ERROR (BIT #14)
2552 SC= 100000 ;SPECIAL CONDITION (BIT #15)
2553
2554 ;STATUS REGISTER (RHDS1) (#01)
2555
2556 DFS= 1 ;DRIVE FORWARD 5"/SEC. (BIT #0)
2557 DFF20= 2 ;DRIVE FORWARD 20"/SEC. (BIT #1)
2558 DIGB= 4 ;DRIVE TO INNER GAVRD BAND (BIT #2)
2559 GRV= 10 ;GO REVERSE (BIT #3)
2560 DL64= 20 ;DIFFERENCE LESS THAN 64 (BIT #4)
2561 DE1= 40 ;DIFFERENCE EQUALS 1 (BIT #5)
2562 VV= 100 ;VOLUME VALID (BIT #6)
2563 DRY= 200 ;DRIVE READY (BIT #7)
2564 DPR= 400 ;DRIVE PRESENT (BIT #8)
2565 PROG= 1000 ;PROGRAMABLE (BIT #9)
2566 LST= 2000 ;LAST SECTOR TRANSFERRED (BIT #10)
2567 WRL= 4000 ;WRITE LOCK (BIT #11)
2568 MOL= 10000 ;MEDIUM ON-LINE (BIT #12)
2569 PIP= 20000 ;POSITIONING OPERATION IN PROGRESS (BIT #13)
2570 ERR= 40000 ;COMPOSIT ERROR. (BIT #14)
2571 ATA= 100000 ;ATTENTION ACTIVE (BIT #15)
2572
2573 ;ERROR REGISTER #01 (RHER1) (#02)
2574 ILF= 1 ;ILLEGAL FUNCTION (BIT #0)
2575 ILR= 2 ;ILLEGAL REGISTER (BIT #1)
2576 RMR= 4 ;REGISTER MODIFICATION REFUSED (BIT #2)
2577 PAR= 10 ;PARITY ERROR (BIT #3)
2578 FER= 20 ;FORMAT ERROR (BIT #4)
2579 WCF= 40 ;WRITE CLOCK FAIL (BIT #5)
2580 ECH= 100 ;ECC HARD ERROR (BIT #6)
2581 HCE= 200 ;HEADER COMPARE ERROR (BIT #7)
2582 HCRC= 400 ;HEADER CRC ERROR (BIT #8)
2583 AOE= 1000 ;ADDRESS OVERFLOW ERROR (BIT #9)
2584 IAE= 2000 ;INVALID ADDRESS ERROR (BIT #10)
2585 WLE= 4000 ;WRITE LOCK ERROR (BIT #11)
2586 DTE= 10000 ;DRIVE TIMING ERROR (BIT #12)
2587 OPI= 20000 ;OPERATION INCOMPLETE (BIT #13)
2588 UNS= 40000 ;DRIVE UNSAFE (BIT #14)
2589 DCK= 100000 ;DATA CHECK ERROR (BIT 15)
2590
```

2591		;MAINTAINABILITY REGISTER (RHMR)(#03)	
2592			
2593	000001	DMD= 1	;DIAGNOSTIC MODE (BIT #0)
2594	000002	MCLK= 2	;MAINTAINABILITY CLOCK (BIT #1)
2595	000004	MINX= 4	;MAINTAINABILITY INDEX (BIT #2)
2596	000010	MSTCK= 10	;MAINTAINABILITY SECTOR CLOCK (BIT #3)
2597	000020	MRD= 20	;MAINTAINABILITY READ (BIT #4)
2598	000040	MWR= 40	;MAINTAINABILITY WRITE (BIT #5)
2599	000200	DENVL= 200	;DATA ENVELOPE (BIT #7)
2600	000400	ZER= 400	;ZERO DETECT (BIT #8)
2601	001000	DTSY= 1000	;MAINTAINABILITY SYNC DETECTED (BIT #9)
2602			
2603		;ATTENTION SUMMARY PSEUDO-REGISTER (RHAS) (#04)	
2604			
2605	000001	AT0= 1	;DEVICE 0 (BIT #0)
2606	000002	AT1= 2	;DEVICE 1 (BIT #1)
2607	000004	AT2= 4	;DEVICE 2 (BIT #2)
2608	000010	AT3= 10	;DEVICE 3 (BIT #3)
2609	000020	AT4= 20	;DEVICE 4 (BIT #4)
2610	000040	AT5= 40	;DEVICE 5 (BIT #5)
2611	000100	AT6= 100	;DEVICE 6 (BIT #6)
2612	000200	AT7= 200	;DEVICE 7 (BIT #7)
2613			
2614			
2615			
2616			
2617			
2618		;DESIRED SECTOR/TRACK ADDRESS REGISTER (RHDST) (#1)	
2619		;EACH BIT IS CALLED BY BIT NUMBER	
2620			
2621			
2622			
2623			
2624			
2625		;DRIVE TYPE REGISTER (RHDT) (#06)	
2626		;EACH BIT IS CALLED BY BIT NUMBER	
2627			
2628			
2629			
2630			
2631			
2632		;LOOK-AHEAD REGISTER (RHLA) (#07)	
2633			
2634	000001	EXT1= 1	;EXTENSION 1 (BIT #0)
2635	000002	EXT2= 2	;EXTENSION 2 (BIT #1)
2636	000004	EXT4= 4	;EXTENSION 3 (BIT #2)
2637	000010	EXT10= 10	;EXTENSION 4 (BIT #3)
2638	000020	EXT20= 20	;EXTENSION 5 (BIT #4)
2639	000040	EXT40= 40	;EXTENSION 6 (BIT #5)
2640	000100	SC1= 100	;SECTOR COUNT FIELD 0 (BIT #6)
2641	000200	SC2= 200	;SECTOR COUNT FIELD 1 (BIT #7)
2642	000400	SC4= 400	;SECTOR COUNT FIELD 2 (BIT #8)
2643	001000	SC10= 1000	;SECTOR COUNT FIELD 3 (BIT #9)
2644	002000	SC20= 2000	;SECTOR COUNT FIELD 4 (BIT #10)
2645	004000	TRK1= 4000	;TRACK FIELD 1 (BIT #11)
2646	010000	TRK2= 10000	;TRACK FIELD 2 (BIT #12)

2647	020000	TRK4= 20000	;TRACK FIELD 3 (BIT #13)
2648	040000	TRK10= 40000	;TRACK FIELD 4 (BIT #14)
2649	100000	TRK20= 100000	;TRACK FIELD 5 (BIT #15)
2650			
2651		;ERROR REGISTER #2 (RMER2) (#10)	
2652			
2653	000001	WCU= 1	;WRITE CURRENT UNSAFE (BIT #0)
2654	000002	CSF= 2	;CURRENT SINK FAILURE (BIT #1)
2655	000004	WSU= 4	;WRITE SELECT UNSAFE (BIT #2)
2656	000010	CSU= 10	;CURRENT SWITCH UNSAFE (BIT #3)
2657	000020	MSE= 20	;MOTOR SEQUENCE ERROR (BIT #4)
2658	000040	TDF= 40	;TRANSITIONS DETECTOR FAILURE (BIT #5)
2659	000100	TUF= 100	;TRANSITIONS UNSAFE (BIT #6)
2660	000200	FEN= 200	;FAILSAFE ENABLED (BIT #7)
2661	000400	WRU= 400	;WRITE READY UNSAFE (BIT #8)
2662	001000	MHS= 1000	;MULTIPLE HEAD SELECT (BIT #9)
2663	002000	NHS= 2000	;NO HEAD SELECTION (BIT #10)
2664	004000	IXE= 4000	;INDEX ERROR (BIT #11)
2665	010000	VU30= 10000	;30VOLT UNSAFE (BIT #12)
2666	020000	PLU= 20000	;PLO UNSAFE (BIT #13)
2667	100000	ACU= 100000	;ACUNSAFE (BIT #15)
2668			
2669		;OFFSET REGISTER (RHOF) (#11)	
2670			
2671	000001	OF25= 1	;OFFSET 25 MICRO INCHES (BIT #0)
2672	000002	OF50= 2	;OFFSET 50 MICRO INCHES (BIT #1)
2673	000004	OF100= 4	;OFFSET 100 MICRO INCHES (BIT #2)
2674	000010	OF200= 10	;OFFSET 200 MICRO INCHES (BIT #3)
2675	000020	OF400= 20	;OFFSET 400 MICRO INCHES (BIT #4)
2676	000040	OF800= 40	;OFFSET 800 MICRO INCHES (BIT #5)
2677			
2678	000200	OFREV= 200	;OFFSET NEGATIVE (REVERSE) (BIT #7)
2679	002000	HCI= 2000	;HEADER COMPARE INHIBIT (BIT #10)
2680	004000	ECI= 4000	;ERROR CORRECTION CODE INHIBIT (BIT #11)
2681	010000	FMT22= 10000	;FORMAT BIT (BIT #12)
2682			
2683			
2684			
2685			
2686			
2687		;DESIRED CYLINDER ADDRESS (RHCA) (#12)	
2688		;EACH BIT IS CALLED BY BIT NUMBER.	
2689			
2690			
2691			
2692			
2693			
2694		;CURRENT CYLINDER ADDRESS (RHCC) (#13)	
2695		;EACH BIT IS CALLED BY BIT NUMBER	
2696			
2697			
2698			
2699			
2700			
2701		;SERIAL NUMBER REGISTER (RMSN) (#14)	
2702		;EACH IS CALLED BY BIT NUMBER	

2703
2704
2705
2706
2707
2708
2709
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2720
2721
2722
2723
2724
2725
2726
2727
2728

000001
000002
000010
000020
000040
000100
040000
100000

;ERROR REGISTER #03 (RHER3) (#15)

PSU=	1	;PACK SPEED UNSAFE (BIT #0)
VUF=	2	;VELOCITY UNSAFE (BIT #1)
UWR=	10	;ANY UNSAFE EXCEPT READ/WRITE (BIT #3)
PRE=	20	;DISK PACK ROTATION ERROR (BIT #4)
ACL=	40	;AC LOW (BIT #5)
DCL=	100	;DC LOW (BIT #6)
SKI=	40000	;SEEK INCOMPLETE (BIT #14)
OCYL=	100000	;OFF CYLINDER (BIT #15)

;FCC POSITION REGISTER (RHEC1) (#16)
;EACH BIT IS CALLED BY BIT NUMBER

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

2729
2730
2731
2732
2733
2734
2735
2736
2737
2738
2739
2740
2741
2742
2743
2744
2745
2746
2747
2748
2749
2750
2751
2752
2753
2754
2755
2756
2757
2758
2759
2760
2761
2762
2763
2764
2765
2766
2767
2768
2769
2770
2771
2772

014744 000254

014746 176722
014750 176702
014752 176704
014754 176710
014756 176700
014760 176714
014762 176706
014764 176740
014766 176732
014770 176734
014772 176742
014774 176716
014776 176724
015000 176712
015002 176726
015004 176730
015006 176744
015010 176746
015012 176720
015014 176736

.SBTTL REGISTER ADDRESSES

:RP04/5/6 VECTOR ADDRESS

RPVEC: 254

:RP04/5/6 VECTOR ADDRESS

:NOTE: THE CONTENTS OF THESE LOCATIONS WILL BE DIFFERENT
: IF THE "CHANGE BASE ADDRESS" ROUTINE IS USED.
: THIS ROUTINE STARTS AT LOCATION TAGGED "BASECH"

RHDB:	176722	:DATA BUFFER
RHWC:	176702	:WORD COUNT
RHBA:	176704	:BUS ADDRESS
RHCS2:	176710	:CONTROL AND STATUS
RHCS1:	176700	:CONTROL AND STATUS 1 SEE NOTE ABOVE
RHER1:	176714	:ERROR #1 SEE NOTE ABOVE
RHDST:	176706	:DESIRED SECTOR / TRACK ADDRESS
RHER2:	176740	:ERROR #2
RHOF:	176732	:OFFSET
RHCA:	176734	:DESIRED CYLINDER ADDRESS
RHER3:	176742	:ERROR #3
RHAS:	176716	:ATTENTION SUMMARY SEE NOTE ABOVE
RHMR:	176724	:MAINTAINABILITY
RHDS1:	176712	:DRIVE STATUS
RHDT:	176726	:DRIVE TYPE
RHSN:	176730	:SERIAL NUMBER SEE NOTE ABOVE
RHEC1:	176744	:ECC POSITION
RHEC2:	176746	:ECC PATTERN
RHLA:	176720	:LOOK AHEAD
RHCC:	176736	:CURRENT CYLINDER ADDRESS

:ADDITIONAL REGISTERS LOCATED IN THE RH70 CONTROLLER

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

```
2773
2774      ;THE FOLLOWING LOCATIONS ARE RESERVED FOR REGISTER SAVES
2775      ;ANY TIME THERE IS AN ERROR ALL THESE WILL BE FILLED
2776      ;ONLY SOME MAY BE PRINTED BUT ALL WILL BE FILLED TRUE
2777      ;FOR THE TIME JUST AFTER THE 'ERROR' ERROR COMMAND
2778
2779 015022 000000      DB:      0      ;DATA BUFFER
2780 015024 000000      WC:      0      ;WORD COUNT
2781 015026 000000      BA:      0      ;BUS ADDRESS
2782 015030 000000      CS2:     0      ;CONTROL AND STATUS 2
2783
2784
2785 015032 000000      CS1:      0      ;CONTROL AND STATUS 1
2786 015034 000000      ER1:      0      ;ERROR #1
2787 015036 000000      DST:      0      ;DESIRED SECTOR/TRACK ADDRESS
2788 015040 000000      ER2:      0      ;ERROR #2
2789 015042 000000      OF:      0      ;OFFSET
2790 015044 000000      CA:      0      ;DESIRED CYLINDER ADDRESS
2791 015046 000000      ER3:      0      ;ERROR #3
2792 015050 000000      AS:      0      ;ATTENTION SUMMARY
2793 015052 000000      MR:      0      ;MAINTAINABILITY
2794 015054 000000      DS1:      0      ;DRIVE STATUS
2795 015056 000000      DT:      0      ;DRIVE TYPE
2796 015060 000000      SN:      0      ;SERIAL NUMBER
2797 015062 000000      EC1:      0      ;ECC POSITION
2798 015064 000000      EC2:      0      ;ECC PATTERN
2799 015066 000000      LA:      0      ;LOOK-AHEAD
2800 015070 000000      CC:      0      ;CURRENT CYLINDER ADDRESS
2801
2802
```

2803 :FLAGS AND INTERNAL PROGRAM CONTROL WORDS
2804
2805
2806
2807 015072 000010 UNITS: .BLKW 8. ;THIS IS FILLED WITH -1
2808 015112 000000 UNIT: .WORD 0 ;UNIT UNDER TEST
2809 015114 000000 NUNIT: .WORD 0 ;NUMBER OF UNITS PRESENT
2810 ;USED TO KEEP TRACK OF UNIT UNDER TEST
2811 015116 000000 NUNIT: .WORD 0 ;USED TO DETERMIN IF THERE ARE MORE
2812 ;THAN ONE UNIT
2813 015120 000000 SELECT: .WORD 0 ;ALL ONES INDICATE UNIT TO BE SELECTED
2814 015122 000000 UNITSL: .WORD 0 ;UNIT NO. SELECTED
2815
2816 015124 000000 ERFLGS: 0 ;ERROR FLAG
2817
2818 015126 000000 SAVDT: 0 ;SAVE DRIVE TYPE REGISTER
2819 ;FOR COMPARISON IN DRIVE CLEAR TEST
2820 ;AND RH INIT TEST
2821 015130 000000 SAVSN: 0 ;SAVE SERIAL NUMBER REGISTER
2822 ;FOR COMPARISON IN DRIVE CLEAR TEST
2823 ;AND RH INIT TEST
2824
2825 015132 000000 PCJSR: 0 ;SAVE PC OF JSR WHICH GAVE THE ERROR
2826
2827 015134 000000 ATTENT: 0 ;ATTENTION BIT FOR PRESENT UNIT
2828 015136 000000 TOTALAT: 0 ;TOTAL ATTENTION BITS
2829
2830 015140 000000 TMPILL: 0 ;TEMPORARY ILLEGAL FUNCTION
2831
2832 015142 000000 TSECC: 0 ;FLAG TO SAY IF ECC TEST OR NOT
2833 ;WHEN =177777 IT IS AN ECC TEST
2834 ;WHEN =0 IT IS NOT AN ECC TEST
2835
2836 015144 000000 TESDTE: 0 ;FLAG TO SAY IF DRIVE TIMING ERROR OR NOT
2837 ;WHEN = 177777 IT IS A DTE TEST
2838 ;WHEN = 0 IT IS NOT A DTE TEST
2839
2840 015146 000000 TAGDTE: 0 ;TEMPORARY TAG USED IN DRIVE TIMING
2841 ;ERROR TEST
2842

```

2843
2844
2845                ;FUNCTION EQUATES
2846
2847                ;TABLE OF COMMAND FUNCTIONS FOR RHCS1
2848                ;THEN "GO" BIT HAS TO BE SET
2849
2850 015150          FUTABL:
2851 015150          NOPERA: 0                ;NO OPERATION
2852 015152          UNLOAD: 2                ;UNLOAD (STAND BY)
2853 015154          RECAL: 6                ;RECALIBRATE
2854 015156          DCLEAR: 10              ;DRIVE CLEAR
2855 015160          RELEAS: 12              ;RELEASE (DUAL-PORT OPERATION)
2856 015162          SERCH: 30              ;SEARCH COMMAND
2857 015164          WRCHK: 50              ;WRITE CHECK DATA
2858 015166          WRCHDT: 52             ;WRITE CHECK HEADER AND DATA
2859 015170          WRIDAT: 60             ;WRITE DATA
2860 015172          WRIFOR: 62             ;WRITE HEADER AND DATA (FORMAT)
2861 015174          READAT: 70             ;READ DATA
2862 015176          REFOR: 72              ;READ HEADER AND DATA
2863 015200          SEECOM: 4               ;SEEK COMMAND
2864 015202          OFSETC: 14             ;OFFSET COMMAND
2865 015204          RETCL: 16              ;RETURN TO CENTERLINE
2866 015206          PKACK: 22              ;PACK ACKNOWLEDGE
2867 015210          READIN: 20             ;READ IN
2868 015212          ILLEGL: .WORD          ;COMPUTED ILLEGAL FUNCTION
2869
2870
2871                ;DATA BUFFER FOR READ WRITE
2872
2873
2874
2875 015220          WRFROM: .BLKW 274.        ;WRITE FROM THIS BUFFER
2876 016264          REINTO: .BLKW 274.        ;READ INTO THIS BUFFER
2877
2878
2879 017330          TSTNM: 0                ;TEST NUMBER
2880 017332          FIRST: 0                ;IF ZERO WILL TYPE HEADER
2881                                           ;IF ONES WILL NOT TYPE HEADER
2882
2883 017334          RH70: 0                  ;FLAG = 1 FOR RH70 CONTROLLER
2884                                           ;FLAG = 0 FOR RH11
2885 017336          SILOSZ: .WORD 0          ;RH SILO SIZE
2886
2887
2888
2889
2890                ;TABLE FOR ATTENTION BITS
2891                ;ATTENTION TABLE
2892
2893 017340          001      002      004      ATABLE: .BYTE 1,2,4,10,20,40,100,200
2894 017343          010      020      040
2895 017346          100      200

```

```
2896
2897
2898
2899
2900
2901 017350 012737 177777 015120 BEGIN2: MOV    #-1,@#SELECT    ;SELECT UNIT
2902 017356 000402          BR      START
2903 017360 005037 015120  BEGIN:  CLR      @#SELECT    ;DO NOT SELECT UNIT
2904                                     ;NORMAL RUN
2905
2906 017364          START:
2907 017364 000005          RESET
2908                                     .SBTTL  INITIALIZE THE COMMON TAGS
2909                                     ;;CLEAR THE-COMMON TAGS ($CMTAG)-AREA
2910 017366 012706 001100          MOV    # $CMTAG,R6    ;;FIRST LOCATION TO BE CLEARED
2911 017372 005026          CLR      (R6)+    ;;CLEAR MEMORY LOCATION
2912 017374 022706 001140          CMP    #SWR,R6    ;;DONE?
2913 017400 001374          BNE      #-6    ;;LOOP BACK IF NO
2914 017402 012706 001000          MOV    #STACK,SP    ;;SETUP THE STACK POINTER
2915                                     ;;INITIALIZE A FEW VECTORS
2916 017406 012737 057530 000020          MOV    # $SCOPE,@#IOTVEC    ;;IOT VECTOR FOR SCOPE ROUTINE
2917 017414 012737 000340 000022          MOV    #340,@#IOTVEC+2    ;;LEVEL 7
2918 017422 012737 061746 000030          MOV    # $ERROR,@#EMTVEC    ;;EMT VECTOR FOR ERROR ROUTINE
2919 017430 012737 000340 000032          MOV    #340,@#EMTVEC+2    ;;LEVEL 7
2920 017436 012737 062516 000034          MOV    # $TRAP,@#TRAPVEC    ;;TRAP VECTOR FOR TRAP CALLS
2921 017444 012737 000340 000036          MOV    #340,@#TRAPVEC+2    ;;LEVEL 7
2922 017452 012737 062606 000024          MOV    # $PWRDN,@#PWRVEC    ;;POWER FAILURE VECTOR
2923 017460 012737 000340 000026          MOV    #340,@#PWRVEC+2    ;;LEVEL 7
2924 017466 005037 001212          CLR      $TIMES    ;;INITIALIZE NUMBER OF ITERATIONS
2925 017472 005037 001214          CLR      $ESCAPE    ;;CLEAR THE ESCAPE ON ERROR ADDRESS
2926 017476 012737 000001 001115          MOVB   #1,$ERMAX    ;;ALLOW ONE ERROR PER TEST
2927 017504 012737 017504 001106          MOV    #.,$LPADR    ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
2928 017512 012737 017512 001110          MOV    #.,$LPERR    ;;SETUP THE ERROR LOOP ADDRESS
2929                                     ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
2930                                     ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
2931 017520 013746 000004          MOV    @#ERRVEC,-(SP)    ;;SAVE ERROR VECTOR
2932 017524 012737 017560 000004          MOV    #64$,@#ERRVEC    ;;SET UP ERROR VECTOR
2933 017532 012737 177570 001140          MOV    #DSWR,SWR    ;;SETUP FOR A HARDWARE SWICH REGISTER
2934 017540 012737 177570 001142          MOV    #DDISP,DISPLAY    ;;AND A HARDWARE DISPLAY REGISTER
2935 017546 022777 177777 161364          CMP    #-1,@SWR    ;;TRY TO REFERENCE HARDWARE SWR
2936 017554 001012          BNE      66$    ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
2937                                     ;;AND THE HARDWARE SWR IS NOT = -1
2938 017556 000403          BR      65$    ;;BRANCH IF NO TIMEOUT
2939 017560 012716 017566          64$:  MOV    #65$,(SP)    ;;SET UP FOR TRAP RETURN
2940 017564 000002          RTI
2941 017566 012737 000176 001140          65$:  MOV    #SWREG,SWR    ;;POINT TO SOFTWARE SWR
2942 017574 012737 000174 001142          MOV    #DISPREG,DISPLAY
2943 017602 012637 000004          66$:  MOV    (SP)+,@#ERRVEC    ;;RESTORE ERROR VECTOR
2944
2945
2946
2947
2948 017606 012737 000000 177776  STARTA: MOV    #0,PS    ;SET PROCESSOR STATUS TO 0
2949 017614 012777 057442 175122          MOV    #RPVECT,@RPVEC    ;THIS IS FOR UNTIMELY DRIVE INTERRUPTS
2950 017622 004737 060506          JSR    PC,@$TKINT    ;INITIALIZE TTY KEYBOARD
2951 017626 005737 017332          TST    @#FIRST    ;IS THIS FIRST TIME ROUND ?
```

```
2952 017632 001001      BNE      1$      ;DON'T TYPE HEADER IF NOT
2953 017634 000402      BR       2$      ;TYPE HEADER IF SO
2954
2955 017636 000137 020660 1$:      JMP      @#SND1
2956
2957 017642      2$:
2958 017642 104401 017650      TYPE      ,65$      ;;TYPE ASCII STRING
2959 017646 000435      BR       64$      ;;GET OVER THE ASCII
2960      ;;65$: .ASCIIZ <15><12>?RP04/5/6 DISKLESS CONTROLLER TEST - PART II - CZRJH-D?
2961 017742      64$:
2962
2963 017742 104401 017750      TYPE      ,67$      ;;TYPE ASCII STRING
2964 017746 000415      BR       66$      ;;GET OVER THE ASCII
2965      ;;67$: .ASCIIZ <15><12>/REVISION DATE: DEC-78/<15><12>
2966 020002      66$:
2967
2968 020002 104401 020010      TYPE      ,69$      ;;TYPE ASCII STRING
2969 020006 000433      BR       68$      ;;GET OVER THE ASCII
2970      ;;69$: .ASCIIZ <15><12>/ALL DCL'S UNDER TEST MUST BE LOCKED ON CORRECT PORT/
2971 020076      68$:
2972 020076 104401 020104      TYPE      ,71$      ;;TYPE ASCII STRING
2973 020102 000433      BR       70$      ;;GET OVER THE ASCII
2974      ;;71$: .ASCIIZ <15><12>/IF CHANGES ARE REQUIRED ON PORT SWITCH, A CYCLE UP/
2975 020172      70$:
2976 020172 104401 020200      TYPE      ,73$      ;;TYPE ASCII STRING
2977 020176 000436      BR       72$      ;;GET OVER THE ASCII
2978      ;;73$: .ASCIIZ <15><12>/SEQUENCE IS REQUIRED FOR STROBING THE PORT SELECT FLOP/<15><12>
2979 020274      72$:
2980 020274 104401 020302      TYPE      ,75$      ;;TYPE ASCII STRING
2981 020300 000431      BR       74$      ;;GET OVER THE ASCII
2982      ;;75$: .ASCIIZ <15><12>/ALL DCL'S NOT UNDER TEST MUST BE SWITCHED OFF/<15><12>
2983 020364      74$:
2984 020364 104401 020372      TYPE      ,77$      ;;TYPE ASCII STRING
2985 020370 000427      BR       76$      ;;GET OVER THE ASCII
2986      ;;77$: .ASCIIZ <15><12>/*****/
2987 020450      76$:
2988 020450 104401 020456      TYPE      ,79$      ;;TYPE ASCII STRING
2989 020454 000427      BR       78$      ;;GET OVER THE ASCII
2990      ;;79$: .ASCIIZ <15><12>/IF THIS IS NOT DONE, ERRORS WILL RESULT ON/
2991 020534      78$:
2992 020534 104401 020542      TYPE      ,81$      ;;TYPE ASCII STRING
2993 020540 000415      BR       80$      ;;GET OVER THE ASCII
2994      ;;81$: .ASCIIZ <15><12>/'NED' TESTS (T21 & T36)/
2995 020574      80$:
2996 020574 104401 020602      TYPE      ,83$      ;;TYPE ASCII STRING
2997 020600 000427      BR       82$      ;;GET OVER THE ASCII
2998      ;;83$: .ASCIIZ <15><12>/*****/
2999 020660      82$:
3000
3001 020660 012737 177777 017332 SND1: MOV      #-1,@#FIRST      ;NEXT TIME DO NOT GIVE HEADER
3002
3003      .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
3004 020666 005737 000042      TST      @#42      ;;ARE WE RUNNING UNDER XXDP/ACT?
3005 020672 001006      BNE      64$      ;;BRANCH IF YES
3006 020674 023727 001140 000176      CMP      SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
3007 020702 001005      BNE      65$      ;;BRANCH IF NO
```

CZRJHDO,RP04/5/6 DSKLS CTRLR2 MACY11 30A(1052) 24-MAY-79 15:07 K 5
CZRJHD.P11 28-MAR-79 09:16 GET VALUE FOR SOFTWARE SWITCH REGISTER PAGE 63

SEQ 0062

```
3008 020704 104406          GTSWR          ;;GET SOFT-SWR SETTINGS
3009 020706 000403          BR          65$
3010 020710 112737 000001 001134 64$:      MOVB      #1,$AUTOB      ;;SET AUTO-MODE INDICATOR
3011 020716          65$:
3012
3013 020716 032777 010000 160214 RH70CK: BIT      #SW12,@SWR          ;LOOK TO SEE IF USING RH70
3014 020724 001403          BEQ      3$          ;IF SW12 = 0, SKIP NEXT
3015 020726 012737 000001 017334          MOV      #1,@#RH70          ;IF SW12 = 1, CU IS AN RH70
3016
3017 020734 005737 015120          3$:      TST      @#SELECT          ;200 START?
3018 020740 001434          BEQ      TST1          ;GO TO FIRST TEST IF STARTING FROM 200 -----)
3019 020742 104401 020750          TYPE      ,65$          ;;TYPE ASCII STRING
3020 020746 000422          BR          64$          ;;GET OVER THE ASCII
3021          ;;65$:      .ASCIIZ <15><12>/SELECT UNIT NUMBER TO BE TESTED ?/
3022          64$:
3023 021014          RDOCT
3024 021016 042716 177770          BIC      #177770,(SP)          ;ONLY KEEP LAST 3 BITS
3025 021022 011637 015112          MOV      (SP),@#UNIT          ;SAVE UNIT TO BE TESTED
3026 021026 012637 015122          MOV      (SP)+,@#UNITSL        ;SAVE UNIT TO BE TESTED
3027
3028
3029
```

```

3030
3031      ;*****
3032      ;*TEST 1      REFERENCE EACH REGISTER
3033
3034      ;*      REFERENCE EACH REGISTER BY A MOVE INSTRUCTION
3035
3036      ;*****
3037 021032 000004      TST1: SCOPE
3038 021034 012737 000001 001212      MOV #1,$TIMES      ;;DO 1 ITERATION
3039 021042 012706 001000      MOV #STACK, SP      ;SET UP STACK POINTER
3040 021046 012737 000001 017330      MOV #TTNO,@#TSTNM      ;THIS SAVES TEST NUMBER
3041 021054 012737 061756 000030      MOV #REGSA1,@#EMTVEC ;ERROR VECTOR SO THAT
3042      ;NO REGISTERS ARE SAVED
3043 021062 012737 021110 000004      MOV #2$,@#ERRVEC      ;SET UP FOR BUS TIMEOUT
3044 021070 012700 000024      MOV #24,R0      ;THERE ARE 24 REG TO TEST
3045 021074 012701 014746      MOV #RHDB,R1      ;R1 NOW HAS ADDR OF ADDR OF FIRST REG.
3046 021100 013102      1$: MOV @ (R1)+,R2      ;READ HARDWARE REG.
3047 021102 005300      DEC R0      ;COUNT DOWN
3048 021104 001375      BNE 1$      ;BRANCH IF 24 NOT DONE
3049 021106 000454      BR 3$      ;BRANCH IF 24 DONE
3050 021110 012737 000006 000004      2$: MOV #ERRVEC+2,@#ERRVEC ;RESTORE TRAP CATCHER
3051 021116 022626      CMP (SP)+,(SP)+      ;CLEAN STACK
3052 021120 016137 177776 001200      MOV -2(R1), $TMP1      ;STORE FAILING REG ADDR
3053 021126 104015      ERROR 1$      ;REGISTER NON EXISTANT
3054 021130 032777 020000 160002      BIT #SW13,@SWR      ;INHIBIT ERROR PRINTOUT ?
3055 021136 001036      BNE 4$      ;BRANCH IF YES
3056 021140 104401 021146      TYPE ,65$      ;;TYPE ASCIZ STRING
3057 021144 000427      BR 64$      ;;GET OVER THE ASCIZ
3058      ;;65$: .ASCIZ <15><12>/TO CHANGE BASE ADDRESS, RESTART AT ADDRESS /
3059      64$:
3060 021224 012746 000204      MOV #ADDMOD,-(SP)      ;GET READY TO TYPE STARTING ADDRESS
3061      ;OF "CHANGE OF BASE ADDRESS" ROUTINE
3062
3063 021230 104402      TYPOC
3064 021232 000000      HALT      ;STOP TO FORCE THE RESTART!
3065 021234 000137 044712      4$: JMP @#$EOP      ;GO TO END OF PROGRAM ----->
3066
3067 021240 012737 021320 000004      3$: MOV #TRP,@#4      ;INITIALIZE VECTOR
3068 021246 005037 017334      CLR RH70      ;INIT RH INDICATOR ++ C.W
3069 021252 005777 173542      TST @RHBAE      ;ADDRESS RPB AE (RH11/RH70?)
3070 021256 005237 017334      INC RH70      ;FOUND AN RH70
3071 021262 104401 021270      TYPE ,67$      ;;TYPE ASCIZ STRING
3072 021266 000413      BR 66$      ;;GET OVER THE ASCIZ
3073      ;;67$: .ASCIZ <15><12>/RH70 CONTROLLER /
3074      66$:
3075 021316 000420      BR RTN      ;SET MASK AND GET OUT
3076 021320 005726      TRP: TST (SP)+      ;ADJUST THE STACK
3077 021322 005726      TST (SP)+
3078 021324 104401 021332      TYPE ,65$      ;;TYPE ASCIZ STRING
3079 021330 000413      BR 64$      ;;GET OVER THE ASCIZ
3080      ;;65$: .ASCIZ <15><12>/RH11 CONTROLLER /
3081      64$:
3082 021360 012737 061746 000030      RTN: MOV #SERROR,@#EMTVEC ;RESTORE ERROR VECTOR
3083      ;SO THAT REGISTERS ARE SAVED
3084 021366 012737 000006 000004      MOV #ERPVEC+2,@#ERRVEC ;RESTORE TRAP CATCHER
3085

```

```

3086 ;FIND THE SILO SIZE
3087 ;IF ITS A RH70C MODIFY TESTS 50 AND 51
3088
3089 021374 004737 046116 JSR PC, @#CLDISK ;CONTROLLER CLEAR
3090 021400 005037 017336 CLR SILOSZ ;CLEAR SILO COUNTER
3091 021404 013777 017336 173334 13$: MOV SILOSZ, @RHDB ;LOAD SILO
3092 021412 005237 017336 INC SILOSZ ;KEEP COUNT
3093 021416 032777 000100 173330 BIT #1R, @RHCS2 ;IS THE SILO FULL?
3094 021424 001367 BNE 13$ ;BRANCH IF NO
3095 021426 022737 000406 017336 CMP #262., SILOSZ ;RH70C?
3096 021434 001031 BNE 14$ ;BRANCH IF NO
3097 021436 005037 032320 CLR VAR1+2 ;VAR1 IN TEST 50
3098 021442 012737 015220 032326 MOV #WRFROM,VAR2+2 ;VAR2 IN TEST 50
3099 021450 062737 000400 032326 ADD #256., VAR2+2
3100 021456 062737 000400 032326 ADD #256., VAR2+2
3101 021464 005037 032766 CLR VAR3+2 ;VAR3 IN TEST 51
3102 021470 012737 015220 032774 MOV #WRFROM, VAR4+2 ;VAR4 IN TEST 51
3103 021476 062737 000404 032774 ADD #260., VAR4+2
3104 021504 062737 000404 032774 ADD #260., VAR4+2
3105 021512 012737 052737 033024 MOV #52737,VAR5 ;VAR5 IN TEST 51
3106 021520 004737 046116 14$: JSR PC, @#CLDISK ;CONTROLLER CLEAR
3107
3108

```

```

3109
3110
3111
3112
3113
3114
3115
3116
3117 021524 000004
3118 021526 012737 000001 001212
3119 021534 012706 001000
3120 021540 012737 000002 017330
3121
3122
3123
3124 021546 005737 017334
3125 021552 001402
3126 021554 000137 021602
3127 021560
3128
3129 021560 013737 014754 021574
3130 021566 004537 045664
3131 021572 020017
3132 021574 000000
3133 021576 104001
3134 021600 000207
3135
3136
3137
3138
3139
3140
3141
3142
3143
3144 021602 000004
3145 021604 012737 000001 001212
3146 021612 012737 000003 017330
3147
3148 021620 013701 014774
3149 021624 012711 177777
3150 021630 011137 001126
3151 021634 105737 001126
3152 021640 001405
3153 021642 005037 001124
3154 021646 010137 045662
3155 021652 104001
3156
3157

;*****
;*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 PROGRAM IS RUNNING WITH AN RH70

      TST    @#RH70          ;TEST FOR RH70 CONTROLLER
      BEQ    30$             ;IF FLAG = 1, SKIP THIS TEST
      JMP    TST3            ;JUMP TO NEXT TEST -----)
30$:   ;IF FLAG = 0, DO THIS TEST

      MOV    @#RHCS2,@#UN+2
      JSR    R5,@#BITST      ;TEST BITS IN REGISTER
      UN:    20017            ;ONLY THESE BITS ARE TEST READ/WRITE
      .WORD  0               ;ADDRESS OF REG. BEING TESTED
      ERROR  1               ;IN CORRECT DATA RECEIVED
      RTS    PC              ;RETURN TO BLT3 ROUTINE

;*****
;*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,@#$BDDAT    ;TEST DATA
      TSTB   @#$BDDAT
      BEQ    TST4            ;BRANCH IF GOOD
      CLR    @#$GDDAT        ;GOOD DATA
      MOV    R1,@#REGADR     ;FAILING REG. RHAS
      ERROR  1               ;RHAS DOES NOT CLEAR
                          ;WITH ONES MOVED INTO IT

```

```
3158
3159
3160 *****
3161 *TEST 4 TEST FOR DRIVES PRESENT USING RHAS AND RHCS2
3162 *****
3162 021654 000004 000001 001212 TST4: SCOPE
3163 021656 012737 000001 001212 MOV #1,STIMES ;DO 1 ITERATION
3164 021664 000005 RESET ;START WITH AN INIT
3165 021666 004737 060506 JSP PC,#STKINT ;INITILIZE TTY KEYBOARD
3166
3167 021672 032777 020000 157240 BIT #SW13,#SWR ;INHIBIT ERROR TYPEOUT ?
3168 021700 001026 BNE 4$ ;SKIP NEXT IF SO
3169 021702 104401 021710 TYPE ,65$ ;TYPE ASCIZ STRING
3170 021706 000423 BR 64$ ;GET OVER THE ASCIZ
3171 ;65$: .ASCIZ <15><12><15><12>/LOOKING AT RHAS - DRIVES PRESENT/
3172 64$:
3173 021756 013701 014774 4$: MOV @#RHAS,R1 ;R1 HAS ADDR. OF RHAS
3174 021762 013702 014754 MOV @#RHCS2,R2 ;R2 HAS ADDR. OF RHCS2
3175 021766 005012 CLR @R2 ;CLEAR RHCS2
3176 021770 012700 000010 MOV #8,R0 ;COUNT
3177 021774 013704 014760 MOV @#RHER1,R4 ;R4 HAS ADDR. OF RHER1
3178
3179 022000 012714 177777 1$: MOV #-1,@R4 ;MOVE ERRORS INTO RHER1
3180 022004 005212 INC @R2 ;INCREMENT UNIT NO.
3181 022006 005300 DEC R0 ;COUNT
3182 022010 001373 BNE 1$ ;BRANCH IF 8 NOT DONE
3183 022012 111137 015136 MOVB @R1,@#TOTALAT ;SAVE TOTAL ATTENTION
3184 ;USED IN DRIVE CLEAR TEST
3185 022016 105037 015137 CLRB @#TOTALAT+1 ;CLEAR UPPER BYTE
3186 022022 105711 TSTB @R1 ;TEST FOR ANY DRIVES PRESENT
3187 022024 001402 BEQ 2$ ;NONE RESPONDING - TYPE THE MESSAGE
3188 022026 000137 022416 JMP XE2 ;SOME THERE - GO FILL 'UNITS' TABLE
3189
3190 022032 032777 020000 157100 2$: BIT #SW13,#SWR ;INHIBIT ERROR TYPE OUT?
3191 022040 001402 BEQ 3$ ;'NO DRIVES' MESSAGE IF NO
3192 022042 000137 022754 JMP SELTST ;CHECK FOR SELECTED UNIT START AND LOAD
3193 ;'UNITS' TABLE WITH DESIRED DRIVE IF SO
3194
3195 022046 3$:
3196 022046 104401 022054 TYPE ,67$ ;TYPE ASCIZ STRING
3197 022052 000412 BR 66$ ;GET OVER THE ASCIZ
3198 ;67$: .ASCIZ <15><12>/NO DRIVES-RHAS=0/
3199 66$:
3200 022100 TYPE ,69$ ;TYPE ASCIZ STRING
3201 022100 104401 022106 BR 68$ ;GET OVER THE ASCIZ
3202 022104 000436 ;69$: .ASCIZ <15><12>/WRITING ONES INTO ERROR REGISTER #1 FOR ALL UNIT NUMBERS/
3203 68$:
3204 022202 TYPE ,71$ ;TYPE ASCIZ STRING
3205 022202 104401 022210 BR 70$ ;GET OVER THE ASCIZ
3206 022206 000441 ;71$: .ASCIZ <15><12>/DOES NOT SET ANY BIT IN THE ATTENTION REGISTER SO ABORT PROGRAM
3207 70$:
3208 022312 TYPE ,73$ ;TYPE ASCIZ STRING
3209 022312 104401 022320 BR 72$ ;GET OVER THE ASCIZ
3210 022316 000435 ;73$: .ASCIZ <15><12>/TO LOOP ON THIS TEST WO PRINTOUT SET SWITCHES 13, 8 & 2/
3211 72$:
3212
3213 022412 000137 044712 JMP @#SEOP ;GO OUT----->
```

```
3214
3215
3216                ;*SET UP UNITS TABLE
3217
3218 022416          XE2:
3219 022416 012700 000010 2$: MOV #8.,R0 ;COUNTER
3220 022422 012703 015072 3$: MOV #UNITS,R3 ;POINTER
3221 022426 012723 177777 3$: MOV #-1,(R3)+ ;PRESET BLOCK TO ALL ONES
3222 022432 005300 3$: DEC R0 ;COUNT
3223 022434 001374 3$: BNE 3$ ;BRANCH IF 8 NOT DONE
3224 022436 012703 015072 3$: MOV #UNITS,R3 ;POINTER
3225 022442 005005 3$: CLR R5 ;NO. OF UNITS PRESENT
3226 022444 005037 015114 3$: CLR @#NOUNIT ;COUNTER
3227 022450 012700 000010 3$: MOV #8.,R0 ;TEMPORARY STORAGE
3228 022454 011137 001176 4$: MOV @R1,@#STMP0 ;SET CARRY IF ONE IN 0 BIT
3229 022460 006037 001176 4$: ROR @#STMP0
3230
3231 022464 103120 3$: BCC 5$
3232 022466 010577 172262 3$: MOV R5,@RHCS2 ;INSERT UNIT NUMBER
3233 022472 022777 024020 172302 3$: CMP #24020,@RHDT ;IS THIS A DUAL PORT RP04 ?
3234 022500 001503 3$: BEQ 6$ ;TYPE DRIVE NO. IF SO
3235 022502 022777 020020 172272 3$: CMP #20020,@RHDT ;IS THIS A SINGLE PORT RP04 ?
3236 022510 001477 3$: BEQ 6$ ;TYPE DRIVE NO. IF YES
3237
3238
3239 ;:*****
3240 022512 022777 024021 172262 3$: CMP #24021,@RHDT ;IS THIS A DUAL PORT RP05 ?
3241 022520 001473 3$: BEQ 6$ ;TYPE UNIT NO. OUT
3242 022522 022777 020021 172252 3$: CMP #20021,@RHDT ;IS THIS A SINGLE PORT RP05 ?
3243 022530 001467 3$: BEQ 6$ ;TYPE UNIT NO. IF SO
3244
3245 022532 022777 024022 172242 3$: CMP #24022,@RHDT ;IS THIS A DUAL PORT RP06 ?
3246 022540 001463 3$: BEQ 6$ ;TYPE THE NO IF SO
3247 022542 022777 020022 172232 3$: CMP #20022,@RHDT ;IS THIS A SINGLE PORT RP06 ?
3248 022550 001457 3$: BEQ 6$ ;TYPE THE NO IF SO
3249 ;:*****
3250
3251
3252 ;*NO...IT'S NOT AN RP04/RP05/RP06 DEVICE
3253 ;*SO TYPE OUT THE DEVICE TYPE
3254
3255 022552 104401 022560 3$: TYPE ,65$ ;:TYPE ASCII STRING
3256 022556 000410 3$: BR 64$ ;:GET OVER THE ASCII
3257 ;:65$: .ASCII <15><12>/UNIT NUMBER /
3258 64$:
3259 022600 3$: MOV R5,-(SP) ;GET READY TO TYPE UNIT NUMBER
3260 022602 010546 3$: TYPDS
3261 022604 104401 022612 3$: TYPE ,67$ ;:TYPE ASCII STRING
3262 022610 000406 3$: BR 66$ ;:GET OVER THE ASCII
3263 ;:67$: .ASCII /, RHDT = /
3264 66$:
3265 022626 017746 172150 3$: MOV @RHDT,-(SP) ;GET READY TO TYPE RHDT
3266 022632 104402 3$: TYPDC
3267 022634 104401 022642 3$: TYPE ,69$ ;:TYPE ASCII STRING
3268 022640 000422 3$: BR 68$ ;:GET OVER THE ASCII
3269 ;:69$: .ASCII ? - NOT AN RP04/RP05/RP06 DEVICE !!!
```

```

3270 022706          68$:
3271 022706 000407      BR      5$          ;NO RP04/5/6 FOUND SO BRANCH
3272 022710 010523      6$:      MOV      R5,(R3)+
3273 022712 104401 001223      TYPE     ,%CRLF
3274 022716 010546      MOV      R5,-(SP)      ;PUT DRIVE NO. ON STACK
3275 022720 104405      TYPDS      ;TYPE DRIVE NO.
3276 022722 005237 015114      INC      @#NUNIT ;INCR TOTAL NO. OF UNITS
3277
3278 022726 005205      5$:      INC      R5      ;'RHCS2' UNIT ADDRESS
3279 022730 005300      DEC      R0      ;DRIVE COUNTER DOWN ONE
3280 022732 001252      BNE      4$      ;TEST AND DO NEXT UNIT IF 8 NOT DONE
3281
3282 022734 013737 015072 015112      MOV      @#UNITS,@#UNIT ;SET UNIT NO. TO FIRST ONE FOUND/OR 0
3283 022742 013737 015114 015116      MOV      @#NUNIT,@#NUNIT ;SAVE NO. OF UNITS
3284 022750 005337 015116      DEC      @#NUNIT ;IF NUNIT = 0 THEN ONLY ONE UNIT
3285                                     ;IF NUNIT > 0 THEN MORE THAN ONE UNIT
3286
3287 022754 005737 015120      SELTST: TST     @#SELECT ;STARTING ADDRESS 200 ?
3288 022760 001403      BEQ      TST5      ;BRANCH IF STARTING FROM 200
3289 022762 013737 015122 015112      MOV      @#UNITSL,@#UNIT ;CHANGE UNIT NUMBER TO SELECTED ONE
3290

```

```

3291
3292
3293
3294
3295
3296
3297
3298
3299
3300
3301 022770 000004
3302 022772 012737 000001 001212
3303 023000 012737 023450 001106
3304
3305 023006 004737 046116
3306 023012 005037 015134
3307
3308
3309
3310 023016 005737 015112
3311 023022 001022
3312 023024 012700 000041
3313 023030 122710 000011
3314 023034 001015
3315 023036 005737 015120
3316
3317 023042 001012
3318
3319
3320
3321
3322 023044 012700 015072
3323 023050 005720
3324
3325 023052 022710 177777
3326 023056 001404
3327 023060 011037 015112
3328 023064 005337 015114
3329 023070 013700 015112
3330
3331
3332
3333
3334
3335
3336
3337
3338
3339
3340
3341
3342
3343
3344
3345
3346 023074 116037 017340 015134

```

```

*****
;*TEST 5      TYPE SERIAL NUMBER AND DRIVE TYPE
*****
;*
;*      READ SERIAL NUMBER REGISTER AND DRIVE TYPE REGISTER
;*      TYPE IT OUT AND PROCEED
*****
;*      TO LOOP HERE SET SWITCH 8 AND THIS TEST NO AND RESTART
*****
1ST5:  SCOPE
      MOV    #1,$TIMES      ;;DO 1 ITERATION
      MOV    #1$,$LPADR     ;;SET SCOPE LOOP ADDRESS
      JSR    PC,@#CLDISK    ;FILL UNIT NO.
      CLR    @#ATTENT       ;CLEAR
      ;*TEST FOR UNIT #0
      TST    @#UNIT         ;IS UNIT #0 NEXT IN THE UNITS TABLE ?
      BNE    10$            ;IF NOT, TEST THIS UNIT
      MOV    #41,$RO        ;IF SO, CHECK THE LOAD MEDIA LOCATION
      CMPB   #11,($RO)      ;WAS IS AN RP04/5/6 ?
      BNE    10$            ;NO... GO AHEAD AND TEST UNIT #0
      TST    @#SELECT       ;WAS UNIT #0 SELECTED ?
                          ;(IE. WAS IT A 210 START ?)
      BNE    10$            ;IF SO...TEST UNIT #0
      ;*INCREMENT THE UNITS TABLE TO NEXT DRIVE (IF ANY)
      ;* 8 DECREMENT THE 'NOUNITS' PRESENT (TO BE TESTED)
      MOV    #UNITS,$RO     ;LOAD UNITS TABLE POINTER
      TST    ($RO)+         ;SELECT THE NEXT UNIT IN THE TABLE
                          ;(DOUBLE INCREMENT THE POINTER)
      CMP    #-1,($RO)      ;IS THERE ANOTHER TABLE ENTRY PRESENT ?
      BEQ    10$            ;IF NOT (LOC = -1) ...MUST USE UNIT #0
      MOV    ($RO),@#UNIT    ;SET UP TO BE THE UNIT UNDER TEST
      DEC    @#NOUNITS       ;DECREMENT BECAUSE UNIT #0 WON'T BE TESTED
10$:   MOV    @#UNIT,$RO     ;RO CONTAINS THE UNIT UNDER TEST
*****
      CLR    @#RP06         ;CLEAR RP06 DEVICE TYPE FLAG
      CMP    #24022,@RHDT   ;DUAL PORT RP06 ?
      BEQ    2$             ;YES...SET THE FLAG
      CMP    #20022,@RHDT   ;SINGLE PORT RP06 ?
      BEQ    2$             ;YES...SET FLAG
      BR     3$             ;DON'T SET THE RP06 FLAG
2$:   MOV    #-1,@#RP06     ;SET THE FLAG
3$:
*****
;ASSUME THE NEXT UNIT IS AN RP04
*****
      MOVB   ATABLE($RO),@#ATTENT ;SET APPROPRIATE ATTENTION BIT

```

```

3347 023102 104401 023110      TYPE      ,65$      ;;TYPE ASCII STRING
3348 023106 000414      BR      64$      ;;GET OVER THE ASCII
3349      ;;65$: .ASCIIZ <15><12>/TESTING DRIVE NUMBER/
3350 023140      64$:
3351 023140 013746 015112      MOV      @#UNIT,-(SP)      ;UNIT NO. TO STACK
3352 023144 104405      TYPDS      ;TYPE DRIVE NO.
3353 023146 104401 023154      TYPE      ,67$      ;;TYPE ASCII STRING
3354 023152 000410      BR      66$      ;;GET OVER THE ASCII
3355      ;;67$: .ASCIIZ <15><12>/SERIAL NO. = /
3356 023174      66$:
3357 023174 017746 171604      MOV      @RHSN,-(SP)      ;;SAVE @RHSN FOR TYPEOUT
3358 023200 104402      TYPOC      ;GO TYPE--OCTAL ASCII(ALL DIGITS)
3359 023202 104401 023210      TYPE      ,69$      ;;TYPE ASCII STRING
3360 023206 000410      BR      68$      ;;GET OVER THE ASCII
3361      ;;69$: .ASCIIZ <15><12>/DRIVE TYPE = /
3362 023230      68$:
3363 023230 017746 171546      MOV      @RHDT,-(SP)      ;;SAVE @RHDT FOR TYPEOUT
3364 023234 104402      TYPOC      ;GO TYPE--OCTAL ASCII(ALL DIGITS)
3365
3366
3367
3368
3369
3370      ;;*****
3371 023236 022777 024020 171536      CMP      #24020,@RHDT      ;DUAL PORT RP04 ?
3372 023244 001425      BEQ      4$      ;TYPE ASCII MSG OUT
3373 023246 022777 020020 171526      CMP      #20020,@RHDT      ;SINGLE PORT RP04 ?
3374 023254 001421      BEQ      4$      ;TYPE THE MESSAGE
3375
3376 023256 022777 024021 171516      CMP      #24021,@RHDT      ;DUAL PORT RP05 ?
3377 023264 001434      BEQ      5$      ;TYPE THE MESSAGE
3378 023266 022777 020021 171506      CMP      #20021,@RHDT      ;SINGLE PORT RP05 ?
3379 023274 001430      BEQ      5$      ;TYPE THE MESSAGE
3380
3381 023276 022777 024022 171476      CMP      #24022,@RHDT      ;DUAL PORT RP06 ?
3382 023304 001443      BEQ      6$      ;TYPE THE MESSAGE
3383 023306 022777 020022 171466      CMP      #20022,@RHDT      ;SINGLE PORT RP06 ?
3384 023314 001437      BEQ      6$      ;TYPE IT OUT
3385 023316 000454      BR      1$      ;DRIVE IS NOT RP04/5/6 - SO
3386      ;DO NOT TYPE ANY MESSAGE OUT
3387
3388      ;-SHOULD NEVER HAPPEN AT THIS POINT
3389      ;UNLESS DRIVE GOT SICK WHILE TESTING
3390      ;WAS IN PROGRESS
3391
3392      4$:
3393 023320      TYPE      ,71$      ;;TYPE ASCII STRING
3394 023320 104401 023326      BR      70$      ;;GET OVER THE ASCII
3395      ;;71$: .ASCIIZ <15><12>/DRIVE IS AN RP04/<15><12>
3396 023354      70$:
3397 023354 000435      BR      1$      ;SKIP NEXT ONES
3398 023356      5$:
3399 023356 104401 023364      TYPE      ,73$      ;;TYPE ASCII STRING
3400 023362 000413      BR      72$      ;;GET OVER THE ASCII
3401      ;;73$: .ASCIIZ <15><12>/DRIVE IS AN RP05/<15><12>
3402 023412      72$:

```

CZRJHDO,RP04/5/6 DSKLS CTRLR2 MACY11 30A(1052) 24-MAY-79 15:07 PAGE 72
CZRJHD.P11 28-MAR-79 09:16 TS TYPE SERIAL NUMBER AND DRIVE TYPE

SEQ 0071

```
3403 023412 000416          BR      1$          ;SKIP NEXT
3404 023414
3405 023414 104401 023422    6$:      TYPE      ,75$          ;;TYPE ASCII STRING
3406 023420 000413          BR      74$          ;;GET OVER THE ASCII
3407          ;;75$: .ASCII <15><12>/DRIVE IS AN RP06/<15><12>
3408 023450          74$:
3409          ;;*****
3410
3411 023450 005777 171330    1$:      1ST      @RHSN          ;READ SERIAL NO. AND DRIVE TYPE
3412 023454 005777 171322          1ST      @RHDT          ;THESE TWO ARE TO HELP SCOPE LOOPS
3413 023460 017737 171320 015130    MOV      @RHSN,@#SAVSN    ;SAVE TO CHECK IF CLR RHCS2 BIT 5 CLEARS ANY BITS
3414 023466 017737 171310 015126    MOV      @RHDT,@#SAVDT    ;SAVE TO CHECK IF CLR RHCS2 BIT 5 CLEARS ANY BITS
```

```
3415
3416
3417
3418
3419
3420
3421
3422
3423
3424
3425 023474 000004
3426 023476 012737 000006 017330
3427 023504 004737 046116
3428 023510 032713 010000
3429 023514 001550
3430
3431 023516 104401 023524
3432 023522 000421
3433
3434 023566
3435 023566 104401 023574
3436 023572 000424
3437
3438 023644
3439 023644 104401 023652
3440 023650 000430
3441
3442 023732
3443
3444 023732 032713 010000
3445 023736 001375
3446 023740 104401 023746
3447 023744 000434
3448
3449 024036
3450
3451
3452

*****
*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
*****
1516:  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$: .ASCIZ  <15><12>/PROGRAM WILL HANG TESTING MOL TILL MOL IS LOW/
68$:   TYPE     ,71$               ;;TYPE ASCIZ STRING
      BR       70$               ;;GET OVER THE ASCIZ
;;71$: .ASCIZ  <15><12>/GOOD - MOL IS NOW LOW . PROGRAM WILL NOW BE EXECUTED/
70$:   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$: .ASCIZ  <15><12>/GOOD - MOL IS NOW LOW . PROGRAM WILL NOW BE EXECUTED/
70$:
```

```
3453
3454
3455
3456
3457
3458
3459
3460
3461
3462
3463 024036 000004
3464 024040 012706 001000
3465 024044 012737 000007 017330
3466
3467 024052 004737 046116
3468
3469 024056 012777 000001 170712
3470
3471 024064 013777 015206 170664
3472
3473
3474 024072 004037 046616
3475 024076 014750
3476 024100 016264
3477 024102 000023
3478
3479
3480 024104 052777 000001 170644
3481
3482
3483 024112 052737 000100 016314
3484
3485
3486
3487
3488 024120 004037 046616
3489 024124 014750
3490 024126 015220
3491 024130 000023
3492
3493
3494
3495
3496 024132 113737 016311 015245
3497
3498
3499
3500
3501 024140 004037 047020
3502 024144 016264
3503 024146 015220
3504 024150 000023
3505 024152 024160
3506 024154 024160
3507 024156 024200
3508 024160 013705 053130

*****
; * TEST 7 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
*****
1ST7: 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
MOV @#PKACK,@RHCS1 ;LOAD PACK ACKNOWLEDGE 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 PACK ACKNOWLEDGE COMMAND
BIS #GO,@RHCS1 ;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
WRFROM
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 PACK ACKNOWLEDGE COMMAND
;WITH AFTER GO
JSR RO,@#COMPAR ;COMPARE
REINTO ;GOOD BUFFER
WRFROM ;TEST BUFFER
19. ;NUMBER
1$ ;RETURN FOR ERROR
1$ ;SAME
2$ ;RETURN FOR GOOD COMPARISON
1$: MOV @#ERWORD,R5 ;GETTING READY TO INDEX
```

```

3509 024164 060505          ADD    R5,R5          ;DOUBLE ERROR WORD
3510 024166 016537 014746 045662      MOV    RHWC-2(R5),@#REGADR ;FAILING REGISTER ADDRESS
3511
3512 024174 104001          ERROR  1              ;IMPROPER REGISTER CHANGE
3513                                     ;AFTER PACK ACKNOWLEDGE COMMAND
3514                                     ;WITH GO IS GIVEN
3515 024176 000207          RTS     PC              ;RETURN TO COMPARISION
3516
3517 024200          28:
3518
3519          ;*****
3520          ;*TEST 10      MAKE CURRENT CYLINDER = 0
3521          ;*****
3522 024200 000004          TST10:  SCOPE
3523 024202 012706 001000      MOV    #STACK,SP      ;RESET STACK
3524 024206 012737 000010 017330      MOV    #TINO,@#TSTNM ;THIS SAVES TEST NUMBER
3525 024214 004737 046116          JSR    PC,@#CLDISK  ;INIT DRIVE
3526 024220 012777 000001 170550      MOV    #DMD,@#RHRM  ;SET DIAGNOSTIC MODE
3527 024226 004037 050476          JSR    R0,@#MAKECYL ;SUBROUTINE TO GIVE A SEEK
3528                                     ;COMMAND FOLOWED BY AN INIT
3529 024232 000000          0              ;THIS SHUOLD CHANGE RHCC TO 0
3530
3531
3532          .SBTTL
3533          .SBTTL ***DIAGNOSTIC CODE***
3534          .SBTTL
  
```

```
3535          000004          TIMOT=4
3536          .REM %
3537          THE FOLLOWING TESTS WILL ATTEMPT TO CATCH THOSE BUGS WHICH FAULT
3538          INSERTION HAS SHOWN US WE MISSED IN THE FIRST PART OF THIS TEST.
3539
3540          THIS TEST WILL ASCERTAIN THAT THE ALL RH11 REGISTERS WILL
3541          RESPOND TO I.E. NOT CAUSE A NON-EXISTANT MEMORY ERROR WHEN ACCESSED
3542          BY A "TST" INSRUCTION.
3543          %
3544          ;*****
3545          ;*TEST 11          BCTA LEGAL REGISTER RESPONSE TEST
3546          ;*****
3547          024234 000004          TST11:  SCOPE
3548          024236 012737 000001 001212          MOV      #1,$TIMES          ;;DO 1 ITERATION
3549          024244 012737 177777 017330          MOV      #TIMO,@#TSTNM          ;THIS SAVES TEST NUMBER
3550          024252 012777 000040 170474          MOV      #CLR,@RHCS2          ;CLEAR RH11 CONTROLLER
3551
3552          024260 012705 014756          MOV      #RHCS1,R5          ;R5=LIST POINTER.
3553
3554          024264 013737 000004 001176          MOV      @#TIMOT,$TMP0          ;SAVE TIMEOUT VECTOR.
3555          024272 012737 024330 000004          MOV      #E00,@#TIMOT          ;SET VECTOR TO E00
3556
3557          024300 012706 001000          L00:  MOV      #STACK,SP          ;SET STACK POINTER.
3558
3559          024304 011502          100:  MOV      (R5),R2          ;R2=RH11 ADDRESS.
3560          024306 010237 045662          MOV      R2,REGADR
3561          024312 005712          TST      (R2)          ;DOES THE RH11 REGISTER RESPOND?
3562
3563          024314 062705 000002          ADD      #2,R5          ;UPDATE ADDRESS
3564          024320 020527 015010          CMP      R5,#RHEC2          ;AT THE END OF LEGAL RH11 REGISTERS?
3565          024324 101767          BLOS     100          ;NOPE
3566          024326 000401          BR       000          ;YES! GOTO 000.
3567
3568          024330 104042          E00:  ERROR  42
3569
3570          024332 013737 001176 000004 000:  MOV      $TMP0,@#TIMOT          ;RESTORE TIMEOUT VECTOR.
```

CZRJHDO,RP04/5/6 DSKLS CTRLR2 MACY11 30A(1052) 24-MAY-79 15:07 L 6
CZRJHD.P12 01-MAR-79 09:10 T11 BCTA LEGAL REGISTER RESPONSE TEST

SEQ 0076

3571

3572
3573
3574
3575
3576
3577
3578
3579
3580
3581
3582
3583
3584
3585
3586
3587
3588
3589
3590
3591
3592
3593
3594
3595
3596
3597
3598
3599
3600
3601
3602
3603
3604
3605
3606
3607
3608

.REM %
THE FOLLOWING 6 TESTS WILL TEST THE BYTE LOGIC FOR THE RH11 REGISTERS
NO ATTEMPT IS MADE TO DETERMINE IF ALL POSSIBLE DATA PATTERNS CAN BE
LOADED, ONLY THAT THE RH11 HARDWARE WILL ALLOW THE PROGRAM TO SELECTIVLY
MANIPULATE BYTES USING THE FOLLOWING COMMANDS:

- 1). MOVE BYTE BOTH TO AND FROM THE WORD COUNT REGISTER
- 2). BIT SET INTO THE WORD COUNT REGISTER.
- 3). BIT CLEAR INTO THE WORD COUNT REGISTER.

%

```
*****  
*TEST 12      BCTA MOV B LO BYTE TO WC  
*****  
TST12:  SCOPE  
        MOV     #1,$TIMES      ;;DO 1 ITERATION  
        MOV     #TINO,@#TSTNM  ;THIS SAVES TEST NUMBER  
        MOV     #CLR,@RHCS2    ;CLEAR RH11 CONTROLLER  
  
        MOV     #STACK,SP      ;SET STACK POINTER.  
  
        MOV     RHC,R2         ;R2=RH11 WC ADDRESS.  
        MOV     R2,REGADR      ;REGISTER ADDRESS TO REGADR FOR TYPING.  
        MOV     #377,$GDDAT    ;$GDDAT=S/B.  
  
L02:    CLR     (R2)           ;CLEAR RH11 WC.  
  
I02:    MOVB    #377,(R2)      ;SET WC TO LO BYTE.  
        MOV     (R2),$BDDAT    ;$BDDAT=ACTUAL WAS  
  
        CMP     $BDDAT,$GDDAT  ;COMPARE RESULTS  
        BEQ     TST13          ;NEXT TEST  
        ERROR   44
```

```
3609
3610
3611      ;*****
3612      ;*TEST 13      BCTA MOVB HI BYTE TO WC
3613      ;*****
3613      TST13:  SCOPE
3614      MOV      #1,$TIMES      ;;DO 1 ITERATION
3615      MOV      #TINO,@#TSTNM  ;THIS SAVES TEST NUMBER
3616      MOV      #CLR,@RHCS2    ;CLEAR RH11 CONTROLLER
3617
3618      MOV      #STACK,SP      ;SET STACK POINTER.
3619
3620      MOV      RHWC,R2         ;R2=RH11 WC HI BYTE ADDRESS.
3621      MOV      R2,REGADR
3622      MOV      #177400,$GDDAT ;$GDDAT=S/B.
3623
3624      L03:  CLR      (R2)      ;CLEAR RH11 WC.
3625
3626      INC      R2             ;R2=HI BYTE ADDRESS.
3627
3628      I03:  MOVB     #377,(R2) ;SET WC HI BYTE.
3629
3630      DEC      R2             ;R2=FULL WORD ADDRESS.
3631
3632      MOV      (R2),$BDDAT     ;$BDDAT=ACTUAL.
3633
3634      CMP      $BDDAT,$GDDAT   ;COMPARE RESULTS.
3635      BEQ      TST14          ;NEXT TEST
3636      ERROR    44
3637
3638
```

```
3639
3640
3641
3642
3643 024530 000004
3644 024532 012737 000001 001212
3645 024540 012737 000014 017330
3646 024546 012777 000040 170200
3647
3648 024554 012706 001000
3649
3650 024560 013702 014750
3651 024564 010237 045662
3652 024570 012737 000377 001124
3653
3654 024576 005012
3655
3656 024600 012712 000252
3657 024604 152712 000125
3658
3659 024610 011237 001126
3660
3661 024614 023737 001126 001124
3662 024622 001401
3663 024624 104045
3664
3665

;*****
;TEST 14 BCTA BISB LO BYTE TO WC
;*****
TST14: SCOPE
MOV #1,$TIMES ;DO 1 ITERATION
MOV #TINO,@TSTNM ;THIS SAVES TEST NUMBER
MOV #CLR,@RHCS2 ;CLEAR RH11 CONTROLLER
MOV #STACK,SP ;SET STACK POINTER.
MOV RHC,R2 ;R2=RH11 WC ADDRESS
MOV R2,REGADR
MOV #377,$GDDAT ;$GDDAT=S/B.
L04: CLR (R2) ;CLEAR RH11 WC.
104: MOV #252,(R2) ;SET UP WC
BISB #125,(R2) ;DO A BISB
MOV (R2),$BDDAT ;$BDDAT=WAS.
CMP $BDDAT,$GDDAT ;COMPARE RESULTS
BEQ TST15 ;NEXT TEST
ERROR 45
```

```

3666
3667
3668
3669
3670 024626 000004
3671 024630 012737 000001 001212
3672 024636 012737 000015 017330
3673 024644 012777 000040 170102
3674
3675 024652 012706 001000
3676
3677 024656 013702 014750
3678 024662 010237 045662
3679 024666 012737 177400 001124
3680
3681 024674 005012 L05: CLR (R2) ;CLEAR RH11 WC.
3682
3683 024676 005202 INC R2 ;R2 =HI BYTE.
3684
3685 024700 112712 000125
3686 024704 152712 000252 105: MOVB #125,(R2) ;SET UP RH11 WC.
3687 BISB #252,(R2) ;DO A BISB.
3688 024710 005302 DEC R2 ;R2=FULL WORD ADDRESS.
3689
3690 024712 011237 001126 MOV (R2),$BDDAT ;$BDDAT=WAS.
3691
3692 024716 023737 001126 001124 CMP $BDDAT,$GDDAT ;COMPARE RESULTS.
3693 024724 001401 BEQ TST16 ;NEXT TEST
3694 024726 104045 ERROR 45
3695
3696
  
```

3697
3698
3699
3700
3701
3702
3703
3704
3705
3706
3707
3708
3709
3710
3711
3712
3713
3714
3715
3716
3717
3718
3719
3720
3721
3722
3723
3724

024730 000004
024732 012737 000001 001212
024740 012737 000016 017330
024746 012777 000040 170000

024754 012706 001000

024760 013702 014750
024764 010237 045662
024770 012737 000252 001124

024776 005012

025000 012712 000377

025004 142712 000125

025010 011237 001126

025014 023737 001126 001124
025022 001401
025024 104046

```
*****  
:TEST 16      BCTA BICB LO-BYTE TO WC  
*****  
TST16:  SCOPE  
        MOV    #1,$TIMES      ;;DO 1 ITERATION  
        MOV    #TINO,@#TSTNM  ;THIS SAVES TEST NUMBER  
        MOV    #CLR,@RHCS2    ;CLEAR RH11 CONTROLLER  
  
        MOV    #STACK,SP      ;SET STACK POINTER.  
  
        MOV    RHWC,R2        ;R2=RH11 WC ADDRESS.  
        MOV    R2,REGADR  
        MOV    #252,$GDDAT    ;$GDDAT=S/B.  
  
L06:    CLR     (R2)          ;CLEAR RH11 WC.  
  
        MOV    #377,(R2)      ;SET WC=0000377  
  
106:    BICB    #125,(R2)     ;DO A BICB  
  
        MOV    (R2),$BDDAT    ;$BDDAT=WAS.  
  
        CMP    $BDDAT,$GDDAT  ;COMPARE RESULTS.  
        BEQ    TST17          ;NEXT TEST  
        ERROR  46
```

```
3725
3726
3727 *****
3728 : *TEST 17 BCTA BICB HI- BYTE TO WC
3729 : *****
3730 025026 000004 1ST17: SCOPE
3731 025030 012737 000001 001212 MOV #1,$TIMES ;;DO 1 ITERATION
3732 025036 012737 000017 017330 MOV #TINO,@#TSTNM ;THIS SAVES TEST NUMBER
3733 025044 012777 000040 167702 MOV #CLR,@RHCS2 ;CLEAR RH11 CONTROLLER
3734 025052 012706 001000 MOV #STACK,SP ;SET STACK POINTER.
3735
3736 025056 013702 014750 MOV RHWC,R2 ;R2=RH11 WC ADDRESS.
3737 025062 010237 045662 MOV R2,REGADR
3738 025066 012737 125000 001124 MOV #125000,$GDDAT ;$GDDAT=S/B.
3739
3740 025074 005012 L07: CLR (R2) ;CLEAR RH11 WC.
3741
3742 025076 005202 INC R2 ;R2=HI BYTE.
3743
3744 025100 112712 000377 MOVB #377,(R2) ;SET WC-177400
3745 025104 142712 000125 107: BICB #125,(R2) ;DO A BICB
3746
3747 025110 005302 DEC R2 ;R2=FULL WORD.
3748
3749 025112 011237 001126 MOV (R2),$BDDAT ;$BDDAT=WAS.
3750
3751 025116 023737 001126 001124 CMP $BDDAT,$GDDAT ;COMPARE RESULTS.
3752 025124 001401 BEQ TST20 ;NEXT TEST
3753 025126 104046 ERROR 46
3754
3755
```

```
3756
3757
3758
3759
3760
3761
3762
3763
3764
3765
3766
3767
3768
3769
3770
3771
3772 025130 000004
3773 025132 012737 000001 001212
3774 025140 012737 000020 017330
3775 025146 012777 000040 167600
3776
3777 025154 005737 017334
3778 025160 001403
3779 025162 013702 015016
3780 025166 000402
3781
3782 025170 013702 015010
3783 025174 062702 000002
3784 025200 010237 045662
3785 025204 005037 001126
3786 025210 005037 001124
3787
3788 025214 013737 000004 001176
3789 025222 012737 025246 000004
3790
3791 025230 012706 001000
3792
3793 025234 005712
3794
3795 025236 062702 000024
3796 025242 005712
3797 025244 104047
3798
3799 025246 013737 001176 000004 010:
3800
```

```

      .REM
      THIS TEST ATTEMPT TO ACCESS AN ILLEGAL REGISTER WITHIN THE RANGE
      OF ADDRESSES SELECTED BY THE XOR'S. THE RH11 ADDRESS DECODE LOGIC WILL ALLOW UP TO 32
      CONTIGUOUS REGISTERS TO EXIST, IN OUR CONFIGURATION ONLY 16 WILL REALLY
      EXIST, THIS TEST ATTEMPTS TO ACCESS THE 20(8) REGISTER - IF IT RESPONDS
      THE TEST WILL TYPE OUT AN ERROR.

      FOR THE CASE OF THE RH70, THE CONFIGURATION ALLOWS 18 OR 32 REGISTERS, SO
      WE WILL ATTEMPT TO ADDRESS REGISTER 23(8), 33(8) OR AS BEFORE THE LAST
      REGISTER + 2.

      *****
      :TEST 20          BCTA ILLEGAL REGISTER TEST
      *****
      TST20:  SCOPE
               MOV      #1,$TIMES          ;;DO 1 ITERATION
               MOV      #TINO,@#TSTNM      ;THIS SAVES TEST NUMBER
               MOV      #CLR,@RHCS2        ;CLEAR RH11 CONTROLLER

               TST      @#RH70             ;CHECK TO SEE IF RUNNING WITH RH70
               BEQ      1$                 ;IF NOT...SKIP NEXT & DO FOLLOWING
               MOV      RHCS3,R2           ;R2 = LAST LEGAL RH70 REG ADDRESS
               BR       2$                 ;SKIP NEXT

               1$:      MOV      RHEC2,R2   ;R2 = LAST LEGAL RH11 REG ADDRESS.
               2$:      ADD      #2,R2      ;R2 = FIRST ILLEGAL ADDRESS.
               MOV      R2,REGADR
               CLR      $BDDAT             ;$BDDAT=WAS.
               CLR      $GDDAT             ;$GDDAT=S/B.

               MOV      @#TIMOT,$TMP0      ;SAVE TIMEOUT VECTOR.
               MOV      #010,@#TIMOT      ;SET TIMEOUT VECTOR TO 010.

               L10:      MOV      #STACK,SP ;SET THE STACK POINTER.

               TST      (R2)               ;TEST ADDRESS.

               ADD      #24,R2             ;THIS MIGHT BE THE CASE OF 32 REGISTERS
               TST      (R2)               ;TEST IT AGAIN
               ERROR    47

               010:      MOV      $TMP0,@#TIMOT ;RESTORE TIMEOUT VECTOR.
```

```
3801
3802
3803
3804
3805
3806
3807
3808
3809
3810
3811
3812
3813
3814
3815
3816 025254 000004
3817 025256 012737 000001 001212
3818 025264 012737 000021 017330
3819 025272 012777 000040 167454
3820
3821 025300 123727 015136 000377
3822 025306 001150
3823
3824 025310 023727 001100 000000
3825
3826
3827 025316 001002
3828 025320 000137 025732
3829
3830 025324
3831 025324 104401 025332
3832 025330 000426
3833
3834 025406
3835 025406 104401 025414
3836 025412 000426
3837
3838 025470
3839 025470 104401 025476
3840 025474 000425
3841
3842 025550
3843 025550 104401 025556
3844 025554 000425
3845
3846 025630
3847
3848 025630 012706 001000
3849
3850 025634 013702 014754
3851 025640 010237 045662
3852
3853 025644 013700 015136
3854 025650 005001
3855 025652 006000
3856 025654 103423

:
: NOW WE ATTACK BTCB
:
: REM %
: THE FOLLOWING TWO TESTS DETERMINE IF ALL NON DRIVES SET THE
: NED BIT AND THAT ALL EXISTING DRIVES CLEAR THE NED BIT.
: THE TESTS LOOK AT TOTALAT TO DETERMINE WHICH DRIVES EXIST AND TEST THAT
: THESE DRIVES WILL CLEAR THE NED BIT. AND ALSO THAT NON EXISTANT DRIVES
: WILL SET THE NED BIT.
: ON FIRST PASS ONLY, IF ALL DRIVES ARE ON LINE A MESSAGE WILL SO ADVISE THE
: OPERATOR. THE TEST WILL CONTINUE REGARDLESS OF THE OPERATORS ACTION.
:
: %
: *****
: *TEST 21 BCTB (NED) NON-EXISTANT DRIVE TEST. (SET)
: *****
: TEST21: SCOPE
: MOV #1,$TIMES ;DO 1 ITERATION
: MOV #TINO,@#TSTNM ;THIS SAVES TEST NUMBER
: MOV #CLR,@RHCS2 ;CLEAR RH11 CONTROLLER
:
: CMPB @#TOTALAT,#377 ;ARE ALL DRIVES ON LINE.
: BNE U11 ;NO GO RUN THE TEST.
:
: CMP $PASS,#0 ;IF PASS #0 THEN TYPE MESSAGE
: ;ADVISING OPERATOR
: ;THAT THIS TEST WILL NOT BE RUN
:
: BNE Y11
: JMP X11
:
: Y11:
: TYPE ,65$ ;:TYPE ASCII STRING
: BR 64$ ;:GET OVER THE ASCII
: ;:65$: .ASCII <15><12>/ALL DRIVES APPEAR TO BE ON LINE THEREFORE/
: 64$:
: TYPE ,67$ ;:TYPE ASCII STRING
: BR 66$ ;:GET OVER THE ASCII
: ;:67$: .ASCII <15><12>/THE NED NON-EXISTANT DRIVE TEST CANNOT BE/
: 66$:
: TYPE ,69$ ;:TYPE ASCII STRING
: BR 68$ ;:GET OVER THE ASCII
: ;:69$: .ASCII <15><12>/RUN. TO RUN THIS TEST TAKE ONE OR MORE/
: 68$:
: TYPE ,71$ ;:TYPE ASCII STRING
: BR 70$ ;:GET OVER THE ASCII
: ;:71$: .ASCII <15><12>/DRIVES OFF-LINE AND RESTART THE PROGRAM/
: 70$:
:
: U11: MOV #STACK,SP ;SET STACK POINTER.
:
: MOV RHCS2,R2 ;R2=RH11 RHCS2 ADDRESS.
: MOV R2,REGADR
:
: MOV @#TOTALAT,R0
: CLR R1
: S11: ROR R0
: BCS N11
```

CZRJHDO,RP04/5/6 DSKLS CTRLR2 MACY11 30A(1052) 24-MAY-79 15:07 PAGE 87
CZRJHD.P12 01-MAR-79 09:10 121 BCTB (NED) NON-EXISTANT DRIVE TEST. (SET)

SEQ 0085

```
3857
3858 025656 010137 001124 R11: MOV R1,$GDDAT ;$GDDAT=S/B.
3859 025662 052737 010000 001124 BIS #NED,$GDDAT
3860
3861 025670 013705 014756 MOV RHCS1,R5 ;ADDRESS OF A DEVICE REGISTER.
3862
3863 025674 010112 L11: MOV R1,(R2) ;LOAD A NON-EXISTANT DRIVE.
3864
3865 025676 011537 001176 MOV (R5),$TMP0 ;ATTEMPT TO READ FROM DEVICE REGISTER.
3866
3867 025702 011237 001126 MOV (R2),$BDDAT ;$BDDAT=WAS.
3868 025706 042737 167770 001126 BIC #^C<NED!US4.US2!US1>,$BDDAT;$BDDAT- SAVED DATA.
3869
3870 025714 023737 001126 001124 CMP $BDDAT,$GDDAT ;COMPARE RESULTS.
3871 025722 001005 BNE E11
3872
3873 025724 005201 N11: INC R1
3874 025726 020127 000010 CMP R1,#8. ;TESTED ALL DRIVES YET.
3875 025732 X11:
3876 025732 001402 BEQ TST22 ;NEXT TEST
3877 025734 000746 BR S11
3878
3879 025736 104050 E11: ERROR 50
3880
3881
```

CZRJHDO,RP04/5/6 DSKLS CTRLR2
CZRJHD.P12 01-MAR-79 09:10

MACY11 30A(1052) 24-MAY-79 15:07 PAGE 88
121 BCTB (NED) NON-EXISTANT DRIVE TEST. (SET)

SEQ 0086

```
3882
3883
3884 ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
3885 :*TEST 22 BCTB (NED) NON-EXISTANT DRIVE TEST. (CLEARED)
3886 :*::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
3886 025740 000004 TST22: SCOPE
3887 025742 012737 000001 001212 MOV #1,$TIMES ;DO 1 ITERATION
3888 025750 012737 000022 017330 MOV #TNO,@#TSTNM ;THIS SAVES TEST NUMBER
3889 025756 012777 000040 166770 MOV #CLR,@RHCS2 ;CLEAR RH11 CONTROLLER
3890
3891 025764 012706 001000 MOV #STACK,SP ;SET STACK POINTER.
3892
3893 025770 013702 014754 MOV RHCS2,R2 ;R2=RH11 RHCS2 ADDRESS.
3894 025774 010237 045662 MOV R2,REGADR
3895 026000 013700 015136 MOV @#TOTALAT,R0
3896 026004 005001 CLR R1
3897 026006 006000 S12: ROR R0
3898 026010 103020 BCC N12
3899
3900 026012 010137 001124 MOV R1,$GDDAT
3901
3902 026016 013705 014756 MOV RHCS1,R5 ;ADDRESS OF A DEVICE REGISTER.
3903
3904 026022 010112 L12: MOV R1,(R2) ;SELECT UNIT.
3905
3906 026024 011537 001176 MOV (R5),$TMP0 ;ATTEMPT TO READ FROM DEVICE.
3907
3908 026030 011237 001126 MOV (R2),$BDDAT ;$BDDAT=WAS.
3909 026034 042737 167770 001126 BIC #^C<NED!US4!US2!US1>,$BDDAT ;$BDDAT=SAVED DATA.
3910
3911 026042 023737 001126 001124 CMP $BDDAT,$GDDAT ;COMPARE RESULTS.
3912 026050 001005 BNE E12
3913
3914 026052 005201 N12: INC R1
3915 026054 020127 000010 CMP R1,#8. ;TESTED ALL DRIVES.
3916 026060 001402 BEQ TST23 ;NEXT TEST
3917
3918 026062 000751 BR S12
3919 026064 104050 E12: ERROR 50
3920
3921
```

```
3922
3923
3924
3925
3926
3927
3928
3929
3930 026066 000004
3931 026070 012737 000001 001212
3932 026076 012737 000023 017330
3933 026104 012777 000040 166642
3934
3935 026112 012706 001000
3936
3937 026116 013702 014754
3938 026122 012737 000007 001124
3939
3940 026130 013705 014756
3941
3942 026134 012712 000007
3943
3944 026140 013702 014774
3945
3946 026144 011237 001176
3947
3948 026150 005012
3949 026152 013702 014754
3950 026156 010237 045662
3951
3952 026162 011237 001126
3953 026166 042737 167770 001126
3954
3955 026174 023737 001126 001124
3956 026202 001401
3957 026204 104051
3958
3959

      .REM      %
      TEST TO SEE IF WE CAN READ FROM THE "AS" REGISTER AND NOT CAUSE
      A NED NON-EXISTANT DRIVE ERROR
      %
      ;*****
      ;*TEST 23      BCTB AS REGISTER TEST
      ;*****
      TST23:  SCOPE
                MOV      #1,$TIMES      ;;DO 1 ITERATION
                MOV      #TINO,@#TSTNM  ;;THIS SAVES TEST NUMBER
                MOV      #CLR,@RHCS2    ;;CLEAR RH11 CONTROLLER
                MOV      #STACK,SP      ;;SET STACK POINTER.
                MOV      RHCS2,R2       ;;R2=RH11 CS2 ADDRESS.
                MOV      #US4.US2!US1,$GDDAT;$GDDAT=S/B.
                MOV      RHCS1,R5       ;;ADDRESS OF A DEVICE REGISTER.
      L13:      MOV      #US4!US2!US1,(R2);LOAD A NON-EXISTANT DEVICE.
                MOV      RHAS,R2        ;;R2= "AS".
                MOV      (R2),$TMP0     ;;ATTEMPT TO READ FROM DEVICE AS.
                CLR      (R2)           ;;CLEAR AS REGISTER.
                MOV      RHCS2,R2       ;;R2=RH11 CS2 ADDRESS.
                MOV      R2,REGADR
                MOV      (R2),$BDDAT    ;;$BDDAT=WAS.
                BIC      #^C<NED!US4!US2!US1>,$BDDAT;SAVE NED AND DEVICE SELECTED.
                CMP      $BDDAT,$GDDAT  ;;COMPARE RESULTS.
                BEQ      TST24          ;;NEXT TEST
                ERROR    51
```

```

3960 .REM %
3961 THE NEXT THREE TESTS WILL TEST THE BUS ADDRESS REGISTER IN BOTH WORD
3962 AND BYTE MODE. THE PATTERNS ARE CHOSEN TO PICK UP ANY DROPPED OR STUCK
3963 BITS.
3964 %
3965
3966 ::*****
3967 ::*TEST 24 BCTC BUS ADDRESS REGISTER
3968 ::*****
3969 026206 000004 TST24: SCOPE
3970 026210 012737 000001 001212 MOV #1,$TIMES ;;DO 1 ITERATION
3971 026216 012737 000024 017330 MOV #TINO,@TSTNM ;THIS SAVES TEST NUMBER
3972 026224 012777 000040 166522 MOV #CLR,@RHCS2 ;CLEAR RH11 CONTROLLER
3973
3974 026232 012706 001000 MOV #STACK,SP ;SET STACK POINTER.
3975
3976 026236 013702 014752 MOV RHBA,R2 ;R2=RH11 BA ADDRESS.
3977 026242 010237 045662 MOV R2,REGADR
3978 026246 012705 026306 MOV #LST14A,R5 ;R5=TEST LIST ADDRESS.
3979
3980 026252 012537 001124 L14: MOV (R5)+,$GDDAT ;$GDDAT=S/B.
3981
3982 026256 013712 001124 I14: MOV $GDDAT,(R2) ;SET BUS ADDRESS REGISTER.
3983 026262 011237 001126 MOV (R2),$BDDAT ;READ BUS ADDRESS REGISTER.
3984
3985 026266 023737 001126 001124 CMP $BDDAT,$GDDAT ;COMPARE RESULTS.
3986 026274 001401 BEQ N14 ;GET NEXT TEST DATA.
3987 026276 104052 ERROR 52
3988
3989 026300 005715 N14: TST (R5) ;AT END OF LIST
3990 026302 001363 BNE L14 ;NO!
3991 026304 000440 BR TST25 ;NEXT TEST
3992 026306 000002 LST14A: 2
3993 026310 000004 4
3994 026312 000010 10
3995 026314 000020 20
3996 026316 000040 40
3997 026320 000100 100
3998 026322 000200 200
3999 026324 000400 400
4000 026326 001000 1000
4001 026330 002000 2000
4002 026332 004000 4000
4003 026334 010000 10000
4004 026336 020000 20000
4005 026340 040000 40000
4006 026342 100000 100000
4007 026344 177776 177776
4008 026346 177774 177774
4009 026350 177772 177772
4010 026352 177766 177766
4011 026354 177756 177756
4012 026356 177736 177736
4013 026360 177676 177676
4014 026362 177576 177576
4015 026364 177376 177376
  
```

CZRJHDO,RP04/5/6 DSKLS CTRLR2 MACY11 30A(1052) 24-MAY-79 15:07 L 7
CZRJHD.P12 01-MAR-79 09:10 124 BCTC BUS ADDRESS REGISTER PAGE 91

SEQ 0089

4016	026366	176776	176776
4017	026370	175776	175776
4018	026372	173776	173776
4019	026374	167776	167776
4020	026376	157776	157776
4021	026400	137776	137776
4022	026402	077776	077776
4023	026404	000000	0
4024			

```

4025
4026
4027
4028
4029 026406 000004
4030 026410 012737 000001 001212
4031 026416 012737 000025 017330
4032 026424 012777 000040 166322
4033
4034 026432 012706 001000
4035
4036 026436 013702 014752
4037 026442 010237 045662
4038 026446 012705 026510
4039
4040 026452 005012
4041
4042 026454 012537 001124
4043
4044 026460 113712 001124
4045 026464 011237 001126
4046
4047 026470 023737 001126 001124
4048 026476 001401
4049
4050 026500 104053
4051
4052 026502 005715
4053 026504 001362
4054 026506 000417
4055
4056
4057 026510 000002
4058 026512 000004
4059 026514 000010
4060 026516 000020
4061 026520 000040
4062 026522 000100
4063 026524 000200
4064 026526 000376
4065 026530 000372
4066 026532 000366
4067 026534 000356
4068 026536 000336
4069 026540 000276
4070 026542 000176
4071 026544 000000

;*****
;*TEST 25 BCTC BUS ADDRESS REGISTER LO-BYTE
;*****
TST25: SCOPE
MOV #1,$TIMES ;;DO 1 ITERATION
MOV #TINO,@TSTNM ;THIS SAVES TEST NUMBER
MOV #CLR,@RHCS2 ;CLEAR RH11 CONTROLLER

MOV #STACK,SP ;RESET STACK.

MOV RHBA,R2 ;R2=RH11 BA ADDRESS.
MOV R2,REGADR
MOV #LST15A,R5 ;R5=TEST LIST ADDRESS.

L15: CLR (R2) ;CLEAR RH11 BA REGISTER.

MOV (R5)+,$GDDAT ;$GDDAT=S/B.

I15: MOVB $GDDAT,(R2) ;SET BUS ADDRESS REGISTER.
MOV (R2),$BDDAT ;READ BUS ADDRESS REGISTER.

CMP $BDDAT,$GDDAT ;COMPARE RESULTS.
BEQ R15

ERROR 53

R15: TST (R5) ;AT END OF TEST LIST.
BNE L15 ;NO!
BR TST26 ;NEXT TEST
;THIS LIST WILL BE USED TO LOAD THE LOWER BYTE OF THE BA REGISTER.
;
LST15A: 2
4
10
20
40
100
200
376
372
366
356
336
276
176
0

```

```

4072
4073
4074
4075
4076 026546 000004
4077 026550 012737 000001 001212
4078 026556 012737 000026 017330
4079 026564 012777 000040 166162
4080
4081 026572 012706 001000
4082
4083 026576 013702 014752
4084 026602 010237 045662
4085 026606 012705 026660
4086
4087 026612 005012
4088
4089 026614 012537 001124
4090
4091 026620 005202
4092
4093 026622 113712 001124
4094
4095 026626 005302
4096
4097 026630 011237 001126
4098 026634 000337 001124
4099
4100 026640 023737 001126 001124
4101 026646 001401
4102
4103 026650 104053
4104
4105 026652 005715
4106 026654 001356
4107 026656 000420
4108
4109
4110 026660 000002
4111 026662 000004
4112 026664 000010
4113 026666 000020
4114 026670 000040
4115 026672 000100
4116 026674 000200
4117 026676 000376
4118 026700 000374
4119 026702 000376
4120 026704 000366
4121 026706 000356
4122 026710 000336
4123 026712 000276
4124 026714 000176
4125 026716 000000
4126

;*****
;TEST 26 BCTC BUS ADDRESS REGISTER HI-BYTE
;*****
TST26: SCOPE
MOV #1,$TIMES ;DO 1 ITERATION
MOV #TINO,@#TSTNM ;THIS SAVES TEST NUMBER
MOV #CLR,@RHCS2 ;CLEAR RH11 CONTROLLER

MOV #STACK,SP ;RESET STACK.

MOV RHBA,R2 ;R2=RH11 BA ADDRESS.
MOV R2,REGADR
MOV #LST16A,R5 ;R5=TEST LIST ADDRESS.

L16: CLR (R2) ;CLEAR BA REGISTER.

MOV (R5)+,$GDDAT ;$GDDAT=S/B.

INC R2 ;R2=HI BYTE ADDRESS.

I16: MOVB $GDDAT,(R2) ;SET BUS ADDRESS REGISTER HI-BYTE.

DEC R2 ;R2=FULL WORD ADDRESS.

MOV (R2),$BDDAT ;READ BUS ADDRESS.
SWAB $GDDAT ;ADJUST DATA FOR HI BYTE.

CMP $BDDAT,$GDDAT ;COMPARE RESULTS.
BEQ R16 ;OKAY

ERROR 53

R16: TST (R5) ;AT END OF TEST LIST.
BNE L16 ;NOPE
BR TST27 ;NEXT TEST

;THIS LIST WILL BE USED TO LOAD THE UPPER BYTE OF THE BA REGISTER.
;
LST16A: 2
4
10
20
40
100
200
376
374
376
366
356
336
276
176
0

```

```

4127      .REM      %
4128      THIS TEST CAUSE AN RH11 INTERRUPT VIA THE SPECIAL COMMAND SEQUENCE
4129      BIS      #IE,%RHCS2. THIS TEST PROVES ONLY THAT THE HARDWARE CAN HANDLE
4130      INTERRUPTS PROPERLY
4131      %
4132      ;*****
4133      ;TEST 27      RH11 INTERRUPT TEST
4134      ;*****
4135      TEST27:  SCOPE
4136      026720 000004      MOV      #1,%TIMES      ;;DO 1 ITERATION
4137      026722 012737 000001 001212      MOV      #TNO,%TSTNM      ;THIS SAVES TEST NUMBER
4138      026730 012737 000027 017330      MOV      #CLR,%RHCS2      ;CLEAR RH11 CONTROLLER
4139      026736 012777 000040 166010
4140      026744 012706 001000      MOV      #STACK,SP      ;SET STACK POINTER.
4141
4142      026750 013702 014756      MOV      RHCS1,R2      ;R2=RH11 ADDRESS.
4143      026754 010237 045662      MOV      R2,REGADR
4144      026760 005037 001126      CLR      %BDDAT      ;%BDDAT=WAS.
4145      026764 005037 001124      CLR      %GDDAT      ;%GDDAT=S/B.
4146
4147      026770 017737 165750 001176      MOV      %RPVEC,%TMP0      ;SAVE RH11 INTERRUPT VECTOR.
4148      026776 012777 027022 165740      MOV      #021,%RPVEC      ;SET RH11 INTERRUPT VECTOR TO 021.
4149
4150      027004 052712 000100      L21:  BIS      #IE,(R2)      ;CAUSE INTERRUPT.
4151      027010 000240      NOP
4152      027012 000240      NOP      ;WAIT FOR INTERRUPT.
4153      027014 011237 001124      MOV      (R2),%GDDAT      ;SAVE CONTENTS OF REGISTER.
4154
4155      027020 104054      ERROR      54
4156
4157      027022 013777 001176 165714 021:  MOV      %TMP0,%RPVEC      ;RESTORE RH11 INTERRUPT VECTOR.
4158
4159

```

```
4160
4161      ;TEST THAT RESET CAN GENERATE THE SIGNAL CLR L.
4162      .REM      %
4163      THE PROGRAM SETS THE PORT SELECT BIT .THEN DOES A RESE*
4164      IF THE SIGNAL CLR L WAS GENERATED THE PORT SELECT BIT WILL CLEAR.
4165      %
4166      ;*****
4167      ;*TEST 30      BCTD CLR L TEST
4168      ;*****
4169      TST30:  SCOPE
4170      MOV      #1,$TIMES      ;;DO 1 ITERATION
4171      MOV      #TINO,@#TSTNM  ;THIS SAVES TEST NUMBER
4172      MOV      #CLR,@RHCS2    ;CLEAR RH11 CONTROLLER
4173
4174      MOV      RHCS1,R2        ;R2=RH11 CS1 ADDRESS
4175      MOV      R2,REGADR
4176      CLR      $GDDAT         ;$GDDAT=S/B
4177
4178      L22:  MOV      #STACK,SP  ;SET STACK POINTER.
4179      MOV      #PSEL,(R2)      ;SET PORT SELECT FLOP.
4180      RESET                      ;DO A BUSA INIT.
4181
4182      MOV      (R2),$BDDAT      ;$BDDAT=WAS.
4183      BIC      #^C<IE>,$BDDAT  ;SAVE ONLY I.E. BIT.
4184
4185      CMP      $BDDAT,$GDDAT    ;COMPARE RESULTS.
4186      BEQ      TST31           ;NEXT TEST
4187      ERROR    55
4188
4189      027030 000004
4170 027032 012737 000001 001212
4171 027040 012737 000030 017330
4172 027046 012777 000040 165700
4174 027054 013702 014756
4175 027060 010237 045662
4176 027064 005037 001124
4178 027070 012706 001000
4179 027074 012712 002000
4180 027100 000005
4182 027102 011237 001126
4183 027106 042737 177677 001126
4185 027114 023737 001126 001124
4186 027122 001401
4187 027124 104055
```

```
4190      .REM      %  
4191      SET THE MXF FLOP AND READ IT BACK.  
4192      %  
4193      ;*****  
4194      ;*TEST 31      MXF TEST  
4195      ;*****  
4196      TST31:  SCOPE  
4197      MOV      #1,$TIMES      ;;DO 1 ITERATION  
4198      MOV      #TINO,@#TSTNM  ;THIS SAVES TEST NUMBER  
4199  
4200      ;*CHECK TO SEE IF PROGRAM IS RUNNING WITH AN RH70  
4201  
4202      027144  005737  017334      TST      @#RH70      ;TEST FOR RH70 CONTROLLER  
4203      027150  001402              BEQ      30$      ;IF FLAG = 1, SKIP THIS TEST  
4204      027152  000137  027236      JMP      TST32      ;JUMP TO NEXT TEST -----)  
4205      027156              30$:      ;IF FLAG = 0, DO THIS TEST  
4206      027156  012777  000040  165570  MOV      #CLR,@RHCS2      ;CLEAR RH11 CONTROLLER  
4207  
4208      027164  012706  001000      MOV      #STACK,SP      ;SET STACK POINTER.  
4209      027170  013702  014754      MOV      RHCS2,R2      ;R2=RH11 CS2 ADDRESS.  
4210      027174  010237  045662      MOV      R2,REGADR  
4211      027200  012737  001000  001124      MOV      #MXF,$GDDAT      ;$GDDAT=S/B.  
4212  
4213      027206  012712  001000      124:  MOV      #MXF,(R2)      ;LOAD MXF  
4214  
4215      027212  011237  001126      MOV      (R2),$BDDAT      ;$BDDAT=WAS.  
4216  
4217      027216  042737  176777  001126      BIC      #^C<MXF>,$BDDAT ;SAVE ONLY MXF  
4218  
4219      027224  023737  001126  001124      CMP      $BDDAT,$GDDAT      ;COMPARE RESULTS  
4220      027232  001401              BEQ      TST32      ;NEXT TEST  
4221      027234  104056              ERROR  56  
4222  
4223
```

```

4224      .REM      %
4225      SET THE UNIBUS PARITY ERROR FLOP DIRECTLY VIA A MOV #UPE TO RHCS2
4226      ATTEMPT TO READ IT BACK.
4227      %
4228      ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
4229      : *TEST 32      CSRB UNIBUS PARITY ERROR SET TEST
4230      : ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
4231      TST32:  SCOPE
4232      MOV      #1,%TIMES      ;;DO 1 ITERATION
4233      MOV      #TINO,%TSTNM   ;THIS SAVES TEST NUMBER
4234
4235      ; *CHECK TO SEE IF PROGRAM IS RUNNING WITH AN RH70
4236
4237      TST      @#RH70          ;TEST FOR RH70 CONTROLLER
4238      BEQ      30$            ;IF FLAG = 1, SKIP THIS TEST
4239      JMP      TST33          ;JUMP TO NEXT TEST -----)
4240      30$:
4241
4242      MOV      #CLR,@RHCS2     ;CLEAR RH11 CONTROLLER
4243
4244      MOV      #STACK,SP      ;SET STACK POINTER.
4245
4246      MOV      RHCS2,R2        ;R2=RHCS2 ADDRESS.
4247      MOV      R2,REGADR
4248      MOV      #UPE,%GDDAT    ;%GDDAT-S/B.
4249
4250      L30:  MOV      %GDDAT,(R2) ;ATTEMPT TO SET UPE.
4251      NOP
4252      NOP
4253
4254      MOV      (R2),%BDDAT      ;%BDDAT=ACTUAL DATA.
4255      BIC      #^C<UPE>,%BDDAT ;SAVE ONLY UPE.
4256
4257      CMP      %BDDAT,%GDDAT    ;COMPARE RESULTS.
4258      BEQ      TST33            ;NEXT TEST
4259      ERROR    57
4260
4261
  
```

```

4262
4263
4264
4265
4266
4267
4268
4269
4270 027352 000004
4271 027354 012737 000001 001212
4272 027362 012737 000033 017330
4273 027370 012777 000040 165356
4274
4275 027376 012706 001000
4276
4277 027402 013702 014754
4278 027406 010237 045662
4279 027412 005037 001124
4280
4281 027416 012712 020000
4282 027422 000240
4283 027424 000240
4284
4285 027426 012712 000040
4286
4287 027432 011237 001126
4288 027436 042737 157777 001126
4289
4290 027444 023737 001126 001124
4291 027452 001401
4292 027454 104057
4293
4294

      .REM      %
      SET THE UNIBUS PARITY ERROR FLOP AND ATTEMPT TO CLEAR IT WITH A
      CONTROLLER CLEAR.
      %
      ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
      ;*TEST 33      CSRB UNIBUS PARITY ERROR CLEAR TEST
      ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
      TST33:  SCOPE
              MOV      #1,$TIMES      ;;DO 1 ITERATION
              MOV      #TINO,@#TSTNM  ;THIS SAVES TEST NUMBER
              MOV      #CLR,@RHCS2    ;CLEAR RH11 CONTROLLER
              MOV      #STACK,SP      ;SET STACK POINTER.
              MOV      RHCS2,R2       ;R2=RHCS2 ADDRESS.
              MOV      R2,REGADR
              CLR      $GDDAT         ;$GDDAT=S/B.
              L31:  MOV      #UPE,(R2) ;SET UNIBUS PARITY ERROR SET.
              NOP
              NOP
              MOV      #CLR,(R2)      ;CONTROLLER CLEAR.
              MOV      (R2),$BDDAT    ;READ STATUS
              BIC      #^C<UPE>,$BDDAT ;SAVE ERROR.
              CMP      $BDDAT,$GDDAT  ;COMPARE RESULTS.
              BEQ      TST34          ;NEXT TEST
              ERROR   57
  
```

```
4295
4296
4297
4298
4299
4300
4301
4302
4303 027456 000004
4304 027460 012737 000001 001212
4305 027466 012737 000034 017330
4306 027474 012777 000040 165252
4307
4308 027502 012706 001000
4309
4310 027506 013702 014754
4311 027512 010237 045662
4312 027516 005037 001124
4313
4314 027522 012712 000007
4315 027526 000240
4316 027530 000240
4317
4318 027532 012712 000040
4319
4320 027536 011237 001126
4321 027542 042737 177770 001126
4322
4323 027550 023737 001126 001124
4324 027556 001401
4325 027560 104060
4326
4327

      .REM %
      SET THE UNIT SELECT REGISTER TO DRIVE #7 ALL BITS SET. GENERATE A
      CONTROLLER CLEAR AND VERIFY THAT THE UNIT SELECT REGISTER IS CLEARED.
      %
      *****
      :TEST 34 CSRB UNIT SELECT CLEAR TEST
      *****
      TST34: SCOPE
              MOV #1,$TIMES ;DO 1 ITERATION
              MOV #TINO,@TSTNM ;THIS SAVES TEST NUMBER
              MOV #CLR,@RHCS2 ;CLEAR RH11 CONTROLLER
              MOV #STACK,SP ;SET STACK POINTER.
              MOV RHCS2,R2 ;R2=CS2 ADDRESS.
              MOV R2,REGADR
              CLR $GDDAT ;$GDDAT=S/B.
      L34. MOV #US4!US2!US1,(R2) ;LOAD RHCS2 DRIVE SELECT
              NOP
              NOP
              MOV #CLR,(R2) ;GENERATE CLR.
              MOV (R2),$BDDAT ;READ RHCS2
              BIC #^C<US4!US2.US1>,$BDDAT;SAVE DRIVE SELECT BITS.
              CMP $BDDAT,$GDDAT ;COMPARE RESULTS.
              BEQ TST35 ;NEXT TEST
              ERROR 60
```

```

4328
4329
4330      .REM      %
4331      SET THE UPE FLOP UNIBUS PARITY ERROR. VERIFY THAT THE SETTING OF (UPE)
4332      ALSO SETS THE TRANSFER ERROR (TRE) FLOP
4333      %
4334      ;*****
4335      ;*TEST 35      CSRB TRANSFER ERROR (TRE) - UPE
4336      ;*****
4337      TST35:  SCOPE
4338      MOV      #1,$TIMES      ;;DO 1 ITERATION
4339      MOV      #TINO,@#TSTNM  ;THIS SAVES TEST NUMBER
4340
4341      ;*CHECK TO SEE IF PROGRAM IS RUNNING WITH AN RH70
4342      TST      @#RH70          ;TEST FOR RH70 CONTROLLER
4343      BEQ      30$            ;IF FLAG = 1, SKIP THIS TEST
4344      JMP      TST36          ;JUMP TO NEXT TEST -----)
4345      30$:      ;IF FLAG = 0, DO THIS TEST
4346
4347      MOV      #CLW,@RHCS2    ;CLEAR RH11 CONTROLLER
4348
4349      MOV      #STACK,SP      ;SET STACK POINTER.
4350
4351      MOV      RHCS2,R2        ;R2=RHCS2 ADDRESS.
4352      MOV      #TRE,$GDDAT    ;$GDDAT=S/B.
4353
4354      L35:      MOV      #UPE,(R2) ;SET UNIBUS PARITY ERROR.
4355
4356      MOV      RHCS1,R2        ;R2=RHCS1 ADDRESS.
4357      MOV      R2,REGADR
4358
4359      MOV      (R2),$BDDAT     ;READ REGISTER
4360      BIC      #^C<TRE>,$BDDAT ;SAVE TRANSFER ERROR.
4361
4362      CMP      $BDDAT,$GDDAT   ;COMPARE RESULTS
4363      BEQ      TST36          ;NEXT TEST
4364      ERROR    61
4365
4366

```

```

4367
4368
4369
4370
4371
4372
4373
4374 027676 000004
4375 027700 012737 000001 001212
4376 027706 012737 000036 017330
4377 027714 012777 000040 165032
4378
4379 027722 012706 001000
4380
4381 027726 013702 014754
4382 027732 012737 040000 001124
4383
4384 027740 013700 015136
4385 027744 005001
4386 027746 006000
4387 027750 103420
4388 027752 010112
4389
4390 027754 013702 014756
4391 027760 010237 045662
4392
4393 027764 011237 001126
4394 027770 042737 137777 001126
4395
4396 027776 023737 001126 001124
4397 030004 001406
4398 030006 104062
4399 030010 000404
4400
4401 030012 005201
4402 030014 020127 000010
4403 030020 001352
4404
4405

      .REM %
      SET THE NED BIT NON EXISTANT DRIVE VERIFY THAT THIS SETS THE (TRE) BIT.
      %
      *****
      :TEST 36          CSRB TRANSFER ERROR (TRE) NED
      *****
      TST36:  SCOPE
              MOV      #1,$TIMES          ;;DO 1 ITERATION
              MOV      #TINO,@#TSTNM      ;THIS SAVES TEST NUMBER
              MOV      #CLR,@RHCS2        ;CLEAR RH11 CONTROLLER
              MOV      #STACK,SP          ;SET STACK POINTER.
              MOV      RHCS2,R2           ;R2-CS2 ADDRESS.
              MOV      #TRE,$GDDAT        ;$GDDAT-S/B.
              MOV      @#TOTALAT,R0       ;GET ALL AVAILABLE DRIVES DATA
              CLR      R1
      S36:    ROR      R0
              BCS      N36
      L36:    MOV      R1,(R2)             ;SET NED.
              MOV      RHCS1,R2           ;R2=RHCS1 ADDRESS.
              MOV      R2,REGADR
              MOV      (R2),$BDDAT        ;READ REGISTER.
              BIC      #^C<'RE>,$BDDAT   ;SAVE TRE.
              CMP      $BDDAT,$GDDAT     ;COMPARE RESULTS.
              BEQ      TST37              ;NEXT TEST
              ERROR    62
              BR       TST37              ;NEXT TEST
      N36:    INC      R1
              CMP      R1,#8.
              BNE      S36

```

4406
 4407
 4408
 4409
 4410
 4411
 4412
 4413
 4414
 4415
 4416
 4417
 4418
 4419
 4420
 4421
 4422
 4423
 4424
 4425
 4426
 4427
 4428
 4429
 4430
 4431
 4432
 4433
 4434

.REM %
 SET THE PORT SELECT FLOP VERIFY THAT WE CAN READ IT BACK.
 %

 :TEST 37 CSRA PSEL CLEAR TEST
 :*****

TST37: SCOPE
 MOV #1,\$TIMES ;;DO 1 ITERATION
 MOV #TNO,@TSTNM ;THIS SAVES TEST NUMBER
 MOV #CLR,@RHCS2 ;CLEAR RH11 CONTROLLER
 MOV #STACK,SP ;SET STACK POINTER.
 MOV RHCS1,R2 ;R2=RHCS1 ADDRESS.
 MOV R2,REGADR
 CLR \$GDDAT ;\$GDDAT-S/B.
 L40: MOV #PSEL,(R2) ;SET P SELECT.
 MOV #CLR,@RHCS2 ;DO A CONTROLLER CLEAR.
 MOV (R2),\$BDDAT ;READ BACK P SELECT BIT.
 BIC #^C<PSEL>,\$BDDAT;SAVE ONLY PORT SELECT.
 CMP \$BDDAT,\$GDDAT ;COMPARE RESULTS.
 BEQ TST40 ;NEXT TEST
 ERROR 64

030022 000004
 030024 012737 000001 001212
 030032 012737 000037 017330
 030040 012777 000040 164706
 030046 012706 001000
 030052 013702 014756
 030056 010237 045662
 030062 005037 001124
 030066 012712 002000
 030072 012777 000040 164654
 030100 011237 001126
 030104 042737 175777 001126
 030112 023737 001126 001124
 030120 001401
 030122 104064

```
4435 .REM %  
4436 VERIFY THAT CONTROLLER CLEAR WILL CLEAR THE COMMAND REGISTER.  
4437 %  
4438  
4439 ::*****  
4440 :*TEST 40 CSRB COMMAND REGISTER CLEAR TEST  
4441 :*****  
4442 TST40: SCOPE  
4443 030124 000004 MOV #1,$TIMES ;;DO 1 ITERATION  
4444 030126 012737 000001 001212 MOV #TINO,@TSTNM ;THIS SAVES TEST NUMBER  
4445 030134 012737 000040 017330 MOV #CLR,@RHCS2 ;CLEAR RH11 CONTROLLER  
4446 030142 012777 000040 164604  
4447 030150 012706 001000 MOV #STACK,SP ;SET STACK POINTER.  
4448  
4449 030154 013702 014756 MOV RHCS1,R2 ;R2=RHCS1 ADDRESS.  
4450 030160 010237 045662 MOV R2,REGADR  
4451 030164 005037 001124 CLR $GDDAT ;$GDDAT=S/B.  
4452  
4453 030170 012712 000011 L41: MOV #11,(R2) ;ISSUE A DRIVE CLR AND GO.  
4454  
4455 030174 012777 000040 164552 MOV #CLR,@RHCS2 ;CONTROLLER CLEAR.  
4456  
4457 030202 011237 001126 MOV (R2),$BDDAT ;READ COMMAND REGISTER.  
4458 030206 042737 177770 001126 BIC #^C<7>,$BDDAT ;SAVE ONLY THE COMMAND BITS.  
4459  
4460 030214 023737 001126 001124 CMP $BDDAT,$GDDAT ;COMPARE RESULTS.  
4461 030222 001401 BEQ TST41 ;NEXT TEST  
4462 030224 104065 ERROR 65  
4463
```

```

4464
4465      .REM      %
4466      THE COMMAND REGISTER IS SUPPOSED TO BE SELF CLEARING THAT IS WHEN THE
4467      RH11 IS SELECTED WHEATHER OR NOT WE ADDRESS IT DIRECTLY THE LOGIC
4468      SHOULD CLEAR THE COMMAND REGISTER.
4469      THIS IS TESTED BY LOADING A COMMAND INTO IT READING ANOTHER REGISTER
4470      THAN RETURNING TO CHECK THE COMMAND REGISTER TO DETERMINE IF IT CLEARED.
4471      %
4472      ;*****
4473      ;*TEST 41      CSRB COMMAND REGISTER RESELECT CLEAR TEST
4474      ;*****
4475      TST41:  SCOPE
4476      MOV      #1,$TIMES      ;;DO 1 ITERATION
4477      MOV      #TINO,@#TSTNM  ;THIS SAVES TEST NUMBER
4478      MOV      #CLR,@RHCS2    ;CLEAR RH11 CONTROLLER
4479
4480      MOV      #STACK,SP      ;SET STACK POINTER.
4481
4482      MOV      RHCS1,R2        ;R2=RHCS1 ADDRESS.
4483      MOV      R2,REGADR
4484      CLR      $GDDAT          ;$GDDAT=S/B.
4485
4486      MOV      #340,PS         ;SET PRIORITY TO #7.
4487
4488      L42:  MOV      #11,(R2)   ;ISSUE A DRIVE CLR AND GO
4489
4490      MOV      (R2),$TMP0      ;DO A RESELECT OF THE REGISTER.
4491
4492      MOV      (R2),$BDDAT     ;READ THE COMMAND REGISTEP.
4493      BIC      #^C<7>,$BDDAT  ;SAVE ONLY THE COMMAND BITS.
4494
4495      CMP      $BDDAT,$GDDAT   ;COMPARE RESULTS.
4496      BEQ      TST42           ;NEXT TEST
4497      ERROR    66
4498
  
```

```

4499
4500
4501
4502
4503
4504
4505
4506 030334 000004
4507 030336 012737 000001 001212
4508 030344 012737 000042 017330
4509 030352 012777 000040 164374
4510
4511 030360 013702 014756
4512 030364 010237 045662
4513 030370 005037 001126
4514 030374 005037 001124
4515
4516 030400 017737 164340 001176
4517 030406 012777 030442 164330
4518
4519 030414 012706 001000
4520 030420 005037 177776
4521
4522 030424 052712 000300
4523 030430 000240
4524 030432 000240
4525
4526 030434 011237 001124
4527 030440 104067
4528
4529 030442 013777 001176 164274 043:

      .REM      %
      HERE WE TEST TO SEE IF SETTING (IE) AND (RDY) WILL CAUSE AN INTERRUPT.
      %
      ;*****
      ;*TEST 42      CSRB READY AND IE INTERRUPT TEST
      ;*****
      TST42:  SCOPE
                MOV      #1,$TIMES      ;;DO 1 ITERATION
                MOV      #TINO,@#TSTNM  ;;THIS SAVES TEST NUMBER
                MOV      #CLR,@RHCS2    ;;CLEAR RH11 CONTROLLER
                MOV      RHCS1,R2        ;R2=RPCS1 ADDRESS.
                MOV      R2,REGADR
                CLR      $BDDAT         ;$BDDAT=WAS.
                CLR      $GDDAT         ;$GDDAT=S/B.
                MOV      @RPVEC,$TMP0   ;SAVE RH11 INTERRUPT VECTOR.
                MOV      #043,@RPVEC    ;SET RH11 INTERRUPT VECTOR 043
                L43:  MOV      #STACK,SP  ;SET STACK POINTER.
                CLR      PS
                BIS      #IE!RDY,(R2)   ;THIS WILL CAUSE AN INTERRUPT.
                NOP
                NOP
                MOV      (R2),$GDDAT
                ERROR    67
                MOV      $TMP0,@RPVEC   ;RESTORE RH11 INTERRUPT VECTOR.
  
```

```
4530
4531
4532 .REM %
4533 THE (IE) BIT IS SET CAUSING AN INTERRUPT. AT THE COMPLETION OF THE INTERRUPT
4534 WE CHECK TO SEE IF THE (IE) BIT HAS CLEARED.
4535
4536 ::*****
4537 ::*TEST 43 CSRB DOES "IE" CLEAR AFTER AN INTERRUPT
4538 ::*****
4538 030450 000004
4539 030452 012737 000001 001212 TST43: SCOPE
4540 030460 012737 000043 017330 MOV #1,$TIMES ;;DO 1 ITERATION
4541 030466 012777 000040 164260 MOV #TINO,@#TSTNM ;THIS SAVES TEST NUMBER
4542 MOV #CLR,@RHCS2 ;CLEAR RH11 CONTROLLER
4543 030474 013702 014756 MOV RHCS1,R2 ;R2=RPCS1 ADDRESS.
4544 030500 010237 045662 MOV R2,REGADR
4545 030504 005037 001124 CLR $GDDAT ;$GDDAT=S/B.
4546
4547 030510 017737 164230 001176 MOV @RPVEC,$TMP0 ;SAVE RH11 INTERRUPT VECTOR.
4548 030516 012777 030546 164220 MOV #044,@RPVEC ;SET VECTOR TO 044.
4549
4550 030524 012706 001000 L44: MOV #STACK,SP ;SET STACK POINTER.
4551 030530 005037 177776 CLR PS
4552
4553 030534 052712 000100 BIS #IE,(R2) ;THIS WILL CAUSE AN INTERRUPT
4554 030540 000240 NOP
4555 030542 000240 NOP
4556
4557 030544 000411 BR E44 ;NO INTERRUPT.
4558
4559 030546 011237 001126 044: MOV (R2),$BDDAT ;READ RHCS1.
4560 030552 042737 177677 001126 BIC #^C<IE>,$BDDAT ;SAVE ONLY THE "IE" BIT.
4561
4562 030560 023737 001126 001124 CMP $BDDAT,$GDDAT ;COMPARE RESULTS.
4563 030566 001401 BEQ R44 ;OK!
4564
4565 030570 104070 E44: ERROR 70
4566
4567 030572 013777 001176 164144 R44: MOV $TMP0,@RPVEC ;RESTORE RH11 INTERRUPT VECTOR.
4568
```

```
4569
4570
4571      .REM      %
4572      HERE WE VERIFY THAT WRITING INTO THE "AS" REGISTER OWILL NOT CAUSE
4573      AN (NED) ERROR.
4574      %
4575      ;*****
4576      ;TEST 44      ECTB "AS" WRITE TEST
4577      ;*****
4577      TST44:  SCOPE
4578      MOV      #1,$TIMES      ;;DO 1 ITERATION
4579      MOV      #TINO,@#TSTNM      ;THIS SAVES TEST NUMBER
4580      MOV      #CLR,@RHCS2      ;CLEAR RH11 CONTROLLER
4581
4582      MOV      #STACK,SP      ;SET STACK POINTER.
4583
4584      MOV      RHCS2,R2      ;R2=RH11 CS2 ADDRESS.
4585      MOV      #US4!US2!US1,$GDDAT;$GDDAT=S/B.
4586
4587      MOV      RHCS1,R5      ;ADDRESS OF A DEVICE REGISTER.
4588
4589      L45:  MOV      #US4!US2!US1,(R2);LOAD A NON EXISTANT DEVICE.
4590
4591      MOV      RHAS,R2      ;R2="AS" ADDRESS.
4592
4593      MOV      #377,(R2)      ;ATTEMPT TO LOAD "AS".
4594
4595      CLR      (R2)      ;CLEAR "AS".
4596
4597      MOV      RHCS2,R2      ;R2=RH11 CS2 ADDRESS.
4598      MOV      R2,REGADR
4599
4600      MOV      (R2),$BDDAT      ;$BDDAT=WAS.
4601      BIC      #^C<NED!US4!US2!US1>,$BDDAT;SAVE NED AND DEVICE SELECTED.
4602
4603      CMP      $BDDAT,$GDDAT      ;COMPARE RESULTS.
4604      BEQ      TST45      ;NEXT TEST
4605      ERROR      71
4606
```

```
4607 .SBTTL EXTENDED MEMORY, DRIVE TIMING & SECTOR SELECT TESTS
4608
4609
4610
4611 *****
4612 *TEST 45 MAKE CURRENT CYLINDER = 0
4613 *****
4614 TST45: SCOPE
4615 MOV #STACK,SP ;RESET STACK
4616 MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
4617 JSR PC,@#CLDISK ;INIT DRIVE
4618 MOV #DMD,@#RHMR ;SET DIAGNOSTIC MODE
4619 JSR RO,@#MAKECYL ;SUBROUTINE TO GIVE A SEEK
4620 ;COMMAND FOLOWED BY AN INIT
4621 ;THIS SHUOLD CHANGE RHCC TO 0
4622
4623
4624
4625
4626
4627
4628
4629 *****
4630 *TEST 46 RHCS1 - BITS 8 AND 9 - EXTENDED ADDR (A16 & A 17)
4631
4632 * WRITE CYLINDER 0, FORMAT 16 BITS PER WORD
4633 * TRACK 0, SECTOR 0, KEYS 0, NUMBER OF WORDS 256
4634 * DATA IS THE CONTENTS OF THE TIY READER STATUS REGISTER
4635 * THIS WILL USE BITS A16 AND A17 WHEN THERE IS MORE THAN 28K OF MEMORY
4636 *****
4637 TST46: SCOPE
4638
4639 MOV #STACK,SP ;RESET STACK
4640 MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
4641
4642 ;*CHECK TO SEE IF PROGRAM IS RUNNING WITH AN RH70
4643
4644 TST @#RH70 ;TEST FOR RH70 CONTROLLER
4645 BEQ 30$ ;IF FLAG = 1, SKIP THIS TEST
4646 JMP TST47 ;JUMP TO NEXT TEST -----)
4647 30$: ;IF FLAG = 0, DO THIS TEST
4648
4649 JSR RO,@#CLAREA ;CLEAR SIMULATED DISK
4650 .WORD DISK ;FROM
4651 .WORD TOLGAP+16 ;TO
4652 .WORD 0 ;DATA
4653
4654 ;THESE ARE SETUP FOR DISKLESS USE ONLY
4655
4656 MOV #FMT22,@#CYL;CYLINDER 0
4657 ;16 BITS PER WORD
4658 CLR @#SECOTR ;SECTOR 0 TRACK 0
4659 CLR @#KEY1 ;KEY1 0
4660 CLR @#KEY2 ;KEY2 0
4661 MOV #256,@#NOWORD ;NO OF DATA WORDS
4662 MOV #1,@#X ;WRITE DATA
```

4663	031052	004537	047332		JSR	RS,@#CRC	;GO TO CALCULATE CRC
4664	031056	053010			CYL		
4665	031060	054710			WCRC		
4666							
4667							
4668							
4669							
4670	031062	004737	046116		JSR	PC,@#CLDISK	;SETUP GENERAL REGISTERS
4671	031066	012777	177400	163654	MOV	#-256,@#HWC	;256 DATA WORDS
4672	031074	013777	001144	163650	MOV	@#STKS,@#HBA	;STARTING ADDRESS OF WRITE BUFFER
4673	031102	017737	150036	015140	MOV	@#STKS,@#TMPILL	;TEMPORARY STORAGE OF DATA
4674	031110	005077	163646		CLR	@#HDST	;SECTOR 0 TRACK 0
4675	031114	012777	010000	163644	MOV	#FMT22,@#HOF	;16 BITS PER WORD FORMAT
4676	031122	005077	163642		CLR	@#HCA	;CYLINDER 0
4677	031126	004737	046160		JSR	PC,@#CHECKT	;CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DON'T
4678	031132	104401	005123		TYPE	,CPHALT	;CANNOT CONTINUE TESTING IF ANY OF THE
4679							;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
4680	031136	000000			HALT		;STOP THE TEST
4681	031140	013746	015170		MOV	@#WRIDAT,-(SP)	;WRITE DATA=60
4682	031144	052716	001400		BIS	#A16.A17,(SP)	;SET HIGH ORDER UNIBUS BITS
4683	031150	012611			MOV	(SP)+,@#R1	;FILL RHCS1
4684	031152	052777	000010	163574	BIS	#BA1,@#RHCS2	;SET BUS ADDRESS INHIBIT
4685	031160	005037	015124		CLR	@#ERFLGS	;CLEAR ERROR FLAG
4686	031164	004737	052650		JSR	PC,@#COMHD	;WRITE DATA

;THESE ARE REGULAR SETUPS

```
4687
4688
4689
4690
4691
4692
4693
4694
4695 031170 004737 045616 JSR PC,@#PUTREG ;SAVE REGISTERS
4696 031174 005737 015124 TST @#ERFLG$ ;HAVE ANY ERRORS OCCURED?
4697 031200 001042 BNE TST47 ; BRANCH OUT IF YES
4698 031202 013700 015140 MOV @#TMPILL,R0 ;GOOD DATA
4699 031206 012701 054726 MOV #DISK,R1 ;DATA WRITTEN INTO "DISK"
4700 031212 012702 000400 MOV #256.,R2 ;COUNTER
4701 031216 012737 000401 053130 1$: MOV #257.,@#ERWORD ;FOR ERROR WORD
4702 031224 020021 CMP R0,(R1)+ ;COMPARE GOOD DATA WITH DATA ON DISK
4703 031226 001425 BEQ 3$ ;BRANCH IF GOOD
4704 031230 013737 015140 001124 MOV @#TMPILL,@#SGDDAT ;GOOD DATA
4705 031236 014137 001126 MOV -(R1),@#SBDDAT ;BAD DATA
4706 031242 160237 053130 SUB R2,@#ERWORD ;ERROR WORD NO
4707 031246 005737 015124 TST @#ERFLG$ ;ANY ERRORS ALREADY THERE?
4708 031252 001002 BNE 2$ ;BRANCH IF YES
4709 031254 104037 ERROR 37 ;ERROR ON WRITE DATA COMMAND
4710
4711 031256 000401 BR 64$ ;SEE NEXT ERROR COMMENTS
4712 031260 104040 2$: ERROR 40 ;BRANCH TO AVOID PRINTING NEXT ERROR
4713
4714
4715 031262 005721 64$: TST (R1)+ ;WORD NO GIVES WORD IN ERROR
4716 031264 017746 147650 MOV @SWR,-(SP) ;ERROR OCCURED WHILE WRITING
4717 031270 042716 177177 BIC #177177,(SP) ;WITH A16 A17 OF RHCS1 SET
4718 031274 022726 000200 CMP #SW07,(SP)+ ;UNDO -(R1) FOR BAD DATA
4719 031300 001402 BEQ TST47 ;GET SWITCH SETTING
4720 031302 005302 3$: DEC R2 ;KEEP ONLY SWITCH 7 AND 8
4721 031304 001344 BNE 1$ ;IS 7 SET AND 8 RESET
; BRANCH OUT IF YES
; IF NOT COUNT 256 WORDS
;BRANCH IF 256 NOT DONE
```

```
4722
4723
4724
4725
4726
4727
4728
4729
4730
4731
4732
4733
4734
4735
4736 031306 000004
4737 031310 012706 001000
4738 031314 012737 000047 017330
4739
4740
4741
4742 031322 012737 010000 053010
4743
4744 031330 005037 053012
4745
4746 031334 005037 053014
4747 031340 005037 053016
4748 031344 005037 053020
4749 031350 004537 047332
4750 031354 053010
4751 031356 054710
4752
4753
4754
4755 031360 004737 046116
4756 031364 012777 177374 163356
4757 031372 012777 016264 163352
4758 031400 005077 163356
4759
4760 031404 012777 014000 163354
4761
4762 031412 005077 163352
4763 031416 004737 046160
4764 031422 104401 005123
4765
4766 031426 000000
4767 031430 013711 015176
4768
4769
4770
4771 031434 004037 046616
4772 031440 014750
4773 031442 015024
4774 031444 000023
4775
4776
4777
```

```
*****
: *TEST 47      DRIVE TIMING ERROR
:
: *      A READ HEADER AND DATA IS STARTED ON CYLINDER 0, SECTOR
: *      0, TRACK 0, 260 WORDS. AFTER THE HEADER IS READ IN CORRECTLY,
: *      NO SYNC BYTE (DATA SYNC) IS GIVEN.
: *      THEN NORMAL DIAGNOSTIC DATA CLOCKS AND DIAGNOSTIC
: *      SECTOR CLOCKS ARE GIVEN FOR 24 BYTES.
:
: *      THEN 536 BYTES OF SECTOR CLOCKS ONLY ARE GIVEN.
: *      THIS IS TO BRING SECTOR PULSE UP WHICH SHOULD
: *      SET 'DRIVE TIMING ERROR' - 'DTE'
: *****
TST47: SCOPE
MOV      #STACK,SP      ;RESET STACK
MOV      #TINO,@TSTNM   ;THIS SAVES TEST NUMBER

; *THESE ARE TO SETUP FOR DISKLESS USE

MOV      #FMT22,@#CYL   ;16 BITS PER WORD
                        ;CYLINDER 0
CLR      @#SECOTR       ;SECTOR 0
                        ;TRACK 0
CLR      @#KEY1         ;KEY1 = 0
CLR      @#KEY2         ;KEY2 = 0
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      #-260,@RHWC    ;256 DATA WORDS, 4 HEADER WORDS
MOV      #REINTO,@RHBA  ;STARTING ADDRESS OF BUFFER
CLR      @RHDS1         ;TRACK = 0
                        ;SECTOR = 0
MOV      #FMT22!EC1,@RHOF ;16 BITS PER WORD
                        ;ECC CORRECTION INHIBITED
CLR      @RHCA          ;CYLINDER = 0
JSR      PC,@#CHECKT    ;CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DON'T
TYPE      ,CPHALT       ;CANNOT CONTINUE TESTING IF ANY OF THE
                        ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
HALT
MOV      @#REFOR,@R1    ;READ HEADER AND DATA = 72

; *READ & SAVE REGISTERS FOR COMPARISON AFTER SIMULATED 'DTE'

JSR      R0,@#SAVER     ;READ IN SEQUENCE
RHWC          ;FROM HARDWARE REGISTER
WC          ;INTO CORE AT LOCATION
19.          ;NUMBER OF REGISTERS TO READ

; *NOW 'GO' WILL BE GIVEN, EVERYTHING WILL BE TREATED
```

```

4778 ;*NORMALLY FOR THE HEADER. BUT WHEN IT IS TIME TO READ
4779 ;*DATA, ONLY SECTOR CLOCKS WILL BE GIVEN. NO DIAGNOSTIC DATA
4780 ;*CLOCKS WILL BE GIVEN. THIS SHOULD BRING SECTOR PULSE HIGH
4781 ;*WITHOUT PUTTING 'READ' DOWN HENCE 'DTE' WILL COME UP.
4782
4783 031446 012737 177777 015144 MOV # -1, @#TESDTE ;SET DTE TEST
4784 031454 012737 177777 015124 MOV # -1, @#ERFLGS ;THIS WILL BRING THE READ HEADER
4785 ;AND DATA PROCESS OUT AFTER THE
4786 ;HEADER HAS BEEN CORRECTLY READ
4787
4788 031462 004737 052650 JSR PC, @#COMHD ;ISSUE 'GO', SEARCH FOR THE SECTOR
4789 ;AND READ THE HEADER
4790
4791 031466 017737 163264 001176 SETCK1: MOV @RHCS1, @#STMPO ;READ CS1 TO CHECK FOR ANY READ ERRORS
4792 031474 032737 100000 001176 BIT #SC, @#STMPO ;TEST FOR 'SPECIAL CONDITION' - 'SC'
4793 031502 001405 BEQ 8$ ;CONTINUE WITH TEST IF 'SC' = 0 (NO ERRORS)
4794 031504 004737 045616 JSR PC, @#PUTREG ;READ & SAVE ALL REGISTERS AGAIN IF AN
4795 ;UNDEFINED DATA TRANSFER ERROR OCCURRED
4796 031510 104040 ERROR 40 ;READ/WRITE HEADER & DATA ERROR DURING
4797 031512 000137 032000 JMP TST50 ; 'DTE' TEST - ABORT THE TEST
4798
4799 031516 8$: ;NOW THE HEADER HAS BEEN READ OK
4800 ;*NOW 560 SECTOR CLOCKS WILL BE GIVEN
4801 ;*GAP 11 BYTES, SYNC 1 BYTE, DATA 512, ECC 4 BYTES
4802 ;*GAP 2 BYTES, TOLERANCE 28 BYTES, EXTRA 2
4803
4804 ;*THESE 560 SECTOR CLOCKS ARE DIVIDED INTO TWO GROUPS
4805 ;*24 SECTOR CLOCKS WITH NORMAL DIAGNOSTIC DATA CLOCKS,
4806 ;*AND 536 SECTOR CLOCKS WITHOUT ANY DIAGNOSTIC DATA CLOCKS.
4807
4808
4809 ;*THIS GIVES 24 SECTOR CLOCKS WITH DIAGNOSTIC DATA CLOCKS
4810 ;*WHICH EQUALS 3 BYTES OF DATA
4811
4812 031516 012701 000030 MOV #24, R1 ;LOAD COUNTER
4813 031522 013700 014776 MOV @#RHMR, R0 ;GET RHMR ADDRESS
4814 031526 012710 000001 MOV #DMD, @R0 ;SET DIAGNOSTIC MODE
4815 031532 052710 000012 1$: BIS #MSTCK!MCLK, @R0 ;SET SECTOR CLOCK AND DATA CLOCK
4816 031536 042710 000012 BIC #MSTCK!MCLK, @R0 ;CLEAR SECTOR CLOCK AND DATA CLOCK
4817 031542 012702 000007 MOV #7, R2 ;LOAD COUNTER FOR DIAGNOSTIC CLOCKS
4818 031546 052710 000002 4$: BIS #MCLK, @R0 ;SET CLOCK
4819 031552 042710 000002 BIC #MCLK, @R0 ;CLEAR CLOCK
4820 031556 005302 DEC R2 ;COUNT TO 7
4821 031560 001372 BNE 4$ ;BRANCH IF 7 NOT DONE
4822 031562 005301 DEC R1 ;COUNT TO 24
4823 031564 001362 BNE 1$ ;BRANCH IF 24 NOT DONE
4824
4825 ;*THIS GIVES 536 SECTOR CLOCKS WITHOUT DIAGNOSTIC DATA CLOCKS
4826
4827 031566 012701 001030 5$: MOV #536, R1 ;LOAD SECTOR CLOCK COUNTER
4828 031572 052710 000010 BIS #MSTCK, @R0 ;SET SECTOR CLOCK
4829 031576 042710 000010 BIC #MSTCK, @R0 ;CLEAR SECTOR CLOCK
4830 031602 005301 DEC R1 ;COUNT
4831 031604 001372 BNE 5$ ;BRANCH IF 536 NOT DONE
4832
4833

```

```
4834 ;*NOW 'DTE' SHOULD BE SET
4835 ;*CHANGE SAVED REGISTERS TO EXPECTED VALUES
4836
4837 031606 012737 177400 015024 MOV # -256.,@#WC ; SAVED RHWC
4838 031614 012737 016274 015026 MOV #REINTO+<4.*2>,@#BA ; SAVED RHBA
4839 031622 052737 140000 015032 BIS #SC!TRE,@#CS1 ; SAVED RHCS1
4840 031630 052737 010000 015034 BIS #DTE,@#ER1 ; SAVED RHER1
4841 031636 012737 000401 015052 MOV #401,@#MR ; SAVED RHMR
4842 031644 052737 140000 015054 BIS #ATA!ERR,@#DS1 ; SAVED RHDS1
4843 031652 012737 000100 015066 MOV #100,@#LA ; SAVED RHLA
4844 031660 012737 000001 015036 MOV #1,@#DST ; SAVED RHDST
4845 031666 013737 015134 015050 MOV @#ATTENT,@#AS ; SAVED RHAS
4846
4847
4848 ;*NOW READ & SAVE REGISTERS AGAIN SO THAT COMPARISONS
4849 ;*CAN BE DONE (USE THE 'WRFROM' SAVE BUFFER THIS TIME)
4850
4851 031674 004037 046616 JSR RO,@#SAVER ; READ IN SEQUENCE
4852 031700 014750 RHWC ; FROM HARDWARE REGISTER
4853 031702 015220 WRFROM ; INTO CORE BUFFER
4854 031704 000023 19. ; NUMBER OF REGISTERS TO READ
4855
4856 ;*FOR RHAS UPPER BYTE
4857 031706 113737 015051 015245 MOVB @#AS+1,@#WRFROM+25 ; UPPER RHAS
4858
4859 ;*COMPARE THE HEADER READ
4860
4861 031714 004037 047020 JSR RO,@#COMPAR ; COMPARE
4862 031720 053010 CYL ; GOOD BUFFER
4863 031722 016264 REINTO ; TEST BUFFER
4864 031724 000004 4. ; NUMBER
4865 031726 031734 6$ ; RETURN FOR ERROR
4866 031730 031734 6$ ; SAME
4867 031732 031740 7$ ; RETURN FOR GOOD COMPARISON
4868 031734 104010 6$: ERROR 10 ; HEADER READ IN DURING THIS TEST IS
4869 ; IN ERROR
4870
4871 031736 000207 RTS PC ; RETURN
4872
4873 031740 7$: ; GOOD COMPARISON, CONTINUE
4874
4875
4876 ;*COMPARE REGISTERS BEFORE COMMAND WITH REGISTERS AFTER COMMAND
4877
4878 031740 004037 047020 JSR RO,@#COMPAR ; COMPARE
4879 031744 015024 WC ; INITIAL SNAPSHOT BUFFER (CHANGED)
4880 031746 015220 WRFROM ; TEST SNAPSHOT BUFFER
4881 031750 000022 18. ; NUMBER OF REGISTERS
4882 031752 031760 2$ ; RETURN FOR ERROR
4883 031754 031760 2$ ; SAME
4884 031756 032000 3$ ; RETURN FOR GOOD COMPARISON
4885
4886 031760 013705 053130 2$: MOV @#ERWORD,R5 ; GETTING READY TO INDEX
4887 031764 060505 ADD R5,R5 ; DOUBLE ERROR WORD
4888 031766 016537 014746 045662 MOV RHWC-2(R5),@#REGADR ; FAILING REGISTER
4889 031774 104001 ERROR 1 ; IMPROPER REGISTER
```

CC

CC

RTS PC

33:

```
;GOOD - REGISTERS OK, GO ON TO NEXT TEST
```

```
4897
4898
4899
4900
4901
4902
4903
4904
4905
4906
4907
4908
4909
4910
4911
4912
4913
4914 032000 000004
4915 032002 012706 001000
4916 032006 012737 000050 017330
4917
4918
4919 032014 012737 010000 053010
4920
4921 032022 005037 053012
4922
4923 032026 005037 053014
4924 032032 005037 053016
4925 032036 012737 177777 053020
4926 032044 004537 047332
4927 032050 053010
4928 032052 054710
4929
4930
4931
4932 032054 004737 046116
4933 032060 012777 177400 162662
4934 032066 012777 015220 162656
4935 032074 005077 162662
4936
4937 032100 012777 010000 162660
4938
4939 032106 005077 162656
4940 032112 004737 046160
4941 032116 104401 005123
4942
4943 032122 000000
4944 032124 013711 015170
4945
4946
4947
4948 032130 004037 046616
4949 032134 014750
4950 032136 015024
4951 032140 000023
4952
```

```
*****
;*TEST 50      DRIVE TIMING ERROR

;*      A WRITE DATA COMMAND IS STARTED ON CYLINDER 0, SECTOR
;*      0, TRACK 0, 256 WORDS.

;*      THE SECTOR IS SEARCHED FOR AND
;*      AFTER THE HEADER IS READ IN CORRECTLY,
;*      NO SYNC BYTE (DATA SYNC) IS GIVEN.
;*      NORMAL DIAGNOSTIC DATA CLOCKS AND DIAGNOSTIC
;*      SECTOR CLOCKS ARE GIVEN FOR 24 BYTES,
;*      THEN 536 BYTES OF SECTOR CLOCKS ONLY, ARE GIVEN.

;*      THIS IS TO BRING SECTOR PULSE UP
;*      WHICH SHOULD SET 'DRIVE TIMING ERROR' - 'DTE'
*****
TST50: SCOPE
MOV      #STACK,SP      ;RESET STACK
MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
;*THESE ARE TO SETUP FOR DISKLESS USE

MOV      #FMT22,@#CYL    ;16 BITS PER WORD
                        ;CYLINDER 0
CLR      @#SECOTR        ;SECTOR 0
                        ;TRACK 0
CLR      @#KEY1          ;KEY1 = 0
CLR      @#KEY2          ;KEY2 = 0
MOV      #-1,@#X         ;THIS IS FOR WRITE DATA COMMAND
JSR      R5,@#CRC        ;GO TO CALCULATE CRC
CYL
WCRC

;* THESE      ARE REGULAR SETUPS & CHECKS

JSR      PC,@#CLDISK     ;SETUP GENERAL REGISTERS
MOV      #-256.,@#RHWC   ;256 DATA WORDS
MOV      #WRFROM,@#HBA   ;STARTING ADDRESS OF BUFFER
CLR      @#RHDS          ;TRACK = 0
                        ;SECTOR = 0
MOV      #FMT22,@#RHOF   ;16 BITS PER WORD
                        ;ECC CORRECTION INHIBITED
CLR      @#RHCA          ;CYLINDER = 0
JSR      PC,@#CHECKT     ;CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DON'T
TYPE      ,CPHALT        ;CANNOT CONTINUE TESTING IF ANY OF THE
                        ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
HALT
MOV      @#WRIDAT,@#R1   ;WRITE DATA = 60

;*READ & SAVE REGISTERS FOR COMPARISON AFTER SIMULATED 'DTE'

JSR      R0,@#SAVER      ;READ REGISTERS IN SEQUENCE
RHWC
WC
19.                      ;FROM HARDWARE REGISTER
                        ;INTO CORE BUFFER LOCATION
                        ;NUMBER OF REGISTERS TO READ
```

```

4953 ;*NOW 'GO' WILL BE GIVEN, EVERYTHING WILL BE TREATED
4954 ;*NORMALLY FOR THE HEADER. WHEN IT IS TIME TO WRITE
4955 ;*DATA, ONLY SECTOR CLOCKS WILL BE GIVEN, NO DIAGNOSTIC DATA
4956 ;*CLOCKS WILL BE GIVEN. THIS SHOULD BRING SECTOR PULSE HIGH
4957 ;*WITHOUT PUTTING READ DOWN, HENCE 'DTE' WILL COME UP.
4958
4959 032142 012737 177777 015144 MOV #1,@#TESDTE ;SET DTE TEST
4960 032150 012737 177777 015124 MOV #1,@#ERFLG$ ;THIS WILL BRING THE READ HEADER
4961 ;AND DATA PROCESS OUT AFTER THE
4962 ;HEADER HAS BEEN CORRECTLY READ
4963
4964 032156 004737 052650 JSR PC,@#COMMD ;ISSUE 'GO', SEARCH FOR SECTOR,
4965 ;READ HEADER AND DATA.
4966
4967 032162 017737 162570 001176 SETCK2: MOV @RHCS1,@#$TMP0 ;READ CS1 TO CHECK FOR READ ERRORS
4968 032170 032737 100000 001176 BIT #SC,@#$TMP0 ;TEST FOR 'SPECIAL CONDITION' - 'SC'
4969 032176 001405 BEQ 6$ ;CONTINUE TESTING IF NO ERROR
4970 032200 004737 045616 JSR PC,@#PUTREG ;READ & SAVE REGISTERS AGAIN IF UNDE-
4971 ;FINED ERROR HAS OCCURRED
4972 032204 104040 ERROR 40 ;THERE WAS A READ/WRITE HEADER ERROR
4973 ;DURING 'DTE' TEST SETUP
4974 032206 000137 032524 JMP TST51 ;ABORT THE TEST
4975
4976 032212 6$: ;NOW THE HEADER HAS BEEN READ,
4977 ;*560 SECTOR CLOCKS WILL BE GIVEN -
4978 ;*GAP 11 BYTES, SYNC 1 BYTE, DATA 512, ECC 4 BYTES
4979 ;*GAP 2 BYTES, TOLERANCE 28 BYTES, EXTRA 2
4980
4981 ;*THESE 560 SECTOR CLOCKS ARE DIVIDED INTO TWO GROUPS
4982 ;*24 SECTOR CLOCKS WITH NORMAL DIAGNOSTIC DATA CLOCKS
4983 ;*AND 536 SECTOR CLOCKS WITHOUT ANY DIAGNOSTIC DATA CLOCKS
4984
4985 ;*THIS GIVES 24 SECTOR CLOCKS WITH DIAGNOSTIC DATA CLOCKS
4986
4987
4988 032212 012701 000030 MOV #24.,R1 ;LOAD SECTOR CLOCK COUNTER
4989 032216 013700 014776 MOV @#RHMR,R0 ;GET RHMR ADDRESS
4990 032222 012710 000001 MOV #DMD,@R0 ;SET DIAGNOSTIC MODE
4991 032226 052710 000012 1$: BIS #MSTCK!MCLK,@R0 ;SET SECTOR CLOCK AND DATA CLOCK
4992 032232 042710 000012 BIC #MSTCK!MCLK,@R0 ;CLEAR SECTOR CLOCK AND DATA CLOCK
4993 032236 012702 000007 MOV #7,R2 ;LOAD COUNTER FOR DIAGNOSTIC DATA CLOCKS
4994 032242 052710 000002 4$: BIS #MCLK,@R0 ;SET CLOCK (DATA)
4995 032246 042710 000002 BIC #MCLK,@R0 ;CLEAR CLOCK (DATA)
4996 032252 005302 DEC R2 ;COUNT
4997 032254 001372 BNE 4$ ;BRANCH IF 7 NOT DONE
4998 032256 005301 DEC R1 ;COUNT
4999 032260 001362 BNE 1$ ;BRANCH IF 24 NOT DONE
5000
5001 ;*THIS GIVES 536 SECTOR CLOCKS WITHOUT DIAGNOSTIC DATA CLOCKS
5002
5003 032262 012701 001030 5$: MOV #536.,R1 ;LOAD SECTOR CLOCK COUNTER
5004 032266 052710 000010 BIS #MSTCK,@R0 ;SET SECTOR CLOCK
5005 032272 042710 000010 BIC #MSTCK,@R0 ;CLEAR SECTOR CLOCK
5006 032276 005301 DEC R1 ;COUNT
5007 032300 001372 BNE 5$ ;BRANCH IF 536 NOT DONE
5008

```

```

5009
5010
5011 ;*ECC PATTERN REGISTER IS NOT CHECKED
5012 032302 017737 162502 015064 MOV @RHEC2,@#EC2 ;RHEC2 IS NOT CHECKED
5013
5014
5015 ;*NOW 'DTE' SHOULD BE SET, CHANGE SAVED REGISTERS TO EXPECTED VALUES
5016
5017 032310 005737 017334 TST @#RH70 ;CHECK FOR RH70 CONTROLLER
5018 032314 001412 BEQ JP1 ;SKIP RH70 CODE AND DO RH11 IF NOT
5019
5020 ;RHC AND RHBA WILL BE MODIFIED FOR RH70C
5021 032316 012737 177416 015024 VAR1: MOV #-242,@#WC ;SAVED RHC
5022 032324 012737 015254 015026 VAR2: MOV #WRFROM+<14.*2>,@#BA ;SAVED RHBA
5023 032332 052737 000300 015030 BIS #IR!OR,@#CS2 ;SAVED RHCS2
5024 032340 000414 BR JP2 ;SKIP NEXT RH11 CODE
5025
5026 032342 012737 177511 015024 JP1: MOV #-183,@#WC ;SAVED RHC
5027 032350 012737 015442 015026 MOV #WRFROM+<73.*2>,@#BA ;SAVED RHBA
5028 032356 042737 000100 015030 BIC #IR,@#CS2 ;SAVED RHCS2
5029 032364 052737 000200 015030 BIS #OR,@#CS2 ;SAVED RHCS2
5030
5031 032372 052737 140000 015032 JP2: BIS #SC!TRE,@#CS1 ;SAVED RHCS1
5032 032400 052737 010000 015034 BIS #DTE,@#ER1 ;SAVED RHER1
5033 032406 012737 000201 015052 MOV #DENVL!DMD,@#MR ;SAVED RHMR
5034 032414 052737 140000 015054 BIS #ATA!ERR,@#DS1 ;SAVED RHDS1
5035 032422 012737 000100 015066 MOV #100,@#LA ;SAVED RHLA
5036 032430 012737 000001 015036 MOV #1,@#DST ;SAVED RHDST
5037 032436 013737 015134 015050 MOV @#ATTENT,@#AS ;SAVED RHAS
5038
5039 ;*NOW READ & SAVE REGISTERS AGAIN SO THAT COMPARISONS
5040 ;CAN BE DONE (USING 'WRFROM' BUFFER THIS TIME)
5041
5042 032444 004037 046616 JSR RO,@#SAVER ;READ IN SEQUENCE
5043 032450 014750 RHC ;FROM HARDWARE REGISTER
5044 032452 016264 REINTO ;INTO CORE BUFFER LOCATION
5045 032454 000023 19. ;NUMBER OF REGISTERS TO READ
5046
5047 ;*FOR RHAS UPPER BYTE
5048 032456 113737 015051 016311 MOVB @#AS+1,@#REINTO+25 ;UPPER RHAS
5049
5050
5051 ;*COMPARE CHANGED REGISTER SNAPSHOT BEFORE COMMAND WITH
5052 ;*SNAPSHOT AFTER COMMAND
5053
5054 032464 004037 047020 JSR RO,@#COMPAR ;COMPARE
5055 032470 015024 WC ;CHANGED INITIAL SNAPSHOT BUFFER
5056 032472 016264 REINTO ;SNAPSHOT BUFFER AFTER COMMAND
5057 032474 000022 18. ;NUMBER OF REGISTERS TO COMPARE
5058 032476 032504 2$ ;RETURN FOR ERROR
5059 032500 032504 2$ ;SAME
5060 032502 032524 3$ ;RETURN FOR GOOD COMPARISON
5061
5062 032504 013705 053130 2$: MOV @#ERWORD,R5 ;GETTING READY TO INDEX
5063 032510 060505 ADD R5,R5 ;DOUBLE ERROR WORD
5064 032512 016537 014746 045662 MOV RHC-2(R5),@#REGADR ;FAILING REGISTER

```

CZRJHDO,RP04/5/6 DSKLS CTRLR2 MACY11 30A(1052) 24-MAY-79 15:07 M 9
CZRJHD.P12 01-MAR-79 09:10 T50 DRIVE TIMING ERROR PAGE 118

SEQ 0116

5065 032520 104001

ERROR 1

;IMPROPER REGISTER
;CHANGE AFTER FORCING
;DATE ERROR
;RETURN

5066

5067

5068 032522 000207

RTS PC

5069

5070 032524

38:

;GOOD, REGISTERS OK - GO ON TO NEXT TEST

5071

5072

```

5073
5074
5075
5076
5077
5078
5079
5080
5081
5082
5083
5084
5085 032524 000004
5086 032526 012706 001000
5087 032532 012737 000051 017330
5088
5089
5090
5091 032540 012737 01000C 056146
5092
5093 032546 005037 056150
5094 032552 005037 056152
5095 032556 005037 056154
5096 032562 012737 000400 056206
5097 032570 004537 047332
5098 032574 056146
5099 032576 056156
5100
5101
5102
5103 032600 004737 046116
5104 032604 012777 177374 162136
5105 032612 012777 015220 162132
5106 032620 005077 162136
5107
5108 032624 012777 014000 162134
5109
5110 032632 005077 162132
5111 032636 004737 046160
5112 032642 104401 005123
5113
5114 032646 000000
5115 032650 013711 015172
5116
5117
5118
5119 032654 004037 046616
5120 032660 014750
5121 032662 015024
5122 032664 000023

*****
; *TEST 51 DRIVE TIMING ERROR
;
; * A WRITE HEADER AND DATA COMMAND IS GIVEN
; * TO CYLINDER 0, SECTOR 0, TRACK 0, 256. WORDS.
;
; * AFTER SECTOR IS FOUND (THE SECTOR FOUND FLOP IS HIGH),
; * NO MORE DIAGNOSTIC DATA CLOCKS ARE GIVEN,
; * ONLY SECTOR CLOCKS ARE GIVEN TILL SECTOR PULSE IS HIGH.
; * THIS SHOULD SET 'DRIVE TIMING ERROR' - 'DTE'
; *****
TST51: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #TINO,@#TSTNM ;THIS SAVES TEST NUMBER

; *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 & CHECKS

JSR PC,@#CLDISK ;SETUP GENERAL REGISTERS
MOV #-260,@#RHWC ;256 DATA WORDS & 4 HEADER WORDS
MOV #WRFROM,@#RHBA ;STARTING ADDRESS OF BUFFER
CLR @#RHDST ;TRACK = 0
;SECTOR = 0
MOV #FMT22!EC1,@#RHOF ;16 BITS PER WORD
;ECC CORRECTION INHIBITED
CLR @#RHCA ;CYLINDER = 0
JSR PC,@#CHECKT ;CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DON'T
TYPE ,CPHALT ;CANNOT CONTINUE TESTING IF ANY OF THE
;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
HALT ;STOP THE TEST
MOV @#WRIFOR,@#R1 ;WRITE HEADER AND DATA = 62

; *READ & SAVE REGISTERS FOR COMPARISON AFTER SIMULATED 'DTE'

JSR R0,@#SAVER ;READ IN SEQUENCE
RHWC ;FROM HARDWARE REGISTER
WC ;INTO CORE BUFFER LOCATION
19. ;NUMBER OF REGISTERS TO READ

```

```

5123
5124 ;*NOW 'GO' WILL BE GIVEN. EVERYTHING WILL BE TREATED
5125 ;*NORMALLY TILL HEADER IS TO BE GIVEN, THEN ONLY
5126 ;*SECTOR CLOCKS WILL BE GIVEN. NO DIAGNOSTIC DATA
5127 ;*CLOCKS WILL BE GIVEN. THIS SHOULD BRING SECTOR PULSE HIGH
5128 ;*WITHOUT PUTTING 'READ' DOWN, HENCE 'DTE' WILL COME UP.
5129
5130 032666 012737 177777 015144 MOV # -1,@#TESDTE ;SET DTE TEST
5131 032674 012737 177777 015124 MOV # -1,@#ERFLG$ ;THIS WILL BRING THE READ HEADER
5132 ;AND DATA PROCESS OUT AFTER THE
5133 ;HEADER HAS BEEN CORRECTLY READ
5134
5135 032702 004737 055772 JSR PC,@#COMWHD ;ISSUE 'GO', SEARCH FOR SECTOR,
5136 ;WRITE HEADER AND DATA.
5137
5138 032706 017737 162044 001176 SETCK3: MOV @RHCS1,@#$TMP0 ;READ CS1 TO CHECK FOR ERRORS DURING WRITE
5139 032714 032737 100000 001176 BIT #SC,@#$TMP0 ;TEST FOR 'SPECIAL CONDITION' - 'SC'
5140 032722 001405 BEQ 4$ ;CONTINUE TEST IF NO ERROR ('SC' = 0)
5141 032724 004737 045616 JSR PC,@#PUTREG ;READ & SAVE REGISTERS AGAIN IF ERROR
5142 032730 104040 ERROR 40 ;THERE WAS A READ/WRITE HEADER ERROR
5143 ;DURING 'DTE' TEST SETUP
5144 032732 000137 033164 JMP TST52 ; ABORT THE TEST AT THAT POINT
5145
5146 032736 4$: ;NOW SECTOR HAS BEEN FOUND OK
5147
5148 ;*609 SECTOR CLOCKS WILL BE GIVEN,
5149 ;*39 BYTES FOR SECTOR GAP,
5150 ;*1 BYTE FOR HEADER SYNC,
5151 ;*8 BYTES FOR HEADER,
5152 ;*GAP 11 BYTES, SYNC 1 BYTE, DATA 512, ECC 4 BYTES
5153 ;*GAP 2 BYTES, TOLERANCE 28 BYTES, EXTRA 3
5154
5155
5156 ;*THIS GIVES 609 SECTOR CLOCKS WITHOUT DIAGNOSTIC DATA CLOCKS
5157
5158 032736 012701 001141 MOV #609.,R1 ;LOAD SECTOR CLOCK COUNTER
5159 032742 052710 C00010 5$: BIS #MSTCK,@R0 ;SET SECTOR CLOCK
5160 032746 042710 000010 BIC #MSTCK,@R0 ;CLEAR SECTOR CLOCK
5161 032752 005301 DEC R1 ;COUNT
5162 032754 001372 BNE 5$ ;BRANCH IF 536 NOT DONE
5163
5164 ;*NOW 'DTE' SHOULD BE SET, CHANGE SAVED REGISTERS
5165 ;TO EXPECTED VALUES
5166
5167 032756 005737 017334 TST @#RH70 ;CHECK FOR RH70 CONTROLLER
5168 032762 001407 BEQ JP3 ;SKIP RH70 CODE AND DO RH11 IF NOT
5169
5170 ;RHC, RHBA AND 1R BIT WILL BE MODIFIED FOR RH70C
5171 032764 012737 177404 015024 VAR3: MOV # -252.,@#WC ;SAVED RHC
5172 032772 012737 015240 015026 VAR4: MOV #WRFROM+<8.*2>,@#BA ;SAVED RHBA
5173 033000 000406 BR JP4 ;SKIP NEXT RH11 CODE
5174
5175 033002 012737 177477 015024 JP3: MOV # -193.,@#WC ;SAVED RHC
5176 033010 012737 015426 015026 MOV #WRFROM+<67.*2>,@#BA ;SAVED RHBA
5177
5178 033016 052737 140000 015032 JP4: BIS #SC!TRE,@#CS1 ;SAVED RHCS1

```

```

5179 033024 042737 000100 015030 VAR5: BIC #1R,@#CS2 ;SAVED RHCS2
5180 033032 052737 000200 015030 BIS #0R,@#CS2 ;SAVED RHCS2
5181 033040 052737 010000 015034 BIS #DTE,@#ER1 ;SAVED RHER1
5182 033046 012737 000401 015052 MOV #401,@#MR ;SAVED RHMR
5183 033054 052737 140000 015054 BIS #ATA!ERR,@#DS1 ;SAVED RHDS1
5184 033062 012737 000100 015066 MOV #100,@#LA ;SAVED RHLA
5185 033070 012737 000001 015036 MOV #1,@#DST ;SAVED RHDST
5186 033076 013737 015134 015050 MOV @#ATTENT,@#AS ;SAVED RHAS
5187
5188 ;*NOW READ & SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN BE DONE
5189
5190 033104 004037 046616 JSR R0,@#SAVER ;READ IN SEQUENCE
5191 033110 014750 RHWC ;FROM HARDWARE REGISTER
5192 033112 016264 REINTO ;INTO CORE BUFFER LOCATION
5193 033114 000023 19. ;NUMBER OF REGISTERS TO READ
5194
5195 ;*FOR RHAS UPPER BYTE
5196 033116 113737 015051 016311 MOVB @#AS+1,@#REINTO+25 ;UPPER RHAS
5197
5198 ;*COMPARE CHANGED REGISTER SNAPSHOT BEFORE COMMAND
5199 ;*WITH REGISTER SNAPSHOT AFTER COMMAND
5200
5201 033124 004037 047020 JSR R0,@#COMPAR ;COMPARE
5202 033130 015024 WC ;CHANGED REGISTER SNAPSHOT
5203 033132 016264 REINTO ;SNAPSHOT AFTER COMMAND
5204 033134 000022 18. ;NUMBER OF REGISTERS TO COMPARE
5205 033136 033144 2$ ;RETURN FOR ERROR
5206 033140 033144 2$ ;SAME
5207 033142 033164 3$ ;RETURN FOR GOOD COMPARISON
5208
5209 033144 013705 053130 2$: MOV @#ERWORD,R5 ;GETTING READY TO INDEX
5210 033150 060505 ADD R5,R5 ;DOUBLE ERROR WORD
5211 033152 016537 014746 045662 MOV RHWC-2(R5),@#REGADR ;FAILING REGISTER
5212 033160 104001 ERROR 1 ;IMPROPER REGISTER
5213 ;CHANGE AFTER FORCING
5214 ;'DTE' ERROR
5215 033162 000207 RTS PC ;RETURN
5216
5217 033164 3$: ;GOOD, REGISTERS COMPARE OK
5218 ;GO ON TO THE NEXT TEST
5219

```

```

5220 ;*****
5221 ;*TEST 52          SECTOR SELECTION
5222
5223 ;*      THE SECTOR SELECTION LOGIC IS CHECKED HERE
5224 ;*      EACH SECTOR ON TRACK ZERO IS WRITTEN INTO.
5225 ;*
5226 ;*      DATA IS - 19 WORDS OF ZEROS - SYNC WORDS, 4 HEADER WORDS
5227 ;*      1 CRC WORD, 5 WORDS OF ZEROS, 1 SYNC WORD, 100 ZEROS
5228 ;*      (DATA), 1 SYNC WORD, 70 SECTOR NUMBER TO VARY
5229 ;*
5230 ;*      THE WRITTEN DATA IS CHECKED IN MEMORY
5231
5232 ;*****
5233 TST52: SCOPE
5234      MOV      #STACK,SP      ;RESET STACK
5235      MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
5236      JSR      PC,@#CLDISK    ;SETUP GENERAL REGISTERS & CLEAR
5237                               ;THE DRIVE
5238      MOV      #22.,@#TAGDTE  ;22 SECTORS
5239                               ;THIS TEST REPEATS
5240                               ;ITSELF 22 TIMES
5241
5242 ;*THE FOLLOWING INITIALIZES FOR SECTOR 0
5243
5244      CLR      @#SS3+2        ;HEADER (SECTOR)
5245      MOV      #21.,@#SS4+2   ;HEADER (KEY1)
5246      MOV      #21.,@#SS5+2   ;HEADER (KEY2)
5247      CLR      @#SS7+2        ;DATA (SECTOR)
5248      CLR      @#SS10+2       ;DATA
5249      CLR      @#SS12+2       ;SECTOR (SIMULATED DISK)
5250      MOV      #21.,@#SS13+2  ;KEY1 (SIMULATED DISK)
5251      MOV      #21.,@#SS14+2  ;KEY2 (SIMULATED DISK)
5252      CLR      @#SS15+2       ;SECTOR (RHDST)
5253
5254 ;*CLEAR SIMULATED DISK AREA
5255      SS1:
5256      1$:      MOV      #SECGAP,R0      ;POINTER
5257              MOV      #304.,R1        ;COUNTER
5258              CLR      (R0)+            ;CLEAR SIMULATED DISK AREA
5259              DEC      R1               ;COUNT
5260              BNE      2$
5261
5262 ;*SETUP WRITE FROM BUFFER
5263      MOV      #WRFROM,R0
5264
5265 ;*HEADER
5266      MOV      #FMT22,(R0)+            ;FORMAT 16 BITS PER WORD
5267
5268      SS3:      MOV      #0,(R0)+       ;CYLINDER 0
5269              MOV      #21.,(R0)+     ;SECTOR TO VARY
5270              SS4:      MOV      #21.,(R0)+ ;KEY1 TO VARY
5271              SS5:      MOV      #21.,(R0)+ ;KEY2 TO VARY
5272
5273 ;*DATA IN WRITE FROM BUFFER ALTHOUGH THIS IS DATA AND NOT
5274 ;*HEADER, THE SECTOR WITH SYNC BYTES WILL BE GIVEN AS DATA.
5275

```

```
5276 ;*DATA IS - 19 WORDS OF ZEROS - SYNC WORDS, 4 HEADER WORDS
5277 ;*1 CRC WORD, 5 WORDS OF ZEROS, 1 SYNC WORD, 100 ZEROS
5278 ;*(DATA), 1 SYNC WORD, 70 SECTOR NUMBER TO VARY
5279
5280 033330 012705 000023      6$: MOV #19.,R5 ;COUNTER
5281 033334 005020          CLR (R0)+ ;19 ZEROS
5282 033336 005305          DEC R5 ;COUNT
5283 033340 001375          BNE 6$ ;19 DONE?
5284 033342 013720 053112      MOV @#RSYNC,(R0)+ ;SYNC = 14400
5285 033346 012720 010000      MOV #FMT22,(R0)+ ;CYLINDER 0
5286 033352 012720 000000      SS7: MOV #0,(R0)+ ;SECTOR TO VARY
5287 033356 005020          CLR (R0)+
5288 033360 005020          CLR (R0)+
5289 033362 004537 047332      JSR R5,@#CRC ;CALCULATE CRC FOR ABOVE 4 WORDS
5290 033366 015270          WRFROM+50 ;4 WORDS START FROM HERE
5291 033370 015300          WRFROM+60 ;PUT CRC HERE
5292
5293 033372 005720          TST (R0)+ ;INCREMENT R0
5294
5295 033374 012705 000005      8$: MOV #5.,R5
5296 033400 005020          CLR (R0)+ ;5 WORDS OF ZEROS
5297 033402 005305          DEC R5 ;COUNT
5298 033404 001375          BNE 8$ ;BRANCH IF 5 NOT DONE
5299
5300 033406 013720 053112      MOV @#RSYNC,(R0)+ ;SYNC = 14400
5301
5302 033412 012705 000144      9$: MOV #100.,R5
5303 033416 005020          CLR (R0)+ ;100 WORDS OF ZEROS
5304 033420 005305          DEC R5
5305 033422 001375          BNE 9$
5306
5307 033424 013720 053112      MOV @#RSYNC,(R0)+ ;SYNC = 14400
5308 033430 012705 000106      MOV #70.,R5
5309 033434 012720 000000      SS10: MOV #0,(R0)+ ;SECTOR TO VARY
5310 033440 005305          DEC R5
5311 033442 001374          BNE SS10
5312
5313 ;*CLEAR REST OF 256 WORDS THAT IS 54 WORDS OF ZEROS
5314
5315 033444 012705 000066      11$: MOV #54.,R5
5316 033450 005020          CLR (R0)+
5317 033452 005305          DEC R5
5318 033454 001375          BNE 11$
5319
5320 ;*THESE ARE TO BE SET UP FOR DISKLESS USE ONLY
5321
5322 033456 012737 010000 056146 MOV #FMT22,@#WCYL ;FORMAT = 16 BIT WORDS
5323 ;CYLINDER = 0
5324 033464 012737 000000 056150 SS12: MOV #0,@#WSECTR ;SECTOR TO VARY
5325 033472 012737 000025 056152 SS13: MOV #21.,@#WKEY1 ;KEY1 TO VARY
5326 033500 012737 000025 056154 SS14: MOV #21.,@#WKEY2 ;KEY2 TO VARY
5327 033506 012737 000312 056206 MOV #202.,@#FNWORD ;202 DATA WORDS
5328 033514 004537 047332      JSR R5,@#CRC ;CALCULATE CRC
5329 033520 056146          WCYL ;FIRST WORD
5330 033522 056156          GCRC ;PUT HERE
5331
```

```

5332                                     ;*THESE ARE REGULAR SETUPS
5333
5334 033524 012777 177400 161216      MOV    #-256.,@RHWC      ;202 DATA, 4 HEADER
5335 033532 012777 015220 161212      MOV    #WRFROM,@RHBA    ;FILL BUS ADDRESS
5336 033540 012777 000000 161214      MOV    #0,@RHDST      ;SECTOR TO VARY
5337 033546 013777 015172 161202      MOV    @#WRIFOR,@RHCS1 ;GET READY TO DO
5338                                     ;WRITE HEADER AND DATA
5339                                     ;WITH 62 FUNCTION CODE IN RHCS1
5340 033554 012777 010000 161204      MOV    #FMT22,@RHOF    ;16 BITS PER WORD FORMAT
5341 033562 005077 161202              CLR    @RHCA          ;CYLINDER = 0
5342
5343 033566 005037 015124              CLR    @#ERFLG$       ;CLEAR ERROR FLAG
5344
5345 033572 004737 046160              JSR    PC,@#CHECKT    ;CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DON'T
5346 033576 104401 005123              TYPE    ,CPHALT      ;CANNOT CONTINUE TESTING IF ANY OF THE
5347                                     ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
5348 033602 000000                    HALT                    ;STOP THE TEST
5349
5350 033604 004737 055772              JSR    PC,@#COMWHD     ;ISSUE 'GO', COUNT SECTOR CLOCKS,
5351                                     ;WRITE HEADER AND DATA
5352 033610 005737 015124              TST    @#ERFLG$       ;HAVE ANY ERRORS OCCURRED ?
5353 033614 001051                    BNE     TST53          ;EXIT IF YES-----)
5354
5355 033616 004737 046350              JSR    PC,@#CHECKE    ;CHECK THAT DVA,RDY,DPR,DRY = 1
5356 033622 104401 005123              TYPE    ,CPHALT      ;CANNOT CONTINUE IF THEY DON'T
5357 033626 000000                    HALT                    ;STOP THE TEST AND RESTART PROGRAM
5358
5359                                     ;*NOW COMPARE 'DISK' BUFFER WITH 'REINTO' BUFFER
5360 033630 004037 047020              JSR    RO,@#COMPAR    ;CHECK
5361 033634 015230                    WRFROM+8.             ;GOOD BUFFER
5362 033636 054726                    DISK                 ;TEST BUFFER
5363 033640 000400                    256.                 ;NUMBER OF WORDS
5364 033642 033650                    16$                  ;RETURN POINT FOR ERROR HEADER
5365 033644 033654                    17$                  ;RETURN POINT FOR ERROR DATA
5366 033646 033660                    18$                  ;RETURN FOR GOOD COMPARISON
5367 033650 104007                    16$: ERROR 7
5368 033652 000207                    RTS    PC
5369 033654 104010                    17$: ERROR 10
5370 033656 000207                    RTS    PC
5371
5372                                     ;*THE FOLLOWING INCREMENTS ARE TO CHANGE THE ABOVE SET UP
5373                                     ;*TO WRITE ON THE NEXT SECTOR
5374
5375 033660 005237 033316              18$: INC    @#SS3+2      ;HEADER (SECTOR)
5376 033664 005337 033322              DEC    @#SS4+2      ;HEADER (KEY1)
5377 033670 005337 033326              DEC    @#SS5+2      ;HEADER (KEY2)
5378 033674 005237 033354              INC    @#SS7+2      ;DATA (SECTOR)
5379 033700 005237 033436              INC    @#SS10+2     ;DATA
5380 033704 005237 033466              INC    @#SS12+2     ;SECTOR (SIMULATED DISK)
5381 033710 005337 033474              DEC    @#SS13+2     ;KEY1 (SIMULATED DISK)
5382 033714 005337 033502              DEC    @#SS14+2     ;KEY2 (SIMULATED DISK)
5383 033720 005237 033542              INC    @#SS15+2     ;SECTOR (RHDST)
5384
5385 033724 005337 015146              SS2: DEC    @#TAGDTE   ;COUNT DOWN FOR 22 SECTORS
5386 033730 001001                    BNE     1$           ;BRANCH IF 22 SECTORS NOT DONE
5387 033732 000402                    BR      TST53        ;ALL DONE - GO TO NEXT TEST

```

CZRJHDO,RP04/5/6 DSKLS CTRLR2 MACY11 30A(1052) 24-MAY-79 15:07 G 10
CZRJHD.P12 01-MAR-79 09:10 T52 SECTOR SELECTION PAGE 125

SEQ 0123

5388 033734 000137 033266 18: JMP @#SS1 ;GO BACK TO NEXT SECTOR
5389
5390
5391

.SBITL DATA TRANSFER TESTS USING ECC

```

;*****
;*TEST 53      WRITE ECC TEST 1

;*      THIS IS A WRITE ECC TEST
;*      WRITE CYLINDER0, FORMAT 16 BITS PER WORD
;*      TRACK 0, SECTOR 1, KEYS 0, NUMBER OF WORDS 256
;*      OF ALL ZEROS.

;*****
TST53:  SCOPE
        MOV      #STACK,SP      ;RESET STACK

        MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
        MOV      #SECGAP,R0     ;POINTER
        MOV      #258.,R1       ;COUNTER
1$:     MOV      #-1,(R0)+       ;FILL SIMULATOR DISK WITH ONES
        DEC      R1
        BNE      1$
        JSR      PC,@#CLDISK    ;THIS IS USED TO SET GENERAL REGISTERS

;*THESE ARE FOR ECC TEST ONLY

        MOV      #-1,@#TSECC    ;THIS IS AN ECC TEST
        CLR      @#POSITI      ;CLEAR ERROR POSITION COUNTER
        MOV      @#NCODE,@#NCOUNT ;TEMPORARY N-CODE COUNTER
        MOV      @#HARDER,@#HADTMP ;TEMPORARY HARD ERROR COUNTER
        CLR      @#GECC1       ;ECC LOW ORDER TO BE GENERATED
        CLR      @#GECC2       ;ECC HIGH ORDER TO BE GENERATED
        CLR      @#DATENV      ;CLEAR DATA ENVELOPE CLOCK COUNT
        CLR      @#ZCODE       ;CLEAR LEADING ZEROS CLOCK COUNT

;*THESE ARE TO BE SETUP FOR DISKLESS USE ONLY

        MOV      #FMT22,@#WCYL  ;FORMAT22=16BIT WORDS AND
                                ;CYLINDER 0
        MOV      #1,@#WSECTR    ;TRACK=0, 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

        MOV      #-260.,@#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

```

```

5448 034126 012720 010000      MOV      #FMT22,(R0)+      ;FORMAT=16 BIT WORDS
5449                                ;CYLINDER=0
5450 034132 012720 000001      2$:  MOV      #1,(R0)+      ;TRACK=0, SECTOR=1, KEYS=0
5451 034136 005020              CLR      (R0)+      ;KEY1=0
5452 034140 005020              CLR      (R0)+      ;KEY2=0
5453 034142 012705 000400      MOV      #256,R5      ;COUNTER
5454 034146 012720 000000      3$:  MOV      #0,(R0)+      ;MOVE ALL ZEROS FOR DATA
5455 034152 005305              DEC      R5
5456 034154 001374              BNE      3$      ;BRANCH IF DATA NOT COMPLETE
5457 034156 012777 000001 160576 MOV      #1,@RHDST      ;TRACK=0 SECTOR=1
5458
5459 034164 004737 046160      JSR      PC,@#CHECKT      ;CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DON'T
5460 034170 104401 005123      TYPE      ,CPHALT      ;CANNOT CONTINUE TESTING IF ANY OF THE
5461                                ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
5462 034174 000000              HALT      ;STOP THE TEST
5463
5464 034176 013711 015172      MOV      @#WRIFOR,@R1      ;GET READY FOR WRITE HEADER AND
5465                                ;DATA WITH 62 IN RHCS1
5466 034202 005037 015124      CLR      @#ERFLG$      ;CLEAR ERROR FLAG
5467 034206 012777 010000 160552 MOV      #FMT22,@RHOF      ;FORMAT BIT=1 (16 BIT WORDS)
5468 034214 005077 160550      CLR      @RHCA      ;CYLINDER =0
5469 034220 004737 055772      JSR      PC,@#COMWHD      ;WRITE HEADER AND DATA
5470
5471                                ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
5472                                ;*FROM THE "COMWHD" ROUTINE THAT MEANS ALL HEADER ON DISK
5473                                ;*IS GOOD IE. ONLY DATA IS TO BE CHECKED TO SEE IF THEY ARE
5474                                ;*ALL ZEROS AND WRITE DATA GAP AND TOLERANCE GAP TO SEE IF
5475                                ;*THEY ARE ALL ZEROS
5476
5477 034224 005737 015124      TST      @#ERFLG$      ;HAS ANY ERRORS OCCURED?
5478                                ;IF WRITE ERROR OCCURS ECC IS NOT CHECKED
5479 034230 001056              BNE      TST54      ;:BRANCH IF YES
5480
5481                                ;*COMPARE SOFTWARE GENERATED ECC WITH THAT GENERATED BY HARDWARE
5482
5483
5484 034232 023737 050620 055726 CMP      @#GECC1,@#WECC1;COMPARE SOFTWARE ECC WITH HARDWARE ECC
5485 034240 001402              BEQ      6$      ;BRANCH IF GOOD
5486 034242 104031              ERROR      31      ;LOW ORDER ECC IN ERROR
5487 034244 000405              BR      7$      ;BRANCH TO CONTINUE
5488 034246 023737 050622 055730 6$: CMP      @#GECC2,@#WECC2;COMPARE SOFTWARE ECC WITH HARDWARE ECC
5489 034254 001401              BEQ      7$      ;BRANCH IF GOOD
5490 034256 104031              ERROR      31      ;HIGH ORDER ECC IN ERROR
5491
5492
5493                                7$:
5494 034260 004737 046350      JSR      PC,@#CHECKE      ;CHECK THAT DVA,RDY,DPR,DRY = 1
5495 034264 104401 005123      TYPE      ,CPHALT      ;CANNOT CONTINUE IF THEY DON'T
5496 034270 000000              HALT      ;STOP THE TEST AND RESTART PROGRAM
5497
5498
5499                                ;*FILL "REINTO" BUFFER WITH EXPECTED DATA
5500
5501
5502 034272 004037 046034      JSR      R0,@#CLAREA      ;FILL REINTO BUFFER
5503 034276 016264              REINTO      ;FROM

```

```

5504 034300 017262          REINTO+<255.*2> ;TO
5505 034302 000000          .WORD 0 ;DATA
5506
5507 034304 013737 050620 017264 MOV @#GECC1,@#REINTO+<256.*2>;FILL ECC1
5508 034312 013737 050622 017266 MOV @#GECC2,@#REINTO+<257.*2>;FILL ECC2
5509 034320 004037 046034 JSR R0,@#CLAREA ;FILL REST
5510 034324 017270 REINTO+<258.*2> ;FROM
5511 034326 017324 REINTO+<272.*2> ;TO
5512 034330 000000 0 ;DATA
5513
5514
5515 034332 005037 015124 CLR @#ERFLG$ ;CLEAR ERROR FLAG
5516
5517
5518 ;*NOW COMPARE 'DISK' BUFFER WITH 'REINTO'
5519
5520 034336 004037 047020 JSR R0,@#COMPAR ;CHECK
5521 034342 016264 REINTO ;GOOD BUFFER
5522 034344 054726 DISK ;TEST BUFFER
5523 034346 000402 258. ;NUMBER OF WORDS CHECKED
5524 034350 034356 4$ ;RETURN POINT FOR ERROR HEADER
5525 034352 034362 5$ ;RETURN POINT FOR ERROR DATA
5526
5527 034354 034366 TST54 ;RETURN FOR GOOD COMPARISON
5528
5529 034356 104007 4$: ERROR 7 ;READ ERROR 10 NEXT
5530 034360 000207 RTS PC ;RETURN TO COMPARE
5531 034362 104010 5$: ERROR 10 ;WORD NOS 1 TO 256 ARE
5532 ;DATA WORDS
5533 ;WORD NOS 257 AND 258
5534 ;ARE ECC WHICH ARE CHECKED
5535 ;WORD NOS 259
5536 ;IS DATA GAP
5537 ;WORD NOS 260 TO 273
5538 ;ARE TOLERANCE GAP
5539 034364 000207 RTS PC ;RETURN TO COMPARE
5540
5541
5542
5543
5544

```

```

5545
5546
5547
5548
5549
5550
5551
5552
5553
5554
5555
5556
5557 034366 000004
5558 034370 012706 001000
5559
5560 034374 012737 000054 017330
5561
5562
5563
5564
5565
5566 034402 012746 000000
5567 034406 012705 000400
5568 034412 012700 054726
5569 034416 011620
5570 034420 005305
5571 034422 001375
5572 034424 005726
5573 034426 022020
5574 034430 012705 000017
5575
5576 034434 005020
5577 034436 005305
5578 034440 001375
5579
5580
5581 034442 004737 051414
5582
5583
5584
5585
5586 034446 012737 177777 015142
5587 034454 005037 050632
5588 034460 013737 050626 050630
5589 034466 013737 050634 050642
5590 034474 005037 050620
5591 034500 005037 050622
5592 034504 005037 050636
5593 034510 005037 050640
5594
5595
5596
5597
5598 034514 012737 010000 053010
5599
5600 034522 112737 000000 053013

;*****
;*TEST 54 READ ECC ENABLED 1A
;
;* THIS IS AN ECC READ DATA TEST
;* ERROR CORRECTION IS ENABLED
;* NO ERROR IS INSERTED
;* GOOD DATA USED IS 256 WORDS OF 0
;* COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
;* TRACK 0, SECTOR 0 KEYS 0 READ HEADER AND DATA
;*****
TST54: 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 #0,-(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)
CMP (R0)+,(R0)+ ;JUMP OVER THE TWO ECC WORDS
MOV #15.,R5 ;1 DATA GAP
;14 TOLERANCE GAP
2$: CLR (R0)+ ;CLEAR DATA GAP, AND
DEC R5 ;TOLERANCE GAP
BNE 2$ ;BRANCH IF NOT COMPLETE
JSR PC,@#FILLEC ;INSERT THE TWO ECC WORDS ON THE DISK
;IN THE CORRECT PLACE
;
;*THESE ARE FOR ECC TEST ONLY
MOV #-1,@#TSECC ;THIS IS AN ECC TEST
CLR @#POSITI ;CLEAR ERROR POSITION COUNTER
MOV @#NCODE,@#NCOUNT ;TEMPORARY N-CODE COUNTER
MOV @#HARDER,@#HADTMP ;TEMPORARY HARD ERROR COUNTER
CLR @#GECC1 ;ECC LOW ORDER TO BE GENERATED
CLR @#GECC2 ;ECC HIGH ORDER TO BE GENERATED
CLR @#DATENV ;CLEAR DATA ENVELOPE CLOCK COUNT
CLR @#ZCODE ;CLEAR LEADING ZEROS CLOCK COUNT
;
;*THESE ARE TO SETUP FOR DISKLESS USE ONLY
MOV #FMT22,@#CYL ;16 BITS PER WORD
;CYLINDER 0, FORMAT 16 BITS
MOVB #0,@#SECOTR+1 ;TRACK 0

```

5601	034530	112737	000000	053012	MOVB	#0,@#SECOTR	;SECTOR 0
5602	034536	012737	000000	053014	MOV	#0,@#KEY1	;KEY1=0
5603	034544	012737	000000	053016	MOV	#0,@#KEY2	;KEY2=0
5604	034552	012737	000400	053070	MOV	#256.,@#DAWORD	;NO. OF DATA WORDS
5605	034560	005037	053020		CLR	@#X	;THIS IS A READ COMMAND
5606	034564	004537	047332		JSR	R5,@#CRC	;GO TO CALCULATE CRC
5607	034570	053010			CYL		
5608	034572	054710			WCRC		
5609							
5610							
5611							
5612							
5613							
5614							
5615	034574	004737	046116		JSR	PC,@#CLDISK	;SETUP GENERAL REGISTERS
5616	034600	012777	177374	160142	MOV	#-256.-4.,@#HWC	;256. DATA 4 HEADER WORDS
5617	034606	012777	016264	160136	MOV	#REINTO,@#HBA	;STARTING ADDRESS OF READ BUFFER
5618	034614	112746	000000		MOVB	#0,-(SP)	;IN LOWER BYTE GET SECTOR
5619	034620	112766	000000	000001	MOVB	#0,1(SP)	;GET TRACK IN HIGHER BYTE
5620	034626	012677	160130		MOV	(SP)+,@#RHDST	;TRACK/SECTOR IN RHDST
5621	034632	012777	010000	160126	MOV	#FMT22,@#HOF	;16 BITS PER WORD
5622							;ECC CORRECTION NOT INHIBIT
5623							;BECAUSE ECC IS NOT GOING
5624							;TO BE CHECKED
5625	034640	005077	160124		CLR	@#HCA	;CYLINDER 0
5626							
5627	034644	004737	046160		JSR	PC,@#CHECKT	;CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DON'T
5628	034650	104401	005123		TYPE	,CPHALT	;CANNOT CONTINUE TESTING IF ANY OF THE
5629							;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
5630	034654	000000			HALT		;STOP THE TEST
5631							
5632	034656	013711	015176		MOV	@#REFOR,@R1	;READ HEADER AND DATA=72
5633	034662	005037	015124		CLR	@#ERFLG\$	;CLEAR ERROR FLAG
5634	034666	004737	052650		JSR	PC,@#COMHD	;READ HEADER AND DATA
5635							;IF THERE ARE READ ERRORS THEN
5636							;ECC WILL NOT BE CHECKED
5637							
5638							
5639							
5640							
5641							
5642							
5643							
5644							
5645							
5646							
5647							
5648							
5649	034672	005737	015124		TST	@#ERFLG\$	;ANY ERRORS ALREADY THERE
5650	034676	001102			BNE	TST55	;BRANCH IF YES
5651	034700	004737	045616		JSR	PC,@#PUTREG	;SAVE REGISTERS
5652	034704	005737	015034		TST	@#ER1	;NO ERRORS SHOULD BE SET
5653	034710	001401			BEQ	6\$	;BRANCH IF NO ERRORS SET
5654	034712	104032			ERROR	32	;32 BIT ECC REGISTER SHOULD BE ZERO
5655							;ONLY 11 OF THE 32 BITS CAN BE SEEN
5656							;IN THE PATERN REGISTER

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

CZRJHDO,RP04/5/6 DSKLS CTRLR2 MACY11 30A(1052) 24-MAY-79 15:07 N 10
CZRJHD.P12 01-MAR-79 09:10 T54 READ ECC ENABLED 1A PAGE 132

SEQ 0130

5713 035102 000207

RTS PC

;RETURN TO "COMPAR"

5714
5715
5716
5717

[illegible]

```

5774 035240 112737 000000 053013      MOV      #0,@#SECOTR+1      ;TRACK 0
5775 035246 112737 000000 053012      MOV      #0,@#SECOTR      ;SECTOR 0
5776 035254 012737 000000 053014      MOV      #0,@#KEY1      ;KEY1=0
5777 035262 012737 000000 053016      MOV      #0,@#KEY2      ;KEY2=0
5778 035270 012737 000400 053070      MOV      #256.,@#DAWORD      ;NO. OF DATA WORDS
5779 035276 005037 053020              CLR      @#X      ;THIS IS A READ COMMAND
5780 035302 004537 047332              JSR      R5,@#CRC      ;GO TO CALCULATE CRC
5781 035306 053010
5782 035310 054710
5783
5784
5785      ;*THIS IS TO INSERT ERROR
5786      ;*THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
5787      ;*THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
5788      ;*THIS MOVE
5789
5790 035312 012737 100000 054730      MOV      #100000,@#DISK+2      ;FORCE ERROR ON BIT NUMBER 32
5791                                     ;SO ERROR POSITION REGISTER WILL SHOW
5792                                     ;22
5793 035320 012737 000026 035474      MOV      #22.,@#8$      ;INSERT POSITION REG.
5794
5795
5796      ;*THESE ARE REGULAR SETUPS
5797
5798 035326 004737 046116              JSR      PC,@#CLDISK      ;SETUP GENERAL REGISTERS
5799 035332 012777 177374 157410      MOV      #-256.-4.,@#RHWC      ;256. DATA 4 HEADER WORDS
5800 035340 012777 016264 157404      MOV      #REINTO,@#RHBA      ;STARTING ADDRESS OF READ BUFFER
5801 035346 112746 000000              MOV      #0,-(SP)      ;IN LOWER BYTE GET SECTOR
5802 035352 112766 000000 000001      MOV      #0,1(SP)      ;GET TRACK IN HIGHER BYTE
5803 035360 012677 157376              MOV      (SP)+,@#RHDST      ;TRACK/SECTOR IN RHDST
5804 035364 012777 010000 157374      MOV      #FMT22,@#RHOF      ;16 BITS PER WORD
5805                                     ;ECC CORRECTION NOT INHIBIT
5806                                     ;BECAUSE ECC IS NOT GOING
5807                                     ;TO BE CHECKED
5808 035372 005077 157372              CLR      @#RHCA      ;CYLINDER 0
5809
5810 035376 004737 046160              JSR      PC,@#CHECKT      ;CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DON'T
5811 035402 104401 005123              TYPE      ,CPHALT      ;CANNOT CONTINUE TESTING IF ANY OF THE
5812                                     ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
5813 035406 000000              HALT      ;STOP THE TEST
5814
5815 035410 013711 015176              MOV      @#REFOR,@#R1      ;READ HEADER AND DATA=72
5816 035414 005037 015124              CLR      @#ERFLG$      ;CLEAR ERROR FLAG
5817 035420 004737 052650              JSR      PC,@#COMHD      ;READ HEADER AND DATA
5818                                     ;IF THERE ARE READ ERRORS THEN
5819                                     ;ECC WILL NOT BE CHECKED
5820
5821
5822      ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
5823      ;*FROM THE "COMHD" ROUTINE THAT MEANS SECTOR GAP,
5824      ;*FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
5825      ;*SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
5826      ;*DETECTED
5827      ;*HEADER AND DATA ARE TO BE CHECKED.
5828      ;*IN CHECKING READ DATA THE WRITE FROM BUFFER
5829      ;*"WRFROM" IS FILLED WITH EXPECTED DATA AND

```

```
5830 ;*COMPARISONS ARE MADE
5831
5832 035424 005737 015124 TST @#ERFLG$ ;ANY ERRORS ALREADY THERE
5833 035430 001077 BNE TST56 ;BRANCH IF YES
5834 035432 004737 045616 JSR PC,@#PUTREG ;SAVE REGISTERS
5835 035436 022737 100000 015034 CMP #DCK,@#ER1 ;ONLY DATA CHECK ERROR SHOULD BE SET
5836 035444 001401 BEQ 6$ ;BRANCH IF YES
5837 035446 104032 ERROR 32 ;32 BIT ECC REGISTER SHOULD BE NON
5838 ;ZERO
5839 ;ONLY 11 OF THE 32 BITS CAN BE SEEN
5840 ;IN THE PATERN REGISTER
5841 ;DCK SHOULD BE SET IN RHER1
5842 035450 013746 050620 6$: MOV @#GECC1,-(SP) ;GET PATTERN REGISTER
5843 035454 042716 174000 BIC #174000,(SP) ;KEEP ONLY 11 BITS
5844 035460 022637 015064 CMP (SP)+,@#EC2 ;COMPARE PATTERN REGISTER
5845 035464 001401 BEQ 7$ ;BRANCH IF GOOD
5846 035466 104032 ERROR 32 ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
5847
5848 035470 004037 051242 7$: JSR RO,@#ECORR ;GO TO ECC CORRECTION PROCESS
5849 035474 000026 8$: 22. ;EXPECTED POSITION REG. WHEN CORRECTION
5850 ;IS COMPLETE
5851
5852
5853
5854 035476 004737 046350 JSR PC,@#CHECKE ;CHECK THAT DVA,RDY,DPR,DRY = 1
5855 035502 104401 005123 TYPE ,CPHALT ;CANNOT CONTINUE IF THEY DON'T
5856 035506 000000 HALT ;STOP THE TEST AND RESTART PROGRAM
5857 035510 012700 015220 MOV #WRFROM,RO ;GETTING READY TO FILL EXPECTED DATA
5858 035514 012720 010000 MOV #0!FMT22,(RO)+ ;CYLINDER 0
5859 035520 112746 000000 MOV #0,-(SP) ;IN LOWER BYTE GET SECTOR
5860 035524 112766 000000 000001 MOV #0,1(SP) ;GET TRACK IN HIGHER BYTE
5861 035532 012620 MOV (SP)+,(RO)+ ;GET TRACK/SECTOR IN BUFFER
5862 035534 012720 000000 MOV #0,(RO)+ ;KEY1 IN BUFFER
5863 035540 012720 000000 MOV #0,(RO)+ ;KEY2 IN BUFFER
5864 035544 012701 000400 MOV #256., R1 ;DATA WORD COUNTER
5865 035550 012702 000000 MOV #0,R2 ;DATA
5866 035554 010220 3$: MOV R2,(RO)+ ;DATA INTO BUFFER
5867 035556 005301 DEC R1 ;COUNT
5868 035560 001375 BNE 3$ ;BRANCH IF 256 NOT DONE
5869
5870 ;*ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
5871 ;*NOW THE INSERTED ERROR WILL BE PUT IN
5872 035562 012737 100000 015232 MOV #100000,@#WRFROM+<5*2> ;INSERTED ERROR
5873
5874
5875
5876 035570 005037 015124 CLR @#ERFLG$ ;CLEAR ERROR FLAG
5877 035574 004737 045616 JSR PC,@#PUTREG ;SAVE REGISTERS
5878
5879
5880 ;*NOW READ DATA BUFFER WILL BE CHECKED
5881
5882 035600 004037 047020 JSR RO,@#COMPAR ;CHECK
5883 035604 015220 WRFROM ;GOOD BUFFER
5884 035606 016264 REINTO ;TEST BUFFER
5885 035610 000404 4*256. ;NUMBER OF WORDS CHECKED
```

5886	035612	035620	4\$		:RETURN POINT FOR ERROR HEADER
5887	035614	035624	5\$		:RETURN POINT FOR ERROR DATA
5888					
5889	035616	035630	TST56		:RETURN FOR GOOD COMPARISON
5890					
5891	035620	104004	4\$:	ERROR 4	:READ NEXT ERROR
5892	035622	000207		RTS PC	:RETURN TO "COMPAR"
5893	035624	104005	5\$:	ERROR 5	:WORD NOS 1 TO 4 ARE
5894					:HEADER WORDS
5895					:5 TO 260 ARE DATA WORDS
5896	035626	000207		RTS PC	:RETURN TO "COMPAR"
5897					

```
5898
5899
5900
5901
5902
5903
5904
5905
5906
5907
5908
5909
5910 035630 000004
5911 035632 012706 001000
5912
5913 035636 012737 000056 017330
5914
5915
5916
5917
5918
5919 035644 012746 000000
5920 035650 012705 000400
5921 035654 012700 054726
5922 035660 011620
5923 035662 005305
5924 035664 001375
5925 035666 005726
5926 035670 022020
5927 035672 012705 000017
5928
5929 035676 005020
5930 035700 005305
5931 035702 001375
5932
5933
5934 035704 004737 051414
5935
5936
5937
5938
5939 035710 012737 177777 015142
5940 035716 005037 050632
5941 035722 013737 050626 050630
5942 035730 013737 050634 050642
5943 035736 005037 050620
5944 035742 005037 050622
5945 035746 005037 050636
5946 035752 005037 050640
5947
5948
5949
5950
5951 035756 012737 010000 053010
5952
5953 035764 112737 000000 053013

*****
*TEST 56      READ ECC ENABLED 1C
*****
*      THIS IS AN ECC READ DATA TEST
*      ERROR CORRECTION IS ENABLED
*      A NON CORRECTABLE ERROR IS INSERTED IN BIT POSITION 21 THRU 32
*      GOOD DATA USED IS 256 WORDS OF 0
*      COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
*      TRACK 0, SECTOR 0 KEYS 0  READ HEADER AND DATA
*****
TST56:  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      #0,-(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)
        CMP      (R0)+,(R0)+   ;JUMP OVER THE TWO ECC WORDS
        MOV      #15.,R5      ;1 DATA GAP
                                ;14 TOLERANCE GAP
2$:     CLR      (R0)+         ;CLEAR DATA GAP, AND
        DEC      R5            ;TOLERANCE GAP
        BNE      2$           ;BRANCH IF NOT COMPLETE

        JSR      PC,@#FILLEC   ;INSERT THE TWO ECC WORDS ON THE DISK
                                ;IN THE CORRECT PLACE

;*THESE ARE FOR ECC TEST ONLY

        MOV      #-1,@#TSECC   ;THIS IS AN ECC TEST
        CLR      @#POSITI      ;CLEAR ERROR POSITION COUNTER
        MOV      @#NCODE,@#NCOUNT ;TEMPORARY N-CODE COUNTER
        MOV      @#HARDER,@#HADTMP ;TEMPORARY HARD ERROR COUNTER
        CLR      @#GECC1       ;ECC LOW ORDER TO BE GENERATED
        CLR      @#GECC2       ;ECC HIGH ORDER TO BE GENERATED
        CLR      @#DATENV      ;CLEAR DATA ENVELOPE CLOCK COUNT
        CLR      @#ZCODE       ;CLEAR LEADING ZEROS CLOCK COUNT

;*THESE ARE TO SETUP FOR DISKLESS USE ONLY

        MOV      #FMT22,@#CYL   ;16 BITS PER WORD
                                ;CYLINDER 0, FORMAT 16 BITS
        MOVB     #0,@#SECOTR+1 ;TRACK 0
```

5954	035772	112737	000000	053012	MOVB	#0,@#SECOTR	;SECTOR 0
5955	036000	012737	000000	053014	MOV	#0,@#KEY1	;KEY1=0
5956	036006	012737	000000	053016	MOV	#0,@#KEY2	;KEY2=0
5957	036014	012737	000400	053070	MOV	#256.,@#DAWORD	;NO. OF DATA WORDS
5958	036022	005037	053020		CLR	@#X	;THIS IS A READ COMMAND
5959	036026	004537	047332		JSR	R5,@#CRC	;GO TO CALCULATE CRC
5960	036032	053010			CYL		
5961	036034	054710			WCRC		
5962							
5963							
5964							
5965							
5966							
5967							
5968	036036	012737	177760	054730	MOV	#177760,@#DISK+2	;FORCE ERROR ON BIT NUMBER 21 THRU 32
5969							;SO ERROR POSITION REGISTER WILL SHOW
5970							;22
5971	036044	012737	010040	036220	MOV	#4128.,@#8\$	;INSERT POSITION REG.
5972							
5973							
5974							
5975							
5976	036052	004737	046116		JSR	PC,@#CLDISK	;SETUP GENERAL REGISTERS
5977	036056	012777	177374	156664	MOV	#-256.-4.,@#RHWC	;256. DATA 4 HEADER WORDS
5978	036064	012777	016264	156660	MOV	#REINTO,@#HBA	;STARTING ADDRESS OF READ BUFFER
5979	036072	112746	000000		MOVB	#0,-(SP)	;IN LOWER BYTE GET SECTOR
5980	036076	112766	000000	000001	MOVB	#0,1(SP)	;GET TRACK IN HIGHER BYTE
5981	036104	012677	156652		MOV	(SP)+,@#RHDST	;TRACK/SECTOR IN RHDST
5982	036110	012777	010000	156650	MOV	#FMT22,@#RHOF	;16 BITS PER WORD
5983							;ECC CORRECTION NOT INHIBIT
5984							;BECAUSE ECC IS NOT GOING
5985							;TO BE CHECKED
5986	036116	005077	156646		CLR	@#RHCA	;CYLINDER 0
5987	036122	004737	046160		JSR	PC,@#CHECKT	;CHECK DVA,RDY,DPR,DRY - 1 AND OTHERS DON'T
5988	036126	104401	005123		TYPE	,CPHALT	;CANNOT CONTINUE TESTING IF ANY OF THE
5989							;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
5990	036132	000000			HALT		;STOP THE TEST
5991	036134	013711	015176		MOV	@#REFOR,@#RI	;READ HEADER AND DATA-72
5992	036140	005037	015124		CLR	@#ERFLG\$	;CLEAR ERROR FLAG
5993	036144	004737	052650		JSR	PC,@#COMHD	;READ HEADER AND DATA
5994							;IF THERE ARE READ ERRORS THEN
5995							;ECC WILL NOT BE CHECKED
5996							

```
5997
5998
5999
6000
6001
6002
6003
6004
6005
6006
6007
6008 036150 005737 015124
6009 036154 001106
6010 036156 004737 045616
6011 036162 022737 100000 015034
6012 036170 001401
6013 036172 104032
6014
6015
6016
6017
6018 036174 013746 050620
6019 036200 042716 174000
6020 036204 022637 015064
6021 036210 001401
6022 036212 104032
6023
6024 036214 004037 051242
6025 036220 000000
6026
6027
6028
6029
6030 036222 004737 045616
6031 036226 022737 100100 015034
6032
6033 036234 001401
6034 036236 104036
6035
6036
6037
6038
6039 036240
6040 036240 004737 046350
6041 036244 104401 005123
6042 036250 000000
6043 036252 012700 015220
6044 036256 012720 010000
6045 036262 112746 000000
6046 036266 112766 000000 000001
6047 036274 012620
6048 036276 012720 000000
6049 036302 012720 000000
6050 036306 012701 000400
6051 036312 012702 000000
6052 036316 010220

; *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
; *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

TST @#ERFLG$ ; ANY ERRORS ALREADY THERE
BNE TST57 ; BRANCH IF YES
JSR PC,@#PUTREG ; SAVE REGISTERS
CMP #DCK,@#ER1 ; ONLY DATA CHECK ERROR SHOULD BE SET
BEQ 6$ ; BRANCH IF YES
ERROR 32 ; 32 BIT ECC REGISTER SHOULD BE NON
; ZERO
; ONLY 11 OF THE 32 BITS CAN BE SEEN
; IN THE PATTERN REGISTER
; DCK SHOULD BE SET IN RHER1
6$: MOV @#GECC1,-(SP) ; GET PATTERN REGISTER
BIC #174000,(SP) ; KEEP ONLY 11 BITS
CMP (SP)+,@#EC2 ; COMPARE PATTERN REGISTER
BEQ 7$ ; BRANCH IF GOOD
ERROR 32 ; 11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
7$: JSR R0,@#ECORR ; GO TO ECC CORRECTION PROCESS
8$: .WORD ; EXPECTED POSITION REG. WHEN CORRECTION
; IS COMPLETE

JSR PC,@#PUTREG ; SAVE REGISTERS
CMP #DCK!ECH,@#ER1 ; WITH ERRORS INSERTED IN BIT POSITION 21
; THRU 32 HARD ERROR BIT SHOULD SET
BEQ 9$ ; BRANCH IF GOOD
ERROR 36 ; WITH ERROR INSERTED IN BIT POSITION 21 THRU
; 32 ECH SHOULD SET

9$: JSR PC,@#CHECKE ; CHECK THAT DVA,RDY,DPR,DRY = 1
TYPE ,CPHALT ; CANNOT CONTINUE IF THEY DON'T
HALT ; STOP THE TEST AND RESTART PROGRAM
MOV #WRFROM,R0 ; GETTING READY TO FILL EXPECTED DATA
MOV #0.FMT22,(R0)+ ; CYLINDER 0
MOVB #0,-(SP) ; IN LOWER BYTE GET SECTOR
MOVB #0,1(SP) ; GET TRACK IN HIGHER BYTE
MOV (SP)+,(R0)+ ; GET TRACK/SECTOR IN BUFFER
MCV #0,(R0)+ ; KEY1 IN BUFFER
MOV #0,(R0)+ ; KEY2 IN BUFFER
MOV #256.,R1 ; DATA WORD COUNTER
MOV #0,R2 ; DATA
3$: MOV R2,(R0)+ ; DATA INTO BUFFER
```

CZRJHD0,RP04/5/6 DSKLS CTRLR2 MACY11 30A(1052) 24-MAY-79 15:07 I 11
CZRJHD.P12 01-MAR-79 09:10 156 READ ECC ENABLED 1C PAGE 140

SEQ 0138

6053 036320 005301
6054 036322 001375
6055

DEC R1
BNE 38

;COUNT
;BRANCH IF 256 NOT DONE

```

6056 ;ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
6057 ;*NOW THE INSERTED ERROR WILL BE PUT IN
6058
6059 036324 012737 177760 015232 MOV #177760,@WRFROM+<5*2> ;INSERTED ERROR
6060
6061
6062
6063 036332 005037 015124 CLR @ERFLG$ ;CLEAR ERROR FLAG
6064 036336 004737 045616 JSR PC,@PUTREG ;SAVE REGISTERS
6065
6066
6067 ;*NOW READ DATA BUFFER WILL BE CHECKED
6068
6069 036342 004037 047020 JSR RO,@COMPAR ;CHECK
6070 036346 015220 WRFROM ;GOOD BUFFER
6071 036350 016264 REINTO ;TEST BUFFER
6072 036352 000404 4+256. ;NUMBER OF WORDS CHECKED
6073 036354 036362 4$ ;RETURN POINT FOR ERROR HEADER
6074 036356 036366 5$ ;RETURN POINT FOR ERROR DATA
6075
6076 036360 036372 TS157 ;RETURN FOR GOOD COMPARISON
6077
6078 036362 104004 4$: ERROR 4 ;READ NEXT ERROR
6079 036364 000207 RTS PC ;RETURN TO 'COMPAR'
6080 036366 104005 5$: ERROR 5 ;WORD NOS 1 TO 4 ARE
6081 ;HEADER WORDS
6082 ;5 TO 260 ARE DATA WORDS
6083 036370 000207 RTS PC ;RETURN TO 'COMPAR'
6084
6085
6086
6087
6088
6089

```

```

6090
6091
6092
6093
6094
6095
6096
6097
6098
6099
6100 036372 000004
6101 036374 012706 001000
6102
6103 036400 012737 000057 017330
6104 036406 012700 054630
6105 036412 012701 000460
6106 036416 005020
6107 036420 005301
6108 036422 001375
6109 036424 004737 046116
6110
6111
6112
6113 036430 012737 177777 015142
6114 036436 005037 050632
6115 036442 013737 050626 050630
6116 036450 013737 050634 050642
6117 036456 005037 050620
6118 036462 005037 050622
6119 036466 005037 050636
6120 036472 005037 050640
6121
6122
6123
6124
6125
6126
6127 036476 012737 010000 056146
6128
6129 036504 012737 000001 056150
6130 036512 005037 056152
6131 036516 005037 056154
6132 036522 012737 000400 056206
6133 036530 004537 047332
6134 036534 056146
6135 036536 056156
6136
6137
6138
6139 036540 012777 177374 156202
6140 036546 012700 015220
6141 036552 010077 156174
6142
6143 036556 012720 010000
6144
6145 036562 012720 0C0001

;*****
;*TEST 57 WRITE ECC TEST 2
;
;* THIS IS A WRITE ECC TEST
;* WRITE CYLINDER 0, FORMAT 16 BITS PER WORD
;* TRACK 0, SECTOR 1, KEYS 0, NUMBER OF WORDS 256
;* OF ALL ONES.
;*****
1ST57: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
MOV #SECGAP,R0 ;POINTER
MOV #304.,R1 ;COUNTER
1$: CLR (R0)+ ;CLEAR SIMULATED DISK AREA
DEC R1
BNE 1$
JSR PC,CLDISK ;THIS IS USED TO SET GENERAL REGISTERS
;
; *THESE ARE FOR ECC TEST ONLY
MOV #-1,@#TSECC ;THIS IS AN ECC TEST
CLR @#POSIT1 ;CLEAR ERROR POSITION COUNTER
MOV @#NCODE,@#NCOUNT ;TEMPORARY N-CODE COUNTER
MOV @#HARDER,@#HADTMP ;TEMPORARY HARD ERROR COUNTER
CLR @#GECC1 ;ECC LOW ORDER TO BE GENERATED
CLR @#GECC2 ;ECC HIGH ORDER TO BE GENERATED
CLR @#DATENV ;CLEAR DATA ENVELOPE CLOCK COUNT
CLR @#ZCODE ;CLEAR LEADING ZEROS CLOCK COUN
;
; *THESE ARE TO BE SETUP FOR DISKLESS USE ONLY
MOV #FMT22,@#WCYL ;FORMAT22=16BIT WORDS AND
;CYLINDER 0
MOV #1,@#WSECTR ;TRACK=0, 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
MOV #-260.,@#RHWC ;256 DATA WORDS 4 HEADER WORDS
MOV #WRFROM,R0 ;THESE TWO INSTRUCTIONS GETS
MOV R0,@#HBA ;ADDR. OF WRFROM INTO R0 AND
;BUS ADDRESS REGISTER
MOV #FMT22,(R0)+ ;FORMAT=16 BIT WORDS
;CYLINDER=0
2$: MOV #1,(R0)+ ;TRACK=0, SECTOR=1, KEYS=0

```

```

6146 036566 005020          CLR      (R0)+      ;KEY1=0
6147 036570 005020          CLR      (R0)+      ;KEY2=0
6148 036572 012705 000400    MOV      #256.,R5    ;COUNTER
6149 036576 012720 177777    3$: MOV      #-1,(R0)+ ;MOVE ALL ONES FOR DATA
6150 036602 005305          DEC      R5
6151 036604 001374          BNE      3$          ;BRANCH IF DATA NOT COMPLETE
6152 036606 012777 000001 156146 MOV      #1,@RHDST ;TRACK=0 SECTOR=1
6153
6154 036614 004737 046160    JSR      PC,@#CHECKT ;CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DON'T
6155 036620 104401 005123    TYPE      ,CPHALT ;CANNOT CONTINUE TESTING IF ANY OF THE
6156                                ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
6157 036624 000000          HALT                ;STOP THE TEST
6158
6159 036626 013711 015172    MOV      @#WRIFOR,@R1 ;GET READY FOR WRITE HEADER AND
6160                                ;DATA WITH 62 IN RHCS1
6161 036632 005037 015124    CLR      @#ERFLG$ ;CLEAR ERROR FLAG
6162 036636 012777 010000 156122 MOV      #FMT22,@RHOF ;FORMAT BIT=1 (16 BIT WORDS)
6163 036644 005077 156120    CLR      @RHCA ;CYLINDER =0
6164 036650 004737 055772    JSR      PC,@#COMWHD ;WRITE HEADER AND DATA
6165
6166                                ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
6167                                ;*FROM THE "COMWHD" ROUTINE THAT MEANS ALL HEADER ON DISK
6168                                ;*IS GOOD IE. ONLY DATA IS TO BE CHECKED TO SEE IF THEY ARE
6169                                ;*ALL ONES AND WRITE DATA GAP AND TOLERANCE GAP TO SEE IF
6170                                ;*THEY ARE ALL ZEROS
6171
6172 036654 005737 015124    TST      @#ERFLG$ ;HAS ANY ERRORS OCCURED?
6173                                ;IF WRITE ERROR OCCURS ECC IS NOT CHECKED
6174 036660 001056          BNE      TST60 ;:BRANCH IF YES
6175
6176                                ;*COMPARE SOFTWARE GENERATED ECC WITH THAT GENERATED BY HARDWARE
6177
6178                                ;*COMPARE SOFTWARE GENERATED ECC WITH THAT GENERATED BY HARDWARE
6179 036662 023737 050620 055726 CMP      @#GECC1,@#WECC1;COMPARE SOFTWARE ECC WITH HARDWARE ECC
6180 036670 001402          BEQ      6$          ;BRANCH IF GOOD
6181 036672 104031          ERROR    31          ;LOW ORDER ECC IN ERROR
6182 036674 000405          BR      7$          ;BRANCH TO CONTINUE
6183 036676 023737 050622 055730 6$: CMP      @#GECC2,@#WECC2;COMPARE SOFTWARE ECC WITH HARDWARE ECC
6184 036704 001401          BEQ      7$          ;BRANCH IF GOOD
6185 036706 104031          ERROR    31          ;HIGH ORDER ECC IN ERROR
6186
6187                                ;*FILL "REINTO" BUFFER WITH EXPECTED DATA
6188                                ;*FILL "REINTO" BUFFER WITH EXPECTED DATA
6189 036710          7$: JSR      PC,@#CHECKE ;CHECK THAT DVA,RDY,DPR,DRY = 1
6190 036714 104401 005123    TYPE      ,CPHALT ;CANNOT CONTINUE IF THEY DON'T
6191 036720 000000          HALT                ;STOP THE TEST AND RESTART PROGRAM
6192
6193                                ;*FILL "REINTO" BUFFER WITH EXPECTED DATA
6194                                ;*FILL "REINTO" BUFFER WITH EXPECTED DATA
6195                                ;*FILL "REINTO" BUFFER WITH EXPECTED DATA
6196                                ;*FILL "REINTO" BUFFER WITH EXPECTED DATA
6197                                ;*FILL "REINTO" BUFFER WITH EXPECTED DATA
6198 036722 004037 046034    JSR      RO,@#CLAREA ;FILL REINTO BUFFER
6199 036726 016264          REINTO ;FROM
6200 036730 017262          REINTO+<255.*2> ;TO
6201 036732 177777          .WORD    -1 ;DATA

```

```

6202
6203 036734 013737 050620 017264      MOV    @#GECC1,@#REINTO+<256.*2>;FILL ECC1
6204 036742 013737 050622 017266      MOV    @#GECC2,@#REINTO+<257.*2>;FILL ECC2
6205 036750 004037 046034              JSR     RO,@#CLAREA      ;FILL REST
6206 036754 017270              REINTO+<258.*2>      ;FROM
6207 036756 017324              REINTO+<272.*2>      ;TO
6208 036760 000000              0          ;DATA
6209
6210
6211 036762 005037 015124      CLR     @#ERFLG$      ;CLEAR ERROR FLAG
6212
6213
6214              ;*NOW COMPARE 'DISK' BUFFER WITH 'REINTO'
6215
6216 036766 004037 . 047020      JSR     RO,@#COMPAR      ;CHECK
6217 036772 016264              REINTO          ;GOOD BUFFER
6218 036774 054726              DISK          ;TEST BUFFER
6219 036776 000402              258.          ;NUMBER OF WORDS CHECKED
6220 037000 037006              4$          ;RETURN POINT FOR ERROR HEADER
6221 037002 037012              5$          ;RETURN POINT FOR ERROR DATA
6222
6223 037004 037016              TST60          ;RETURN FOR GOOD COMPARISON
6224
6225 037006 104007              4$:  ERROR 7          ;READ ERROR 10 NEXT
6226 037010 000207              RTS  PC          ;RETURN TO COMPARE
6227 037012 104010              5$:  ERROR 10         ;WORD NOS 1 TO 256 ARE
6228                                     ;DATA WORDS
6229                                     ;WORD NOS 257 AND 258
6230                                     ;ARE ECC WHICH ARE CHECKED
6231                                     ;WORD NOS 259
6232                                     ;IS DATA GAP
6233                                     ;WORD NOS 260 TO 273
6234                                     ;ARE TOLERANCE GAP
6235 037014 000207              RTS  PC          ;RETURN TO COMPARE
6236
6237
6238
6239
6240
  
```

```

6241
6242
6243      ;*****
6244      ;*TEST 60      READ ECC ENABLED 2A
6245      ;*
6246      ;*      THIS IS AN ECC READ DATA TEST
6247      ;*      ERROR CORRECTION IS ENABLED
6248      ;*      NO ERROR IS INSERTED
6249      ;*      GOOD DATA USED IS 256 WORDS OF 177777
6250      ;*      COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
6251      ;*      TRACK 0, SECTOR 0 KEYS 0 READ HEADER AND DATA
6252      ;*****
6253      037016 000004      TST60: SCOPE
6254      037020 012706 001000      MOV      #STACK,SP      ;RESET STACK
6255
6256      037024 012737 000060 017330      MOV      #TTNO,@#TSTNM      ;THIS SAVES TEST NUMBER
6257
6258
6259      ;      SETUP FOR WHAT IS TO BE READ
6260      ;      HEADER CRC IS RESTORED FROM A SUBROUTINE
6261
6262      037032 012746 177777      MOV      #-1,-(SP)      ;DATA TO BE READ
6263      037036 012705 000400      MOV      #256.,R5      ;COUNTER
6264      037042 012700 054726      MOV      #DISK,R0      ;START OF SIMULATED DISK DATA
6265      037046 011620      1$: MOV      (SP),(R0)+      ;MOVE IN DATA ON TO SIMULATED DISK
6266      037050 005305      DEC      R5      ;COUNT
6267      037052 001375      BNE      1$      ;BRANCH IF 256 NOT COMPLETE
6268      037054 005726      TST      (SP)+      ;UNDO -(SP)
6269      037056 022020      CMP      (R0)+,(R0)+      ;JUMP OVER THE TWO ECC WORDS
6270      037060 012705 000017      MOV      #15.,R5      ;1 DATA GAP
6271
6272      037064 005020      2$: CLR      (R0)+      ;14 TOLERANCE GAP
6273      037066 005305      DEC      R5      ;CLEAR DATA GAP, AND
6274      037070 001375      BNE      2$      ;TOLERANCE GAP
6275
6276
6277      037072 004737 051414      JSR      PC,@#FILLEC      ;INSERT THE TWO ECC WORDS ON THE DISK
6278
6279
6280      ;*THESE ARE FOR ECC TEST ONLY
6281
6282      037076 012737 177777 015142      MOV      #-1,@#TSECC      ;THIS IS AN ECC TEST
6283      037104 005037 050632      CLR      @#POSITI      ;CLEAR ERROR POSITION COUNTER
6284      037110 013737 050626 050630      MOV      @#NCODE,@#NCOUNT      ;TEMPORARY N-CODE COUNTER
6285      037116 013737 050634 050642      MOV      @#HARDER,@#HADTMP      ;TEMPORARY HARD ERROR COUNTER
6286      037124 005037 050620      CLR      @#GECC1      ;ECC LOW ORDER TO BE GENERATED
6287      037130 005037 050622      CLR      @#GECC2      ;ECC HIGH ORDER TO BE GENERATED
6288      037134 005037 050636      CLR      @#DATENV      ;CLEAR DATA ENVELOPE CLOCK COUNT
6289      037140 005037 050640      CLR      @#ZCODE      ;CLEAR LEADING ZEROS CLOCK COUNT
6290
6291
6292      ;*THESE ARE TO SETUP FOR DISKLESS USE ONLY
6293
6294      037144 012737 010000 053010      MOV      #FMT22,@#CYL      ;16 BITS PER WORD
6295
6296      037152 112737 000000 053013      MOV      #0,@#SECOTR+1      ;CYLINDER 0, FORMAT 16 BITS
        ;TRACK 0
  
```

```

6297 037160 112737 000000 053012      MOV#B  #0,2#SECOTR      ;SECTOR 0
6298 037166 012737 000000 053014      MOV  #0,2#KEY1        ;KEY1=0
6299 037174 012737 000000 053016      MOV  #0,2#KEY2        ;KEY2=0
6300 037202 012737 000400 053070      MOV  #256.,2#DAWORD   ;NO. OF DATA WORDS
6301 037210 005037 053020              CLR  2#X              ;THIS IS A READ COMMAND
6302 037214 004537 047332              JSR  R5,2#CRC         ;GO TO CALCULATE CRC
6303 037220 053010
6304 037222 054710
6305
6306
6307
6308
6309
6310
6311 037224 004737 046116              JSR  PC,2#CLDISK      ;SETUP GENERAL REGISTERS
6312 037230 012777 177374 155512      MOV  #-256.-4.,2#RHWC ;256. DATA 4 HEADER WORDS
6313 037236 012777 016264 155506      MOV  #REINTO,2#RHBA  ;STARTING ADDRESS OF READ BUFFER
6314 037244 112746 000000              MOV#B  #0,-(SP)       ;IN LOWER BYTE GET SECTOR
6315 037250 112766 000000 000001      MOV#B  #0,1(SP)      ;GET TRACK IN HIGHER BYTE
6316 037256 012677 155500              MOV  (SP)+,2#RHDST    ;TRACK/SECTOR IN RHDST
6317 037262 012777 010000 155476      MOV  #FMT22,2#RHOF   ;16 BITS PER WORD
6318
6319
6320
6321 037270 005077 155474              CLR  2#RHCA          ;ECC CORRECTION NOT INHIBIT
6322
6323 037274 004737 046160              JSR  PC,2#CHECKT      ;BECAUSE ECC IS NOT GOING
6324 037300 104401 005123              TYPE ,CPHALT        ;TO BE CHECKED
6325
6326 037304 000000                      HALT                ;CYLINDER 0
6327
6328 037306 013711 015176              MOV  2#REFOR,2#R1    ;CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DON'T
6329 037312 005037 015124              CLR  2#ERFLG$        ;CANNOT CONTINUE TESTING IF ANY OF THE
6330 037316 004737 052650              JSR  PC,2#COMHD      ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
6331
6332
6333
6334
6335
6336
6337
6338
6339
6340
6341
6342
6343
6344
6345 037322 005737 015124              TST  2#ERFLG$        ;STOP THE TEST
6346 037326 001102                      BNE  TST61            ;READ HEADER AND DATA=72
6347 037330 004737 045616              JSR  PC,2#PUTREG      ;CLEAR ERROR FLAG
6348 037334 005737 015034              TST  2#ER1           ;READ HEADER AND DATA
6349 037340 001401                      BEQ  6$              ;IF THERE ARE READ ERRORS THEN
6350 037342 104032                      ERROR 32            ;ECC WILL NOT BE CHECKED
6351
6352
; *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
; *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
; *ANY ERRORS ALREADY THERE
; *BRANCH IF YES
; *SAVE REGISTERS
; *NO ERRORS SHOULD BE SET
; *BRANCH IF NO ERRORS SET
; *32 BIT ECC REGISTER SHOULD BE ZERO
; *ONLY 11 OF THE 32 BITS CAN BE SEEN
; *IN THE PATTERN REGISTER

```

```

6353
6354 037344 013746 050620      6$: MOV    @#GECC1,-(SP)    ;DCK SHOULD BE SET IN RHER1
6355 037350 042716 174000      BIC    #174000,(SP)    ;GET PATTERN REGISTER
6356 037354 022637 015064      CMP    (SP)+,@#EC2    ;KEEP ONLY 11 BITS
6357 037360 001401      BEQ    7$      ;COMPARE PATTERN REGISTER
6358 037362 104032      ERROR  32      ;BRANCH IF GOOD
                                           ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
6359
6360
6361
6362
6363      ;*ADD 16 MAINTENANCE CLOCKS TO
6364      ;*BRING EBL DOWN
6365
6366 037364 012700 000020      7$: MOV    #16.,R0      ;COUNTER
6367 037370 052777 000002 155400 8$: BIS    #MCLK,@RHMR    ;SET CLOCK
6368 037376 042777 000002 155372 BIC    #MCLK,@RHMR    ;CLEAR CLOCK
6369 037404 005300      DEC    R0      ;COUNT
6370 037406 001370      BNE    8$      ;BRANCH IF 16 CLOCKS NOT DONE
6371 037410 004737 046350      JSR    PC,@#CHECKE    ;CHECK THAT DVA,RDY,DPR,DRY = 1
6372 037414 104401 005123      TYPE    ,CPHALT    ;CANNOT CONTINUE IF THEY DON'T
6373 037420 000000      HALT                    ;STOP THE TEST AND RESTART PROGRAM
6374 037422 012700 015220      MOV    #WRFROM,R0    ;GETTING READY TO FILL EXPECTED DATA
6375 037426 012720 010000      MOV    #0!FMT22,(R0)+    ;CYLINDER 0
6376 037432 112746 000000      MOV    #0,-(SP)    ;IN LOWER BYTE GET SECTOR
6377 037436 112766 000000 000001 MOV    #0,1(SP)    ;GET TRACK IN HIGHER BYTE
6378 037444 012620      MOV    (SP)+,(R0)+    ;GET TRACK/SECTOR IN BUFFER
6379 037446 012720 000000      MOV    #0,(R0)+    ;KEY1 IN BUFFER
6380 037452 012720 000000      MOV    #0,(R0)+    ;KEY2 IN BUFFER
6381 037456 012701 000400      MOV    #256.,R1    ;DATA WORD COUNTER
6382 037462 012702 177777      MOV    #-1,R2    ;DATA
6383
6384 037466 010220      3$: MOV    R2,(R0)+    ;DATA INTO BUFFER
6385 037470 005301      DEC    R1      ;COUNT
6386 037472 001375      BNE    3$      ;BRANCH IF 256 NOT DONE
6387 037474 005037 015124      CLR    @#ERFLG$    ;CLEAR ERROR FLAG
6388 037500 004737 045616      JSR    PC,@#PUTREG    ;SAVE REGISTERS
6389
6390      ;NOW READ DATA BUFFER WILL BE CHECKED
6391
6392 037504 004037 047020      JSR    R0,@#COMPAR    ;CHECK
6393 037510 015220      WRFROM    ;GOOD BUFFER
6394 037512 016264      REINTO    ;TEST BUFFER
6395 037514 000404      4+256.    ;NUMBER OF WORDS CHECKED
6396 037516 037524      4$      ;RETURN POINT FOR ERROR HEADER
6397 037520 037530      5$      ;RETURN POINT FOR ERROR DATA
6398
6399 037522 037534      TST61    ;RETURN FOR GOOD COMPARISON
6400
6401 037524 104004      4$: ERROR  4      ;READ NEXT ERROR
6402 037526 000207      RTS    PC      ;RETURN TO 'COMPAR'
6403 037530 104005      5$: ERROR  5      ;WORD NOS 1 TO 4 ARE
6404      ;HEADER WORDS
6405      ;5 TO 260 ARE DATA WORDS
6406 037532 000207      RTS    PC      ;RETURN TO 'COMPAR'
6407
6408

```

CZRJHDO,RP04/5/6 DSKLS CTRLR2 MACY11 30A(1052) 24-MAY-79 15:07 D 12
CZRJHD.P12 01-MAR-79 09:10 T60 READ ECC ENABLED 2A PAGE 148

SEQ 0146

6409
6410

[illegible]

6467	037670	112737	000000	053013	MOVB	#0, @#SECOTR+1	; TRACK 0
6468	037676	112737	000000	053012	MOVB	#0, @#SECOTR	; SECTOR 0
6469	037704	012737	000000	053014	MOV	#0, @#KEY1	; KEY1=0
6470	037712	012737	000000	053016	MOV	#0, @#KEY2	; KEY2=0
6471	037720	012737	000400	053070	MOV	#256., @#DAWORD	; NO. OF DATA WORDS
6472	037726	005037	053020		CLR	@#X	; THIS IS A READ COMMAND
6473	037732	004537	047332		JSR	R5, @#CRC	; GO TO CALCULATE CRC
6474	037736	053010			CYL		
6475	037740	054710			WCRC		
6476							
6477							
6478							
6479							
6480							
6481							
6482							
6483	037742	012737	077777	054730	MOV	#77777, @#DISK+2	; FORCE ERROR ON BIT NUMBER 32
6484							; SO ERROR POSITION REGISTER WILL SHOW
6485							; 22
6486	037750	012737	000026	040124	MOV	#22., @#8\$	; INSERT POSITION REG.
6487							
6488							
6489							
6490							
6491	037756	004737	046116		JSR	PC, @#CLDISK	; SETUP GENERAL REGISTERS
6492	037762	012777	177374	154760	MOV	#-256.-4., @#RHC	; 256. DATA 4 HEADER WORDS
6493	037770	012777	016264	154754	MOV	#REINTO, @#RHB	; STARTING ADDRESS OF READ BUFFER
6494	037776	112746	000000		MOVB	#0, -(SP)	; IN LOWER BYTE GET SECTOR
6495	040002	112766	000000	000001	MOVB	#0, 1(SP)	; GET TRACK IN HIGHER BYTE
6496	040010	012677	154746		MOV	(SP)+, @#RHDST	; TRACK/SECTOR IN RHDST
6497	040014	012777	010000	154744	MOV	#FM*22, @#RHO	; 16 BITS PER WORD
6498							; ECC CORRECTION NOT INHIBIT
6499							; BECAUSE ECC IS NOT GOING
6500							; TO BE CHECKED
6501	040022	005077	154742		CLR	@#RHCA	; CYLINDER 0
6502	040026	004737	046160		JSR	PC, @#CHECKT	; CHECK DVA, RDY, DPR, DRY = 1 AND OTHERS DON'T
6503	040032	104401	005123		TYPE	, CPHALT	; CANNOT CONTINUE TESTING IF ANY OF THE
6504							; ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
6505	040036	000000			HALT		; STOP THE TEST
6506	040040	013711	015176		MOV	@#REFOR, @R1	; READ HEADER AND DATA=7?
6507	040044	005037	015124		CLR	@#ERFLG\$	; CLEAR ERROR FLAG
6508	040050	004737	052650		JSR	PC, @#COMHD	; READ HEADER AND DATA
6509							; IF THERE ARE READ ERRORS THEN
6510							; ECC WILL NOT BE CHECKED
6511							
6512							
6513							
6514							
6515							
6516							
6517							
6518							
6519							
6520							
6521							
6522							

; *THIS IS TO INSERT ERROR
 ; *THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
 ; *THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
 ; *THIS MOVE

; *THESE ARE REGULAR SETUPS

; *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 SYNCs WERE CORRECTLY
 ; *DETECTED
 ; *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

6523	040054	005737	015124		TST	@#ERFLG\$	; ANY ERRORS ALREADY THERE
6524	040060	001077			BNE	TST62	; BRANCH IF YES
6525	040062	004737	045616		JSR	PC,@#PUTREG	; SAVE REGISTERS
6526	040066	022737	100000	015034	CMP	#DCK,@#ER1	; ONLY DATA CHECK ERROR SHOULD BE SET
6527	040074	001401			BEQ	6\$	; BRANCH IF YES
6528	040076	104032			ERROR	32	; 32 BIT ECC REGISTER SHOULD BE NON
6529							; ZERO
6530							; ONLY 11 OF THE 32 BITS CAN BE SEEN
6531							; IN THE PATTERN REGISTER
6532							; DCK SHOULD BE SET IN RHER1
6533	040100	013746	050620	6\$:	MOV	@#GECC1,-(SP)	; GET PATTERN REGISTER
6534	040104	042716	174000		BIC	#174000,(SP)	; KEEP ONLY 11 BITS
6535	040110	022637	015064		CMP	(SP)+,@#EC2	; COMPARE PATTERN REGISTER
6536	040114	001401			BEQ	7\$	; BRANCH IF GOOD
6537	040116	104032			ERROR	32	; 11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
6538							
6539	040120	004037	051242	7\$:	JSR	RO,@#ECORR	; GO TO ECC CORRECTION PROCESS
6540	040124	000026		8\$:	22.		; EXPECTED POSITION REG. WHEN CORRECTION
6541							; IS COMPLETE
6542							
6543							
6544							
6545	040126	004737	046350		JSR	PC,@#CHECKE	; CHECK THAT DVA,RDY,DPR,DRY = 1
6546	040132	104401	005123		TYPE	,CPHALT	; CANNOT CONTINUE IF THEY DON'T
6547	040136	000000			HALT		; STOP THE TEST AND RESTART PROGRAM
6548	040140	012700	015220		MOV	#WRFROM,RO	; GETTING READY TO FILL EXPECTED DATA
6549	040144	012720	010000		MOV	#0!FMT22,(RO)+	; CYLINDER 0
6550	040150	112746	000000		MOVB	#0,-(SP)	; IN LOWER BYTE GET SECTOR
6551	040154	112766	000000	000001	MOVB	#0,1(SP)	; GET TRACK IN HIGHER BYTE
6552	040162	012620			MOV	(SP)+,(RO)+	; GET TRACK/SECTOR IN BUFFER
6553	040164	012720	000000		MOV	#0,(RO)+	; KEY1 IN BUFFER
6554	040170	012720	000000		MOV	#0,(RO)+	; KEY2 IN BUFFER
6555	040174	012701	000400		MOV	#256,R1	; DATA WORD COUNTER
6556	040200	012702	177777		MOV	#-1,R2	; DATA
6557	040204	010220		3\$:	MOV	R2,(RO)+	; DATA INTO BUFFER
6558	040206	005301			DEC	R1	; COUNT
6559	040210	001375			BNE	3\$	; BRANCH IF 256 NOT DONE
6560							
6561							; *ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
6562							; *NOW THE INSERTED ERROR WILL BE PUT IN
6563							
6564	040212	012737	077777	015232	MOV	#77777,@#WRFROM+<5*2>	; INSERTED ERROR
6565	040220	004737	045616		JSR	PC,@#PUTREG	; SAVE REGISTERS
6566	040224	005037	015124		CLR	@#ERFLG\$	; CLEAR ERROR FLAG
6567							
6568							
6569							; *NOW READ DATA BUFFER WILL BE CHECKED
6570							
6571	040230	004037	047020		JSR	RO,@#COMPAR	; CHECK
6572	040234	015220			WRFROM		; GOOD BUFFER
6573	040236	016264			REINTO		; TEST BUFFER
6574	040240	000404			4+256.		; NUMBER OF WORDS CHECKED
6575	040242	040250			4\$		; RETURN POINT FOR ERROR HEADER
6576	040244	040254			5\$		; RETURN POINT FOR ERROR DATA
6577							
6578	040246	040260			TST62		; RETURN FOR GOOD COMPARISON

CZRJHDO,RP04/5/6 DSKLS CTRLR2 MACY11 30A(1052) 24-MAY-79 15:07 H 12
CZRJHD.P12 01-MAR-79 09:10 T61 READ ECC ENABLED 2B PAGE 1>2

SEQ 0150

6579					
6580	040250	104004	4\$:	ERROR	4
6581	040252	000207		RTS	PC
6582	040254	104005	5\$:	ERROR	5
6583					
6584					
6585	040256	000207		RTS	PC
6586					

;READ NEXT ERROR
;RETURN TO "COMPAR"
;WORD NOS 1 TO 4 ARE
;HEADER WORDS
;5 TO 260 ARE DATA WORDS
;RETURN TO "COMPAR"

```

6587
6588
6589
6590
6591
6592
6593
6594
6595
6596
6597
6598
6599 040260 000004
6600 040262 012706 001000
6601
6602 040266 012737 000062 017330
6603
6604
6605
6606
6607
6608 040274 012746 177777
6609 040300 012705 000400
6610 040304 012700 054726
6611 040310 011620
6612 040312 005305
6613 040314 001375
6614 040316 005726
6615 040320 022020
6616 040322 012705 000017
6617
6618 040326 005020
6619 040330 005305
6620 040332 001375
6621
6622
6623 040334 004737 051414
6624
6625
6626
6627
6628 040340 012737 177777 015142
6629 040346 005037 050632
6630 040352 013737 050626 050630
6631 040360 013737 050634 050642
6632 040366 005037 050620
6633 040372 005037 050622
6634 040376 005037 050636
6635 040402 005037 050640
6636
6637
6638
6639
6640 040406 012737 010000 053010
6641
6642 040414 112737 000000 053013

```

```

*****
; *TEST 62      READ ECC ENABLED 2C
; *
; * THIS IS AN ECC READ DATA TEST
; * ERROR CORRECTION IS ENABLED
; * A NON CORRECTABLE ERROR IS INSERTED IN BIT POSITION 32 AND 21
; * GOOD DATA USED IS 256 WORDS OF 177777
; * COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
; * TRACK 0, SECTOR 0 KEYS 0  READ HEADER AND DATA
; *
*****
TST62: 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)
        CMP      (R0)+,(R0)+   ;JUMP OVER THE TWO ECC WORDS
        MOV      #15.,R5      ;1 DATA GAP
                                ;14 TOLERANCE GAP
2$:      CLR      (R0)+         ;CLEAR DATA GAP, AND
        DEC      R5            ;TOLERANCE GAP
        BNE      2$           ;BRANCH IF NOT COMPLETE

        JSR      PC,@#FILLEC   ;INSERT THE TWO ECC WORDS ON THE DISK
                                ;IN THE CORRECT PLACE

; *THESE ARE FOR ECC TEST ONLY
        MOV      #-1,@#TSECC   ;THIS IS AN ECC TEST
        CLR      @#POSIT1     ;CLEAR ERROR POSITION COUNTER
        MOV      @#NCODE,@#NCOUNT ;TEMPORARY N-CODE COUNTER
        MOV      @#HARDER,@#HADTMP ;TEMPORARY HARD ERROR COUNTER
        CLR      @#GECC1      ;ECC LOW ORDER TO BE GENERATED
        CLR      @#GECC2      ;ECC HIGH ORDER TO BE GENERATED
        CLR      @#DATENV     ;CLEAR DATA ENVELOPE CLOCK COUNT
        CLR      @#ZCODE      ;CLEAR LEADING ZEROS CLOCK COUNT

; *THESE ARE TO SETUP FOR DISKLESS USE ONLY
        MOV      #FMT22,@#CYL  ;16 BITS PER WORD
                                ;CYLINDER 0, FORMAT 16 BITS
        MOVB     #0,@#SECTR+1  ;TRACK 0

```

```

6643 040422 112737 000000 053012      MOVB    #0,@#SECOTR      ;SECTOR 0
6644 040430 012737 000000 053014      MOV     #0,@#KEY1       ;KEY1=0
6645 040436 012737 000000 053016      MOV     #0,@#KEY2       ;KEY2=0
6646 040444 012737 000400 053070      MOV     #256.,@#DAWORD ;NO. OF DATA WORDS
6647 040452 005037 053020              CLR     @#X              ;THIS IS A READ COMMAND
6648 040456 004537 047332              JSR     R5,@#CRC        ;GO TO CALCULATE CRC
6649 040462 053010                      CYL
6650 040464 054710                      WCRC
6651
6652
6653      ;*THIS IS TO INSERT ERROR
6654      ;*THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
6655      ;*THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
6656      ;*THIS MOVE
6657
6658 040466 012737 077757 054730      MOV     #77757,@#DISK+2 ;FORCE ERROR ON BIT NUMBER 32 AND 21
6659                                      ;SO ERROR POSITION REGISTER WILL SHOW
6660                                      ;22
6661 040474 012737 010040 040650      MOV     #4128.,@#8$     ;INSERT POSITION REG.
6662
6663
6664      ;*THESE ARE REGULAR SETUPS
6665
6666 040502 004737 046116              JSR     PC,@#CLDISK     ;SETUP GENERAL REGISTERS
6667 040506 012777 177374 154234      MOV     #-256.-4.,@#RHWC ;256. DATA 4 HEADER WORDS
6668 040514 012777 016264 154230      MOV     @#REINTO,@#RHBA ;STARTING ADDRESS OF READ BUFFER
6669 040522 112746 000000              MOVB    #0,-(SP)        ;IN LOWER BYTE GET SECTOR
6670 040526 112766 000000 000001      MOVB    #0,1(SP)       ;GET TRACK IN HIGHER BYTE
6671 040534 012677 154222              MOV     (SP)+,@#RHDST   ;TRACK/SECTOR IN RHDST
6672 040540 012777 010000 154220      MOV     @#FMT22,@#RHOF ;16 BITS PER WORD
6673                                      ;ECC CORRECTION NOT INHIBIT
6674                                      ;BECAUSE ECC IS NOT GOING
6675                                      ;TO BE CHECKED
6676 040546 005077 154216              CLR     @#RHCA          ;CYLINDER 0
6677
6678 040552 004737 046160              JSR     PC,@#CHECKT     ;CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DON'T
6679 040556 104401 005123              TYPE    ,CPHALT       ;CANNOT CONTINUE TESTING IF ANY OF THE
6680                                      ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
6681 040562 000000                      HALT                    ;STOP THE TEST
6682
6683 040564 013711 015176              MOV     @#REFOR,@#R1   ;READ HEADER AND DATA=72
6684 040570 005037 015124              CLR     @#ERFLG$      ;CLEAR ERROR FLAG
6685 040574 004737 052650              JSR     PC,@#COMHD     ;READ HEADER AND DATA
6686                                      ;IF THERE ARE READ ERRORS THEN
6687                                      ;ECC WILL NOT BE CHECKED
6688
6689
6690      ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
6691      ;*FROM THE 'COMHD' ROUTINE THAT MEANS SECTOR GAP,
6692      ;*FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
6693      ;*SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
6694      ;*DETECTED
6695      ;*HEADER AND DATA ARE TO BE CHECKED.
6696      ;*IN CHECKING READ DATA THE WRITE FROM BUFFER
6697      ;*'WRFROM' IS FILLED WITH EXPECTED DATA AND
6698      ;*COMPARISONS ARE MADE

```

```

6699
6700 040600 005737 015124      TST    @#ERFLG$      ;ANY ERRORS ALREADY THERE
6701 040604 001106              BNE     TST63          ;BRANCH IF YES
6702 040606 004737 045616      JSR     PC,@#PUTREG    ;SAVE REGISTERS
6703 040612 022737 100000 015034 CMP     #DCK,@#ER1    ;ONLY DATA CHECK ERROR SHOULD BE SET
6704 040620 001401              BEQ     6$             ;BRANCH IF YES
6705 040622 104032              ERROR  32             ;32 BIT ECC REGISTER SHOULD BE NON
6706                                ;ZERO
6707                                ;ONLY 11 OF THE 32 BITS CAN BE SEEN
6708                                ;IN THE PATERN REGISTER
6709                                ;DCK SHOULD BE SET IN RHER1
6710 040624 013746 050620      6$:  MOV     @#GECC1,-(SP)    ;GET PATTERN REGISTER
6711 040630 042716 174000      BIC     #174000,(SP)    ;KEEP ONLY 11 BITS
6712 040634 022637 015064      CMP     (SP)+,@#EC2    ;COMPARE PATTERN REGISTER
6713 040640 001401              BEQ     7$             ;BRANCH IF GOOD
6714 040642 104032              ERROR  32             ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
6715
6716 040644 004037 051242      7$:  JSR     R0,@#ECORR    ;GO TO ECC CORRECTION PROCESS
6717 040650 000000      8$:  .WORD    ;EXPECTED POSITION REG. WHEN CORRECTION
6718                                ;IS COMPLETE
6719
6720 040652 004737 045616      JSR     PC,@#PUTREG    ;SAVE REGISTERS
6721
6722
6723 040656 022737 100100 015034 CMP     #DCK!ECH,@#ER1 ;WITH ERRORS INSERTED IN BIT POSITION 21
6724                                ;AND 32 HARD ERROR BIT SHOULD SET
6725 040664 001401              BEQ     9$             ;BRANCH IF GOOD
6726 040666 104036              ERROR  36             ;WITH ERROR INSERTED IN BIT POSITION 21 THRU
6727                                ;32 HCE SHOULD SET
6728
6729
6730
6731
6732 040670      9$:
6733 040670 004737 046350      JSR     PC,@#CHECKE    ;CHECK THAT DVA,RDY,DPR,DY = 1
6734 040674 104401 005123      TYPE    ,CPHALT        ;CANNOT CONTINUE IF THEY DON'T
6735 040700 000000              HALT                    ;STOP THE TEST AND RESTART PROGRAM
6736 040702 012700 015220      MOV     #WRFROM,R0      ;GETTING READY TO FILL EXPECTED DATA
6737 040706 012720 010000      MOV     #0!FMT22,(R0)+    ;CYLINDER 0
6738 040712 112746 000000      MOV     #0,-(SP)         ;IN LOWER BYTE GET SECTOR
6739 040716 112766 000000 000001 MOV     #0,1(SP)      ;GET TRACK IN HIGHER BYTE
6740 040724 012620              MOV     (SP)+,(R0)+    ;GET TRACK/SECTOR IN BUFFER
6741 040726 012720 000003      MOV     #0,(R0)+        ;KEY1 IN BUFFER
6742 040732 012720 000000      MOV     #0,(R0)+        ;KEY2 IN BUFFER
6743 040736 012701 000400      MOV     #256.,R1      ;DATA WORD COUNTER
6744 040742 012702 177777      MOV     #-1,R2       ;DATA
6745 040746 010220      3$:  MOV     R2,(R0)+        ;DATA INTO BUFFER
6746 040750 005301              DEC     R1             ;COUNT
6747 040752 001375              BNE     3$             ;BRANCH IF 256 NOT DONE
6748
6749                                ;*ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
6750                                ;*NOW THE INSERTED ERROR WILL BE PUT IN
6751
6752 040754 012737 077757 015232 MOV     #77757,@#WRFROM+<5*2> ;INSERTED ERROR
6753 040762 004737 045616      JSR     PC,@#PUTREG    ;SAVE REGISTERS
6754 040766 005037 015124      CLR     @#ERFLG$      ;CLEAR ERROR FLAG

```

```
6755
6756
6757 ;*NOW READ DATA BUFFER WILL BE CHECKED
6758
6759 040772 004037 047020 JSR RO,#COMPAP ;CHECK
6760 040776 015220 WRFROM ;GOOD BUFFER
6761 041000 016264 REINTO ;TEST BUFFER
6762 041002 000404 4+256. ;NUMBER OF WORDS CHECKED
6763 041004 041012 4$ ;RETURN POINT FOR ERROR HEADER
6764 041006 041016 5$ ;RETURN POINT FOR ERROR DATA
6765
6766 041010 041022 TST63 ;RETURN FOR GOOD COMPARISON
6767
6768 041012 104004 4$: ERROR 4 ;READ NEXT ERROR
6769 041014 000207 RTS PC ;RETURN TO 'COMPAR'
6770 041016 104005 5$: ERROR 5 ;WORD NOS 1 TO 4 ARE
6771 ;HEADER WORDS
6772 ;5 TO 260 ARE DATA WORDS
6773 041020 000207 RTS PC ;RETURN TO 'COMPAR'
```

```

6774
6775
6776
6777
6778
6779
6780
6781
6782
6783
6784 041022 000004
6785 041024 012706 001000
6786
6787 041030 012737 000063 017330
6788 041036 012700 054630
6789 041042 012701 000460
6790 041046 005020
6791 041050 005301
6792 041052 001375
6793 041054 004737 046116
6794
6795
6796
6797 041060 012737 177777 015142
6798 041066 005037 050632
6799 041072 013737 050626 050630
6800 041100 013737 050634 050642
6801 041106 005037 050620
6802 041112 005037 050622
6803 041116 005037 050636
6804 041122 005037 050640
6805
6806
6807
6808
6809
6810
6811 041126 012737 010000 056146
6812
6813 041134 012737 000001 056150
6814 041142 005037 056152
6815 041146 005037 056154
6816 041152 012737 000400 056206
6817 041160 004537 047332
6818 041164 056146
6819 041166 056156
6820
6821
6822
6823 041170 012777 177374 153552
6824 041176 012700 015220
6825 041202 010077 153544
6826
6827 041206 012720 010000
6828
6829 041212 012720 000001

```

```

*****
; *TEST 63      WRITE ECC TEST 3
; *
; *      THIS IS A WRITE ECC TEST
; *      WRITE CYLINDER 0, FORMAT 16 BITS PER WORD
; *      TRACK 0, SECTOR 1, KEYS 0, NUMBER OF WORDS 256
; *      OF ALL 52525.
*****
1$T63:  SCOPE
        MOV      #STACK,SP      ;RESET STACK
        MOV      #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
        MOV      #SECGAP,R0     ;POINTER
        MOV      #304.,R1      ;COUNTER
1$:     CLR      (R0)+          ;CLEAR SIMULATED DISK AREA
        DEC      R1
        BNE      1$
        JSR      PC,CLDISK      ;THIS IS USED TO SET GENERAL REGISTERS
; *THESE ARE FOR ECC TEST ONLY
        MOV      #-1,@#TSECC    ;THIS IS AN ECC TEST
        CLR      @#POSITI      ;CLEAR ERROR POSITION COUNTER
        MOV      @#NCODE,@#NCOUNT ;TEMPORARY N-CODE COUNTER
        MOV      @#HARDER,@#HADTMP ;TEMPORARY HARD ERROR COUNTER
        CLR      @#GECC1       ;ECC LOW ORDER TO BE GENERATED
        CLR      @#GECC2       ;ECC HIGH ORDER TO BE GENERATED
        CLR      @#DATENV      ;CLEAR DATA ENVELOPE CLOCK COUNT
        CLR      @#ZCODE       ;CLEAR LEADING ZEROS CLOCK COUNT
; *THESE ARE TO BE SETUP FOR DISKLESS USE ONLY
        MOV      #FMT22,@#WCYL ;FORMAT22=16BIT WORDS AND
                                ;CYLINDER 0
        MOV      #1,@#WSECTR   ;TRACK=0, 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
        MOV      #-260.,@#RHWC ;256 DATA WORDS 4 HEADER WORDS
        MOV      #WRFROM,R0     ;THESE TWO INSTRUCTIONS GETS
        MOV      R0,@#HBA      ;ADDR. OF WRFROM INTO R0 AND
                                ;BUS ADDRESS REGISTER
        MOV      #FMT22,(R0)+   ;FORMAT=16 BIT WORDS
                                ;CYLINDER=0
2$:     MOV      #1,(R0)+       ;TRACK=0, SECTOR=1, KEYS=0

```

```

6830 041216 005020          CLR      (R0)+      ;KEY1=0
6831 041220 005020          CLR      (R0)+      ;KEY2=0
6832 041222 012705 000400    MOV      #256,R5      ;COUNTER
6833 041226 012720 052525    3$: MOV      #52525,(R0)+ ;MOVE ALL 52525 FOR DATA
6834 041232 005305          DEC      R5
6835 041234 001374          BNE      3$          ;BRANCH IF DATA NOT COMPLETE
6836 041236 012777 000001 153516 MOV      #1, @RHDS1      ;TRACK=0 SECTOR=1
6837
6838 041244 004737 046160    JSR      PC,@#CHECKT ;CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DON'T
6839 041250 104401 005123    TYPE      ,CPHALT ;CANNOT CONTINUE TESTING IF ANY OF THE
6840                                     ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
6841 041254 000000          HALT      ;STOP THE TEST
6842
6843 041256 013711 015172    MOV      @#WRIFOR,@R1 ;GET READY FOR WRITE HEADER AND
6844                                     ;DATA WITH 62 IN RHCS1
6845 041262 005037 015124    CLR      @#ERFLG$ ;CLEAR ERROR FLAG
6846 041266 012777 010000 153472 MOV      #FMT22,@RHOF ;FORMAT BIT=1 (16 BIT WORDS)
6847 041274 005077 153470    CLR      @RHCA ;CYLINDER =0
6848 041300 004737 055772    JSR      PC,@#COMWHD ;WRITE HEADER AND DATA
6849
6850                                     ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
6851                                     ;*FROM THE "COMWHD" ROUTINE THAT MEANS ALL HEADER ON DISK
6852                                     ;*IS GOOD IE. ONLY DATA IS TO BE CHECKED TO SEE IF THEY ARE
6853                                     ;*ALL 52525 AND WRITE DATA GAP AND TOLERANCE GAP TO SEE IF
6854                                     ;*THEY ARE ALL ZEROS
6855
6856 041304 005737 015124    TST      @#ERFLG$ ;HAS ANY ERRORS OCCURED?
6857                                     ;IF WRITE ERROR OCCURS ECC IS NOT CHECKED
6858 041310 001056          BNE      TST64 ;BRANCH IF YES
6859
6860
6861                                     ;COMPARE SOFTWARE GENERATED ECC WITH THAT GENERATED BY HARDWARE
6862 041312 023737 050620 055726 CMP      @#GECC1,@#WECC1 ;COMPARE SOFTWARE ECC WITH HARDWARE ECC
6863 041320 001402          BEQ      6$          ;BRANCH IF GOOD
6864 041322 104031          ERROR    31          ;LOW ORDER ECC IN ERROR
6865 041324 000405          BR       7$          ;BRANCH TO CONTINUE
6866 041326 023737 050622 055730 6$: CMP      @#GECC2,@#WECC2 ;COMPARE SOFTWARE ECC WITH HARDWARE ECC
6867 041334 001401          BEQ      7$          ;BRANCH IF GOOD
6868 041336 104031          ERROR    31          ;HIGH ORDER ECC IN ERROR
6869
6870
6871                                     7$:
6872 041340 004737 046350    JSR      PC,@#CHECKE ;CHECK THAT DVA,RDY,DPR,DRY = 1
6873 041344 104401 005123    TYPE      ,CPHALT ;CANNOT CONTINUE IF THEY DON'T
6874 041350 000000          HALT      ;STOP THE TEST AND RESTART PROGRAM
6875
6876
6877
6878
6879                                     ;*FILL "REINTO" BUFFER WITH EXPECTED DATA
6880
6881 041352 004037 046034    JSR      R0,@#CLAREA ;FILL REINTO BUFFER
6882 041356 016264          REINTO ;FROM
6883 041360 017262          REINTO+<255.*2> ;TO
6884 041362 052525          .WORD    52525 ;DATA
6885

```

```
6886 041364 013737 050620 017264 MOV @#GECC1,@#REINTO+<256.*2>;FILL ECC1
6887 041372 013737 050622 017266 MOV @#GECC2,@#REINTO+<257.*2>;FILL ECC2
6888 041400 004037 046034 JSR RO,@#CLAREA ;FILL REST
6889 041404 017270 REINTO+<258.*2> ;FROM
6890 041406 017324 REINTO+<272.*2> ;TO
6891 041410 000000 0 ;DATA
6892
6893
6894 041412 005037 015124 CLR @#ERFLG$ ;CLEAR ERROR FLAG
6895
6896
6897 ;*NOW COMPARE 'DISK' BUFFER WITH 'REINTO'
6898
6899 041416 004037 047020 JSR RO,@#COMPAR ;CHECK
6900 041422 016264 REINTO ;GOOD BUFFER
6901 041424 054726 DISK ;TEST BUFFER
6902 041426 000402 258. ;NUMBER OF WORDS CHECKED
6903 041430 041436 4$ ;RETURN POINT FOR ERROR HEADER
6904 041432 041442 5$ ;RETURN POINT FOR ERROR DATA
6905
6906 041434 041446 TST64 ;RETURN FOR GOOD COMPARISON
6907
6908 041436 104007 4$: ERROR 7 ;READ ERROR 10 NEXT
6909 041440 000207 RTS PC ;RETURN TO COMPARE
6910 041442 104010 5$: ERROR 10 ;WORD NOS 1 TO 256 ARE
6911 ;DATA WORDS
6912 ;WORD NOS 257 AND 258
6913 ;ARE ECC WHICH ARE CHECKED
6914 ;WORD NOS 259
6915 ;IS DATA GAP
6916 ;WORD NOS 260 TO 273
6917 ;ARE TOLERANCE GAP
6918 041444 000207 RTS PC ;RETURN TO COMPARE
```

```
6919
6920
6921
6922
6923
6924
6925
6926
6927
6928
6929
6930
6931 041446 000004
6932 041450 012706 001000
6933 041454 012737 000064 017330
6934
6935
6936
6937
6938
6939 041462 012746 052525
6940 041466 012705 000400
6941 041472 012700 054726
6942 041476 011620
6943 041500 005305
6944 041502 001375
6945 041504 005726
6946 041506 022020
6947 041510 012705 000017
6948
6949 041514 005020
6950 041516 005305
6951 041520 001375
6952
6953 041522 004737 051414
6954
6955
6956
6957
6958 041526 012737 177777 015142
6959 041534 005037 050632
6960 041540 013737 050626 050630
6961 041546 013737 050634 050642
6962 041554 005037 050620
6963 041560 005037 050622
6964 041564 005037 050636
6965 041570 005037 050640
6966
6967
6968
6969
6970 041574 012737 010000 053010
6971
6972 041602 112737 000000 053013
6973 041610 112737 000000 053012
6974 041616 012737 000000 053014

*****
*TEST 64      READ ECC ENABLED 3A
*****
*      THIS IS AN ECC READ DATA TEST
*      ERROR CORRECTION IS ENABLED
*      NO ERROR IS INSERTED
*      GOOD DATA USED IS 256 WORDS OF 52525
*      COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
*      TRACK 0, SECTOR 0 KEYS 0 READ HEADER AND DATA
*****
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      #52525,-(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)
        CMP      (R0)+,(R0)+     ;JUMP OVER THE TWO ECC WORDS
        MOV      #15.,R5        ;1 DATA GAP
                                ;14 TOLERANCE GAP
2$:     CLR      (R0)+           ;CLEAR DATA GAP, AND
        DEC      R5              ;TOLERANCE GAP
        BNE      2$              ;BRANCH IF NOT COMPLETE

        JSR      PC,#FILLEC      ;INSERT THE TWO ECC WORDS ON THE DISK
                                ;IN THE CORRECT PLACE

        ;*THESE ARE FOR ECC TEST ONLY

        MOV      #-1,#TSECC      ;THIS IS AN ECC TEST
        CLR      #POSIT1         ;CLEAR ERROR POSITION COUNTER
        MOV      #NCODE,#NCOUNT ;TEMPORARY N-CODE COUNTER
        MOV      #HARDER,#HADTMP  ;TEMPORARY HARD ERROR COUNTER
        CLR      #GECC1          ;ECC LOW ORDER TO BE GENERATED
        CLR      #GECC2          ;ECC HIGH ORDER TO BE GENERATED
        CLR      #DATENV         ;CLEAR DATA ENVELOPE CLOCK COUNT
        CLR      #ZCODE          ;CLEAR LEADING ZEROS CLOCK COUNT

        ;*THESE ARE TO SETUP FOR DISKLESS USE ONLY

        MOV      #FMT22,#CYL     ;16 BITS PER WORD
                                ;CYLINDER 0, FORMAT 16 BITS
        MOVB     #0,#SECOTR+1     ;TRACK 0
        MOVB     #0,#SECOTR      ;SECTOR 0
        MOV      #0,#KEY1        ;KEY1=0
```

6975	041624	012737	000000	053016	MOV	#0,@#KEY2	:KEY2=0
6976	041632	012737	000400	053070	MOV	#256.,@#DAWORD	:NO. OF DATA WORDS
6977	041640	005037	053020		CLR	@#X	:THIS IS A READ COMMAND
6978	041644	004537	047332		JSR	R5,@#CRC	:GO TO CALCULATE CRC
6979	041650	053010			CYL		
6980	041652	054710			WCRC		
6981							
6982							
6983							
6984							
6985							
6986							
6987	041654	004737	046116		JSR	PC,@#CLDISK	:SETUP GENERAL REGISTERS
6988	041660	012777	177374	153062	MOV	#-256.-4.,@#RHW	:256. DATA 4 HEADER WORDS
6989	041666	012777	016264	153056	MOV	#REINTO,@#RHB	:STARTING ADDRESS OF READ BUFFER
6990	041674	112746	000000		MOVB	#0,-(SP)	:IN LOWER BYTE GET SECTOR
6991	041700	112766	000000	000001	MOVB	#0,1(SP)	:GET TRACK IN HIGHER BYTE
6992	041706	012677	153050		MOV	(SP)+,@#RHDST	:TRACK/SECTOR IN RHDST
6993	041712	012777	010000	153046	MOV	#FMT22,@#RHO	:16 BITS PER WORD
6994							:ECC CORRECTION NOT INHIBIT
6995							:BECAUSE ECC IS NOT GOING
6996							:TO BE CHECKED
6997	041720	005077	153044		CLR	@#RHC	:CYLINDER 0
6998							
6999	041724	004737	046160		JSR	PC,@#CHECKT	:CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DON'T
7000	041730	104401	005123		TYPE	,CPHALT	:CANNOT CONTINUE TESTING IF ANY OF THE
7001							:ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
7002	041734	000000			HALT		:STOP THE TEST
7003							
7004	041736	013711	015176		MOV	@#REFOR,@#R1	:READ HEADER AND DATA=72
7005	041742	005037	015124		CLR	@#ERFLG	:CLEAR ERROR FLAG
7006	041746	004737	052650		JSR	PC,@#COMHD	:READ HEADER AND DATA
7007							:IF THERE ARE READ ERRORS THEN
7008							:ECC WILL NOT BE CHECKED
7009							
7010							
7011							:*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
7012							:*FROM THE 'COMHD' ROUTINE THAT MEANS SECTOR GAP,
7013							:*FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
7014							:*SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
7015							:*DETECTED
7016							:*HEADER AND DATA ARE TO BE CHECKED.
7017							:*IN CHECKING READ DATA THE WRITE FROM BUFFER
7018							:* 'WRFROM' IS FILLED WITH EXPECTED DATA AND
7019							:*COMPARISONS ARE MADE
7020							
7021	041752	005737	015124		TST	@#ERFLG	:ANY ERRORS ALREADY THERE
7022	041756	001102			BNE	TST65	:BRANCH IF YES
7023	041760	004737	045616		JSR	PC,@#PUTREG	:SAVE REGISTERS
7024	041764	005737	015034		TST	@#ER1	:NO ERRORS SHOULD BE SET
7025	041770	001401			BEQ	65	:BRANCH IF NO ERRORS SET
7026	041772	104032			ERROR	32	:32 BIT ECC REGISTER SHOULD BE ZERO
7027							:ONLY 11 OF THE 32 BITS CAN BE SEEN
7028							:IN THE PATTERN REGISTER
7029							:DCK SHOULD BE SET IN RHER1
7030	041774	013746	050620	65:	MOV	@#GECC1,-(SP)	:GET PATTERN REGISTER

7031	042000	042716	174000		BIC	#174000,(SP)	;KEEP ONLY 11 BITS
7032	042004	022637	015064		CMP	(SP)+,@#EC2	;COMPARE PATTERN REGISTER
7033	042010	001401			BEQ	7\$	;BRANCH IF GOOD
7034	042012	104032			ERROR	32	;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
7035							
7036							
7037							
7038							
7039							
7040							
7041							
7042	042014	012700	000020	7\$:	MOV	#16.,R0	;COUNTER
7043	042020	052777	000002	152750 8\$:	BIS	#MCLK,@RHMR	;SET CLOCK
7044	042026	042777	000002	152742	BIC	#MCLK,@RHMR	;CLEAR CLOCK
7045	042034	005300			DEC	R0	;COUNT
7046	042036	001370			BNE	8\$	;BRANCH IF 16 CLOCKS NOT DONE
7047	042040	004737	046350		JSR	PC,@#CHECKE	;CHECK THAT DVA,RDY,DPR,DRY - 1
7048	042044	104401	005123		TYPE	,CPHALT	;CANNOT CONTINUE IF THEY DON'T
7049	042050	000000			HALT		;STOP THE TEST AND RESTART PROGRAM
7050	042052	012700	015220		MOV	#WRFROM,R0	;GETTING READY TO FILL EXPECTED DATA
7051	042056	012720	010000		MOV	#0!FMT22,(R0)+	;CYLINDER 0
7052	042062	112746	000000		MOVB	#0,-(SP)	;IN LOWER BYTE GET SECTOR
7053	042066	112766	000000	000001	MOVB	#0,1(SP)	;GET TRACK IN HIGHER BYTE
7054	042074	012620			MOV	(SP)+,(R0)+	;GET TRACK/SECTOR IN BUFFER
7055	042076	012720	000000		MOV	#0,(R0)+	;KEY1 IN BUFFER
7056	042102	012720	000000		MOV	#0,(R0)+	;KEY2 IN BUFFER
7057	042106	012701	000400		MOV	#256.,R1	;DATA WORD COUNTER
7058	042112	012702	052525		MOV	#52525,R2	;DATA
7059	042116	010220		3\$:	MOV	R2,(R0)+	;DATA INTO BUFFER
7060	042120	005301			DEC	R1	;COUNT
7061	042122	001375			BNE	3\$	;BRANCH IF 256 NOT DONE
7062	042124	005037	015124		CLR	@#ERFLG\$	;CLEAR ERROR FLAG
7063	042130	004737	045616		JSR	PC,@#PUTREG	;SAVE REGISTERS
7064							
7065							
7066							
7067	042134	004037	047020		JSR	R0,@#COMPAR	;CHECK
7068	042140	015220			WRFROM		;GOOD BUFFER
7069	042142	016264			REINTO		;TEST BUFFER
7070	042144	000404			4+256.		;NUMBER OF WORDS CHECKED
7071	042146	042154			4\$		;RETURN POINT FOR ERROR HEADER
7072	042150	042160			5\$		;RETURN POINT FOR ERROR DATA
7073							
7074	042152	042164			TST65		;RETURN FOR GOOD COMPARISON
7075							
7076	042154	104004		4\$:	ERROR	4	;READ NEXT ERROR
7077	042156	000207			RTS	PC	;RETURN TO "COMPAR"
7078	042160	104005		5\$:	ERROR	5	;WORD NOS 1 TO 4 ARE
7079							;HEADER WORDS
7080							;5 TO 260 ARE DATA WORDS
7081	042162	000207			RTS	PC	;RETURN TO "COMPAR"

```
7082
7083
7084
7085
7086
7087
7088
7089
7090
7091
7092
7093
7094
7095 042164 000004
7096 042166 012706 001000
7097 042172 012737 000065 017330
7098
7099
7100
7101
7102 042200 012746 052525
7103 042204 012705 000400
7104 042210 012700 054726
7105 042214 011620
7106 042216 005305
7107 042220 001375
7108 042222 005726
7109 042224 022020
7110 042226 012705 000017
7111
7112 042232 005020
7113 042234 005305
7114 042236 001375
7115
7116
7117 042240 004737 051414
7118
7119
7120
7121
7122
7123 042244 012737 177777 015142
7124 042252 005037 050632
7125 042256 013737 050626 050630
7126 042264 013737 050634 050642
7127 042272 005037 050620
7128 042276 005037 050622
7129 042302 005037 050636
7130 042306 005037 050640
7131
7132
7133
7134
7135 042312 012737 010000 053010
7136
7137 042320 112737 000000 053013
```

```
*****
*TEST 65      READ ECC ENABLED 3B
*****
*      THIS IS AN ECC READ DATA TEST
*      ERROR CORRECTION IS ENABLED
*      A CORRECTABLE ERROR IS INSERTED IN BIT POSITION 4128
*      THIS IS THE LAST BIT OF THE ECC
*      GOOD DATA USED IS 256 WORDS OF 52525
*      COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
*      TRACK 0, SECTOR 0 KEYS 0  READ HEADER AND DATA
*****
TST65: 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      #52525,-(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)
CMP      (R0)+,(R0)+    ;JUMP OVER THE TWO ECC WORDS
MOV      #15., R5       ;1 DATA GAP
                          ;14 TOLERANCE GAP
2$: CLR   (R0)+          ;CLEAR DATA GAP, AND
DEC      R5             ;TOLERANCE GAP
BNE      2$            ;BRANCH IF NOT COMPLETE

JSR      PC,@#FILLEC    ;INSERT ECC IN PROPER PLACE ON DISK

;*THESE ARE FOR ECC TEST ONLY
MOV      #-1,@#TSECC    ;THIS IS AN ECC TEST
CLR      @#POSITI      ;CLEAR ERROR POSITION COUNTER
MOV      @#NCODE,@#NCOUNT ;TEMPORARY N-CODE COUNTER
MOV      @#HARDER,@#HADTMP ;TEMPORARY HARD ERROR COUNTER
CLR      @#GECC1        ;ECC LOW ORDER TO BE GENERATED
CLR      @#GECC2        ;ECC HIGH ORDER TO BE GENERATED
CLR      @#DATENV       ;CLEAR DATA ENVELOPE CLOCK COUNT
CLR      @#ZCODE        ;CLEAR LEADING ZEROS CLOCK COUNT

;*THESE ARE TO SETUP FOR DISKLESS USE ONLY
MOV      #FMT22,@#CYL   ;16 BITS PER WORD
                          ;CYLINDER 0, FORMAT 16 BITS
MOVB     #0,@#SECTR+1   ;TRACK 0
```

```

7138 042326 112737 000000 053012      MOVB    #0,@#SECOTR      ;SECTOR 0
7139 042334 012737 000000 053014      MOV     #0,@#KEY1       ;KEY1=0
7140 042342 012737 000000 053016      MOV     #0,@#KEY2       ;KEY2=0
7141 042350 012737 000400 053070      MOV     #256.,@#DAWORD  ;NO. OF DATA WORDS
7142 042356 005037 053020      CLR     @#X             ;THIS IS A READ COMMAND
7143 042362 004537 047332      JSR     RS,@#CRC        ;GO TO CALCULATE CRC
7144 042366 053010      CYL
7145 042370 054710      WCRC

7146
7147
7148      ;*THIS IS TO INSERT ERROR
7149      ;*THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
7150      ;*THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
7151      ;*THIS MOVE
7152      ;*THIS CHANGES THE LAST BIT OF THE ECC
7153
7154 042372 013746 055730      MOV     @#WECC2,-(SP)   ;GET LAST ECC
7155 042376 005116      COM     (SP)          ;INVERT ALL BITS OF WECC2
7156 042400 042716 077777      BIC     #^C100000,(SP) ;KEEP BIT 16
7157 042404 042737 100000 055730      BIC     #100000,@#WECC2 ;CLEAR BIT 16 IN ECC
7158 042412 052637 055730      BIS     (SP)+,@#WECC2  ;THIS WILL SET BIT 16 IF IT WAS 0
7159                                ;OR WILL SET NOTHING IF IT WAS A 1
7160
7161
7162 042416 012737 010026 042572      MOV     #4118.,@#8$    ;INSERT POSITION REG.
7163
7164
7165      ;*THESE ARE REGULAR SETUPS
7166
7167 042424 004737 046116      JSR     PC,@#CLDISK    ;SETUP GENERAL REGISTERS
7168 042430 012777 177374 152312      MOV     #-256.-4.,@#RHWC ;256. DATA 4 HEADER WORDS
7169 042436 012777 016264 152306      MOV     @#REINTO,@#RHBA ;STARTING ADDRESS OF READ BUFFER
7170 042444 112746 000000      MOVB    #0,-(SP)       ;IN LOWER BYTE GET SECTOR
7171 042450 112766 000000 000001      MOVB    #0,1(SP)       ;GET TRACK IN HIGHER BYTE
7172 042456 012677 152300      MOV     (SP)+,@#RHDST  ;TRACK/SECTOR IN RHDST
7173 042462 012777 010000 152276      MOV     @#FMT22,@#RHOF ;16 BITS PER WORD
7174                                ;ECC CORRECTION NOT INHIBIT
7175                                ;BECAUSE ECC IS NOT GOING
7176                                ;TO BE CHECKED
7177 042470 005077 152274      CLR     @#RHCA        ;CYLINDER 0
7178
7179 042474 004737 046160      JSR     PC,@#CHECKT    ;CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DON'T
7180 042500 104401 005123      TYPE    ,CPHALT      ;CANNOT CONTINUE TESTING IF ANY OF THE
7181                                ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
7182 042504 000000      HALT      ;STOP THE TEST
7183
7184 042506 013711 015176      MOV     @#REFOR,@#R1   ;READ HEADER AND DATA=72
7185 042512 005037 015124      CLR     @#ERFLG$       ;CLEAR ERROR FLAG
7186 042516 004737 052650      JSR     PC,@#COMHD     ;READ HEADER AND DATA
7187                                ;IF THERE ARE READ ERRORS THEN
7188                                ;ECC WILL NOT BE CHECKED
7189
7190
7191      ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
7192      ;*FROM THE 'COMHD' ROUTINE THAT MEANS SECTOR GAP.
7193      ;*FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND

```

```

7194      ;*SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
7195      ;*DETECTED
7196      ;*HEADER AND DATA ARE TO BE CHECKED.
7197      ;*IN CHECKING READ DATA THE WRITE FROM BUFFER
7198      ;*'WRFROM' IS FILLED WITH EXPECTED DATA AND
7199      ;*COMPARISONS ARE MADE
7200
7201 042522 005737 015124      TST      @#ERFLG$      ;ANY ERRORS ALREADY THERE
7202 042526 001074      BNE      TST66      ;BRANCH IF YES
7203 042530 004737 045616      JSR      PC,@#PUTREG      ;SAVE REGISTERS
7204 042534 022737 100000 015034      CMP      #DCK,@#ER1      ;ONLY DATA CHECK ERROR SHOULD BE SET
7205 042542 001401      BEQ      6$      ;BRANCH IF YES
7206 042544 104032      ERROR      32      ;32 BIT ECC REGISTER SHOULD BE NON
7207      ;ZERO
7208      ;ONLY 11 OF THE 32 BITS CAN BE SEEN
7209      ;IN THE PATERN REGISTER
7210      ;DCK SHOULD BE SET IN RHER1
7211 042546 013746 050620      6$:      MOV      @#GECC1,-(SP)      ;GET PATTERN REGISTER
7212 042552 042716 174000      BIC      #174000,(SP)      ;KEEP ONLY 11 BITS
7213 042556 022637 015064      CMP      (SP)+,@#EC2      ;COMPARE PATTERN REGISTER
7214 042562 001401      BEQ      7$      ;BRANCH IF GOOD
7215 042564 104032      ERROR      32      ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
7216
7217 042566 004037 051242      7$:      JSR      R0,@#ECORR      ;GO TO ECC CORRECTION PROCESS
7218 042572 010026      8$:      4118.      ;EXPECTED POSITION REG. WHEN CORRECTION
7219      ;IS COMPLETE
7220
7221
7222
7223 042574 004737 046350      JSR      PC,@#CHECKE      ;CHECK THAT DVA,RDY,DPR,DRY = 1
7224 042600 104401 005123      TYPE      ,CPHALT      ;CANNOT CONTINUE IF THEY DON'T
7225 042604 000000      HALT      ;STOP THE TEST AND RESTART PROGRAM
7226 042606 012700 015220      MOV      #WRFROM,R0      ;GETTING READY TO FILL EXPECTED DATA
7227 042612 012720 010000      MOV      #0:FM122,(R0)+      ;CYLINDER 0
7228 042616 112746 000000      MOV      #0,-(SP)      ;IN LOWER BYTE GET SECTOR
7229 042622 112766 000000 000001      MOV      #0,1(SP)      ;GET TRACK IN HIGHER BYTE
7230 042630 012620      MOV      (SP)+,(R0)+      ;GET TRACK/SECTOR IN BUFFER
7231 042632 012720 000000      MOV      #0,(R0)+      ;KEY1 IN BUFFER
7232 042636 012720 000000      MOV      #0,(R0)+      ;KEY2 IN BUFFER
7233 042642 012701 000400      MOV      #256.,R1      ;DATA WORD COUNTER
7234 042646 012702 052525      MOV      #52525,R2      ;DATA
7235 042652 010220      3$:      MOV      R2,(R0)+      ;DATA INTO BUFFER
7236 042654 005301      DEC      R1      ;COUNT
7237 042656 001375      BNE      3$      ;BRANCH IF 256 NOT DONE
7238
7239      ;*ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
7240      ;*NOW THE INSERTED ERROR WILL BE PUT IN
7241      ;*BUT INSERTED ERROR IS IN ECC SO DATA IS NOT WRONG
7242
7243
7244 042660 004737 045616      JSR      PC,@#PUTREG      ;SAVE REGISTERS
7245 042664 005037 015124      CLR      @#ERFLG$      ;CLEAR ERROR FLAG
7246
7247
7248      ;*NOW READ DATA BUFFER WILL BE CHECKED
7249

```

CZRJHDO,RP04/5/6 DSKLS CTRLR2 MACY11 30A(1052) 24-MAY-79 15:07 I 13
CZRJHD.P12 01-MAR-79 09:10 T65 READ ECC ENAB'D 3B PAGE 166

SEQ 0164

7250	042670	004037	047020	JSR	RG,2#COMPAR	;CHECK
7251	042674	015220		WRFROM		;GOOD BUFFER
7252	042676	016264		REINT0		;TEST BUFFER
7253	042700	000404		4+256.		;NUMBER OF WORDS CHECKED
7254	042702	042710		4\$		;RETURN POINT FOR ERROR HEADER
7255	042704	042714		5\$		;RETURN POINT FOR ERROR DATA
7256						
7257	042706	042720		TST66		;RETURN FOR GOOD COMPARISON
7258						
7259	042710	104004	4\$:	ERROR	4	;READ NEXT ERROR
7260	042712	000207		RTS	PC	;RETURN TO "COMPAR"
7261	042714	104005	5\$:	ERROR	5	;WORD NOS 1 TO 4 ARE
7262						;HEADER WORDS
7263						;5 TO 260 ARE DATA WORDS
7264	042716	000207		RTS	PC	;RETURN TO "COMPAR"
7265						

```

7266
7267
7268
7269
7270
7271
7272
7273
7274
7275
7276
7277
7278
7279 042720 000004
7280 042722 012706 001000
7281 042726 012737 000066 017330
7282
7283
7284
7285
7286
7287 042734 012746 052525
7288 042740 012705 000400
7289 042744 012700 054726
7290 042750 011620
7291 042752 005305
7292 042754 001375
7293 042756 005726
7294 042760 022020
7295 042762 012705 000017
7296
7297 042766 005020
7298 042770 005305
7299 042772 001375
7300
7301
7302 042774 004737 051414
7303
7304
7305
7306
7307 043000 012737 177777 015142
7308 043006 005037 050632
7309 043012 013737 050626 050630
7310 043020 013737 050634 050642
7311 043026 005037 050620
7312 043032 005037 050622
7313 043036 005037 050636
7314 043042 005037 050640
7315
7316
7317
7318
7319 043046 012737 010000 053010
7320
7321 043054 112737 000000 053013
  
```

```

*****
; *TEST 66      READ ECC ENABLED 3C
; *
; *  THIS IS AN ECC READ DATA TEST
; *  ERROR CORRECTION IS ENABLED
; *  A NON CORRECTABLE ERROR IS INSERTED IN BIT POSITION 296 THRU 308
; *  THIS IS IN WORD NUMBER 19 AND 20
; *  GOOD DATA USED IS 256 WORDS OF 52525
; *  COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
; *  TRACK 0, SECTOR 0 KEYS 0  READ HEADER AND DATA
*****
TST66: SCOPE
MOV     #STACK,SP      ;RESET STACK
MOV     #TINO,@#TSTNM  ;THIS SAVES TEST NUMBER

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

MOV     #52525,-(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)
CMP     (R0)+,(R0)+     ;JUMP OVER THE TWO ECC WORDS
MOV     #15., R5        ;1 DATA GAP
                     ;14 TOLERANCE GAP
2$: CLR  (R0)+          ;CLEAR DATA GAP, AND
DEC     R5              ;TOLERANCE GAP
BNE     2$              ;BRANCH IF NOT COMPLETE

JSR     PC,@#FILLEC     ;INSERT THE TWO ECC WORDS ON THE DISK
                     ;IN THE CORRECT PLACE

; *THESE ARE FOR ECC TEST ONLY

MOV     #-1,@#TSECC     ;THIS IS AN ECC TEST
CLR     @#POSITI        ;CLEAR ERROR POSITION COUNTER
MOV     @#NCODE,@#NCOUNT ;TEMPORARY N-CODE COUNTER
MOV     @#HARDER,@#HADTMP ;TEMPORARY HARD ERROR COUNTER
CLR     @#GECC1         ;ECC LOW ORDER TO BE GENERATED
CLR     @#GECC2         ;ECC HIGH ORDER TO BE GENERATED
CLR     @#DATENV        ;CLEAR DATA ENVELOPE CLOCK COUNT
CLR     @#ZCODE         ;CLEAR LEADING ZEROS CLOCK COUNT

; *THESE ARE TO SETUP FOR DISKLESS USE ONLY

MOV     #FMT22,@#CYL    ;16 BITS PER WORD
                     ;CYLINDER 0, FORMAT 16 BITS
MOVB    #0,@#SECOTR+1   ;TRACK 0
  
```

```

7322 043062 112737 000000 053012      MOVB    #0,@#SECOTR      ;SECTOR 0
7323 043070 012737 000000 053014      MOV     #0,@#KEY1       ;KEY1=0
7324 043076 012737 000000 053016      MOV     #0,@#KEY2       ;KEY2=0
7325 043104 012737 000400 053070      MOV     #256.,@#DAWORD  ;NO. OF DATA WORDS
7326 043112 005037 053020              CLR     @#X              ;THIS IS A READ COMMAND
7327 043116 004537 047332              JSR     R5,@#CRC        ;GO TO CALCULATE CRC
7328 043122 053010              CYL
7329 043124 054710              WCRC
7330
7331
7332      ;*THIS IS TO INSERT ERROR
7333      ;*THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
7334      ;*THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
7335      ;*THIS MOVE
7336
7337 043126 012737 152652 054772      MOV     #152652,@#DISK+44;INSERT ERROR IN POSITION 296 THRU 304
7338                                ;IN WORD NUMBER 19
7339 043134 012737 052532 054774      MOV     #52532,@#DISK+46;INSERT ERROR IN POSITION 305 THRU 308
7340                                ;IN WORD NUMBER 20
7341 043142 012737 010040 043316      MOV     #4128.,@#8$     ;INSERT POSITION REG.
7342
7343
7344      ;*THESE ARE REGULAR SETUPS
7345
7346 043150 004737 046116              JSR     PC,@#CLDISK     ;SETUP GENERAL REGISTERS
7347 043154 012777 177374 151566      MOV     #-256.-4.,@#RHWC ;256. DATA 4 HEADER WORDS
7348 043162 012777 016264 151562      MOV     #REINTO,@#RHBA  ;STARTING ADDRESS OF READ BUFFER
7349 043170 112746 000000              MOVB    #0,-(SP)        ;IN LOWER BYTE GET SECTOR
7350 043174 112766 000000 000001      MOVB    #0,1(SP)       ;GET TRACK IN HIGHER BYTE
7351 043202 012677 151554              MOV     (SP)+, @#RHDST   ;TRACK/SECTOR IN RHDST
7352 043206 012777 010000 151552      MOV     #FMT22,@#RHOF   ;16 BITS PER WORD
7353                                ;ECC CORRECTION NOT INHIBIT
7354                                ;BECAUSE ECC IS NOT GOING
7355                                ;TO BE CHECKED
7356 043214 005077 151550              CLR     @#RHCA          ;CYLINDER 0
7357 043220 004737 046160              JSR     PC,@#CHECKT     ;CHECK DVA,RDY,DPR,DYR = 1 AND OTHERS DON'T
7358 043224 104401 005123              TYPE    ,CPHALT      ;CANNOT CONTINUE TESTING IF ANY OF THE
7359                                ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
7360 043230 000000              HALT
7361 043232 013711 015176              MOV     @#REFOR,@#R1   ;READ HEADER AND DATA=72
7362 043236 005037 015124              CLR     @#ERFLG$      ;CLEAR ERROR FLAG
7363 043242 004737 052650              JSR     PC,@#COMHD     ;READ HEADER AND DATA
7364                                ;IF THERE ARE READ ERRORS THEN
7365                                ;ECC WILL NOT BE CHECKED
7366
7367
7368      ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
7369      ;*FROM THE 'COMHD' ROUTINE THAT MEANS SECTOR GAP,
7370      ;*FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
7371      ;*SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
7372      ;*DETECTED
7373      ;*HEADER AND DATA ARE TO BE CHECKED.
7374      ;*IN CHECKING READ DATA THE WRITE FROM BUFFER
7375      ;*'WRFROM' IS FILLED WITH EXPECTED DATA AND
7376      ;*COMPARISONS ARE MADE
7377

```

7378	043246	005737	015124		TST	2#ERFLG\$	;ANY ERRORS ALREADY THERE
7379	043252	001111			BNE	TST67	;BRANCH IF YES
7380	043254	004737	045616		JSR	PC,2#PUTREG	;SAVE REGISTERS
7381	043260	022737	100000	015034	CMP	#DCK,2#ER1	;ONLY DATA CHECK ERROR SHOULD BE SET
7382	043266	001401			BEQ	6\$	;BRANCH IF YES
7383	043270	104032			ERROR	32	;32 BIT ECC REGISTER SHOULD BE NON
7384							;ZERO
7385							;ONLY 11 OF THE 32 BITS CAN BE SEEN
7386							;IN THE PATERN REGISTER
7387							;DCK SHOULD BE SET IN RHER1
7388	043272	013746	050620		6\$: MOV	2#GECC1,-(SP)	;GET PATTERN REGISTER
7389	043276	042716	174000		BIC	#174000,(SP)	;KEEP ONLY 11 BITS
7390	043302	022637	015064		CMP	(SP)+,2#EC2	;COMPARE PATTERN REGISTER
7391	043306	001401			BEQ	7\$	;BRANCH IF GOOD
7392	043310	104032			ERROR	32	;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
7393							
7394	043312	004037	051242		7\$: JSR	RO,2#ECORR	;GO TO ECC CORRECTION PROCESS
7395	043316	000000			8\$: .WORD		;EXPECTED POSITION REG. WHEN CORRECTION
7396							;IS COMPLETE
7397							
7398							
7399							
7400	043320	004737	045616		JSR	PC,2#PUTREG	;SAVE REGISTERS
7401	043324	022737	100100	015034	CMP	#DCK!ECH,2#ER1	;WITH ERRORS INSERTED IN BIT POSITION 21
7402							;THRU 32 HARD ERROR BIT SHOULD SET
7403	043332	001401			BEQ	9\$	;BRANCH IF GOOD
7404	043334	104036			ERROR	36	;WITH ERROR INSERTED IN BIT POSITION 21 THRU
7405							;32 HCE SHOULD SET
7406							
7407							
7408							
7409	043336				9\$: JSR	PC,2#CHECKE	;CHECK THAT DVA,RDY,DPR,DRY = 1
7410	043336	004737	046350		TYPE	,CPHALT	;CANNOT CONTINUE IF THEY DON'T
7411	043342	104401	005123		HALT		;STOP THE TEST AND RESTART PROGRAM
7412	043346	000000			MOV	#WRFROM,RO	;GETTING READY TO FILL EXPECTED DATA
7413	043350	012700	015220		MOV	#0!FMT22,(RO)+	;CYLINDER 0
7414	043354	012720	010000		MOV	#0,-(SP)	;IN LOWER BYTE GET SECTOR
7415	043360	112746	000000		MOVB	#0,1(SP)	;GET TRACK IN HIGHER BYTE
7416	043364	112766	000000	000001	MOVB	(SP)+,(RO)+	;GET TRACK/SECTOR IN BUFFER
7417	043372	012620			MOV	#0,(RO)+	;KEY1 IN BUFFER
7418	043374	012720	000000		MOV	#0,(RO)+	;KEY2 IN BUFFER
7419	043400	012720	000000		MOV	#256,R1	;DATA WORD COUNTER
7420	043404	012701	000400		MOV	#52525,R2	;DATA
7421	043410	012702	052525		MOV	R2,(RO)+	;DATA INTO BUFFER
7422	043414	010220			3\$: DEC	R1	;COUNT
7423	043416	005301			BNE	3\$	;BRANCH IF 256 NOT DONE
7424	043420	001375					
7425							
7426							;*ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
7427							;*NOW THE INSERTED ERROR WILL BE PUT IN
7428							
7429	043422	012737	152652	015274	MOV	#152652,2#WRFROM+54;	INSERT ERROR IN POSITION 296 THRU 304
7430							;IN WORD NUMBER 19 IN DATA
7431	043430	012737	052532	015276	MOV	#52532,2#WRFROM+56;	INSERT ERROR IN POSITION 305 THRU 308
7432							;IN WORD NUMBER 20 IN DATA
7433							

```
7434 043436 004737 045616 JSR PC,@#PUTREG ;SAVE REGISTERS
7435 043442 005037 015124 CLR @#ERFLG$ ;CLEAR ERROR FLAG
7436
7437
7438 ;*NOW READ DATA BUFFER WILL BE CHECKED
7439
7440 043446 004037 047020 JSR RO,@#COMPAR ;CHECK
7441 043452 015220 WRFROM ;GOOD BUFFER
7442 043454 016264 REINTO ;TEST BUFFER
7443 043456 000404 4+256. ;NUMBER OF WORDS CHECKED
7444 043460 043466 4$ ;RETURN POINT FOR ERROR HEADER
7445 043462 043472 5$ ;RETURN POINT FOR ERROR DATA
7446
7447 043464 043476 TST67 ;RETURN FOR GOOD COMPARISON
7448
7449 043466 104004 4$: ERROR 4 ;READ NEXT ERROR
7450 043470 000207 RTS PC ;RETURN TO "COMPAR"
7451 043472 104005 5$: ERROR 5 ;WORD NOS 1 TO 4 ARE
7452 ;HEADER WORDS
7453 ;5 TO 260 ARE DATA WORDS
7454 043474 000207 RTS PC ;RETURN TO "COMPAR"
7455
```

```

7456
7457
7458
7459
7460
7461
7462
7463
7464
7465
7466
7467
7468
7469 043476 000004
7470 043500 012706 001000
7471
7472 043504 012737 000067 017330
7473
7474
7475
7476
7477
7478 043512 012746 052525
7479 043516 012705 000400
7480 043522 012700 054726
7481 043526 011620
7482 043530 005305
7483 043532 001375
7484 043534 005726
7485 043536 022020
7486 043540 012705 000017
7487
7488 043544 005020
7489 043546 005305
7490 043550 001375
7491
7492
7493 043552 004737 051414
7494
7495
7496
7497
7498 043556 012737 177777 015142
7499 043564 005037 050632
7500 043570 013737 050626 050630
7501 043576 013737 050634 050642
7502 043604 005037 050620
7503 043610 005037 050622
7504 043614 005037 050636
7505 043620 005037 050640
7506
7507
7508
7509
7510 043624 012737 010000 053010
7511

```

```

MOVW    #0,2#SECOTR+1      ;TRACK 0
MOVW    #0,2#SECOTR        ;SECTOR 0
MOVW    #0,2#KEY1          ;KEY1=0
MOVW    #0,2#KEY2          ;KEY2=0
MOVW    #256.,2#DAWORD     ;NO. OF DATA WORDS
CLR      2#X                ;THIS IS A READ COMMAND
JSR      R5,2#CRC           ;GO TO CALCULATE CRC
CYL
WCRCL

```

```
;*THIS IS TO INSERT ERROR
;*THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
;*THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
;*THIS MOVE
```

```
MOV    #152525, @NDISK+2    ;FORCE ERROR ON BIT NUMBER 32
MOV    #152525, @NDISK+<255.*2>;FORCE ERROR IN BIT 4096
MOV    #4128., @#8$        ;INSERT POSITION REG.
```

: * THESE ARE REGULAR SETUPS

```

JSR    PC,@#CLDISK      ;SETUP GENERAL REGISTERS
MOV     #-256.-4.,@RHWC  ;256. DATA 4 HEADER WORDS
MOV     #REINT0,@RHBA    ;STARTING ADDRESS OF READ BUFFER
MOVB    #0,-(SP)         ;IN LOWER BYTE GET SECTOR
MOVB    #0,1(SP)         ;GET TRACK IN HIGHER BYTE
MOV     (SP)+, @RHDST    ;TRACK/SECTOR IN RHDST
MOV     #FMT22,@RHOF     ;16 BITS PER WORD
                        ;ECC CORRECTION NOT INHIBIT
                        ;BECAUSE ECC IS NOT GOING
                        ;TO BE CHECKED
CLR     @RHCA            ;CYLINDER 0

JSR     PC,@#CHECKT      ;CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DON'T
TYPE    ,CPHALT          ;CANNOT CONTINUE TESTING IF ANY OF THE
                        ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
HALT                                ;STOP THE TEST

```

```

MOV    @#REFOR,@R1      ;READ HEADER AND DATA=72
CLR    @#ERFLG$         ;CLEAR ERROR FLAG
JSR     PC,@#COMHD       ;READ HEADER AND DATA
                          ;IF THERE ARE READ ERRORS THEN
                          ;ECC WILL NOT BE CHECKED

```

```
; *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 SYNCs WERE CORRECTLY
; *DETECTED
; *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
```

```
7568
7569 044024 005737 015124      TST    @#ERFLG$      ;ANY ERRORS ALREADY THERE
7570 044030 001111              BNE     TST70             ;      BRANCH IF YES
7571 044032 004737 045616      JSR     PC,@#PUTREG    ;SAVE REGISTERS
7572 044036 022737 100000 015034  CMP     #DCK,@#ER1    ;ONLY DATA CHECK ERROR SHOULD BE SET
7573 044044 001401              BEQ     6$              ;BRANCH IF YES
7574 044046 104032              ERROR  32              ;32 BIT ECC REGISTR LR SHOULD BE NON
7575                                ;ZERO
7576                                ;ONLY 11 OF THE 32 BITS CAN BE SEEN
7577                                ;IN THE PATERN REGISTER
7578                                ;DCK SHOULD BE SET IN RHER1
7579 044050 013746 050620      6$:  MOV     @#GECC1,-(SP)    ;GET PATTERN REGISTER
7580 044054 042716 174000      BIC     #174000,(SP)    ;KEEP ONLY 11 BITS
7581 044060 022637 015064      CMP     (SP)+,@#EC2    ;COMPARE PATTERN REGISTER
7582 044064 001401              BEQ     7$              ;BRANCH IF GOOD
7583 044066 104032              ERROR  32              ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
7584
7585 044070 004037 051242      7$:  JSR     R0,@#ECORR    ;GO TO ECC CORRECTION PROCESS
7586 044074 000000      8$:  .WORD    ;EXPECTED POSITION REG. WHEN CORRECTION
7587                                ;IS COMPLETE
7588
7589
7590
7591 044076 004737 045616      JSR     PC,@#PUTREG    ;SAVE REGISTERS
7592 044102 022737 100100 015034  CMP     #DCK.ECH,@#ER1 ;WITH ERRORS INSERTED IN BIT POSITION 32
7593                                ;AND 4096 HARD ERROR BIT SHOULD SET
7594 044110 001401              BEQ     9$              ;BRANCH IF GOOD
7595 044112 104036              ERROR  36              ;WITH ERROR INSERTED IN BIT POSITION 21 THRU
7596                                ;32 HCE SHOULD SET
7597
7598
7599
7600
7601 044114      9$:
7602 044114 004737 046350      JSR     PC,@#CHECKE    ;CHECK THAT DVA,RDY,DPR,DRY = 1
7603 044120 104401 005123      TYPE    ,CPHALT        ;CANNOT CONTINUE IF THEY DON'T
7604 044124 000000      HALT                    ;STOP THE TEST AND RESTART PROGRAM
7605 044126 012700 015220      MOV     #WRFROM,R0      ;GETTING READY TO FILL EXPECTED DATA
7606 044132 012720 010000      MOV     #0!FMT22,(R0)+    ;CYLINDER 0
7607 044136 112746 000000      MOV     #0,-(SP)         ;IN LOWER BYTE GET SECTOR
7608 044142 112766 000000 000001  MOV     #0,1(SP)      ;GET TRACK IN HIGHER BYTE
7609 044150 012620      MOV     (SP)+,(R0)+    ;GET TRACK/SECTOR IN BUFFER
7610 044152 012720 000000      MOV     #0,(R0)+    ;KEY1 IN BUFFER
7611 044156 012720 000000      MOV     #0,(R0)+    ;KEY2 IN BUFFER
7612 044162 012701 000400      MOV     #256,R1        ;DATA WORD COUNTER
7613 044166 012702 052525      MOV     #52525,R2      ;DATA
7614 044172 010220      3$:  MOV     R2,(R0)+    ;DATA INTO BUFFER
7615 044174 005301      DEC     R1              ;COUNT
7616 044176 001375      BNE     3$              ;BRANCH IF 256 NOT DONE
7617
7618      ;*ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
7619      ;*NOW THE INSERTED ERROR WILL BE PUT IN
7620
7621 044200 012737 152525 015232  MOV     #152525,@#WRFROM+<5*2> ;INSERTED ERROR IN BIT 32
7622 044206 012737 152525 016226  MOV     #152525,@#WRFROM+<259*2> ;INSERT ERROR IN BIT 4096
7623
```

```
7624 044214 005037 015124 CLR @#ERFLG$ ;CLEAR ERROR FLAG
7625 044220 004737 045616 JSR PC,@#PUTREG ;SAVE REGISTERS
7626
7627 ;*NOW READ DATA BUFFER WILL BE CHECKED
7628
7629 044224 004037 047020 JSR RO,@#COMPAR ;CHECK
7630 044230 015220 WRFROM ;GOOD BUFFER (CHANGED)
7631 044232 016264 REINTO ;TEST BUFFER
7632 044234 000404 4+256. ;NUMBER OF WORDS CHECKED
7633 044236 044244 4$ ;RETURN POINT FOR ERROR HEADER
7634 044240 044250 5$ ;RETURN POINT FOR ERROR DATA
7635
7636 044242 044254 TST70 ;RETURN FOR GOOD COMPARISON
7637
7638 044244 104004 4$: ERROR 4 ;READ NEXT ERROR
7639 044246 000207 RTS PC ;RETURN TO 'COMPAR'
7640 044250 104005 5$: ERROR 5 ;WORD NOS 1 TO 4 ARE
7641 ;HEADER WORDS
7642 ;5 TO 260 ARE DATA WORDS
7643 044252 000207 RTS PC ;RETURN TO 'COMPAR'
```

```

7644                                     .SBTTL   CURSORY INTERRUPT LOGIC TESTS
7645
7646
7647
7648                                     ;*****
7649                                     ;*TEST 70          PROGRAM INTERRUPT
7650
7651                                     ;*      PROGRAM INTERRUPT IS TESTED BY SETTING RDY AND IE
7652                                     ;*      IN RHCS1 AT THE SAME TIME
7653                                     ;*      THIS SHOULD INTERRUPT THROUGH LOCATION 254
7654                                     ;*      THE PROCESSOR PRIORITY IS SET TO 4
7655                                     ;*****
7656    044254    000004               TST70:  SCOPE
7657
7658    044256    012737    000070    017330           MOV     #TTNO,@#TSTNM       ;THIS SAVES TEST NUMBER
7659    044264    012706    001000                       MOV     #STACK,SP         ;RESET STACK
7660
7661    044270    004737    046116                       JSR      PC,@#CLDISK        ;CLEAR DISK
7662    044274    013700    014744                       MOV     @#RPVEC,R0         ;GET VECTOR ADDRESS
7663    044300    012720    044346                       MOV     #RPTRP1,(R0)+      ;SET INTERRUPT VECTOR
7664    044304    012710    000340                       MOV     #340,(R0)         ;SET SERVICE ROUTINE PRIORITY
7665    044310    012737    000200    177776           MOV     #200,PS          ;SET PROCESSOR PRIORITY
7666    044316    012711    000300    177776           MOV     #RDY!IE,@R1       ;RDY, IE IN RHSC1 SHOULD CAUSE INTERRUPT
7667    044322    013737    046474    001200           MOV     @#TIMCNT,@#$TMP1  ;COUNTER
7668    044330    005337    001200           1$: DEC     @#$TMP1             ;WAIT FOR INTERRUPT
7669    044334    001375                                BNE     1$              ;BRANCH IF NOT ZERO
7670                                ;BEFORE THIS IS ZERO INTERRUPT SHOULD
7671                                ;OCCUR
7672    044336    004737    045616           JSR      PC,@#PUTREG      ;SAVE REGISTERS
7673    044342    104021                                ERROR    21              ;INTERRUPT DID NOT OCCUR
7674
7675    044344    000410                                BR       TST71            ;      BRANCH TO NEXT TEST
7676
7677    044346    022626           RPTRP1: CMP     (SP)+,(SP)+      ;RESTORE STACK
7678    044350    004737    045616           JSR      PC,@#PUTREG      ;SAVE REGISTERS
7679    044354    022737    004200    015032           CMP     #DVA!RDY,@#CS1  ;'IE' SHOULD BE LOW
7680    044362    001401                                BEQ      TST71            ;      BRANCH IF GOOD
7681    044364    104021                                ERROR    21              ;INTERRUPT OCCURRED BUT
7682                                ;'IE' FAILED TO RESET

```

```
7683
7684
7685 .....
7686 : * TEST 71          INTERRUPT AT PROCESSOR AND DISK PRIORITY SAME
7687 :
7688 : *      PROCESSOR PRIORITY IS SET AT 5 (SAME AS THE DISK)
7689 : *      IE AND RDY IS SET. THIS SHOULD NOT INTERRUPT
7690 : .....
7691 044366 000004      TST71: SCOPE
7692
7693 044370 012737 000071 017330      MOV      #TTNO,@#TSTNM      ;THIS SAVES TEST NUMBER
7694 044376 012706 001000      MOV      #STACK,SP      ;RESET STACK
7695
7696 044402 004737 046116      JSR      PC,@#CLDISK      ;CLEAR DISK
7697 044406 013700 014744      MOV      @#RPVEC,R0      ;GET VECTOR ADDRESS
7698 044412 012720 044452      MOV      #RPTRP2,(R0)+      ;SET INTERRUPT VECTOR
7699 044416 012710 000340      MOV      #340,(R0)      ;SET SERVICE ROUTINE PRIORITY
7700 044422 012737 000240 177776      MOV      #240,PS      ;SET PROCESSOR PRIORITY
7701 044430 012711 000300      MOV      #RDY!IE,@R1      ;RDY, IE IN RHSC1 SHOULD CAUSE INTERRUPT
7702 044434 013737 046474 001200      MOV      @#TIMCNT,@#STMP1 ;COUNTER
7703 044442 005337 001200      1$: DEC      @#STMP1      ;WAIT FOR INTERRUPT
7704 044446 001375      BNE      1$      ;BRANCH IF NOT ZERO
7705      ;BEFORE THIS IS ZERO INTERRUPT SHOULD
7706      ;OCCUR
7707 044450 000404      BR      TST72      ; NO INTERRUPT SO BRANCH
7708
7709 044452 022626      RPTRP2: CMP      (SP)+,(SP)+      ;RESTORE STACK
7710 044454 004737 045616      JSR      PC,@#PUTREG      ;SAVE REGISTERS
7711 044460 104021      ERROR      21      ;INTERRUPT OCCURRED WITH
7712      ;PROCESSOR STATUS SAME
7713      ;AS DISK
7714
7715
7716
7717
```

CZRJHDO,RP04/5/6 DSKLS CTRLR2
CZRJHD.P12 01-MAR-79 09:10

MACY11 30A(1052) 24-MAY-79 15:07 PAGE 177
171 INTERRUPT AT PROCESSOR AND DISK PRIORITY SAME

SEQ 0175

```
7718
7719
7720
7721
7722
7723
7724
7725
7726
7727
7728 044462 000004
7729 044464 012737 000001 001212
7730 044472 004737 046116
7731
7732 044476 012737 000000 177776
7733 044504 104401 044512
7734 044510 000425
7735
7736 044564
7737 044564 013746 015112
7738 044570 104405
7739 044572 104401 044600
7740 044576 000402
7741
7742 044604
7743 044604 013746 001112
7744 044610 104405
7745 044612 005037 001112
7746 044616 005037 001102
7747 044622 005737 015120
7748 044626 001413
7749
7750 044630 005237 001100
7751 044634 104401 045017
7752 044640 013746 001100
7753 044644 104405
7754 044646 104401 045014
7755 044652 000137 022770
7756
7757 044656 005337 015114
7758 044662 001413
7759 044664 013700 015112
7760 044670 012701 015072
7761 044674 022100
7762 044676 001401
7763 044700 000775
7764 044702 011137 015112
7765 044706 000137 022770

*****
: * TEST 72      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
*****
1ST72: SCOPE
      MOV      #1,$TIMES      ;;DO 1 ITERATION
      JSR      PC,@#CLDISK
      MOV      #0,PS          ;REINSTATE PS TO 0
      TYPE     ,65$           ;;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      @#SFRTTL        ;CLEAR TOTAL NUMBER OF ERRORS
      CLR      @#STSTNM        ;CLEAR TEST NUMBER
      TST      @#SELECT        ;STARTING FROM 200 ?
      BEQ      3$             ;TEST NEXT DRIVE IF SO
      INC      @#$PASS         ;INCREASE PASS COUNT
      TYPE     ,SENDMG         ;TYPE END PASS #
      MCV      @#$PASS,-(SP)
      TYPDS
      TYPE     ,SENULL
      JMP      @#TST5          ;CONTINUE TESTING THIS DRIVE ----->
      3$: DEC      @#NOUNITS    ;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          ;TEST NEXT DRIVE ----->
```

```

7766
7767
7768 .SBTTL
7769 .SBTTL ***SUBROUTINES***
7770 .SBTTL
7771
7772
7773
7774 .SBTTL END OF PASS ROUTINE
7775
7776 ;*****
7777 ;*INCREMENT THE PASS NUMBER ($PASS)
7778 ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
7779 ;*IF THERES A MONITOR GO TO IT
7780 ;*IF THERE ISN'T JUMP TO TST1
7781
7782 $EOP:
7783 044712 000004 SCOPE
7784 044714 005037 001102 CLR $TSTNM ;;ZERO THE TEST NUMBER
7785 044720 005037 001212 CLR $TIMES ;;ZERO THE NUMBER OF ITERATIONS
7786 044724 005237 001100 INC $PASS ;;INCREMENT THE PASS NUMBER
7787 044730 042737 100000 001100 BIC #100000,$PASS ;;DON'T ALLOW A NEG. NUMBER
7788 044736 005327 DEC (PC)+ ;;LOOP?
7789 044740 000001 $EOPCT: .WORD 1
7790 044742 003022 BGT $DOAGN ;;YES
7791 044744 012737 MOV (PC)+,@(PC)+ ;;RESTORE COUNTER
7792 044746 000001 $ENDCT: .WORD 1
7793 044750 044740 $EOPCT
7794 044752 104401 045017 TYPE , $ENDMG ;;TYPE "END PASS #"
7795 044756 013746 001100 MOV $PASS,-(SP) ;;SAVE $PASS FOR TYPEOUT
7796 044762 104405 TYPDS ;;GO TYPE--DECIMAL ASCII WITH SIGN
7797 044764 104401 045014 TYPE , $ENULL ;;TYPE A NULL CHARACTER
7798 044770 013700 000042 $GET42: MOV @#42,R0 ;;GET MONITOR ADDRESS
7799 044774 001405 BEQ $DOAGN ;;BRANCH IF NO MONITOR
7800 044776 000005 RESET ;;CLEAR THE WORLD
7801 045000 004710 $ENDAD: JSR PC,(R0) ;;GO TO MONITOR
7802 045002 000240 NOP ;;SAVE ROOM
7803 045004 000240 NOP ;;FOR
7804 045006 000240 NOP ;;ACT11
7805 045010 $DOAGN:
7806 045010 000137 JMP @(PC)+ ;;RETURN
7807 045012 021032 $RTNAD: .WORD TST1
7808 045014 377 377 000 $ENULL: .BYTE -1,-1,0 ;;NULL CHARACTER STRING
7809 045017 015 042412 042116 $ENDMG: .ASCIIZ <15><12>/END PASS #/
7810 045024 050040 051501 020123
7811 045032 000043
7812
7813
7814
7815
7816
7817 ;*****
7818 ;*HERE IS A DETAILED EXPLANATION OF HOW THE LOOP ON ERROR WORKS.
7819 ;*ON HITTING AN ERROR IF THE LOOP ON ERROR SWITCH IS SET, THE
7820 ;*PROGRAM GOES BACK - USUALLY BACK TO THE BEGINNING OF THE TEST.
7821

```

```

7822 ;*WHEN THIS OPERATOR SELECTABLE SCOPE LOOP IS USED THEN THE POINT
7823 ;*THE PROGRAM GOES BACK TO CAN BE CHANGED.
7824 ;*THE RESTRICTIONS TO THE POINT WHERE THE PROGRAM CAN GO ARE: -
7825 ;*1. IT MUST BE WITHIN THE TEST UNDER CONSIDERATION
7826 ;*2. LOOP ON ERROR SWITCH MUST BE SET
7827 ;*3. THE ERROR MUST OCCUR WITHIN THE TEST UNDER CONSIDERATION
7828 ;*IF THE ERROR DOES NOT OCCUR WITHIN THE TEST UNDER CONSIDERATION
7829 ;*THE PROGRAM WILL REVERT TO NORMAL OPERATION. HOWEVER, IF LOOP ON
7830 ;*TEST SWITCH IS SET AND THIS OPERATOR SELECTABLE SCOPE LOOP IS USED
7831 ;*THEN THE PROGRAM WILL LOOP BACK TO THE SELECTED POINT WHEN IT
7832 ;*COMES TO THE END OF THE TEST UNDER CONSIDERATION.
7833 ;*
7834 ;*AFTER LOOPING FOR SOME TIME IF THE LOOP SWITCH IS PUT DOWN THEN
7835 ;*NORMAL OPERATION WILL CONTINUE.
7836 ;*****
7837
7838
7839 045034 000000 TESTAD: 0 ;FIRST ADDRESS OF TEST
7840
7841 045036 OPERSEL:
7842 045036 005037 177776 CLR PS ;MAKE PROCESSOR STATUS ZERO
7843 045042 104401 045050 TYPE ,65$ ;;TYPE ASCII STRING
7844 045046 000421 BR 64$ ;;GET OVER THE ASCIIZ
7845
7846 045112 ;;65$: .ASCIIZ <15><12>/THE PROGRAM WAS IN TEST NUMBER /
7847 045112 013746 017330 64$: MOV @#TSTNM,-(SP) ;GET READY TO TYPE TEST
7848 045116 104402 TYPOC ;NUMBER
7849 045120 104401 045126 TYPE ,67$ ;;TYPE ASCII STRING
7850 045124 000414 BR 66$ ;;GET OVER THE ASCIIZ
7851
7852 045156 ;;67$: .ASCIIZ <15><12>/THE LOOP BACK PC WAS /
7853 045156 013746 001110 66$: MOV @#SLPERR,-(SP) ;GET READY TO TYPE LOOP BACK PC
7854 045162 104402 TYPOC
7855 045164 104401 001223 TYPE ,69$
7856 045170 104401 045176 TYPE ,69$ ;;TYPE ASCII STRING
7857 045174 000426 BR 68$ ;;GET OVER THE ASCIIZ
7858
7859 045252 ;;69$: .ASCIIZ <15><12>/SET LOOP ON ERROR OR LOOP ON TEST SWITCH/
7860 045252 104401 045260 68$: TYPE ,71$ ;;TYPE ASCII STRING
7861 045256 000420 BR 70$ ;;GET OVER THE ASCIIZ
7862
7863 045320 ;;71$: .ASCIIZ <15><12>/TYPE THE FIRST PC OF THE TEST/
7864 045320 104401 045326 70$: TYPE ,73$ ;;TYPE ASCII STRING
7865 045324 000423 BR 72$ ;;GET OVER THE ASCIIZ
7866
7867 045374 ;;73$: .ASCIIZ <15><12>/ FOLLOWED BY A CARRIAGE RETURN /<15><12>
7868 045374 104412 72$: RDOCT
7869 045376 062716 000002 ADD #2,(SP) ;GET LPADR
7870 045402 012637 001106 MOV (SP)+,@#SLPADR
7871 045406 104401 045414 TYPE ,75$ ;;TYPE ASCII STRING
7872 045412 000417 BR 74$ ;;GET OVER THE ASCIIZ
7873
7874 045452 ;;75$: .ASCIIZ <15><12>/TYPE THE PC WHERE YOU WANT/
7875 045452 104401 045460 74$: TYPE ,77$ ;;TYPE ASCII STRING
7876 045456 000441 BR 76$ ;;GET OVER THE ASCIIZ
7877
7877 ;;77$: .ASCIIZ <15><12>/ THE PROGRAM TO LOOP BACK TO FOLLOWED BY A CARRIAGE RETURN /<1

```

```
7878 045562          768:      RDOCT
7879 045562 104412      MOV      (SP)+, @#SLPERR ;GET LPERR
7880 045564 012637 001110      MOV      @#SLPADR, -(SP)
7881 045570 013746 001106      ;THIS CLEARS UP GARBAGE
7882                                CLR      @#NOSYNC      ;CLEAR FLAG FOR HEADER ERROR COMMANDS
7883 045574 005037 053124      CLR      @#TSECC      ;CLEAR FLAG FOR ECC TEST
7884 045600 005037 015142      ;WHEN =177777 IT IS AN ECC TEST
7885                                ;WHEN =0 IT IS NOT AN ECC TEST
7886
7887                                CLR      @#TSECCG      ;EVEN IN AN ECC TEST EVERY CLOCK
7888 045604 005037 050624      ;IS NOT TO GENERATE ECC
7889                                ;IF =177777 GENERATE ECC
7890                                ;IF =0 DO NOT GENERATE ECC
7891                                ;DRIVE TIMING ERROR TEST
7892 045610 005037 015144      CLR      @#TESDTE
7893 045614 000002      RTI
```

```

7894
7895          .SBTTL  SAVE REGISTERS ROUTINE
7896
7897          ;;*****
7898          ;THIS SAVES THE CONTENTS OF ALL HARDWARE REGISTERS
7899          ;IN MEMORY LOCATIONS TAGED FROM 'WC' TO 'EC2'
7900          ;
7901          ;THIS IS DONE SO THAT COMPARES ARE DONE WITH SAVED LOCATIONS
7902          ;AND NOT THE REGISTERS THEMSELVES.  THIS WILL MAKE
7903          ;ERROR PRINTOUTS FOR GOOD AND BAD DATA ALWAYS DIFFRENT
7904          ;;*****
7905
7906
7907  PUTREG:
7908  045616      MOV      R0,-(SP)          ;;PUSH RC ON STACK
7909  045616      MOV      R1,-(SP)          ;;PUSH R1 ON STACK
7910  045620      MOV      R2,-(SP)          ;;PUSH R2 ON STACK
7911  045622      MOV      #RHWC,R0          ;STARTING ADDRESS OF REG
7912  045624      MOV      #WC,R1           ;STARTING ADDRESS OF SAVE LOCATIONS
7913  045630      MOV      #RHCC-RHWC+2/2,R2 ;NUMBER OF REG. INTO R2
7914  045634      MOV      @ (R0)+,(R1)+    ;SAVE HARDWARE REG.
7915  045640      DEC      R2
7916  045642      BNE     10$
7917  045644      MOV      (SP)+,R2          ;;POP STACK INTO R2
7918  045646      MOV      (SP)+,R1          ;;POP STACK INTO R1
7919  045650      MOV      (SP)+,R0          ;;POP STACK INTO R0
7920  045654      RTS      PC
  
```

```

7921
7922
7923
7924
7925
7926
7927
7928
7929 045656 000000
7930 045660 000000
7931 045662 000000
7932
7933 045664 012537 045656
7934 045670 012504
7935 045672 010437 045662
7936 045676 010537 045660
7937 045702 062705 000004
7938 045706 012703 000001
7939 045712 004737 045734
7940 045716 004737 045734
7941 045722 000241
7942 045724 006103
7943 045726 005703
7944 045730 001570
7945 045732 000205
7946 045734 005103
7947 045736 012737 045744 046076
7948 045744 010337 001124
7949 045750 005137 045656
7950 045754 043737 045656 001124
7951 045762 005137 045656
7952 045766 013714 001124
7953 045772 011437 001126
7954 045776 005137 045656
7955 046002 043737 045656 001126
7956 046010 005137 045656
7957 046014 023737 001124 001126
7958 046022 001403
7959 046024 004777 177630
7960 046030 104413
7961 046032 000207

.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: MOV R3,@#SGDDAT ;STORE GOOD DATA
      COM @#MASK ;AND MASK WITH PATTERN
      BIC @#MASK,@#SGDDAT ;CLEAR THE REST
      COM @#MASK ;RESTORE MASK
      MOV @#SGDDAT,(R4) ;OUTPUT TO REGISTER
      MOV (R4),@#SBDDAT ;INPUT FROM REGISTER
      COM @#MASK
      BIC @#MASK,@#SBDDAT ;AND MASK OUT RECEIVED DATA
      COM @#MASK ;RESTORE MASK
      CMP @#SGDDAT,@#SBDDAT ;IS DATA CORRECT
      BEQ 1$ ;BRANCH IF GOOD
      JSR PC, @LERR ;GO TO REPORT ERROR
      SCOP1 ;LOCAL SCOPE LOOP
1$: RTS PC

```

7962
7963
7964
7965
7966
7967
7968
7969
7970
7971
7972
7973
7974
7975
7976
7977
7978
7979
7980
7981
7982
7983
7984
7985
7986
7987
7988
7989
7990
7991
7992
7993
7994
7995
7996
7997

046034
046034 010146
046036 010246
046040 010346
046042 012001
046044 012002
046046 012003
046050 160102
046052 062702 000002
046056 010321
046060 005302
046062 005302
046064 001374
046066 012603
046070 012602
046072 012601
046074 000200

.SBTTL CLFAR MEMORY ROUTINE

```

*****
; THIS CLEARS ANY BLOCK OF MEMORY
; FILLING IT WITH ANY DATA
;
; CALL
; JSR      RO,CLAREA
; X
; Y
; Z
; DATA TO BE FILLED
; *R1 WILL HAVE STARTING ADDRESS OF BLOCK TO BE FILLED
; *R2 AFTER SUBTRACTION WILL HAVE TWICE NUMBER OF LOCATIONS
; *R3 WILL HAVE DATA TO BE FILLED
; *TO AVOID DIVIDE ROUTINE TWO DECREMENT R2 WILL BE USED
*****

```

```

CLAREA:
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      ;DATA
SUB      R1,R2          ;NO. OF LOCATIONS MINUS TWO
ADD      #2,R2          ;GET TWICE NO OF LOCATIONS
1$:      MOV      R3,(R1)+ ;MOVE IN DATA
DEC      R2
DEC      R2
BNE      1$            ;BRANCH IF NOT COMPLETE
MOV      (SP)+,R3      ;;POP STACK INTO R3
MOV      (SP)+,R2      ;;POP STACK INTO R2
MOV      (SP)+,R1      ;;POP STACK INTO R1
RTS      R0            ;RETURN

```

```

7998
7999 046076 000000          LAD: 0
8000
8001 046100 032777 001000 133032 T.SCOPI: BIT    #SW09, @SWR
8002 046106 001402          BEQ    1$
8003 046110 013716 046076          MOV    @#LAD, (SP)
8004 046114 000002          1$:    RTI
8005
8006          ;*EXAMPLE OF THE USE OF THE ABOVE
8007          ;*THIS WILL LOOP BETWEEN X: AND SCOP1 PROVIDED THERE IS NO 'NEWTSI'
8008          ;*MOV    #X,    @#LAD
8009          ;*X:    ---    ---
8010          ;*      ---    ---
8011          ;*      ---    ---
8012          ;*      SCOP1
8013
8014          .SBTTL  CLEAR DISK ROUTINE
8015
8016          CLDISK: MOV    @#RHCS1,R1    ;R1 WILL BE CONTROL AND STATUS1
8017 046116 013701 014756          MOV    @#RHCS2,R2    ;R2 WILL BE CONTROL AND STATUS2
8018 046122 013702 014754          MOV    @#RHDS1,R3    ;R3 WILL BE DISK STATUS REGISTER1
8019 046126 013703 015000          MOV    @#RHER1,R4    ;R4 WILL BE ERROR REGISTER #1
8020 046132 013704 014760
8021
8022 046136 012712 000040          MOV    #CLR,@R2    ;CLEAR ALL REG.
8023 046142 013712 015112          MOV    @#UNIT,@R2    ;REINSTATE UNIT NO.
8024 046146 005011          CLR    @R1    ;CLEAR FUNCTION BITS
8025 046150 012777 177777 146616  MOV    #-1,@RHAS    ;CLEAR ATTENTION BITS
8026 046156 000207          RTS    PC
8027

```

```

8028
8029
8030
8031
8032
8033
8034
8035
8036
8037
8038 046160 011637 015132 CHECKT: MOV (SP),@#PCJSR ;SAVE PC OF JSR+4
8039 046164 162737 000004 015132 SUB #4,@#PCJSR ;GET PC OF JSR
8040 046172 004737 045616 JSR PC,@#PUTREG ;SAVE REGISTERS
8041 046176 022737 004200 015032 CMP #DVA!RDY,@#CS1 ;RHCS1 SHOULD HAVE DEVICE AVAILABLE
8042 ;AND BE READY
8043 046204 001423 BEQ 3$ ;BRANCH IF GOOD TO RHDS1 CHECK
8044
8045 046206 032737 004000 015032 BIT #DVA,@#CS1 ;BAD SO TEST DEVICE AVAILABLE
8046 046214 001004 BNE 1$ ;TEST READY IF DVA THERE
8047 046216 010137 001122 MOV R1,@#SBDADR ;ADDRESS OF BAD REGISTER (RHCS1)
8048 046222 104026 ERROR 26 ;RHCS1 DID NOT HAVE DEVICE
8049 ;AVAILABLE AT START OF TEST
8050 046224 000413 BR 3$ ;BRANCH TO RHDS1 CHECK
8051
8052 046226 032737 000200 015032 1$: BIT #RDY,@#CS1 ;TEST READY
8053 046234 001003 BNE 2$ ;IF RDY THERE BRANCH
8054 046236 010137 001122 MOV R1,@#SBDADR ;ADDRESS OF BAD REGISTER (RHCS1)
8055 046242 104026 ERROR 26 ;RHCS1 DID NOT HAVE READY
8056 ;AT THE START OF TEST
8057 046244 000403 2$: BR 3$ ;BRANCH TO NEXT COMPARE
8058 046246 010137 001122 MOV R1,@#SBDADR ;ADDRESS OF BAD REGISTER (RHCS1)
8059 046252 104026 ERROR 26 ;RHCS1 HAD SOME BITS OTHER
8060 ;THAN DVA AND RDY SET
8061 ;ALL OTHER BITS SHOULD BE 0
8062 ;AT START OF TEST
8063
8064 046254 013746 015054 3$: MOV @#DS1,-(SP) ;GET RHDS1
8065 046260 042716 001100 BIC #VV!PROG,(SP) ;CLEAR VV AND PROGRAMABLE BIT
8066 046264 022726 000600 CMP #DPR!DRY,(SP)+ ;RHDS1 SHOULD HAVE THESE SET
8067 046270 001424 BEQ 8$ ;RETURN TO TEST IF GOOD
8068
8069 046272 032737 000400 015054 4$: BIT #DPR,@#DS1 ;BAD SO TEST DRIVE PRESENT
8070 046300 001004 BNE 5$ ;CHECK DRY IF GOOD
8071 046302 010337 001122 MOV R3,@#SBDADR ;ADDRESS OF BAD REGISTER (RHDS1)
8072 046306 104026 ERROR 26 ;RHDS1 DOES NOT HAVE DPR
8073 046310 000413 BR 7$ ;BRANCH OUT
8074 046312 032737 000200 015054 5$: BIT #DRY,@#DS1 ;TEST DRIVE READY
8075 046320 001004 BNE 6$ ;IF DPR WAS THERE SO BRANCH
8076 046322 010337 001122 MOV R3,@#SBDADR ;ADDRESS OF BAD REGISTER (RHDS1)
8077 046326 104026 ERROR 26 ;RHDS1 DOES NOT HAVE DRY
8078 046330 000403 BR 7$ ;BRANCH OUT
8079 046332 010337 001122 6$: MOV R3,@#SBDADR ;ADDRESS OF BAD REGISTER (RHDS1)
8080 046336 104026 ERROR 26 ;RHDS1 HAS SOME BITS OTHER
8081 ;THAN MOL, DRY, DPR, SET
8082 ;ALL OTHER BITS SHOULD BE 0
8083 046340 000207 7$: RTS PC ;RETURN TO TEST AND HALT - FATAL EPROR

```

```
8084
8085 046342 062716 000006      8$:  ADD    #6,(SP)      ;ADJUST STACK PTR TO GET OVER HALT IN TEST
8086 046346 000207              RTS    PC          ;RETURN TO TEST AND CONTINUE TESTING
8087
8088
8089
8090
8091
8092
8093
8094
8095 046350 011637 015132      CHECKE: MOV    (SP),@#PCJSR    ;SAVE PC OF JSR+4
8096 046354 162737 000004 015132 SUB    #4,@#PCJSR    ;GET PC OF JSR
8097 046362 004737 045616      JSR    PC,@#PUTREG    ;SAVE REGISTERS
8098 046366 032737 000200 015032 BIT    #RDY,@#CS1    ;RHCS1 SHOULD HAVE DEVICE AVAILABLE
8099                                ;AND BE READY
8100 046374 001004              BNE    1$          ;BRANCH IF GOOD
8101 046376 010137 001122      MOV    R1,@#SBDADR    ;FAILING REGISTER
8102 046402 104026              ERROR 26          ;RHCS1 IS IN ERROR
8103                                ;DOES NOT HAVE DVA, RDY
8104 046404 0004c              BR     4$          ;BRANCH
8105
8106 046406 032737 004000 015032 1$: BIT    #DVA,@#CS1    ;RHCS1 SHOULD HAVE DEVICE AVAILABLE
8107                                ;AND BE READY
8108 046414 001004              BNE    2$          ;BRANCH IF GOOD
8109 046416 010137 001122      MOV    R1,@#SBDADR    ;FAILING REGISTER
8110 046422 104026              ERROR 26          ;RHCS1 IS IN ERROR
8111                                ;DOES NOT HAVE DVA, RDY
8112 046424 000417              BR     4$          ;BRANCH OUT
8113 046426 032737 000200 015054 2$: BIT    #DRY,@#DS1    ;RHDS1 SHOULD HAVE DPR,DRY
8114 046434 001004              BNE    3$          ;BRANCH IF THERE
8115 046436 010337 001122      MOV    R3,@#SBDADR    ;FAILING REGISTER RHDS1
8116 046442 104026              ERROR 26          ;RHDS1 DOES NOT HAVE DPR,DRY
8117 046444 000407              BR     4$          ;BRANCH OUT
8118 046446 032737 000400 015054 3$: BIT    #DPR,@#DS1    ;RHDS1 SHOULD HAVE DPR,DRY
8119 046454 001004              BNE    5$          ;BRANCH IF THERE
8120 046456 010337 001122      MOV    R3,@#SBDADR    ;FAILING REGISTER RHDS1
8121 046462 104026              ERROR 26          ;RHDS1 DOES NOT HAVE DPR,DRY
8122 046464 000207              4$:  RTS    PC          ;RETURN TO TEST AND HALT - FATAL ERROR
8123
8124 046466 062716 000006      5$:  ADD    #6,(SP)      ;ADJUST STACK TO GET OVER HALT IN TEST
8125 046472 000207              RTS    PC          ;RETURN TO TEST AND CONTINUE TESTING
8126
8127
8128
8129
8130
8131
8132
8133
8134
8135 046474 177777              TIMCNT: 177777          ;WAITING COUNT
8136
8137 046476 010046              WAIT.T: MOV    R0,-(SP)    ;SAVE R0
8138 046500 016600 000002      MOV    2(SP),R0        ;GET ADDRESS OF REG. ADDRESS
8139 046504 010037 001204      MOV    R0,@#SIMP3    ;WAT PC+2 IN $IMP3
```

```

8140 046510 162737 000002 001204 SUB #2,#STMP3 ;WAT PC FOR TYPEOUT
8141 046516 012037 001176 MOV (R0)+,#STMP0 ;WAIT REGISTER ADDRESS
8142 046522 012037 001200 MOV (R0)+,#STMP1 ;WAIT ON BIT
8143 046526 010066 000002 MOV R0,2(SP) ;RESTORE RETURN ON STACK
8144 046532 012600 MOV (SP)+,R0 ;RESTORE R0
8145 046534 013737 046474 001202 MOV #TIMCNT,#STMP2;TEMPORARY COUNT
8146
8147 046542 033777 001200 132426 1$: BIT #STMP1,#STMP0 ;IS REQUIRED BIT THERE?
8148 046550 001021 BNE 2$ ;BRANCH IF YES
8149 046552 005337 001202 DEC #STMP2 ;COUNT
8150 046556 001371 BNE 1$ ;BRANCH IF NOT TIME UP
8151 046560 013737 046474 001202 MOV #TIMCNT,#STMP2;TEMPORARY COUNT
8152 046566 033777 001200 132402 3$: BIT #STMP1,#STMP0 ;IS REQUIRED BIT THERE?
8153 046574 001007 BNE 2$ ;BRANCH IF YES
8154 046576 005337 001202 DEC #STMP2 ;COUNT
8155 046602 001371 BNE 3$ ;BRANCH IF NOT TIME UP
8156 046604 017737 132366 001126 MOV #STMP0,#$BDDAT ;REGISTER CONTENTS
8157 046612 104016 ERROR 16 ;WAITED ON BIT FAILED TO SET
8158 046614 000002 2$: RTI
8159
8160
8161 ;* CALL FOR THE ABOVE WAITLOOP IS
8162 ;*
8163 ;* MOV #A,#XS ;A CONTAINS REGISTER ADDRESS
8164 ;* - - - ;HENCE XS WILL HAVE ABSOLUTE REG. ADR.
8165 ;* - - -
8166 ;* - - -
8167 ;* WAT
8168 ;*XS: 0 ;ABSOLUTE REG. ADDRESS UNDER WAIT
8169 ;* .WORD 0 ;BIT WAITED FOR
8170 ;* ;CONTINUE

```

```

8171
8172
8173
8174
8175
8176
8177
8178
8179
8180
8181
8182
8183
      .SBTTL  SAVE ROUTINE
      ;*****
      ;THIS IS A SUBROUTINE TO READ & SAVE REGISTERS
      ;IN THE REGISTER TABLE TO ANY LOCATION
      ;THE CALL IS
      ;JSR    R0,@#SAVER
      ;FROM
      ;TO
      ;NUMBER OF WORDS SAVED
      ;*****
  
```

```

8184 046616
8185 046616 010146
8186 046620 010246
8187 046622 010346
8188 046624 012001
8189 046626 012002
8190 046630 012003
8191 046632 013122
8192 046634 005303
8193 046636 001375
8194 046640 012603
8195 046642 012602
8196 046644 012601
8197 046646 000200
      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
  
```

```

8198
8199
8200
8201
8202
8203
8204
8205
8206
8207
8208
8209
8210
      .SBTTL  WRITE CHECK ROUTINE
      ;*****
      ;THIS IS A SUBROUTINE TO DO WRITE CHECK HEADER AND DATA
      ;CYLINDER 0, TRACK 1, SECTOR 1, KEYS 0
      ;*****
  
```

```

8209
8210
      ;THESE ARE TO SET UP FOR DISKLESS USE ONLY
8211 046650 012737 010000 053010 WRCHMD: MOV    #FMT22,@#CYL      ;CYLINDER 0 FORMAT 16 BIT WORDS
8212 046656 112737 000001 053013      MOV    #1,@#SECOTR+1    ;TRACK=1
8213 046664 112737 000001 053012      MOV    #1,@#SECOTR      ;SECTOR=1
8214 046672 005037 053014      CLR     @#KEY1          ;KEY1=0
8215 046676 005037 053016      CLR     @#KEY2          ;KEY2=0
8216 046702 012737 000044 053070      MOV    #36.,DAWORD      ;NO OF DATA WORDS
8217 046710 005037 053020      CLR     @#X              ;THIS IS A READ OPERATION
8218 046714 004537 047332      JSR     R5,@#CRC          ;GO TO CALCULATE CRC
8219 046720 053010      CYL
8220 046722 054710      WCRC
  
```

```

8221
8222
8223
8224 046724 004737 046116      JSR     PC,@#CLDISK      ;SET UP GENERAL REGISTERS
8225
8226 046730 012777 177730 146012      MOV    #-40.,@RHW      ;AND CLEAR DISK REGISTERS
      ;36 DATA WORDS 4 HEADER WORDS
  
```

CZRJHDO,RP04/5/6 DSKLS CTRLR2 MACY11 30A(1052) 24-MAY-79 15:07 F 15
CZRJHD.P12 01-MAR-79 09:10 WRITE CHECK ROUTINE PAGE 189

SEQ 0187

8227	046736	012777	016264	146006	MOV	#REINTO,@RHBA	;STARTING ADDRESS OF READ BUFFER
8228	046744	112746	000001		MOVB	#1,-(SP)	;SECTOR=1
8229	046750	112766	000001	000001	MOVB	#1,1(SP)	;TRACK=1 IN UPPER BYTE
8230	046756	012677	146000		MOV	(SP)+,@RHDS	;TRACK=1, SECTOR=1 IN RHDS
8231	046762	012777	014000	145776	MOV	#FMT22!EC1,@RHOF	;16 BIT WORDS
8232							;ECC CORRECTION INHIBIT BECAUSE
8233							;ECC LOGIC IS NOT CHECKED YET
8234	046770	005077	145774		CLR	@RHCA	;CYLINDER=0
8235	046774	004737	046160		JSR	PC,@#CHECKT	;CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DON'T
8236	047000	104401	005123		TYPE	,CPHALT	;CANNOT CONTINUE TESTING IF ANY OF THE
8237							;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
8238	047004	000000			HALT		;STOP THE TEST
8239	047006	013711	015166		MOV	@#WRCHDT,@R1	;WRITE CHECK HEADER AND DATA=52
8240							;INTO RHCS1
8241	047012	004737	052650		JSR	PC,@#COMHD	;WRITE CHECK HEADER AND DATA
8242							;SAME AS READ HEADER AND DATA
8243							
8244	047016	000207			RTS	PC	;RETURN TO WRITE CHECK TEST

8245
8246
8247
8248
8249
8250
8251
8252
8253
8254
8255
8256
8257
8258
8259
8260
8261
8262
8263
8264
8265
8266
8267
8268
8269
8270
8271
8272
8273
8274
8275
8276
8277
8278
8279
8280
8281
8282
8283
8284
8285
8286
8287
8288
8289
8290
8291
8292
8293
8294
8295
8296
8297

047020
047020 010146
047022 010246
047024 010346
047026 010446
047030 010546
047032 012001
047034 012002
047036 012003
047040 012037 001176
047044 012037 001200
047050 011000
047052 010304
047054 005204
047056 010437 053130
047062 022122
047064 001426

047066 011137 001124
047072 014237 001126
047076 160337 053130
047102 005737 015124
047106 001003
047110 004777 132062
047114 000402
047116 004777 132056
047122 022122
047124 017746 132010
047130 042716 177177
047134 022726 000200
047140 001402
047142 005303
047144 001344
047146
047146 012605
047150 012604
047152 012603
047154 012602
047156 012601
047160 000200

.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

```

```

8298 .SBTTL WRITE CHECK DATA
8299
8300 ::*****
8301 ;THIS IS A SUBROUTINE TO DO WRITE CHECK DATA
8302 ;CYLINDER 0, TRACK 1, SECTOR 1, KEYS 0
8303 ::*****
8304
8305 ;THESE ARE TO SET UP FOR DISKLESS USE ONLY
8306
8307 WRCHDA: MOV #FMT22,@#CYL ;CYLINDER 0 FORMAT 16 BIT WORDS
8308 047162 012737 010000 053010 MOV #1,@#SECOTR+1 ;TRACK=1
8309 047170 112737 000001 053013 MOV #1,@#SECOTR ;SECTOR=1
8310 047176 112737 000001 053012 CLV @#KEY1 ;KEY1=0
8311 047204 005037 053014 CLV @#KEY2 ;KEY2=0
8312 047210 005037 053016 MOV #32,@#DAWORD ;NO OF DATA WORDS
8313 047214 012737 000040 053070 CLV @#X ;THIS IS A READ OPERATION
8314 047222 005037 053020
8315
8316 JSR R5,@#CRC ;GO TO CALCULATE CRC
8317 047232 053010 CYL
8318 047234 054710 WCRC
8319
8320 ;THESE ARE REGULAR SETUPS
8321
8322 JSR PC,@#CLDISK ;SET UP GENERAL REGISTERS
8323 047236 004737 046116 ;AND CLEAR DISK REGISTERS
8324
8325 MOV #-32,@#RHW ;36 DATA WORDS 4 HEADER WORDS
8326 047250 012777 016264 145500 MOV #REINTO,@#RHB ;STARTING ADDRESS OF READ BUFFER
8327 047256 112746 000001 145474 MOV #1,-(SP) ;SECTOR=1
8328 047262 112766 000001 000001 MOV #1,1(SP) ;TRACK=1 IN UPPER BYTE
8329 047270 012677 145466 MOV (SP)+,@#RHDST ;TRACK=1, SECTOR=1 IN RHDST
8330 047274 012777 014000 145'64 MOV #FMT22!ECI,@#RHO ;16 BIT WORDS
8331
8332 ;ECC CORRECTION INHIBIT BECAUSE
8333 047302 005077 145462 CLV @#RHCA ;ECC LOGIC IS NOT CHECKED YET
8334 047306 004737 046160 JSR PC,@#CHECKT ;CYLINDER=0
8335 047312 104401 005123 TYPE ,CPHALT ;CHECK DVA,RDY,DPR,DRY - 1 AND OTHERS DON'T
8336 ;CANNOT CONTINUE TESTING IF ANY OF THE
8337 047316 000000 HALT ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
8338 047320 013711 015164 MOV @#WRCHK,@#R1 ;STOP THE TEST
8339 047324 004737 052650 JSR PC,@#COMHD ;WRITE CHECK DATA=50 INTO RHCS1
8340 ;WRITE CHECK HEADER AND DATA
8341 ;SAME AS READ HEADER AND DATA
8342 047330 000207 RTS PC ;RETURN TO WRITE CHECK TEST
8343

```

.SBTTL CRC GENERATION ROUTINE

```

;*****
;THIS IS A SUBROUTINE TO CALCULATE CRC FOR THE FOUR
;HEADER WORDS AND STORE THEM IN 'WCRC' AND 'GCRC'
;R1 - REGISTER FOR CRC, INCREMENTED CRC VALUE IS HERE
;R2 - THIS HAS BIT POSITION 2 VALUE C
;R3 - THIS HAS BIT POSITION 16 I.E. OUTPUT BIT VALUE B
;R4 - THIS HAS BIT POSITION 15 VALUE E
;STMP0 - NUMBER OF WORDS
;STMP2 - NUMBER OF BITS PER WORD = 16
;STMP3 - TEMPORARY REG.
;STMP4 - TEMPORARY REG TO TRANSFER CARRY
;STMP5 - THIS HAS DATA BIT VALUE D

;FETCH DATA BIT D
;B = D XOR 16
;C = B XOR 2
;E = B XOR 15
;ROTATE RIGHT ONE POSITION
;B GOES TO POSITION 1
;C GOES TO POSITION 3
;E GOES TO POSITION 16
;REPEAT 64 TIMES

;CALL JSR R5,@WCRC
;X ;FIRST LOCATION AT
;Y ;PUT CRC IN WCRC FOR READ GCRC FOR WRITE
;*****
  
```

CRC:

```

MOV R0,-(SP) ;PUSH R0 ON STACK
MOV (R5)+,R0 ;GET POINTER TO CYL NO.
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
CLR R1 ;CLEAR WORKING LOCATION
CLR @STMP5
MOV #4,@STMP0 ;WORD COUNT
MOV (R0)+,@STMP3 ;TEMPORARY WORD STORAGE
MOV #16,@STMP2 ;BIT COUNT
MOV @STMP3,@STMP4 ;TEMPORARY WORD STORAGE
ROR @STMP3 ;GET LSB INTO 'C'
ROR @STMP5 ;GET ABOVE 'C' INTO STMP5
BIT #BIT0,R1 ;IS POSITION 15 HIGH
BEQ 1$ ;BRANCH IF POSITION 16 LOW
MOV #BIT15,R3 ;GET POSITION 16
BR 2$
1$: CLR R3 ;GET POSITION 16
2$: ADD @STMP5,R3 ;XOR POSITION 16 WITH D
;TO GIVE B
BIT #BIT14,R1 ;IS POSITION 2 HIGH
BEQ 3$ ;BRANCH IF POSITION 2 LOW
MOV #BIT15,R2 ;GET POSITION 2
BR 4$
  
```

```

8344
8345
8346
8347
8348
8349
8350
8351
8352
8353
8354
8355
8356
8357
8358
8359
8360
8361
8362
8363
8364
8365
8366
8367
8368
8369
8370
8371
8372
8373
8374 047332
8375 047332 010046
8376 047334 012500
8377 047336 010146
8378 047340 010246
8379 047342 010346
8380 047344 010446
8381 047346 005001
8382 047350 005037 001210
8383 047354 012737 000004 001176
8384 047362 012037 001204 16$:
8385 047366 012737 000020 001202
8386 047374 013737 001204 001206
8387 047402 006037 001204 15$:
8388 047406 006037 001210
8389 047412 032701 000001
8390 047416 001403
8391 047420 012703 100000
8392 047424 000401
8393 047426 005003 1$:
8394 047430 063703 001210 2$:
8395
8396 047434 032701 040000
8397 047440 001403
8398 047442 012702 100000
8399 047446 000401
  
```

8400	047450	005002		3\$: CLR	R2	:GET POSITION 2
8401	047452	060302		4\$: ADD	R3,R2	:XOR B WITH POSITION 2
8402						:TO GIVE C
8403	047454	032701	000002		BIT #BIT1,R1	:IS POSITION 15 HIGH
8404	047460	001403			5\$: BEQ	:BRANCH IF POSITION 15 LOW
8405	047462	012701	100000		MOV #BIT15,R4	:GET POSITION 15
8406	047466	000401			6\$: BR	
8407	047470	005004		5\$: CLR	R4	:GET POSITION 15
8408	047472	060304		6\$: ADD	R3,R4	:XOR POSITION 15 WITH B
8409						:TO GIVE E
8410	047474	006037	001206		ROR @#STMP4	:GET LSB INTO "C"
8411	047500	006001			ROR R1	:GET ABOVE C INTO R1
8412	047502	005703			TST R3	:TEST B
8413	047504	100403			BMI 7\$	:BRANCH IF B=1
8414	047506	042701	100000		BIC #BIT15,R1	:SET B IN POSITION 1
8415	047512	000402			10\$: BR	
8416	047514	052701	100000	7\$: BIS	#BIT15,R1	:SET B IN POSITION 1
8417	047520	005702		10\$: TST	R2	:TEST C
8418	047522	100403			BMI 11\$	:BRANCH IF C=1
8419	047524	042701	020000		BIC #BIT13,R1	:GET C IN POSITION 3
8420	047530	000402			12\$: BR	
8421	047532	052701	020000	11\$: BIS	#BIT13,R1	:GET C IN POSITION 3
8422	047536	005704		12\$: TST	R4	:TEST E
8423	047540	100403			BMI 13\$	:BRANCH IF E=1
8424	047542	042701	000001		BIC #BIT0,R1	:GET E IN POSITION 16
8425	047546	000402			14\$: BR	
8426	047550	052701	000001	13\$: BIS	#BIT0,R1	:GET E IN POSITION 16
8427	047554	005337	001202	14\$: DEC	@#STMP2	:BIT COUNTER
8428	047560	001310			BNE 15\$	:BRANCH IF 16 NOT DONE
8429	047562	005337	001176		DEC @#STMP0	:WORD COUNTER
8430	047566	001275			BNE 16\$	:BRANCH IF 4 NOT DONE
8431	047570	010135			MOV R1,@(R5)+	:PUT CRC WHERE DESIRED
8432	047572	012604			MOV (SP)+,R4	:POP STACK INTO R4
8433	047574	012603			MOV (SP)+,R3	:POP STACK INTO R3
8434	047576	012602			MOV (SP)+,R2	:POP STACK INTO R2
8435	047600	012601			MOV (SP)+,R1	:POP STACK INTO R1
8436	047602	012600			MOV (SP)+,R0	:POP STACK INTO R0
8437	047604	000205			RTS R5	

```

8438 .SBTTL SIMULATED DISK SETUP
8439
8440 ;*****
8441 ;THIS IS A SUBROUTINE TO SET UP THE SIMULATOR DISK FOR
8442 ;CYLINDER 0 (16 BITS PER WORD)
8443 ;TRACK 1, SECTOR 1
8444 ;KEY1 1
8445 ;KEY2 1
8446 ;CRC THROUGH THE JSR R5,@#CRC
8447 ;256 WORDS OF 177400
8448
8449 ;CALL JSR PC,@#SETDSK
8450 ;*****
8451
8452 SETDSK:
8453     MOV     R0,-(SP)      ;;PUSH R0 ON STACK
8454     MOV     R1,-(SP)      ;;PUSH R1 ON STACK
8455     MOV     R2,-(SP)      ;;PUSH R2 ON STACK
8456     MOV     #177400,R0    ;DATA IN THE DISK
8457     MOV     #256.,R1      ;COUNTER
8458     MOV     #DISK,R2      ;START OF SIMULATOR DISK
8459 1$:     MOV     R0,(R2)+    ;MOVE IN DATA
8460     DEC     R1            ;COUNT FOR 256
8461     BNE     1$           ;BRANCH IF 256 NOT COMPLETE
8462     MOV     #17.,R1      ;2 ECC WORDS, 1 DATA GAP
8463     ;14 TOLERANCE GAP
8464 2$:     CLR     (R2)+      ;CLEAR ECC,DATA GAP AND
8465     ;TOLERANCE GAP
8466     DEC     R1            ;COUNT
8467     BNE     2$           ;BRANCH IF NOT COMPLETE
8468
8469     ;NOW SET UP FOR DISKLESS USE
8470
8471     MOV     #FMT22,@#CYL  ;CYLINDER 0 (16 BIT WORDS)
8472     MOVB    #1,@#SECOTR+1 ;TRACK=1
8473     MOVB    #1,@#SECOTR   ;SECTOR=1
8474     MOV     #1,@#KEY1     ;KEY1=1
8475     MOV     #1,@#KEY2     ;KEY2=1
8476     MOV     256.,@#DAWORD ;NO. OF DATA WORDS
8477     JSR     R5,@#CRC      ;GO TO CALCULATE CRC
8478     CYL     ;FIRST CRC WORD
8479     WCRC    ;PUT CALCULATED CRC
8480     MOV     (SP)+,R2      ;;POP STACK INTO R2
8481     MOV     (SP)+,R1      ;;POP STACK INTO R1
8482     MOV     (SP)+,R0      ;;POP STACK INTO R0
8483     RTS     PC
8484
8471 047650 012737 010000 053010
8472 047656 112737 000001 053013
8473 047664 112737 000001 053012
8474 047672 012737 000001 053014
8475 047700 012737 000001 053016
8476 047706 013737 000400 053070
8477 047714 004537 047332
8478 047720 053010
8479 047722 054710
8480 047724 012602
8481 047726 012601
8482 047730 012600
8483 047732 000207

```

```
8485 .SBTTL CHECK HCE ROUTINE
8486
8487 ;*****
8488 ;THIS IS A SUBROUTINE TO CHECK HEADER COMPARE ERROR
8489 ;(BIT #7) AND CRC ERROR (BIT #8)
8490
8491 ;CALL JSR RO,@#HCCRCE
8492
8493 ;      COM      ;COMMAND-READ HEADER AND DATA
8494 ;      C      ;      -WRITE DATA
8495 ;      S      ;CYLINDER
8496 ;      T      ;SECTOR
8497 ;      -N.    ;TRACK
8498 ;      B      ;WORD COUNT
8499 ;      X      ;RHBA BUFFER START
8500 ;      H      ;1=WRITE DATA 0=READ
8501 ;             ;H=1 HEADER CHECK, H=0 CRC CHECK
8502 ;*****
8503
8504 047734 010037 015132 HCCRCE: MOV RO,@#PCJSR ;SAVE PC OF JSR+4
8505 047740 162737 000004 015132 SUB #4,@#PCJSR ;GET PC OF JSR
8506 047746 004737 046116 JSR PC,@#CLDISK ;INIT AND SETUP GENERAL REG.
8507 047752 004737 046160 JSR PC,@#CHECKT ;CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DON'T
8508 047756 104401 005123 TYPE ,CPHALT ;CANNOT CONTINUE TESTING IF ANY OF THE
8509 ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
8510 047762 000000 HALT ;STOP THE TEST
8511 047764 011037 001210 MOV (RO),@#STMP5 ;SAVE COMMAND
8512 047770 012011 MOV (RO)+,@R1 ;COMMAND
8513 047772 012077 144772 MOV (RO)+,@RHCA ;CYLINDER
8514 047776 112046 MOV (RO)+,@R1 ;SECTOR
8515 050000 105720 TSTB (RO)+ ;UP DATE RO
8516 050002 112066 000001 MOV (RO)+,@R1 ;TRACK
8517 050006 105720 TSTB (RO)+ ;UPDATE RO
8518 050010 012677 144746 MOV (SP)+,@RHDST ;TRACK SECTOR
8519 050014 012077 144730 MOV (RO)+,@RHWC ;NO. OF DATA WORDS +4 HEADER
8520 ;IF A READ HEADER AND DATA
8521 050020 012077 144726 MOV (RO)+,@RHBA ;STARTING ADDRESS OF BUFFER
8522 050024 012037 053020 MOV (RO)+,@#X ;X=0 READ HEADER AND DATA
8523 ;X=1 WRITE DATA
8524 050030 012777 014000 144730 MOV #FMT22!EC1,@RHOF ;16 BITS PER WORD
8525 ;ECC CORRECTION INHIBIT
8526 050036 005037 015124 CLR @#ERFLG$ ;CLEAR ERROR FLAG
8527 050042 004737 052650 JSR PC,@#COMHD ;COMMAND
8528
8529 ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
8530 ;FROM THE 'COMHD' ROUTINE THAT MEANS SECTOR GAP,
8531 ;FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
8532 ;SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
8533 ;DETECTED
8534 ;HEADER AND DATA ARE TO BE CHECKED.
8535
8536 050046 004737 045616 JSR PC,@#PUTREG ;SAVE REGISTERS
8537 050052 005737 015124 TST @#ERFLG$ ;ANY ERRORS ALREADY THERE
8538 050056 001034 BNE 10$ ;BRANCH IF YES
8539 050060 005737 053020 TST @#X ;IS THIS A READ
8540 050064 001015 BNE 3$ ;IF A WRITE DATA BRANCH
```

```

8541
8542      ;NOW THE READ BUFFER WILL BE CHECKED
8543      ;HEADER SHOULD BE COMPLETELY READ AS WRITTEN
8544      ;NO DATA WORDS SHOULD BE READ
8545      ;REINTO BUFFER HAS BEEN FILLED WITH 0
8546      ;WRFROM BUFFER HAS BEEN FILLED WITH EXPECTED DATA
8547
8548 050066 004037 047020      JSR      RO,@#COMPAR      ;CHECK
8549 050072 015220      WRFROM      ;GOOD DATA
8550 050074 016264      REINTO      ;TEST BUFFER
8551 050076 000400      256.      ;4 HEADER 252 DATA
8552 050100 050106      1$      ;RETURN POINT FOR ERROR HEADER
8553 050102 050112      2$      ;RETURN POINT FOR ERROR DATA
8554 050104 050150      10$     ;RETURN FOR GOOD COMPARISON
8555 050106 104004      1$:      ERROR      4      ;READ NEXT ERROR 5
8556 050110 000207      RTS      PC      ;RETURN TO COMPARISON SUBROUTINE
8557 050112 104005      2$:      ERROR      5      ;WORD NO 1 THRU 4 ARE
8558                                     ;HEADER WORDS AND HENCE
8559                                     ;SHOULD BE READ AS WRITTEN ON
8560                                     ;DISK, WORD NOS. 5 ONWARDS
8561                                     ;SHOULD NOT BE READ AND HENCE
8562                                     ;READ INTO BUFFER
8563                                     ;SHOULD BE UNCHANGED
8564 050114 000207      RTS      PC      ;RETURN TO COMPARISON
8565
8566 050116 000414      BR      10$      ;JUMP OUT
8567
8568      ;NOW THE DISK WILL BE CHECKED
8569      ;NO DATA SHOULD BE WRITTEN
8570      ;REINTO BUFFER HAS BEEN FILLED WITH EXPECTED DATA
8571      ;DISK HAS BEEN FILLED WITH 177400
8572      ;WRFROM HAS BEEN FILLED WITH 125252
8573
8574 050120 004037 047020      3$:      JSR      RO,@#COMPAR      ;CHECK
8575 050124 016264      REINTO      ;GOOD DATA BUFFER
8576 050126 054726      DISK      ;TEST BUFFER
8577 050130 000400      256.
8578 050132 050140      4$      ;RETURN POINT FOR ERROR HEADER
8579 050134 050144      5$      ;RETURN POINT FOR ERROR DATA
8580 050136 050150      10$     ;RETURN POINT FOR GOOD COMPARISON
8581 050140 104004      4$:      ERROR      4      ;READ NEXT ERROR 5
8582 050142 000207      RTS      PC      ;RETURN TO COMPARISON SUBROUTINE
8583 050144 104005      5$:      ERROR      5      ;WORD NO ARE ALL DATA
8584                                     ;WORDS THE SHOULD NOT
8585                                     ;HAVE BEEN CHANGED BY THE
8586                                     ;WRITE COMMAND
8587 050146 000207      RTS      PC      ;RETURN TO COMPARISON SUBROUTINE
8588 050150 005720      10$:      TST      (RO)+      ;IS THIS A HCRC ON HCE CHECK?
8589 050152 001442      BEQ      6$      ;BRANCH IF HCRC
8590 050154 022737 000072 001210      CMP      #72,@#STMP5      ;IS THIS A READ COMMAND
8591 050162 001417      BEQ      11$     ;BRANCH IF YES
8592 050164 017737 144570 001126      MOV      @RHER1,@#SBDDAT      ;TEST DATA
8593 050172 022737 000200 001126      CMP      #HCE,@#SBDDAT      ;ONLY HEADER COMPARE BIT?
8594                                     ;SHOULD BE SET
8595 050200 001470      BEQ      7$      ;BRANCH IF GOOD
8596 050202 013737 014760 045662      MOV      @#RHER1,@#REGADR      ;REGISTER ADDRESS RHER1

```

```

8597 050210 012737 000200 001124      MOV    #HCE,@#SGDDAT ;GOOD DATA
8598 050216 104027              ERROR  27      ;AFTER AN ERROR ON THE
8599                                ;HEADER ONLY HCE SHOULD
8600 050220 000460              BR      7$      ;BE SET
8601 050222              11$:
8602 050222 017737 144532 001126      MOV    @RHER1,@#SBDDAT ;TEST DATA
8603 050230 022737 100200 001126      CMP    #DCK!HCE,@#SBDDAT ;ONLY HEADER COMPARE BIT?
8604                                ;SHOULD BE SET
8605                                ;DCK IS SET BECAUSE ECC IS NOT READ
8606 050236 001451              BEQ      7$      ;BRANCH IF GOOD
8607 050240 013737 014760 045662      MOV    @RHER1,@#REGADR ;REGISTER ADDRESS RHER1
8608 050246 012737 100200 001124      MOV    #DCK!HCE,@#SGDDAT ;GOOD DATA
8609 050254 104027              ERROR  27      ;AFTER AN ERROR ON THE
8610                                ;HEADER ONLY HCE SHOULD
8611 050256 000441              BR      7$      ;BE SET
8612 050260 022737 000072 001210 6$:    CMP    #72,@#STMP5 ;IS THIS A READ COMMAND?
8613 050266 001417              BEQ      12$     ;BRANCH IF A READ
8614 050270 017737 144464 001126      MOV    @RHER1,@#SBDDAT ;TEST DATA
8615 050276 022737 000400 001126      CMP    #HCRC,@#SBDDAT ;ONLY CRC ERROR SHOULD BE THERE
8616 050304 001426              BEQ      7$
8617 050306 013737 014760 045662      MOV    @RHER1,@#REGADR ;REG. ADDR = RHER1
8618 050314 012737 000400 001124      MOV    #HCRC,@#SGDDAT ;GOOD DATA
8619 050322 104027              ERROR  27      ;AFTER A CRC ERROR ONLY CRC
8620                                ;SHOULD BE SET
8621 050324 000416              BR      7$      ;BRANCH OUT
8622 050326 017737 144426 001126 12$:    MOV    @RHER1,@#SBDDAT ;TEST DATA
8623
8624 050334 022737 100400 001126      CMP    #DCK!HCRC,@#SBDDAT;HCRC AND DCK SHOULD BE SET
8625                                ;DCK IS SET BECAUSE ECC IS NOT READ
8626 050342 001407              BEQ      7$      ;BRANCH IF GOOD
8627 050344 012737 100400 001124      MOV    #DCK!HCRC,@#SGDDAT;GOOD DATA
8628 050352 013737 014760 045662      MOV    @RHER1,@#REGADR;FAILING REGISTER RHER1
8629 050360 104027              ERROR  27      ;AFTER A CRC ERROR ON A READ
8630                                ;DCK AND HCRC SHOULD BE SET
8631                                ;DCK IS SET BECAUSE ECC IS NOT READ
8632 050362 000200              7$:    RTS      R0      ;RETURN TO MAIN TEST
  
```

```

8633 .SBTTL EXIT WRT HEADER & DATA ROUTINE
8634
8635 ;:*****
8636 ;THIS IS A SUBROUTINE TO LEAVE AT THE MIDDLE OF
8637 ;A WRITE HEADER AND DATA COMMAND
8638 ;IT TRYS TO GET SECTOR 10, TRACK 0, CYLINDER 0
8639 ;BUT COMES OUT AFTER ONE SECTOR
8640 ;THE COMMAND OS JSR PC,@#MIDDLE
8641 ;BAI IS SET
8642 ;:*****
8643
8644 MIDDLE:
8645 050364 010046      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
8646 050366 010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
8647 050370 013777 015172 144360      MOV      @#WRTFOR,@RHCS1 ;;WRITE HEADER AND DATA=62
8648                                     ;IN RHCS1
8649 050376 012777 177766 144344      MOV      #-10,@RHWC      ;10 WORDS
8650 050404 012777 015220 144340      MOV      #WRTFROM,@RHBA ;;BUS ADDRESS=WRTFROM
8651 050412 012777 000010 144342      MOV      #10,@RHDS1    ;;DESIRED TRACK=0 SECTOR=10
8652 050420 052777 000010 144326      BIS      #BAI,@RHCS2    ;;BUS ADDRESS INCREMENT INHIBIT
8653 050426 012777 010000 144332      MOV      #FMT22,@RHOF    ;;FORMAT 16 BIT WORDS
8654 050434 005077 144330      CLR      @RHCA      ;CYLINDER=0
8655 050440 012737 000001 050466      MOV      #1,@#MID      ;SECTOR IS SET TO 1 SO THAT
8656                                     ;WE CAN GET OUT AT THE
8657                                     ;MIDDLE OF AN OPERATION
8658                                     ;LOOKING FOR SECTOR 10
8659 050446 012777 000001 144322      MOV      #DMD,@RHMR    ;SET DIAGNOSTIC MODE
8660 050454 052777 000001 144274      BIS      #GO,@RHCS1    ;GO TO RHCS1 WITH 62
8661 050462 004137 057056      JSR      R1,@#SEARCH
8662 050466 000000      MID:      .WORD      0      ;SECTOR
8663 050470 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
8664 050472 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
8665 050474 000207      RTS      PC

```

8666
8667
8668
8669
8670
8671
8672
8673
8674
8675
8676
8677
8678
8679
8680
8681
8682
8683
8684
8685
8686
8687
8688
8689
8690
8691
8692
8693
8694
8695
8696
8697
8698
8699
8700
8701
8702
8703
8704

050476
050476 010546
050500 010037 015132
050504 162737 000004 015132
050512 012005
050514 010577 144250
050520 005077 144236
050524 013777 015200 144224
050532 012777 000001 144236
050540 052777 000001 144210
050546 000240
050550 000240
050552 000240
050554 000240
050556 004737 046116
050562 017737 144226 001126
050570 020537 001126
050574 001406
050576 010537 001124
050602 013737 015014 045662
050610 104030
050612
050612 012605
050614 000200

.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      R0,@#MAKECYL
*      XC
*                                ;DESIRED VALUE OF CURRENT CYLINDER
*****

MAKECYL:
      MOV      R5,-(SP)          ;;PUSH R5 ON STACK
      MOV      R0,@#PCJSR       ;PC OF JSR+4
      SUB      #4,@#PCJSR       ;SAVE PC OF JSR
      MOV      (R0)+,R5         ;GETTING READY TO FILL DESIRED CYLINDER
      MOV      R5,@RHCA         ;FILL DESIRED CYLINDER REGISTER
      CLR      @RHDS1          ;MAKE SURE DESIRED SECTOR TRACK IS NOT ILLEGAL
      MOV      @#SEECOM,@RHCS1 ;FILL SEEK COMMAND
      MOV      #DMD,@RHMR      ;SET DIAGNOSTIC MODE
      BIS      #GO,@RHCS1      ;GO TO SEEK
      NOP
      NOP                      ;ALLOW TIME FOR SEEK TO HANG UP
      NOP                      ;ALLOW TIME FOR SEEK TO HANG UP
      NOP                      ;ALLOW TIME FOR SEEK TO HANG UP
      NOP                      ;ALLOW TIME FOR SEEK TO HANG UP
      JSR      PC,@#CLDISK      ;GIVE INIT
      MOV      @RHCC,@#SBDDAT   ;TEST DATA
      CMP      R5,@#SBDDAT      ;COMPARE CURRENT CYLINDER
      BEQ      1$              ;BRANCH IF GOOD
      MOV      R5,@#SGDDAT      ;GOOD VALUE OF RHCC
      MOV      @#RHCC,@#REGADR  ;FAILING REGISTER ADDRESS
      ERROR    30              ;CURRENT CYLINDER DOES NOT MATCH DESIRED CYLINDER
                                ;REGISTER AFTER A SEEK AND AN INIT

1$:
      MOV      (SP)+,R5         ;;POP STACK INTO R5
      RTS      R0

```

```

8705
8706
8707
8708
8709
8710
8711
8712
8713
8714
8715
8716      100000      PIE1      =100000
8717      040000      PIE2      =40000
8718      020000      PIE3      =20000
8719      010000      PIE4      =10000
8720      004000      PIE5      =4000
8721      002000      PIE6      =2000
8722      001000      PIE7      =1000
8723      000400      PIE8      =400
8724      000200      PIE9      =200
8725      000100      PIE10     =100
8726      000040      PIE11     =40
8727      000020      PIE12     =20
8728      000010      PIE13     =10
8729      000004      PIE14     =4
8730      000002      PIE15     =2
8731      000001      PIE16     =1
8732      100000      PIE17     =100000
8733      040000      PIE18     =40000
8734      020000      PIE19     =20000
8735      010000      PIE20     =10000
8736      004000      PIE21     =4000
8737      002000      PIE22     =2000
8738      001000      PIE23     =1000
8739      000400      PIE24     =400
8740      000200      PIE25     =200
8741      000100      PIE26     =100
8742      000040      PIE27     =40
8743      000020      PIE28     =20
8744      000010      PIE29     =10
8745      000004      PIE30     =4
8746      000002      PIE31     =2
8747      000001      PIE32     =1
8748
8749      050616      000000      EC DATA: 0      ;DATA BIT FOR ECC
8750
8751
8752
8753      050620      000000      GECC1: 0      ;LOW ORDER ECC WORD TO BE GENERATED HERE
8754
8755
8756      050622      000000      GECC2: 0      ;HIGH ORDER ECC WORD TO BE GENERATED HERE
8757
8758
8759      050624      000000      TSECCG: 0      ;IF =177777 GENERATE AND TEST ECC FOR THIS BIT
8760
      ;IF =0 DO NOT GENERATE AND TEST ECC FOR THIS BIT

```

8761			NCODE:	38859.		;N-CODE WORD
8762	050626	113713	NCOUNT:	0		;TEMPORARY N CODE
8763	050630	000000	POSIT1:	0		;POSITION REGISTER
8764	050632	000000	HARDER:	4129.		;HARD ERROR COUNT
8765	050634	010041				;TRUE COUNT IS 4128 BUT AS COMPARES ARE
8766						;DONE ONE STAGE LATER SO 4129
8767			DATENV:	0		;DATA ENVELOPE FOR TYPE OUT
8768	050636	000000				;MAX FOR WRITE IS 4096
8769						;MAX FOR READ IS 4128
8770			ZLODE:	0		;LEADING ZEROS ENVELOPE FOR TYPE OUT
8771	050640	000000				;THIS IS SHUT OFF WHEN POSITION COUNTER
8772						;IN ENABLED
8773						;MAX COUNT IS 38859
8774						
8775						
8776						
8777						
8778	050642	000000	HADTMP:	0		;TEMPORARY HARD ERROR COUNT
8779	050644	000000	P3:	0		
8780	050646	000000	P12:	0		
8781	050650	000000	P22:	0		
8782	050652	000000	P24:	0		
8783						
8784						
8785						
8786						
8787						
8788	050654		ECTEST:			
8789	050654	010046	MOV	R0,-(SP)	::PUSH R0 ON STACK	
8790	050656	010146	MOV	R1,-(SP)	::PUSH R1 ON STACK	
8791	050660	010246	MOV	R2,-(SP)	::PUSH R2 ON STACK	
8792	050662	010346	MOV	R3,(SP)	::PUSH R3 ON STACK	
8793	050664	010446	MOV	R4,-(SP)	::PUSH R4 ON STACK	
8794	050666	010546	MOV	R5,-(SP)	::PUSH R5 ON STACK	
8795	050670	013701	MOV	@#GECC1,R1	:ECC1 WORD	
8796	050674	013702	MOV	@#GECC2,R2	:ECC2 WORD	
8797	050700	005737	TST	@#ECDATA	:IS CURRENT BIT A ONE	
8798	050704	001406	BFC	2\$	:BRANCH IF CURRENT DATA D=0	
8799						
8800						
8801						
8802						
8803	050706	010103	1\$:	MOV	R1,R3	
8804	050710	052703		BIS	#^CPIE32,R3	
8805	050714	005103		COM	R3	
8806	050716	010300		MOV	R3,R0	
8807	050720	000404		BR	3\$	
8808						
8809						
8810						
8811	050722	010103	2\$:	MOV	R1,R3	
8812	050724	042703		BIC	#^CPIE32,F3	
8813	050730	010300		MOV	R3,R0	
8814						
8815	050732	000241	3\$:	CLC		
8816	050734	006000		ROR	R0	

CZRJHDO,RP04/5/6 DSKLS CTRLR2
CZRJHD.P12 01-MAR-79 09:10

MACY11 30A(1052) 24-MAY-79 15:07 PAGE 202
ECC GENERATION AND COMPARISON ROUTINE

F 16

SEQ 0200

8817	050736	006000		ROR	R0	
8818	050740	005700		TST	R0	
8819	050742	001462		BEQ	10\$	:BRANCH IF R0=0
8820				;INVERT	X2	
8821						
8822	050744	010203		MOV	R2,R3	
8823	050746	052703	137777	BIS	#^CPIE2,R3	
8824	050752	005103		COM	R3	
8825	050754	010337	050644	MOV	R3,@#P3	
8826	050760	006237	050644	ASR	@#P3	
8827						
8828				;INVERT	X11	
8829						
8830						
8831	050764	010203		MOV	R2,R3	
8832	050766	052703	177737	BIS	#^CPIE11,R3	
8833	050772	005103		COM	R3	
8834	050774	010337	050646	MOV	R3,@#P12	
8835	051000	006237	050646	ASR	@#P12	
8836						
8837				;INVERT	X21	
8838						
8839	051004	010103		MOV	R1,R3	
8840	051006	052703	173777	BIS	#^CPIE21,R3	
8841	051012	005103		COM	R3	
8842	051014	010337	050650	MOV	R3,@#P22	
8843	051020	006237	050650	ASR	@#P22	
8844						
8845				;INVERT	X23	
8846						
8847	051024	010103		MOV	R1,R3	
8848	051026	052703	176777	BIS	#^CPIE23,R3	
8849	051032	005103		COM	R3	
8850	051034	010337	050652	MOV	R3,@#P24	
8851	051040	006237	050652	ASR	@#P24	
8852						
8853				;NOW THAT R0 FOR POSITION 1		
8854				:	P3 FOR POSITION 3	
8855				:	P12 FOR POSITION 12	
8856				:	P22 FOR POSITION 22	
8857				:	P24 FOR POSITION 24	
8858				;ARE KNOWN THE ROTATE WILL BE DONE AND		
8859				;THESE BITS JAMED IN		
8860						
8861	051044	006002		ROR	R2	
8862	051046	006001		ROR	R1	
8863	051050	053700	050644	BIS	@#P3,R0	
8864	051054	053700	050646	BIS	@#P12,R0	
8865	051060	042702	120020	BIC	#PIE1:PIE3:PIE12,R2	
8866	051064	050002		BIS	R0,R2	
8867						
8868	051066	005000		CLR	R0	
8869	051070	053700	050650	BIS	@#P22,R0	
8870	051074	053700	050652	BIS	@#P24,R0	
8871	051100	042701	002400	BIC	#PIE22:PIE24,R1	
8872	051104	050001		BIS	R0,R1	

```

8873 051106 000404          BR      12$
8874
8875          ;THE PROGRAM COMES HERE IF R0=0
8876          ;SO AFTER ROTATE R0 GETS PUT INTO POSITION 1
8877
8878 051110 006002          10$:  ROR      R2
8879 051112 006001          ROR      R1
8880 051114 042702 100000    BIC      #PIE1,R2
8881 051120 010137 050620    12$:  MOV      R1,@#GECC1      ;SAVE ECC1
8882 051124 010237 050622    MOV      R2,@#GECC2      ;SAVE ECC2
8883 051130 005737 050624    TST      @#TSECCG      ;IS HARDWARE TO BE CHECKED
8884          ;IF =1777777 TEST HARDWARE
8885          ;IF = 0 DO NOT TEST HARDWARE
8886 051134 001432          BEQ      14$      ;BRANCH IF HARDWARE NOT TO BE CHECKED
8887
8888
8889          ;*CHECK HARDWARE
8890 051136 032777 000400 127774  BIT      #SW8,@SWR      ;IS SWITCH 8 SET
8891 051144 001005          BNE      15$      ;BRANCH IF SW8 IS SET
8892 051146 032777 000100 127764  BIT      #SW6,@SWR      ;IS SWITCH 6 SET
8893 051154 001401          BEQ      15$      ;BRANCH IF SW6 IS NOT SET
8894 051156 000421          BR       14$      ;IF SWITCH 8 IS NOT SET AND
8895          ;SWITCH 6 IS SET THEN
8896          ;DO NOT DO COMPARES
8897 051160 010146          15$:  MOV      R1,-(SP)      ;GOOD PATTERN REGISTER
8898 051162 042716 174000    BIC      #174000,(SP)      ;GET ONLY PATTERN BITS
8899 051166 022677 143616    CMP      (SP)+,@RHEC2      ;COMPARE PATTERN REGISTER
8900 051172 001404          BEQ      13$      ;BRANCH IF GOOD
8901          ;TO SAVE TIME
8902 051174 004737 045616    JSR      PC,@#PUTREG      ;SAVE REGISTERS
8903 051200 104035          ERROR    35      ;PATTERN REGISTER IN 11 BITS IN ERROR
8904 051202 000407          BR       14$      ;BRANCH OUT
8905 051204 023777 050632 143574 13$:  CMP      @#POSIT1,@RHEC1 ;COMPARE POSITION REGISTER
8906 051212 001403          BEQ      14$      ;BRANCH IF GOOD
8907          ;TO SAVE TIME
8908 051214 004737 045616    JSR      PC,@#PUTREG      ;SAVE REGISTERS
8909 051220 104035          ERROR    35      ;POSITION REGISTER IN ERROR
8910          ;"DATA ENVELOP" GIVES NUMBER OF CLOCK
8911          ;PULSES FROM BEGINING OF COMMAND
8912          ;THAT IS THE CLOCKS IN THE R/W DATA FIELD ENVELOPE
8913          ;
8914          ;IN A WRITE THERE ARE 10000 OCTAL CLOCKS
8915          ;IN A READ THERE ARE 10040 OCTAL CLOCKS
8916          ;
8917          ;
8918          ;"N-CODE ZEROS" GIVE THE NUMBER OF CLOCKS
8919          ;GIVEN FOR THE LEADING ZEROS FIELD
8920          ;MAX COUNT IS 113713 OCTAL
8921          ;
8922          ;"GOOD POSITION" GIVES NUMBER OF CLOCKS
8923          ;GIVEN AFTER LEADING ZEROS WHICH IS FOR THE DATA
8924          ;FIELD
8925          ;MAX COUNT IS 10040 OR 10041 OCTAL
8926
8927
8928 051222          14$:

```

```

8929 051222 012605      MOV      (SP)+,R5      ;;POP STACK INTO R5
8930 051224 012604      MOV      (SP)+,R4      ;;POP STACK INTO R4
8931 051226 012603      MOV      (SP)+,R3      ;;POP STACK INTO R3
8932 051230 012602      MOV      (SP)+,R2      ;;POP STACK INTO R2
8933 051232 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
8934 051234 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
8935 051236 000207      RTS        PC
8936                      .SBTTL  ECC GENERATION CONTROL ROUTINE
8937
8938                      ;*****
8939                      ;*THIS SUBROUTINE WILL CONTROL THE ECC GENERATION ROUTINE
8940                      ;*FOR ERROR CORRECTION PROCESS
8941                      ;*CALL JSR, PC,@#ECORR
8942                      ;* XP ;EXPECTED POSITION REGISTER WHEN CORRECTION IS COMPLETE
8943                      ;*****
8944
8945
8946 051240 000000      ERPOS: 0                      ;POSITION REG. WHEN CORRECTION IS COMPLETE
8947
8948
8949
8950 051242 010037 015132 015132  ECORR: MOV      R0,@#PCJSR      ;SAVE PC OF JSR + 4
8951 051246 162737 000004 015132  SUB      #4,@#PCJSR      ;SAVE PC OF JSR
8952 051254 012037 051240      MOV      (R0)+,@#ERPOS      ;GET POSITION REG. WHEN CORRECTION IS COMPLETE
8953 051260 010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
8954 051262 013701 014776      MOV      @#RHMR,R1      ;MAINTENANCE REGISTER
8955 051266 012711 000001      MOV      #DMD,@R1      ;SET DIAGNOSTIC MODE BIT
8956 051272 005037 050616      CLR      @#ECDATA      ;ECC DATA IS ZERO
8957
8958
8959
8960 051276 005737 050632      1$:   TST      @#POSIT1      ;IS SOFTWARE POSITION NON ZERO
8961 051302 001007      BNE      2$      ;BRANCH IF N-CODE S COMPLETE
8962 051304 005337 050630      DEC      @#NCOUNT      ;DECREMENT N-CODE
8963 051310 001001      BNE      6$      ;BRANCH IF N-CODE IS NOT COMPLETE
8964 051312 000403      BR       2$      ;BRANCH AS N-CODE IS COMPLETE
8965 051314 005237 050640      6$:   INC      @#ZCODE      ;INCREMENT CLOCKS GIVEN FOR LEADING ZEROS
8966 051320 000420      BR       3$      ;BRANCH AS N-CODE IS NOT COMPLETE
8967                      ;GO TO GIVE CLOCK AND TEST ECC
8968 051322 005237 050632      2$:   INC      @#POSIT1      ;INCREMENT SOFTWARE POSITION
8969 051326 023737 051240 050632  CMP      @#ERPOS,@#POSIT1 ;HAVE ENOUGH CLOCKS BEEN GIVEN TO DETECT ERROR
8970 051334 103012      BHS      3$      ;BRANCH IF MORE CLOCKS TO BE GIVEN
8971 051336 023737 050642 050632  CMP      @#HADTMP,@#POSIT1 ;HAVE ENOUGH CLOCKS BEEN GIVEN FOR HARD ERROR
8972                      ;THAT IS HAVE 4128 MORE CLOCKS BEEN GIVEN
8973                      BEQ      5$      ;BRANCH IF YES
8974 051344 001415      BIT      #ZER,@R1      ;CHECK ZERO DETECT BIT IN RHMR
8975 051346 032711 000400      BNE      4$      ;BRANCH IS ZER SET
8976 051352 001016      ;TO SAVE TIME
8977 051354 004737 045616      JSR      PC,@#PUTREG      ;SAVE REGISTERS
8978 051360 104034      ERROR 34      ;ZERO DETECT BIT NOT HIGH
8979                      ;WHEN 21 BITS IN ECC 32 BIT REGISTER IS 0
8980
8981
8982 051362 052711 000002      3$:   BIS      #MCLK,@R1      ;SET CLOCK
8983 051366 042711 000002      BIC      #MCLK,@R1      ;CLEAR CLOCK
8984 051372 004737 050654      JSR      PC,@#ECTEST      ;GO TO GENERATE AND TEST ECC

```

CZRJHDO,RP04/5/6 DSKLS CTRLR2 MACY11 30A(1052) 24-MAY-79 15:07 PAGE 205
CZRJHD.P12 01-MAR-79 09:10 ECC GENERATION CONTROL ROUTINE

SEQ 0203

```
8985 051376 000737          BR      18          ;CONTINUE
8986
8987          ;THIS EXTRA CLOCK IS TO BRING ECH HIGH
8988          ;AFTER THIS CLOCK POSITION REGISTER MAY BE 10040 OR 10041 OCTAL
8989
8990 051400 052711 000002    58:    BIS      #MCLK,@R1      ;SET CLOCK
8991 051404 042711 000002    BIC      #MCLK,@R1      ;CLEAR CLOCK
8992
8993 051410          48:
8994 051410 012601    MOV      (SP)+,R1      ;;POP STACK INTO R1
8995 051412 000200    RTS      R0
```

.SBTTL SOFTWARE DISK DATA ECC GEN. ROUTINE

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

```
8996  
8997  
8998  
8999  
9000  
9001  
9002  
9003  
9004 051414  
9005 051414 010046  
9006 051416 010146  
9007 051420 010246  
9008 051422 010346  
9009 051424 010446  
9010 051426 010546  
9011 051430 005037 050632  
9012 051434 005037 050620  
9013 051440 005037 050622  
9014 051444 012701 054726  
9015 051450 012702 000400  
9016 051454 012703 000020  
9017 051460 012104  
9018 051462 006004  
9019 051464 103004  
9020 051466 012737 177777 050616  
9021 051474 000402  
9022 051476 005037 050616  
9023 051502 004737 050654  
9024 051506 005303  
9025 051510 001364  
9026 051512 005302  
9027 051514 001357  
9028 051516 013737 050620 055726  
9029 051524 013737 050622 055730  
9030 051532 012605  
9031 051534 012604  
9032 051536 012603  
9033 051540 012602  
9034 051542 012601  
9035 051544 012600  
9036 051546 000207
```

FILLEC:

```
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 @#POSIT1 ; 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,@#ECCDATA ; ECC DATA BIT IS A ONE  
BR 12% ; BRANCH TO GENERATE ECC  
11%: CLR @#ECCDATA ; 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
```

9037
9038
9039
9040
9041
9042
9043
9044
9045 051550
9046 051550 104401 051556
9047 051554 000425
9048
9049 051630
9050 051630 013746 014756
9051 051634 104402
9052 051636 104401 051644
9053 051642 000425
9054
9055 051716
9056 051716 004737 060506
9057 051722 104412
9058 051724 012700 014746
9059 051730 012701 000026
9060 051734 012737 052534 000004
9061 051742 021637 014756
9062 051746 001431
9063 051750 005776 000000
9064 051754 163716 014756
9065 051760 061620
9066 051762 005301
9067 051764 001375
9068 051766 104401 051774
9069 051772 000417
9070
9071 052032
9072 052032 013746 014744
9073 052036 104402
9074 052040 104401 052046
9075 052044 000437
9076
9077 052144
9078 052144 104412
9079 052146 012637 014744
9080 052152 104401 052160
9081 052156 000416
9082
9083 052214
9084 052214 013746 014756
9085 052220 104402
9086 052222 104401 052230
9087 052226 000416
9088
9089 052264
9090 052264 013746 014744
9091 052270 104402
9092 052272 104401 052300

```
.SBTTL  RH BASE ADDRESS CHANGE ROUTINE

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

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,@#STKINT    ;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 TEST THIS ADDRESS
      CMP      @SP,@#RHCS1    ;NEW CSR?
      BEQ      1$            ;NO, THE OLD ONE WAS RETYPED
      TST      @0(SP)         ;ACCESS THE NEW ADDRESS
      SUB      @#RHCS1,@SP    ;GET THE ADDRESS OFFSET
      ADD      @SP,(R0)+      ;AND PLUG IT IN
      DEC      R1             ;ONE LESS ADDRESS TO CHANGE
      BNE      2$            ;BUT DO SOME MORE
      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
```

```

9093 052276 000417      BR      76$      ;;GET OVER THE ASCIZ
9094      ;;77$: .ASCIZ <15><12>/UNTIL PROGRAM IS RELOADED./
9095 052336      76$:      TYPE      79$      ;;TYPE ASCIZ STRING
9096 052336 104401 052344      BR      78$      ;;GET OVER THE ASCIZ
9097 052342 000402      ;;79$: .ASCIZ <15><12>/ /
9098      78$:      TYPE      81$      ;;TYPE ASCIZ STRING
9099 052350      BR      80$      ;;GET OVER THE ASCIZ
9100 052350 104401 052356      ;;81$: .ASCIZ <15><12>/UNLESS HALTED AND MANUALLY RESTARTED,/
9101 052354 000424      80$:      TYPE      83$      ;;TYPE ASCIZ STRING
9102      BR      82$      ;;GET OVER THE ASCIZ
9103 052426      ;;83$: .ASCIZ <15><12>/PROGRAM WILL AUTOMATICALLY RESTART FROM /
9104 052426 104401 052434      82$:      MOV      #RA,-(SP)
9105 052432 000426      TYP0C
9106      TYPE      85$      ;;TYPE ASCIZ STRING
9107 052510      BR      84$      ;;GET OVER THE ASCIZ
9108 052510 012746 000200      ;;85$: .ASCIZ <15><12>/ /
9109 052514 104402      84$:      JMP      @#BEGIN      ;ALL DONE, NOW START OVER!
9110 052516 104401 052524      ADTIMO: CMP      (SP)+,(SP)+
9111 052522 000402      TYPE      65$      ;;TYPE ASCIZ STRING
9112      BR      64$      ;;GET OVER THE ASCIZ
9113 052530      ;;65$: .ASCIZ <15><12>/SELECTED ADDRESS DID NOT RESPOND. /
9114 052530 000137 017360      64$:      JMP      @#BASECH
9115 052534 022626
9116 052536 104401 052544
9117 052542 000424
9118
9119 052614
9120 052614 000137 051550
9121
9122
9123
9124      ;*THIS IS A LITTLE ROUTINE THAT TESTS NED BIT 11 IN RHCS2
9125      ;*THIS LOOPS HERE FOR EVER
9126      ;*TO BE USED ONLY IF DRIVES PRESENT LOOKING AT NED DOES NOT AGREE
9127      ;*WITH WHAT IS REALY THERE
9128 052620 000000      ERUNIT: 0      ;UNIT UNDER MANUAL TEST
9129 052622 004737 046116      ERSTART:JSR      PC,@#CLDISK      ;SET GENERAL REG.
9130 052626 013712 052620      MOV      @#ERUNIT,@R2      ;SELECT UNIT
9131 052632 005714      1$:      TST      @R4      ;TEST RHER1
9132 052634 032712 010000      BIT      #NED,@R2      ;TEST NED
9133 052640 001401      BEQ      2$      ;BRANCH IF GOOD
9134 052642 000773      BR      1$      ;NED NOT SET
9135 052644 000772      2$:      BR      1$      ;NED SET

```

```
9136 .SBTTL DISK SIMULATION
9137 :*****
9138 :*****
9139 :*IN A WRITE HEADER AND DATA COMMAND FILL THE FOLLOWING
9140 :*WCLY=WITH CYLINDER TO BE ON DISK
9141 :*WSECTR=WITH SECTOR AND TRACK TO BE ON DISK
9142 :*WKEY1= WITH KEY1 TO BE ON DISK
9143 :*WKEY2= WITH KEY2 TO BE ON DISK
9144 :*FNWORD= NO OF DATA WORDS TO BE WRITTEN ON DISK
9145 :*THE COMMAND THEN IS JSR PC,COMWHD
9146 :
9147 :
9148 :
9149 :
9150 :*IN A WRITE DATA COMMAND FILL THE FOLLOWING
9151 :*CYL=WITH CYLINDER TO BE FOUND ON DISK
9152 :*SECOTR= WITH SECTOR AND TRACK TO BE FOUND ON DISK
9153 :*KEY1= WITH KEY1 TO BE FOUND ON DISK
9154 :*KEY2= WITH KEY2 TO BE FOUND ON DISK
9155 :*X= 1 MUST BE ONE
9156 :*NOWORD= WITH NUMBER OF DATA WORDS TO BE WRITTEN
9157 :*THE COMMAND THEN IS JSR PC,COMHD
9158 :
9159 :
9160 :
9161 :
9162 :
9163 :
9164 :*IN A READ HEADER AND DATA COMMAND FILL THE FOLLOWING
9165 :*CYL= WITH CYLINDER TO BE FOUND ON DISK
9166 :*SECOTR= WITH SECTOR AND TRACK TO BE FOUND ON DISK
9167 :*KEY1= WITH KEY1 TO BE FOUND ON DISK
9168 :*KEY2=WITH KEY2 TO BE FOUND ON DISK
9169 :*DAWORD= WITH NUMBER OF WORDS TO BE FOUND ON DISK
9170 :*X=0 MUST BE ZERO
9171 :*THE COMMAND THEN IS JSR PC,COMHD
9172 :
9173 :
9174 :
9175 :
9176 :
9177 :
9178 :
9179 :
9180 :
9181 :
9182 :*IN A READ DATA COMMAND FILL THE FOLLOWING
9183 :*CYL= WITH CYLINDER TO BE FOUND ON DISK
9184 :*SECOTR= WITH SECTOR AND TRACK TO BE FOUND ON DISK
9185 :*KEY1= WITH KEY1 TO BE FOUND ON DISK
9186 :*KEY2=WITH KEY2 TO BE FOUND ON DISK
9187 :*DAWORD= WITH NUMBER OF WORDS TO BE FOUND ON DISK
9188 :*X=0 MUST BE ZERO
9189 :*THE COMMAND THEN IS JSR PC,COMHD
9190 :
```

```
9191
9192
9193
9194
9195
9196
9197
9198
9199
9200
9201
9202
9203
9204
9205
9206
9207
9208
9209
9210
9211
9212
9213
9214
9215
9216
9217
9218
9219
9220
9221
9222
9223 052646 000000
9224 052650 011637 015132
9225 052654 162737 000004 015132
9226 052662 010046
9227 052664 010146
9228 052666 010246
9229 052670 010346
9230 052672 010446
9231 052674 010546
9232
9233 052676 012777 000001 142072
9234 052704 052777 000004 142064
9235 052712 042777 000004 142056
9236 052720 052777 000001 142030
9237
9238 052726 012737 000113 052646
9239 052734 005337 052646
9240 052740 001375
9241
9242 052742 013746 053012
9243 052746 042716 177740
9244 052752 012637 052762
9245 052756 004137 057056
9246

*****
: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 ISSUES DIAGNOSTIC MODE, AND EXTRA DIAGNOSTIC INDEX, AND THE
:'GO' BIT
:IT THEN CALLS THREE OTHER SUBROUTINES, WHICH IN TURN CALL
:OTHER SUBROUTINES. THE SUBROUTINES CALLED HERE ARE:

:      SEARCH      :ISSUES SECTOR CLOCKS TO SET SECTOR FOUND FLOP
:      RDHEAD      :READS THE SECTOR HEADER
:      WRDATA      :WRITES THE SECTOR DATA (WRITE OPERATION)
:      REDATA      :READS THE SECTOR DATA (READ OPERATION)

RUNCTR: .WORD 0
COMMD:  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

        MOV      #DMD,@RHMR      :SET DIAGNOSTIC MODE
        BIS      #MINX,@RHMR      :SET DIAGNOSTIC INDEX
        BIC      #MINX,@RHMR      :CLEAR DIAGNOSTIC INDEX
        BIS      #GO,@RHCS1       :ISSUE 'GO' BIT & STALL 'TILL 'RUN'
                                   :FUNCTION CODE WAS ISSUED BY THE TEST
        MOV      #75.,@#RUNCTR    :LOAD STALL COUNT = APPROX. 450US FOR 11/50 CPU
        DEC      @#RUNCTR         :COUNT DOWN ONE
        BNE      1$              :CONTINUE UNTIL = 0

        MOV      SECOTR, -(SP)     :GET DESIRED SECTOR/TRACK
        BIC      #177740,(SP)      :MAKE ONLY SECTOR
        MOV      (SP)+,@#TRK       :SAVE SECTOR
        JSR      R1,@#SEARCH       :ISSUE SECTOR CLOCKS <----->
```

9247	052762	000000	TRK:	.WORD	0	
9248	052764	012701		MOV	#+NOP,R1	;GOING TO MOVE NOPS
9249	052770	010137		MOV	R1,@#SSYN	;NOP INTO SSYN
9250	052774	010137		MOV	R1,@#HEDGAP	;NOP INTO HEDGAP
9251	053000	010137		MOV	R1,@#HEDSYN	;NOP INTO HEDSYN
9252	053004	004137		JSR	R1,@#RDHEAD	;READ THE HEADER <----->
9253						
9254	053010	000000	CYL:	.WORD	0	;CYLINDER ADDRESS
9255	053012	000000	SECOTR:	.WORD	0	;SECTOR/TRACK ADDRESS
9256	053014	000000	KEY1:	.WORD	0	;KEY1 WORD
9257	053016	000000	KEY2:	.WORD	0	;KEY2 WORD
9258	053020	000000	X:	.WORD	0	;X=1 WRITE COMMAND
9259						;X=0 READ COMMAND
9260						
9261						
9262						;DUMMY ERROR CALL LOCATIONS FOR THE READ HEADER OPERATION
9263						
9264	053022	000240	SSYN:	NOP		;IF 'ERROR 2' INSERTED BY RDHEAD
9265						;SUBROUTINE, THEN THE FIRST SYNC
9266						;IS NOT DETECTED. NO BAD DATA
9267						;IS GIVEN BECAUSE SYNC=144000
9268						;CANNOT BE READ. WORD NUMBER
9269						;IS '1' BECAUSE THIS IS THE FIRST
9270						;WORD TESTED.
9271						
9272	053024	000240	HEDGAP:	NOP		;IF 'ERROR 3' INSERTED BY
9273						;RDHEAD SUBROUTINE, THEN THE
9274						;HEADER GAP 0'S WERE NOT
9275						;WRITTEN RIGHT.
9276						
9277						;IF 'WORD NO' CONTAINS, SAY
9278						;3(8), THEN IT IS THE THIRD
9279						;WORD OF A 5 WORD HEADER
9280						;GAP THAT IS WRONG.
9281						
9282						; 'BAD DATA' CONTAINS WHAT IS
9283						;GOING ON THE DISK.
9284						
9285	053026	000240	HEDSYN:	NOP		;IF 'ERROR 3' INSERTED BY RDHEAD
9286						;SUBROUTINE, THEN THE HEADER SYNC
9287						;GENERATED BY DCL IS WRONG,
9288						;OR THE LAST BYTE
9289						;OF THE HEADER GAP 0'S IS WRONG.
9290						
9291						;IN EITHER CASE WORD NO=6,
9292						;RIGHT BYTE IS HEADER 0,
9293						;LEFT BYTE IS SYNC.
9294						
9295						; 'BAD DATA' HAS WHAT IS GOING
9296						;ON DISK.
9297						
9298						
9299	053030	005737	015124	TST	@#ERFLGS	;WERE ANY ERRORS DETECTED ?
9300	053034	001017		BNE	OUT	;IF YES, EXIT <----->
9301						
9302	053036	005737	053020	TST	@#X	;IS IT A DATA WRITE OPERATION ?

```

9303 053042 001410          BEQ    DAREAD    ;NO...THEN DO A DATA READ
9304 053044 005737 053124    TST    @#NOSYNC ;IS THIS FORCED HEADER ERROR COMMAND ?
9305                                ;IF YES NOSYNC=-1 THEN WRITE OR READ
9306                                ;IS SHUT OFF, SO BRANCH OUT
9307                                ;IF NOSYNC=0 THEN CONTINUE
9308 053050 001011          BNE    OUT        ;EXIT IF SET ----->
9309
9310 053052 004137 054376    JSR     R1,@#WRDATA ;WRITE DATA <----->
9311 053056 000000          NOWORD: .WORD 0    ;NO OF WORDS TO BE WRITTEN
9312 053060 000000          Y:      .WORD 0    ;
9313 053062 000404          BR      OUT        ;EXIT ----->
9314
9315 053064 004137 057332    DAREAD: JSR     R1,@#REDATA ;READ DATA <----->
9316 053070 000000          DAWORD: .WORD 0    ;NO OF WORDS TO BE READ
9317 053072 000000          .WORD 0
9318
9319 053074          OUT:      MOV     (SP)+,R5    ;;POP STACK INTO R5
9320 053074 012605          MOV     (SP)+,R4    ;;POP STACK INTO R4
9321 053076 012604          MOV     (SP)+,R3    ;;POP STACK INTO R3
9322 053100 012603          MOV     (SP)+,R2    ;;POP STACK INTO R2
9323 053102 012602          MOV     (SP)+,R1    ;;POP STACK INTO R1
9324 053104 012601          MOV     (SP)+,R0    ;;POP STACK INTO R0
9325 053106 012600          MOV     (SP)+,R0    ;;POP STACK INTO R0
9326 053110 000207          RTS     PC          ;EXIT
9327

```

```
9328
9329 ::*****
9330
9331 :*THE DISK SECTOR IS DEVIDED AS FOLLOWS
9332
9333 :*19 WORDS OF 0, ONE WORD 144000
9334 :*THESE MAKE 39 BYTES FOR SECTOR GAP AND ONE SYNC. BYTE
9335
9336
9337 053112 014400 RSYNC: 14400
9338 053114 000000 RCYL: 0
9339 053116 000000 RSETR: 0
9340 053120 000000 RKEY1: 0
9341 053122 000000 RKEY2: 0
9342
9343
9344 :*5 WORDS OF 0'S, ONE WORD 144000
9345 :*THESE MAKE 11 BYTES FOR HEADER GAP AND ONE SYNC. BYTE
9346 :*THESE ARE DCL GENERATED
9347
9348
9349
9350 :*THERE ARE 256 WORDS OF DATA
9351 :*THERE ARE 2 WORDS FOR ECC GENFRATED BY DCL
9352 :*15 WORDS OF 0'S FOR DATA GAP AND TOLERANCE GAP
9353 ::*****
9354
9355
9356
9357
9358
9359
```

```
9360          ;*****
9361          ;*READ DISK HEADER
9362          ;*****
9363
9364
9365
9366
9367
9368 053124 000000      NOSYNC: 0          ;FORCED HEADER ERROR = -1
9369                                     ;NORMAL = 0
9370 053126 000000      1Y: 0              ;ERROR TYPE NO.
9371 053130 000000      ERWORD: 0         ;ERROR WORD NO.
9372
9373
9374
9375
9376 053132 012137 053114      RDHEAD: MOV    (R1)+, @#RCYL      ;STORE CYLINDER ADDRESS
9377 053136 012137 053116      MOV    (R1)+, @#RSETR      ;STORE SECTOR AND TRACK ADDRESS
9378 053142 012137 053120      MOV    (R1)+, @#RKEY1      ;STORE KEY1
9379 053146 012137 053122      MOV    (R1)+, @#RKEY2      ;STORE KEY2
9380 053152 012137 053722      MOV    (R1)+, @#COMPA      ;STORE COMPARE OR NOT
9381 053156 010146      MOV    R1, -(SP)      ;PUSH R1 ON STACK
9382
9383 053160 013700 014776      MOV    @#RHMR, R0          ;R0 CONTAINS MAINTANENCE REG.
9384 053164 012705 000002      MOV    #2, R5            ;R5 IS A COUNTER FOR WORDS
9385 053170 012710 000001      MOV    #DMD, @RO          ;SET DIAG. MODE
9386 053174 052710 000010      BIS    #MSTCK, @RO        ;SET SECTOR CLOCK FOR FIRST WORD
9387 053200 052710 000002      BIS    #MCLK, @RO         ;SET MAINT. CLOCK FOR FIRST WORD
9388 053204 042710 000012      BIC    #MSTCK!MCLK, @RO    ;RESET THEM
9389
9390 053210 000404          BR    2$                ;DON'T GIVE SECTOR CLOCK FIRST TIME
9391 053212 012710 000013      1$: MOV    #MSTCK!MCLK!DMD, @RO ;SET SECTOR, CLOCK, DIAG. MODE, RESET INDEX
9392 053216 042710 000012      BIC    #MSTCK.MCLK, @RO    ;RESET SECTOR & MAINT. CLOCK
9393
9394 053222 012702 000007      2$: MOV    #7, R2          ;LOAD BYTE COUNTER
9395 053226 052710 000002      3$: BIS    #MCLK, @RO      ;SET MAINT. CLOCK
9396 053232 042710 000002      BIC    #MCLK, @RO         ;RESET IT
9397 053236 005302          DEC    R2                ;BYTE COUNTER
9398 053240 001372          BNE    3$                ;CONTINUE IF BYTE NOT COMPLETE
9399 053242 005305          DEC    R5                ;WORD COUNTER
9400 053244 001362          BNE    1$                ;CONTINUE IF WORD NOT COMPLETE
9401
9402 053246 012702 000022          MOV    #18, R2        ;LOAD NO OF WORDS OF ZEROS
9403 053252 005037 053720      4$: CLR    @#WORD        ;DO 0'S
9404 053256 004737 053724      JSR    PC, @#READ        ;READ 0'S <----->
9405 053262 005302          DEC    R2                ;COUNT DOWN WORDS
9406 053264 001372          BNE    4$                ;CONTINUE
9407
9408 053266 013737 053112 053720      MOV    @#RSYNC, @#WORD ;SYNC. WORD
9409 053274 004737 053724      JSR    PC, @#READ        ;READ IT <----->
9410 053300 032710 001000      BIT    #DISY, @RO        ;SYNC. BYTE DETECTED?
9411 053304 001012          BNE    5$                ;CONTINUE IF SYNC DETECTED
9412 053306 012737 000001 053130      MOV    #1, @#ERWORD ;ERROR WORD NO
9413 053314 013737 053112 001124      MOV    @#RSYNC, @#SGDDAT ;SYNC WORD
9414 053322 012737 104002 053022      MOV    #104002, @#SSYN ;INSERT 'ERROR 2' IN SSYN
9415 053330 000571          BR    13$                ;BRANCH OUT <----->
```

```
9416
9417 053332 013737 053114 053720 5$: MOV @#RCYL,@#WORD ;SETUP CYLINDER
9418 053340 004737 053724 JSR PC,@#READ ;READ IT <----->
9419 053344 013737 053116 053720 MOV @#RSETR,@#WORD ;SETUP SECTOR/TRACK
9420 053352 004737 053724 JSR PC,@#READ ;READ THEM <----->
9421 053356 013737 053120 053720 MOV @#RKEY1,@#WORD ;SETUP KEY1
9422 053364 004737 053724 JSR PC,@#READ ;READ IT <----->
9423 053370 013737 053122 053720 MOV @#RKEY2,@#WORD ;SETUP KEY2
9424 053376 004737 053724 JSR PC,@#READ ;READ IT <----->
9425 053402 013737 054710 053720 MOV @#WCRC,@#WORD ;SETUP CRC
9426 053410 004737 053724 JSR PC,@#READ ;READ IT <----->
9427
9428 053414 005737 015144 TST @#TESDTE ;IS THIS A DRIVE TIMING ERROR ?
9429 053420 001135 BNE 13$ ;BRANCH OUT IF YES <----->
9430 053422 005737 053722 TST @#COMPA ;IS THIS A READ OR WRITE COMMAND ?
9431 053426 001472 BEQ 11$ ;DO READ IF = 0
9432
9433 ;*CONTINUE WITH DIAGNOSTIC WRITE COMMAND
9434
9435 053430 012705 054712 MOV #HEGAP,R5 ;POINTER FOR HEADER GAP
9436 053434 012702 000005 MOV #5,R2 ;NO OF WORDS OF ZEROS
9437 053440 012737 000006 053130 6$: MOV #6,@#ERWORD ;ERROR WORD NO SET
9438 053446 004737 054156 JSR PC,@#WRITE ;FOR HEADER GAP
9439 053452 005737 054154 TST @#WORD ;TEST WRITTEN WORD
9440 053456 001413 BEQ 7$ ;CONTINUE IF GOOD, THAT IS = 0
9441 053460 160237 053130 SUB R2,@#ERWORD ;WORD NO IN ERROR
9442 053464 005037 001124 CLR @#$GDDAT ;GOOD WORD SHOULD BE 0
9443 053470 013737 054154 001126 MOV @#WORD,$BDDAT ;BAD DATA
9444 053476 012737 104003 053024 MOV #104003,@#HEDGAP ;"ERROR 2" GOES IN HEDGAP
9445 053504 000503 BR 13$ ;BRANCH OUT <----->
9446
9447 053506 013725 054154 7$: MOV @#WORD,(R5)+ ;SAVE HEADER GAP
9448 053512 005302 DEC R2
9449 053514 001351 BNE 6$
9450 053516 004737 054156 JSR PC,@#WRITE ;WRITE HEADER (DATA) GAP SYNC
9451 053522 023737 053112 054154 CMP @#RSYNC,@#WORD
9452 053530 001426 BEQ 10$
9453 053532 005737 053124 TST @#NOSYNC ;IS THIS FORCED HEADER ERROR COMMAND ?
9454 ;IF YES NOSYNC=-1 THEN WRITE OR READ
9455 ;IS SHUT OFF SO BRANCH OUT
9456 ;IF NO NOSYNC=0 THEN CONTINUE
9457 053536 001406 BEQ 14$ ;PRINT IT IF TRUE ERROR
9458
9459 053540 005737 054154 TST @#WORD ;IS IT = 0 ?
9460 053544 001420 BEQ 10$ ;CONTINUE IF GOOD
9461 053546 005037 001124 CLR @#$GDDAT ;IT SHOULD BE ZERO
9462 053552 000403 BR 15$ ;BRANCH TO TYPE ERROR
9463 053554 013737 053112 001124 14$: MOV @#RSYNC,@#$GDDAT ;GOOD DATA
9464 053562 013737 054154 001126 15$: MOV @#WORD,@#$BDDAT ;BAD DATA
9465 053570 012737 000006 053130 MOV #6,@#ERWORD
9466 053576 012737 104003 053026 MOV #104003,@#HEDSYN
9467 053604 000443 BR 13$ ;BRANCH OUT <----->
9468
9469 053606 013725 054154 10$: MOV @#WORD,(R5)+ ;SAVE DATA SYNC.
9470 053612 000440 BR 13$ ;EXIT <----->
9471
```

CZRJHDO,RP04/5/6 DSKLS CTRLR2
CZRJHD.P12 01-MAR-79 09:10

MACY11 30A(1052) 24-MAY-79 15:07 PAGE 216
DISK SIMULATION

SEQ 0214

```
9472          ;*READ COMMAND START FROM HERE
9473
9474 053614 012702 000005      11$: MOV    #5,R2
9475 053620 005037 053720      12$: CLR    WORD
9476 053624 004737 053724      JSR    PC,@#READ      ;READ HEADER GAP <----->
9477 053630 005302              DEC    R2              ;ARE 5 HEADER GAP ZEROS COMPLETE ?
9478 053632 001372              BNE    12$              ;IF NOT CONTINUE
9479 053634 013737 053112 053720  MOV    @#RSYNC,@#WORD ;SYNC WORD
9480 053642 004737 053724      JSR    PC,@#READ      ;READ HEADER (DATA) SYNC)
9481 053646 005737 053124      TST    @#NOSYNC        ;FORCED SYNC ERROR ?
9482 053652 001404              BEQ    16$              ;IF NOT ERROR COMMAND CONTINUE
9483 053654 032710 001000      BIT    #DTSY,@R0        ;SYNC. DETECTED
9484 053660 001415              BEQ    13$              ;IF ZERO BRANCH OUT ----->
9485 053662 000403              BR     17$              ;IF NOT ZERO BRANCH TO ERROR
9486
9487 053664 032710 001000      16$: BIT    #DTSY, @R0    ;SYNC. DETECTED ?
9488 053670 001011              BNE    13$              ;EXIT IF YES ----->
9489 053672 012737 000006 053130 17$: MOV    #6,@#ERWORD ;ERROR WORD NO.
9490 053700 013737 053112 001124  MOV    @#RSYNC,@#SGDDAT;SYNC WORD
9491 053706 012737 104002 053026  MOV    #104002,@#HEDSYN;MOVE "ERROR 2"
9492 053714              13$:
9493 053714 012601      MOV    (SP)+,R1      ;POP STACK INTO R1
9494 053716 000201      RTS    R1            ;EXIT ----->
9495
9496
9497
9498
9499
9500
9501
9502
```

```
9503          ;*****
9504          ;*READ ONE WORD IN 'WORD'
9505          ;*****
9506
9507
9508
9509
9510 053720 000000          WORD: 0
9511 053722 000000          COMPA: 0
9512
9513
9514
9515
9516          READ:
9517 053724 010246          MOV      R2,-(SP)          ;;PUSH R2 ON STACK
9518 053726 012705 000002  MOV      #2,R5          ;;WORD COUNTER
9519 053732 012710 000001  MOV      #DMD,@R0          ;;SET DIAG. MODE
9520 053736 006037 053720  ROR      @#WORD          ;;CHECKING IF THERE IS A ONE
9521 053742 103002          BCC      1$          ;;IF NO ONE BRANCH
9522 053744 052710 000020  BIS      #MRD,@R0          ;;SET BIT 4 IF DATA HAS ONE
9523 053750 012702 000007  1$: MOV      #7,R2          ;;BYTE COUNTER
9524 053754 052710 000012  BIS      #MSTCK!MCLK,@R0  ;;SET CLOCK,DATA IF ANY, SECTOR
9525 053760 005737 050624  TST      @#TSECCG          ;;IS THIS BIT TO GENERATE AND TEST ECC ?
9526 053764 001411          BEQ      6$          ;;BRANCH IF NO
9527 053766 032710 000020  BIT      #MRD,@R0          ;;IS DATA BIT A ONE ?
9528 053772 001404          BEQ      5$          ;;BRANCH IF DATA BIT IS 0
9529 053774 012737 177777 050616 MOV      #-1,@#ECDATA          ;;ECC DATA BIT IS A ONE
9530 054002 000402          BR       6$          ;;BRANCH
9531 054004 005037 050616  5$: CLR      @#ECDATA          ;;ECC DATA BIT IS A 0
9532 054010 012746 000001  6$: MOV      #DMD,-(SP)          ;;KEEP ONLY DIAG. MODE
9533 054014 006037 053720  ROR      @#WORD          ;;CHECKING IF THERE IS A ONE
9534 054020 103002          BCC      2$          ;;IF NO ONE BRANCH
9535 054022 012716 000021  MOV      #MRD!DMD,(SP)      ;;KEEP DATA AND DIAG. MODE
9536 054026 012610          2$: MOV      (SP)+,@R0          ;;PUT IN DATA,RESET CLOCK, SECTOR
9537 054030 005737 050624  TST      @#TSECCG          ;;IS ECC TO BE GENERATED FOR THIS BIT
9538 054034 001404          BEQ      3$          ;;BRANCH IF NO
9539 054036 005237 050636  INC      @#DATENV          ;;NUMBER OF CLOCKS GIVEN FOR DATA ENVELOPE
9540 054042 004737 050654  JSR      PC,@#ECTEST          ;;GO TO GENERATE AND TEST ECC
9541 054046 052710 000002  3$: BIS      #MCLK,@R0          ;;SET CLOCK
9542 054052 005737 050624  TST      @#TSECCG          ;;IS THIS BIT TO GENERATE ECC
9543 054056 001411          BEQ      8$          ;;BRANCH IF NO
9544 054060 032710 000020  BIT      #MRD,@R0          ;;IS DATA BIT A ONE
9545 054064 001404          BEQ      7$          ;;BRANCH IF DATA BIT IS = 0
9546 054066 012737 177777 050616 MOV      #-1,@#ECDATA          ;;ECC DATA BIT IS A ONE
9547 054074 000402          BR       8$          ;;BRANCH
9548 054076 005037 050616  7$: CLR      @#ECDATA          ;;ECC DATA BIT IS = 0
9549 054102 012746 000001  8$: MOV      #DMD,-(SP)          ;;KEEP DIAG. MODE
9550 054106 006037 053720  ROR      @#WORD          ;;CHECKING IF THERE IS A ONE
9551 054112 103002          BCC      4$          ;;BRANCH IF NO ONE
9552 054114 012716 000021  MOV      #MRD!DMD,(SP)      ;;KEEP DIAG. MODE AND DATA
9553 054120 012610          4$: MOV      (SP)+,@R0          ;;SET DATA, DIAG. MODE, CLEAR CLOCK
9554 054122 005737 050624  TST      @#TSECCG          ;;IS THIS BIT TO GENERATE ECC
9555 054126 001404          BEQ      9$          ;;BRANCH IF NO
9556 054130 005237 050636  INC      @#DATENV          ;;NUMBER OF CLOCKS FOR DATA ENVELOPE
9557 054134 004737 050654  JSR      PC,@#ECTEST          ;;GO TO GENERATE AND TEST ECC
9558
```

9559	054140	005302	98:	DEC	R2	;BYTE COUNTER
9560	054142	001341		BNE	38	;CONTINUE IF ONE BYTE NOT COMPLETE
9561	054144	005305		DEC	R5	;WORD COUNTER
9562	054146	001300		BNE	18	;CONTINUE IF ONE WORD NOT COMPLETE
9563	054150	012602		MOV	(SP)+,R2	;POP STACK INTO R2
9564	054152	000207		RTS	PC	;EXIT ----->
9565						
9566						
9567						
9568						
9569						
9570						

```
9571                                     ;*****
9572                                     ;*WRITE ONE WORD WHICH COMES BACK IN 'WORD'*
9573                                     ;*****
9574
9575
9576
9577
9578
9579
9580 054154 000000                      WORD: 0
9581
9582
9583
9584
9585 054156                      WRITE:
9586 054156 010046                      MOV R0,-(SP)      ;PUSH R0 ON STACK
9587 054160 010246                      MOV R2,-(SP)      ;PUSH R2 ON STACK
9588 054162 010346                      MOV R3,-(SP)      ;PUSH R3 ON STACK
9589 054164 010546                      MOV R5,-(SP)      ;PUSH R5 ON STACK
9590 054166 012705 000002              MOV #2,R5          ;WORD COUNTER
9591 054172 012710 000001              MOV #1,@R0         ;SET DIAG. MODE
9592 054176 012702 000007              1$: MOV #7,R2       ;BYTE COUNTER
9593 054202 012710 000013              MOV #MSTCK!MCLK!DMD,@R0 ;SET SECTOR & MANT. CLOCKS
9594 054206 032710 000040              BIT #MWR,@R0       ;CHECK WRITEBIT IN MAINT. REG.
9595 054212 001406                      BEQ 2$             ;BRANCH IF ZERO
9596 054214 012737 177777 050616      MOV #-1,@#ECDATA   ;ECC DATA BIT IS A ONE
9597 054222 000261                      SEC               ;SET CARRY
9598 054224 006003                      ROR R3             ;MOVE 1 FORWARD
9599 054226 000404                      BR 3$              ;
9600 054230 005037 050616              2$: CLR @#ECDATA   ;ECC DATA BIT IS = 0
9601 054234 000241                      CLC               ;CLEAR CARRY
9602 054236 006003                      ROR R3             ;MOVE 0 FOR WORD
9603 054240 012710 000001              3$: MOV #DMD,@R0    ;CLEAR SECTOR AND CLOCK
9604 054244 005737 050624              TST @#TSECCG       ;IS THIS BIT TO GENERATE ECC ?
9605 054250 001404                      BEQ 4$             ;BRANCH IF NO
9606 054252 005237 050636              INC @#DATENV        ;NUMBER OF CLOCKS FOR DATA ENVELOPE
9607 054256 004737 050654              JSR PC,@#ECTEST     ;GO TO GENERATE AND TEST ECC <----->
9608
9609 054262 052710 000002              4$: BIS #MCLK,@R0    ;SET CLOCK
9610 054266 032710 000040              BIT #MWR,@R0       ;CHECK WRITE BIT IN MAINT. REG.
9611 054272 001406                      BEQ 5$             ;BRANCH IF ZERO
9612 054274 012737 177777 050616      MOV #-1,@#ECDATA   ;ECC DATA BIT IS A ONE
9613 054302 000261                      SEC               ;SET CARRY
9614 054304 006003                      ROR R3             ;MOVE 1 FOR WORD
9615 054306 000404                      BR 6$              ;
9616 054310 005037 050616              5$: CLR @#ECDATA   ;ECC DATA BIT IS ZERO
9617 054314 000241                      CLC               ;CLEAR CARRY
9618 054316 006003                      ROR R3             ;MOVE 0 FOR WORD
9619 054320 012710 000001              6$: MOV #DMD,@R0    ;CLEAR CLOCK
9620 054324 005737 050624              TST @#TSECCG       ;IS THIS BIT TO GENERATE ECC ?
9621 054330 001404                      BEQ 7$             ;BRANCH IF NO
9622 054332 005237 050636              INC @#DATENV        ;NUMBER OF CLOCKS FOR DATA ENVELOPE
9623 054336 004737 050654              JSR PC,@#ECTEST     ;GO TO GENERATE AND TEST ECC <----->
9624
9625 054342 005302              7$: DEC R2                 ;COUNT FOR BYTE END
9626 054344 001346                      BNE 4$             ;IF NOT BYTE END CONTINUE
```

CZRJHDO,RP04/5/6 DSKLS CTRLR2
CZRJHD.P12 01-MAR-79 09:10

MACY11 30A(1052) 24-MAY-79 15:07 L 1 PAGE 220
DISK SIMULATION

SEQ 0218

9627	054346	005305		DEC	R5	;COUNT FOR WORD END
9628	054350	001312		BNE	18	;IF NOT WORD END CONTINUE
9629	054352	010337	054154	MOV	R3,@#WORD	;STORE THE WORD
9630	054356	012605		MOV	(SP)+,R5	::POP STACK INTO R5
9631	054360	012603		MOV	(SP)+,R3	::POP STACK INTO R3
9632	054362	012602		MOV	(SP)+,R2	::POP STACK INTO R2
9633	054364	012600		MOV	(SP)+,R0	::POP STACK INTO R0
9634	054366	000207		RTS	PC	;EXIT ----->
9635						
9636						
9637						
9638						
9639						
9640						

```
9641
9642
9643
9644
9645
9646
9647
9648
9649
9650
9651 054370 000000
9652 054372 000400
9653 054374 000000
9654 054376
9655 054376 ^11137 054370
9656 054402 012102
9657 054404 012137 053722
9658 054410 010046
9659 054412 010146
9660 054414 010246
9661 054416 010346
9662 054420 010446
9663 054422 012701 000016
9664 054426 012703 055734
9665 054432 012723 177777
9666 054436 005301
9667 054440 001374
9668
9669 054442 013700 014776
9670 054446 013746 054372
9671 054452 163716 054370
9672 054456 011637 054374
9673 054462 012604
9674 054464 005737 015142
9675 054470 001403
9676 054472 012737 177777 050624
9677 054500 012703 054726
9678 054504 004737 054156
9679 054510 013723 054154
9680 054514 005302
9681 054516 001372
9682 054520 005704
9683 054522 001406
9684
9685 054524 004737 054156
9686 054530 013723 054154
9687 054534 005304
9688 054536 001372
9689 054540 005037 050624
9690 054544 012701 000002
9691 054550 004737 054156
9692 054554 013723 054154
9693 054560 005301
9694 054562 001372
9695 054564 004737 054156
9696 054570 013723 054154

*****
*WRITE DATA HOUSEKEEPING ROUTINE
*****

COUNT: 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 NOT, CONTINUE

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$: MOV #DISK,R3 ;SIMULATED DISK AREA
2$: JSR PC,@#WRITE ;WRITE ON SIMULATED DISK
MOV @#WORD,(R3)+ ;STORE ON SIMULATED DISK
DEC R2
BNE 2$
TST R4 ;ANY ZEROS TO BE WRITTEN ?
BEQ 4$ ;BRANCH IF NONE TO BE WRITTEN

3$: JSR PC,@#WRITE ;WRITE ZEROS ON SIMULATED DISK
MOV @#WORD,(R3)+ ;STORE THEM
DEC R4
BNE 3$
4$: CLR @#TSECCG ;NO MORE ECC TO BE GENERATED
MOV #2,R1
5$: JSR PC,@#WRITE ;WRITE ECC1 AND ECC2 ON SIMULATED DISK
MOV @#WORD,(R3)+ ;STORE IN WEEC1 AND WEEC2
DEC R1
BNE 5$
JSR PC,@#WRITE ;WRITE DATA GAP
MOV @#WORD,(R3)+ ;STORE IT
```

CZRJHDO,RP04/5/6 DSKLS CTRLR2
CZRJHD.P12 01-MAR-79 09:10

MACY11 30A(1052) 24-MAY-79 15:07 PAGE 222
DISK SIMULATION

N 1

SEQ 0220

9697	054574	012701	000016		MOV	#14.,R1	
9698	054600	004737	054156	68:	JSR	PC,@#WRITE	;WRITE TOLERANCE GAP ZEROS
9699	054604	013723	054154		MOV	@#WORD,(R3)+	;STORE THEM
9700	054610	005301			DEC	R1	
9701	054612	001372			BNE	68	
9702	054614	012604			MOV	(SP)+,R4	::POP STACK INTO R4
9703	054616	012603			MOV	(SP)+,R3	::POP STACK INTO R3
9704	054620	012602			MOV	(SP)+,R2	::POP STACK INTO R2
9705	054622	012601			MOV	(SP)+,R1	::POP STACK INTO R1
9706	054624	012600			MOV	(SP)+,R0	::POP STACK INTO R0
9707	054626	000201			RTS	R1	;EXIT ----->
9708							
9709							
9710							
9711							
9712							
9713							
9714							
9715							

9716
9717
9718
9719
9720
9721
9722
9723
9724
9725
9726
9727
9728
9729
9730
9731
9732
9733
9734
9735
9736
9737
9738
9739
9740
9741
9742
9743
9744
9745

; *THIS IS THE SIMULATED DISK
; *ONLY ONE SECTOR OF SPACE IS ALLOCATED
;
;

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:		; (ALSO 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

```
9746 ;*****
9747 ;*WRITE HEADER AND DATA
9748 ;*****
9749
9750
9751
9752
9753
9754 ;*THIS SUBROUTINE IS THE FIRST IN A SERIES OF NESTED SUBROUTINES
9755
9756 ;*IT ISSUES DIAGNOSTIC MODE, AN EXTRA DIAGNOSTIC INDEX, AND THE
9757 ;*'GO' BIT
9758
9759 ;*IT THEN CALLS THREE OTHER SUBROUTINES, WHICH IN TURN CALL OTHER
9760 ;*SUBROUTINES. THE THREE SUBROUTINES CALLED HERE ARE:
9761
9762 ;*      SEARCH      ;ISSUES SECTOR CLOCKS TO SET SECTOR FOUND FLOP
9763 ;*      WRHEAD      ;WRITES THE SECTOR HEADER
9764 ;*      WRDATA      ;WRITES THE ACTUAL SECTOR DATA
9765
9766 ;*ALL OF THE ABOVE MENTIONED 'WRITING' IS ACTUALLY DONE INTO A CORE
9767 ;*BUFFER AREA CALLED 'DISK' VIA THE MAINTENANCE REGISTER (RHMR)
9768
9769
9770
9771
9772
9773
9774 055770 000000      RNCTR1: .WORD 0      ;'RUN' LINE STALL COUNTER
9775
9776 055772 011637 015132 COMWHD: MOV (SP),@#PCJSR ;SAVE PC OF JSR + 4
9777 055776 162737 000004 015132 SUB #4,@#PCJSR ;SAVE PC OF JSR
9778 056004 010046      MOV R0,-(SP) ;:PUSH R0 ON STACK
9779 056006 010146      MOV R1,-(SP) ;:PUSH R1 ON STACK
9780 056010 010246      MOV R2,-(SP) ;:PUSH R2 ON STACK
9781 056012 010346      MOV R3,-(SP) ;:PUSH R3 ON STACK
9782 056014 010446      MOV R4,-(SP) ;:PUSH R4 ON STACK
9783 056016 010546      MOV R5,-(SP) ;:PUSH R5 ON STACK
9784
9785 056020 012777 000001 136750 MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE
9786 056026 052777 000004 136742 BIS #MINX,@RHMR ;SET DIAGNOSTIC INDEX
9787 056034 042777 000004 136734 BIC #MINX,@RHMR ;CLEAR DIAGNOSTIC INDEX
9788 056042 052777 000001 136706 BIS #GO,@RHCS1 ;SET 'GO' BIT & STALL 'TILL 'RUN'
9789 ;(FUNCTION CODE WAS SET UP BY THE TEST)
9790 056050 012737 000113 055770 RNWAT1: MOV #75.,@#RNCTR1 ;LOAD STALL COUNTER - APPROX 450US
9791 ;FOR 11/50 CPU
9792 056056 005337 055770 1$: DEC @#RNCTR1 ;COUNT DOWN 1 TIME
9793 056062 001375      BNE 1$ ;CONTINUE UNTIL = 0
9794
9795 056064 013746 056150      MOV @#WSECTR,-(SP) ;GET DESIRED SECTOR/TRACK
9796 056070 042716 177740      BIC #177740,(SP) ;MAKE ONLY SECTOR
9797 056074 012637 056104      MOV (SP)+,@#WTRK ;SAVE SECTOR
9798 056100 004137 057056      2$: JSR R1,@#SEARCH ;ISSUE SECTOR CLOCKS TO GET TO
9799 ;DESIRED SECTOR <----->
9800
9801 056104 000000      WTRK: .WORD 0 ;SECTOR NO.
```

```
9802 056106 012701 000240      MOV      #*NOP,R1      ;GOING TO MOVE NOPS
9803 056112 010137 056160      MOV      R1,#SEGPER    ;NOP INTO SEGAP
9804 056116 010137 056162      MOV      R1,#FSYNER    ;NOP INTO FSYNER
9805 056122 010137 056164      MOV      R1,#ERHEAD    ;NOP INTO ERHEAD
9806 056126 010137 056166      MOV      R1,#ERCRC     ;NOP INTO ERCRC
9807 056132 010137 056170      MOV      R1,#ERHDGP    ;NOP INTO ERHDGAP
9808 056136 010137 056172      MOV      R1,#HDESYN    ;NOP INTO HDESYN
9809
9810 056142 004137 056242      JSR      R1,#WRHEAD    ;WRITE THE HEADER <----->
9811
9812 056146 000000      WCYL: 0      ;CYLINDER
9813 056150 000000      WSECTR: 0     ;SECTOR AND TRACK
9814 056152 000000      WKEY1: 0     ;KEY1
9815 056154 000000      WKEY2: 0     ;KEY2
9816 056156 000000      GCRC: 0      ;GOOD CRC
9817
9818
9819      ;DUMMY ERROR CALL LOCATIONS FOR THE WRITE HEADER OPERATION
9820
9821 056160 000240      SEGP: NOP      ;IF "ERROR 6" INSERTED BY
9822      ;WRHEAD SUBROUTINE, THEN
9823      ;SECTOR GAP GOING ON DISK
9824      ;IS NOT RIGHT.
9825
9826      ;WORD NO. CONTAINS WHICH
9827      ;WORD IS WRONG, THAT IS
9828      ;FIRST OF TENTH, OR WHAT EVER.
9829
9830      ;BAD WORD IS WHAT IS GOING ON DISK
9831
9832 056162 000240      FSYNER: NOP    ;IF "ERROR 6" INSERTED BY
9833      ;WRHEAD SUBROUTINE, THEN
9834      ;THE LAST 0 BYTE OF SECTOR
9835      ;GAP OF FIRST SYNC. BYTE
9836      ;AFTER SECTOR GAP IS IN
9837      ;ERROR.
9838
9839      ;WORD NO. CONTAINS 20
9840      ;RIGHT BYTE IS SECTOR GAP
9841      ;LEFT BYTE IS SYNC. BYTE.
9842
9843      ;BAD WORD IS WHAT IS GOING ON
9844      ;DISK
9845
9846 056164 000240      ERHEAD: NOP    ;IF "ERROR 6" INSERTED BY
9847      ;WRHEAD SUBROUTINE, THEN
9848      ;HEADER GOING ON DISK
9849      ;IS WRONG.
9850
9851      ;WORD NO 1 = CYLINDER NO
9852      ;WORD NO 2 = SECTOR/TRACK
9853      ;WORD NO 3 = KEY1
9854      ;WORD NO 4 = KEY2
9855
9856      ;BAD WORD IS WHAT IS GOING ON
9857
```

```

9858                                     ;DISK
9859
9860 056166 000240          ERCRC: NOP
9861                                     ;IF 'ERROR 6' INSERTED BY
9862                                     ;WRHEAD SUBROUTINE, THEN CRC WRITTEN
9863                                     ;ON DISK IS IN ERROR.
9864
9865                                     ;GOOD DATA IS WHAT SHOULD BE ON DISK
9866                                     ;BAD DATA IS WHAT IS GOING ON DISK.
9867
9868                                     ;WORD NO IS 5
9869
9870 056170 000240          ERHDGP: NOP
9871                                     ;IF 'ERROR 6' INSERTED BY
9872                                     ;WRHEAD SUBROUTINE, THEN HEADER
9873                                     ;GAP GOING ON DISK IS WRONG.
9874
9875                                     ;WORD NO GIVES WHICH OF
9876                                     ;THE HEADER GAP WORDS
9877                                     ;ARE WRONG. FOR EXAMPLE:
9878
9879                                     ;WORD NO 1 = FIRST HEADER
9880                                     ;GAP WORD
9881                                     ;BAD WORD IS WHAT IS GOING ON DISK
9882
9883 056172 000240          HDESYN: NOP
9884                                     ;IF 'ERROR 6' INSERTED BY
9885                                     ;WRHEAD SUBROUTINE, THEN LAST
9886                                     ;HEADER GAP BYTE OR HEADER
9887                                     ;SYNC BYTE GOING ON DISK IS WRONG.
9888
9889                                     ;WORD NO = 5
9890
9891                                     ;BAD DATA IS WHAT IS GOING
9892                                     ;ON DISK, RIGHT BYTE IS HEADER
9893                                     ;GAP 0'S BYTE, LEFT BYTE IS HEADER
9894                                     ;GAP SYNC.
9895
9896
9897 056174 005737 015124          TST      @#ERFLGS      ;ARE ANY ERRORS DETECTED ?
9898 056200 001004          BNE      FOUT                ;IF YES BRANCH
9899 056202 004137 054376          JSR      R1,@#WRDATA    ;WRITE THE DATA
9900
9901 056206 000000          FNWORD: .WORD 0                ;FORMAT COMMAND NO. OF DATA
9902 056210 000000          .WORD 0
9903
9904          FOUT:
9905 056212 012605          MOV      (SP)+,R5              ;;POP STACK INTO R5
9906 056214 012604          MOV      (SP)+,R4              ;;POP STACK INTO R4
9907 056216 012603          MOV      (SP)+,R3              ;;POP STACK INTO R3
9908 056220 012602          MOV      (SP)+,R2              ;;POP STACK INTO R2
9909 056222 012601          MOV      (SP)+,R1              ;;POP STACK INTO R1
9910 056224 012600          MOV      (SP)+,R0              ;;POP STACK INTO R0
9911 056226 000207          RTS      PC                    ;EXIT

```

```
9912
9913
9914
9915
9916
9917
9918
9919
9920
9921
9922
9923
9924
9925
9926
9927
9928
9929
9930 056230 000000
9931 056232 000000
9932 056234 000000
9933 056236 000000
9934 056240 000000
9935
9936
9937 056242 012137 056230
9938 056246 012137 056232
9939 056252 012137 056234
9940 056256 012137 056236
9941 056262 012137 056240
9942 056266 010146
9943
9944 056270 012701 054630
9945 056274 013700 014776
9946 056300 012710 000001
9947 056304 012705 000002
9948 056310 052710 000010
9949 056314 012710 000013
9950
9951 056320 032710 000040
9952 056324 001403
9953 056326 000261
9954 056330 006003
9955 056332 000402
9956 056334 000241
9957 056336 006003
9958 056340 012710 000001
9959 056344 012702 000007
9960 056350 052710 000002
9961 056354 032710 000040
9962 056360 001403
9963
9964 056362 000261
9965 056364 006003
9966 056366 000402
9967 056370 000241

*****
*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, @R0 ; SET DIAG. MODE
MOV #2, R5 ; WORD COUNTER
BIS #MSTCK, @R0 ; SET SECTOR FOR FIRST BYTE
1$: MOV #MSTCK!MCLK!DMD, @R0 ; SET SECTOR, CLOCK, DIAG.
; MODE, RESET INDEX
; CHECK WRITE BIT IN MAINT. REG.
BIT #MWR, @R0
BEQ 2$
SEC ; SET CARRY
ROR R3 ; MOVE ONE FORWARD
BR 3$
2$: CLC ; CLEAR CARRY
ROR R3 ; MOVE ZERO FORWARD
3$: MOV #DMD, @R0 ; CLEAR CLOCK, SECTOR
MOV #7, R2 ; BYTE COUNTER
4$: BIS #MCLK, @R0 ; SET BIT CLOCK
BIT #MWR, @R0 ; CHECK WRITE BIT IN MAINT.REG.
BEQ 5$ ; BRANCH IF ZERO
5$: CLC ; SET CARRY
ROR R3 ; MOVE ONE FORWARD
BR 6$
6$: CLC
```

```

9968 056372 006003
9969 056374 012710 000001      6$: ROR      R3
9970 056400 005302      MOV      #DMD,AR0
9971 056402 001362      DEC      R2
9972 056404 005305      BNE      4$
9973 056406 001342      DFC      R5
9974 056410 010321      BNE      1$
9975 056412 005703      MOV      R3,(R1)+
9976 056414 001414      TST      R3
9977                                BEQ      7$
9978 056416 012737 000001 053130      MOV      #1,ANERWORD
9979 056424 005037 001124      CLR      ANSGDDAT
9980 056430 010337 001126      MOV      R3,AN$BDDAT
9981 056434 012737 104006 056160      MOV      #104006,AN$SEGP
9982 056442 000137 057050      JMP      AN$17$ ;BRANCH OUT ----->
9983
9984 056446 012702 000022      7$: MOV      #18,R2 ;COUNT NO. OF SECTOR GAP
9985 056452 012737 000024 053130 10$: MOV      #20,ANERWORD ;COUNT TO GIVE ERROR WORD
9986 056460 004737 054156      JSR      PC,ANWRITE ;WRITE SECTOR GAP
9987 056464 013721 054154      MOV      AN$WORD,(R1)+ ;STORE SECTOR GAP WORD
9988 056470 001413      BEQ      11$
9989 056472 160237 053130      SUB      R2,ANERWORD ;IF NOT GET ERROR WORD NO.
9990 056476 005037 001124      CLR      ANSGDDAT ;GOOD WORD
9991 056502 013737 054154 001126      MOV      AN$WORD,AN$BDDAT ;BAD WORD
9992 056510 012737 104006 056160      MOV      #104006,AN$SEGP ;STORE "ERROR 6" IN SEGP
9993 056516 000554      BR      17$ ;BRANCH OUT
9994 056520 005302      11$: DEC      R2 ;HAVE 18 WORDS OF ZEROS BEEN WRITTEN ?
9995 056522 001353      BNE      10$ ;IF NOT CONTINUE
9996
9997 ;AT THIS POINT THE SECTOR FOUND FLOP SHOULD
9998 ;BE HIGH, SO THAT THE HEADER SYNC BYTE CAN BE GIVEN.
9999
10000 ;HOWEVER, IN THE DRIVE TIMING ERROR TEST THE REST OF THE ROUTINE
10001 ;IS ABORTED
10002
10003 056524 005737 015144      TST      ANTESDTE ;IS THIS A DRIVE TIMING ERROR TEST ?
10004 056530 001147      BNE      17$ ;BRANCH OUT IF YES ----->
10005
10006 056532 004737 054156      JSR      PC,ANWRITE ;WRITE ONE SECTOR GAP 0 BYTE
10007                                ;AND ONE SYNC. BYTE - 230
10008 056536 013711 054154      MOV      AN$WORD,(R1) ;SAVE 0 BYTE AND SYNC BYTE
10009 056542 023721 053112      CMP      AN$Rsync,(R1)+ ;IF SYNC. BYTE RIGHT
10010 056546 001414      BEQ      12$ ;IF YES CONTINUE OPERATION
10011 056550 012737 000024 053130      MOV      #20,ANERWORD ;IF NOT GET READY FOR ERROR PRINT
10012
10013 056556 013737 053112 001124      MOV      AN$Rsync,AN$GDDAT ;GOOD WORD
10014 056564 014137 001126      MOV      -(R1),AN$BDDAT ;BAD WORD
10015 056570 012737 104006 056162      MOV      #104006,AN$FSYNER ;INSERT "ERROR 6" IN FSYNER
10016 056576 000524      BR      17$ ;BRANCH OUT ----->
10017
10018 056600 012702 000004      12$: MOV      #4,R2 ;FOUR HEADER WORDS
10019 056604 012703 056230      MOV      #SCYL,R3 ;POINTER FOR HEADER TABLE
10020 056610 012737 000005 053130 13$: MOV      #5,ANERWORD ;ERROR WORD NO SET
10021 056616 004737 054156      JSR      PC,ANWRITE ;WRITE 4 HEADER WORDS
10022 056622 013711 054154      MOV      AN$WORD,(R1) ;STORE WRITTEN WORD
10023 056626 022321      CMP      (R3)+,(R1)+ ;IS IT CORRECT ?

```

```
10024 056630 001412          BEQ      14$          ;IF GOOD CONTINUE OPERATION
10025                                     ;IF NOT GET READY FOR ERROR PRINT
10026
10027 056632 160237 053130      SUB      R2,@#ERWORD ;WORD NO
10028 056636 014337 001124      MOV      -(R3),@#SGDDAT ;GOOD DATA
10029 056642 014137 001126      MOV      -(R1),@#SBDDAT ;BAD DATA
10030 056646 012737 104006 056164 MOV      #104006,@#ERHEAD;INSERT 'ERROR 6'
10031 056654 000475          BR       17$          ;BRANCH OUT ----->
10032
10033 056656 005302          14$: DEC      R2          ;ARE 4 HEADER WORDS DONE?
10034 056660 001353          BNE      13$          ;IF NOT CONTINUE
10035 056662 004737 054156      JSR      PC,@#WRITE ;WRITE CRC
10036 056666 013711 054154      MOV      @#WORD,(R1) ;STORE CRC
10037 056672 022137 056156      CMP      (R1)+,@#GCRC ;COMPARE GOOD CRC
10038 056676 001414          BEQ      20$          ;IF GOOD CONTINUE OPERATION
10039                                     ;IF NOT GET READY FOR ERROR PRINT
10040
10041 056700 014137 001126      MOV      -(R1),@#SBDDATA ;BAD CRC WRITTEN
10042 056704 013737 056156 001124 MOV      @#GCRC,@#SGDDAT ;GOOD CRC
10043 056712 012737 000005 053130 MOV      #5,@#ERWORD ;ERROR WORD NO
10044 056720 012737 104006 056166 MOV      #104006,@#ERCRC ;INSERT ERROR 6
10045 056726 000450          BR       17$          ;EXIT ----->
10046
10047 056730 012702 000005      20$: MOV      #5,R2          ;NO OF HEADER GAP
10048 056734 012737 000006 053130 15$: MOV      #6,@#ERWORD ;ERROR WORD NO SET
10049 056742 004737 054156      JSR      PC,@#WRITE ;WRITE HEADER GAP
10050 056746 013721 054154      MOV      @#WORD,(R1)+ ;STORE
10051 056752 001412          BEQ      16$          ;IF GOOD CONTINUE OPERATION
10052                                     ;IF NOT GET READY FOR PRINT
10053
10054 056754 160237 053130      SUB      R2,@#ERWORD ;ERROR WORD NO
10055 056760 005037 001124      CLR      @#SGDDAT ;GOOD DATA
10056 056764 014137 001126      MOV      -(R1),@#SBDDAT ;BAD DATA
10057 056770 012737 104006 056170 MOV      #104006,@#ERHDP ;STORE 'ERROR 6'
10058 056776 000424          BR       17$          ;BRANCH OUT ----->
10059
10060 057000 005302          16$: DEC      R2          ;ARE 5 HEADER GAP ZEROS DONE ?
10061 057002 001354          BNE      15$          ;IF NOT CONTINUE
10062 057004 004737 054156      JSR      PC,@#WRITE
10063 057010 013711 054154      MOV      @#WORD,(R1)
10064 057014 023721 053112      CMP      @#RSYNC,(R1)+
10065 057020 001413          BEQ      17$          ;A-OK, EXIT ----->
10066
10067 057022 012737 000005 053130 MOV      #5,@#ERWORD ;IF NOT GET READY FOR ERROR PRINT
10068 057030 014137 001126      MOV      -(R1),@#SBDDAT
10069 057034 013737 053112 001124 MOV      @#RSYNC,@#SGDDAT
10070 057042 012737 104006 056172 MOV      #104006,@#HDESYN
10071
10072 057050          17$: MOV      (SP)+,R1          ;;POP STACK INTO R1
10073 057050 012601          RTS      R1          ;EXIT ----->
10074 057052 000201
10075
10076
10077
10078
10079
```

CZRJHDO,RP04/5/6 DSKLS CTRLR2 MACY11 30A(1052) 24-MAY-79 15:07 ¹ ² PAGE 230
CZRJHD.P12 01-MAR-79 09:10 DISK SIMULATION

SEQ 0228

10080
10081
10082

```
10083 ;*****
10084 ;*SEARCH SECTOR
10085 ;*****
10086
10087
10088
10089
10090
10091
10092 ;* R0=RHMR ADDRESS
10093 ;* R1=PASSED ARGUMENT (SECTOR SEARCHED FOR)
10094 ;* R2=CLOCK COUNT (PER BYTE)
10095 ;* R3=SECTOR COUNTER FROM R1
10096 ;* R5=BYTES PER WORD COUNT
10097
10098 ;*BEFORE INDEX IS GIVEN TWO SECTOR CLOCKS ARE GIVEN TO RESET
10099 ;*SECTOR PULSE IN CASE IT IS SET.
10100
10101 ;*AT THE BEGINNING OF EACH SECTOR, ONE SECTOR CLOCK HAS TO RISE
10102 ;*BEFORE MAINT. CLOCK, THEN EVERY EIGHT MAINT.CLOCKS, ONE SECTOR CLOCK
10103 ;*IS IDENTICAL WITH THE MAINT. CLOCK
10104 ;*THE SECTOR CLOCKS ARE NUMBERED AS FOLLOWS:
10105
10106 ;*THE SECTOR CLOCK UNDER INDEX - 0
10107 ;*THE NEXT - 1
10108 ;*THE NEXT - 2
10109 ;*ETC.
10110 ;*THEN THE LAST SECTOR CLOCK IN ONE SECTOR HAS NUMBER - 608
10111 ;*THE NEXT SECTOR WILL HAVE 608 SECTOR CLOCKS
10112 ;*THE NEXT SECTOR THEN HAS ANOTHER 608 SECTOR CLOCKS
10113 ;*AND SO ON
10114
10115
10116
10117 057054 000000 SECTR: 0 ;SECTOR SEARCHED FOR
10118
10119 057056 012137 057054 SEARCH: MOV (R1)+,@#SECTR ;SAVE SECTOR SEARCHED FOR
10120 057062 010046 MOV R0,-(SP) ;:PUSH R0 ON STACK
10121 057064 010146 MOV R1,-(SP) ;:PUSH R1 ON STACK
10122 057066 010246 MOV R2,-(SP) ;:PUSH R2 ON STACK
10123 057070 010346 MOV R3,-(SP) ;:PUSH R3 ON STACK
10124 057072 010446 MOV R4,-(SP) ;:PUSH R4 ON STACK
10125 057074 010546 MOV R5,-(SP) ;:PUSH R5 ON STACK
10126 057076 013700 014776 MOV @#RHMR,R0 ;NOW R0 HAS MAINTENANCE REG. ADR.
10127 057102 013703 057054 MOV @#SECTR,R3 ;LOAD SECTOR COUNTER
10128
10129 057106 012710 000001 MOV #DMD,@R0 ;SET DIAGNOSTIC MODE
10130 057112 052710 000010 BIS #MSTCK,@R0 ;SET SECTOR CLOCK
10131 057116 042710 000010 BIC #MSTCK,@R0 ;CLEAR SECTOR CLOCK
10132 057122 052710 000010 BIS #MSTCK,@R0 ;SET SECTOR CLOCK
10133 057126 042710 000010 BIC #MSTCK,@R0 ;CLEAR SECTOR CLOCK
10134 ;THE ABOVE TWO SECTOR CLOCKS ARE GIVEN FOR
10135 ;RESETTING SECTOR PULSE
10136 ;IN CASE IT STARTS SET
10137 057132 052710 000014 BIS #MINX!MSTCK,@R0 ;SET INDEX AND SECTOR CLOCK
10138 057136 042710 000014 BIC #MINX!MSTCK,@R0 ;RESET INDEX AND SECTOR CLOCK
```

```
10139 057142 005703          TST    R3          ;TEST FOR SECTOR 0
10140 057144 001461          BEQ    7$          ;IF SECTOR 0 IS REQUIRED, EXIT ----->
10141
10142                      ;NOW 304 WORDS WILL START (608 BYTES)
10143
10144                      ;*FOR FIRST BYTE MAINT. SECTOR CLOCK WILL GO HIGH, THEN MAINT. CLOCK WILL GO HIGH
10145                      ;*BOTH WILL COME DOWN TOGETHER, THEN SEVEN MAINT. CLOCKS WILL BE GIVEN
10146                      ;*FOR SECOND BYTE, AND ALL OTHER BYTES TILL NEXT SECTOR, SECTOR CLOCK
10147                      ;*WILL BE IDENTICAL WITH ONE MAINT. CLOCK
10148
10149
10150                      ;ONE WORD ONLY
10151
10152 057146 012702 000010      1$:  MOV    #8., R2          ;BYTE COUNTER
10153 057152 012705 000002      MOV    #2., R5          ;BYTES PER WORD
10154 057156 052710 000010      BIS    #MSTCK,@R0        ;SET SECTOR CLOCK
10155 057162 052710 000002      BIS    #MCLK,@R0         ;SET MAINT. CLOCK
10156 057166 000402           BR      3$          ;BRANCH TO CLEAR SECTOR AND CLOCK
10157 057170 052710 000012      2$:  BIS    #MSTCK!MCLK,@R0    ;SET BOTH CLOCKS
10158 057174 042710 000012      3$:  BIC    #MSTCK!MCLK,@R0    ;CLEAR BOTH CLOCKS
10159 057200 052710 000002      8$:  BIS    #MCLK,@R0         ;SET MAINT. CLOCK
10160 057204 042710 000002      BIC    #MCLK,@R0         ;CLEAR MAINT. CLOCK
10161 057210 005302          DEC    R2          ;BYTE COUNTER
10162 057212 001372          BNE    8$          ;CONTINUE IF BYTE NOT COMPLETE
10163
10164 057214 012702 000007      MOV    #7,R2          ;SETUP FOR SECOND BYTE
10165 057220 005305          DEC    R5          ;IS WORD COMPLETE?
10166 057222 001362          BNE    2$          ;CONTINUE IF NOT COMPLETE
10167
10168
10169
10170                      ;NOW 303 WORDS ARE LEFT AND ALL ARE IDENTICAL
10171
10172 057224 012701 000457      MOV    #303.,R1        ;WORDS PER SECTOR COUNTER
10173 057230 012705 000002      4$:  MOV    #2,R5          ;BYTES PER WORD COUNTER
10174 057234 012702 000007      5$:  MOV    #7,R2          ;BIT COUNTER (MAINT.CLOCK COUNTER)
10175 057240 052710 000012      BIS    #MSTCK!MCLK,@R0    ;SET SECTOR CLOCK AND MAINT. CLOCK
10176 057244 042710 000012      BIC    #MSTCK!MCLK,@R0    ;CLEAR CLOCKS
10177 057250 052710 000002      6$:  BIS    #MCLK,@R0         ;SET MAINT. CLOCK
10178 057254 042710 000002      BIC    #MCLK,@R0         ;RESET MAINT. CLOCK
10179 057260 005302          DEC    R2          ;IS BYTE COMPLETE ?
10180 057262 001372          BNE    6$          ;CONTINUE IF NOT COMPLETE
10181
10182 057264 005305          DEC    R5          ;IS WORD COMPLETE ?
10183 057266 001362          BNE    5$          ;CONTINUE IF NOT
10184
10185 057270 005301          DEC    R1          ;IS SECTOR COMPLETE ?
10186 057272 001356          BNE    4$          ;CONTINUE IF NOT
10187
10188 057274 052710 000010      BIS    #MSTCK,@R0        ;SET SECTOR CLOCK 1 MORE TIME (FOR 609)
10189 057300 042710 000010      BIC    #MSTCK,@R0        ;CLEAR SECTOR CLOCK
10190 057304 005303          DEC    R3          ;IS REQUIRED NO OF SECTORS COMPLETE ?
10191 057306 001317          BNE    1$          ;CONTINUE IF NOT
10192
10193
10194 057310 012605      7$:  MOV    (SP)+,R5          ;;POP STACK INTO R5
```

```

10195 057312 012604      MOV      (SP)+,R4      ;;POP STACK INTO R4
10196 057314 012603      MOV      (SP)+,R3      ;;POP STACK INTO R3
10197 057316 012602      MOV      (SP)+,R2      ;;POP STACK INTO R2
10198 057320 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
10199 057322 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
10200 057324 000201      RTS      R1
10201
10202
10203      ;*****
10204      ;*READ ONE SECTOR OF DATA
10205      ;*****
10206
10207 057326 000000      RNO:      0      ;NO. OF WORDS READ
10208 057330 000000      RCOM:     0      ;EXTRA STORAGE
10209
10210
10211
10212 057332 012137 057326  REDATA: MOV      (R1)+,@#RNO      ;SAVE NO. OF WORDS ONLY FOR INFORMATION
10213 057336 012137 057330      MOV      (R1)+,@#RCOM      ;EXTRA WORD ONLY FOR INFORMATION
10214 057342 010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
10215 057344 005737 015142      TST      @#TSECC      ;IS THIS AN ECC TEST
10216 057350 001403      BEQ      1$      ;BRANCH IF NO
10217 057352 012737 177777 050624  MOV      #-1,@#TSECCG      ;THESE BITS ARE TO GENERATE ECC
10218 057360 012702 000402      1$:  MOV      #256.,R2      ;256 WORDS PER SECTOR
10219      ;PLUS 2 ECC WORDS
10220 057364 012703 054726      MOV      #DISK,R3      ;POINTE TO DISK SIMULATION
10221 057370 012337 053720      2$:  MOV      (R3)+,@#WORD      ;READY TO READ CONTENTS
10222 057374 004737 053724      JSR      PC,@#READ      ;READ
10223 057400 005302      DEC      R2      ;IS 256 WORDS DONE?
10224 057402 001372      BNE      2$      ;IF NOT BRANCH
10225 057404 005737 015142      TST      @#TSECC      ;IS THIS AN ECC TEST
10226 057410 001012      BNE      4$      ;BRANCH OUT IF YES
10227 057412 005037 050624      CLR      @#TSECCG      ;NO MORE ECC BITS ARE TO BE GENERATED
10228 057416 012702 000017      MOV      #15.,R2      ;ONE DATA GAP, 14 TOLERANCE GAP
10229 057422 012337 053720      3$:  MOV      (R3)+,@#WORD      ;READY TO READ CONTENTS OF WORD
10230 057426 004737 053724      JSR      PC,@#READ      ;READ
10231 057432 005302      DEC      R2      ;COUNT
10232 057434 001372      BNE      3$      ;BRANCH IF 14 NOT DONE
10233 057436      4$:
10234 057436 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
10235 057440 000201      RTS      R1      ;RETURN

```

MACY11 30A(1052) 24-MAY-79 15:07 "PAGE 234
DISK SIMULATION

11 2

10236					
10237	057442				
10238	057442	104401	057450		
10239	057446	000423			
10240					
10241	057516				
10242	057516	104402			
10243	057520	012777	057442	135216	
10244	057526	000000			
10245					
10246					

```

;*****
RPVECT:
        TYPE      ,65$                ;;TYPE ASCII STRING
        BR        64$                ;;GET OVER THE ASCII
;65$:   .ASCII    /UNEXPECTED RP04 INTERRUPT FROM PC = /
64$:
        TYPOC                      ;TYPE FROM PC
        MOV       #RPVECT,@RPVEC    ;RESTORE TRAP RP04 VECTOR
        HALT                       ;CHANGE TO CONTINUE
;*****

```

```

10247 .SBTTL
10248 .SBTTL ***SYSMAC LIBRARY ROUTINES***
10249 .SBTTL
10250
10251
10252
10253 .SBTTL SCOPE HANDLER ROUTINE
10254
10255 ;*****
10256 ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
10257 ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7.0>)
10258 ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
10259 ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
10260 ;*SW14=1 LOOP ON TEST
10261 ;*SW11=1 INHIBIT ITERATIONS
10262 ;*SW09=1 LOOP ON ERROR
10263 ;*SW08=1 LOOP ON TEST IN SWR<7:0>
10264 ;*CALL
10265 ;* SCOPE ;:SCOPE=10T
10266
10267 $SCOPE:
10268 057530 CKSWR ;:TEST FOR CHANGE IN SOFT-SWR
10269 057530 104407 CLR @#NOSYNC ;:CLEAR FLAG FOR HEADER ERROR COMMANDS
10270 057532 005037 053124 CLR @#TSECC ;:CLEAR FLAG FOR ECC TEST
10271 057536 005037 015142 ;:WHEN =177777 IT IS AN ECC TEST
10272 ;:WHEN =0 IT IS NOT AN ECC TEST
10273
10274 057542 005037 050624 CLR @#TSECCG ;:EVEN IN AN ECC TEST EVERY CLOCK
10275 ;:IS NOT TO GENERATE ECC
10276 ;:IF =177777 GENERATE ECC
10277 ;:IF =0 DO NOT GENERATE ECC
10278 057546 005037 015144 CLR @#TESDTE ;:DRIVE TIMING ERROR TEST
10279 057552
10280 057552 032777 040000 121360 1$: BIT #BIT14,@SWR ;:LOOP ON PRESENT TEST?
10281 057560 001111 BNE $OVER ;:YES IF SW14=1
10282 ;:*****START OF CODE FOR THE XOR TESTER*****
10283 057562 000416 $XTSTR: BR 6$ ;:IF RUNNING ON THE "XOR" TESTER CHANGE
10284 ;:THIS INSTRUCTION TO A "NOP" (NOP=240)
10285 057564 013746 000004 MOV @#ERRVEC,-(SP) ;:SAVE THE CONTENTS OF THE ERROR VECTOR
10286 057570 012737 057610 000004 MOV #5,@#ERRVEC ;:SET FOR TIMEOUT
10287 057576 005737 177060 TST @#177060 ;:TIME OUT ON XOR?
10288 057602 012637 000004 MOV (SP)+,@#ERRVEC ;:RESTORE THE ERROR VECTOR
10289 057606 000463 BR $SVLAD ;:GO TO THE NEXT TEST
10290 057610 022626 5$: CMP (SP)+,(SP)+ ;:CLEAR THE STACK AFTER A TIME OUT
10291 057612 012637 000004 MOV (SP)+,@#ERRVEC ;:RESTORE THE ERROR VECTOR
10292 057616 000423 BR 7$ ;:LOOP ON THE PRESENT TEST
10293 057620 6$;;*****END OF CODE FOR THE XOR TESTER*****
10294 057620 032777 000400 121312 BIT #BIT08,@SWR ;:LOOP ON SPEC. TEST?
10295 057626 001404 BEQ 2$ ;:BR IF NO
10296 057630 127737 121304 001102 CMPB @SWR,$TSTNM ;:ON THE RIGHT TEST? SWR<7:0>
10297 057636 001462 BEQ $OVER ;:BR IF YES
10298 057640 105737 001103 2$: TSTB $ERFLG ;:HAS AN ERROR OCCURRED?
10299 057644 001421 BEQ 3$ ;:BR IF NO
10300 057646 123737 001115 001103 CMPB $ERMAX,$ERFLG ;:MAX. ERRORS FOR THIS TEST OCCURRED?
10301 057654 101015 BHI 3$ ;:BR IF NO
10302 057656 032777 001000 121254 BIT #BIT09,@SWR ;:LOOP ON ERROR?

```

CZRJHDO,RP04/5/6 DSKLS CTRLR2
CZRJHD.P12 01-MAR-79 09:10

MACY11 30A(1052) 24-MAY-79 15:07 PAGE 236
SCOPE HANDLER ROUTINE

SEQ 0234

10303	057664	001404				BEQ	48	::BR IF NO
10304	057666	013737	001110	001106	78:	MOV	\$LPERR,\$LPADR	::SET LOOP ADDRESS TO LAST SCOPE
10305	057674	000443				BR	\$OVER	
10306	057676	105037	001103		48:	CLRB	\$ERFLG	::ZERO THE ERROR FLAG
10307	057702	005037	001212			CLR	\$TIMES	::CLEAR THE NUMBER OF ITERATIONS TO MAKE
10308	057706	000415				BR	18	::ESCAPE TO THE NEXT TEST
10309	057710	032777	004000	121222	38:	BIT	#BIT111,\$SWR	::INHIBIT ITERATIONS?
10310	057716	001011				BNE	18	::BR IF YES
10311	057720	005737	001100			TST	\$PASS	::IF FIRST PASS OF PROGRAM
10312	057724	001406				BEQ	18	::INHIBIT ITERATIONS
10313	057726	005237	001104			INC	\$ICNT	::INCREMENT ITERATION COUNT
10314	057732	023737	001212	001104		CMP	\$TIMES,\$ICNT	::CHECK THE NUMBER OF ITERATIONS MADE
10315	057740	002021				BGE	\$OVER	::BR IF MORE ITERATION REQUIRED
10316	057742	012737	000001	001104	18:	MOV	#1,\$ICNT	::REINITIALIZE THE ITERATION COUNTER
10317	057750	013737	060020	001212		MOV	\$MXCNT,\$TIMES	::SET NUMBER OF ITERATIONS TO DO
10318	057756	105237	001102		\$SVLAD:	INCB	\$TSTNM	::COUNT TEST NUMBERS
10319	057762	011637	001106			MOV	(SP),\$LPADR	::SAVE SCOPE LOOP ADDRESS
10320	057766	011637	001110			MOV	(SP),\$LPERR	::SAVE ERROR LOOP ADDRESS
10321	057772	005037	001214			CLR	\$ESCAPE	::CLEAR THE ESCAPE FROM ERROR ADDRESS
10322	057776	112737	000001	001115		MOVB	#1,\$ERMAX	::ONLY ALLOW ONE(1) ERROR ON NEXT TEST
10323	060004	013777	001102	121130	\$OVER:	MOV	\$TSTNM,\$DISPLAY	::DISPLAY TEST NUMBER
10324	060012	013716	001106			MOV	\$LPADR,(SP)	::FUDGE RETURN ADDRESS
10325	060016	000002				RTI		::FIXES PS
10326	060020	000004			\$MXCNT:	4		::MAX. NUMBER OF ITERATIONS

```
10327 .SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
10328
10329
10330
10331
10332
10333
10334
10335
10336
10337
10338
10339 060022
10340 060022 010046
10341 060024 010146
10342 060026 010246
10343 060030 010346
10344 060032 010546
10345 060034 012746 020200
10346 060040 016605 000020
10347 060044 100004
10348 060046 005405
10349 060050 112766 000055 000001
10350 060056 005000
10351 060060 012703 060236
10352 060064 112723 0C0040
10353 060070 005002
10354 060072 016001 060226
10355 060076 160105
10356 060100 002402
10357 060102 005202
10358 060104 000774
10359 060106 060105
10360 060110 005702
10361 060112 001002
10362 060114 105716
10363 060116 100407
10364 060120 106316
10365 060122 103003
10366 060124 116663 000001 177777
10367 060132 052702 000060
10368 060136 052702 000040
10369 060142 110223
10370 060144 005720
10371 060146 020027 000010
10372 060152 002746
10373 060154 003002
10374 060156 010502
10375 060160 000764
10376 060162 105726
10377 060164 100003
10378 060166 116663 177777 177776
10379 060174 105013
10380 060176 012605
10381 060200 012603
10382 060202 012602

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

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

CZRJHDO,RP04/5/6 DSKLS CTRLR2 MACY11 30A(1052) 24-MAY-79 15:07 PAGE 238
CZRJHD.P12 01-MAR-79 09:10 CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

SEQ 0236

```
10383 060204 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
10384 060206 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
10385 060210 104401 060236  TYPE      ,SDBLK      ;;NOW TYPE THE NUMBER
10386 060214 016666 000002 000004  MOV      2(SP),4(SP)      ;;ADJUST THE STACK
10387 060222 012616      MOV      (SP)+,(SP)
10388 060224 000002      RTI      ;;RETURN TO USER
10389 060226 023420      $DTBL: 10000.
10390 060230 001750      1000.
10391 060232 000144      100.
10392 060234 000012      10.
10393 060236 000004      $DBLK: .BLKW 4
```

```

10394 .SBTTL TYPE ROUTINE
10395
10396 *****
10397 *ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
10398 *THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
10399 *NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
10400 *NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
10401 *NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
10402
10403 *CALL:
10404 *1) USING A TRAP INSTRUCTION
10405 * TYPE ,MESADR ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
10406 *OR
10407 * TYPE
10408 * MESADR
10409 *
10410
10411 060246 105737 001157 $TYPE: TSTB $TPFLG ;;IS THERE A TERMINAL?
10412 060252 100002 BPL 1$ ;;BR IF YES
10413 060254 000000 HALT ;;HALT HERE IF NO TERMINAL
10414 060256 000407 BR 3$ ;;LEAVE
10415 060260 010046 1$: MOV R0,-(SP) ;;SAVE R0
10416 060262 017600 000002 MOV @2(SP),R0 ;;GET ADDRESS OF ASCIZ STRING
10417 060266 112046 2$: MOVB (R0)+,-(SP) ;;PUSH CHARACTER TO BE TYPED ONTO STACK
10418 060270 001005 BNE 4$ ;;BR IF IT ISN'T THE TERMINATOR
10419 060272 005726 TST (SP)+ ;;IF TERMINATOR POP IT OFF THE STACK
10420 060274 012600 60$: MOV (SP)+,R0 ;;RESTORE R0
10421 060276 062716 000002 3$: ADD #2,(SP) ;;ADJUST RETURN PC
10422 060302 000002 RTI ;;RETURN
10423 060304 122716 000011 4$: CMPB #HT,(SP) ;;BRANCH IF <HT>
10424 060310 001430 BEQ 8$
10425 060312 122716 000200 CMPB #CRLF,(SP) ;;BRANCH IF NOT <CRLF>
10426 060316 001006 BNE 5$
10427 060320 005726 TST (SP)+ ;;POP <CR><LF> EQUIV
10428 060322 104401 TYPE ;;TYPE A CR AND LF
10429 060324 001223 $CRLF
10430 060326 105037 060462 CLRB $CHARCNT ;;CLEAR CHARACTER COUNT
10431 060332 000755 BR 2$ ;;GET NEXT CHARACTER
10432 060334 004737 060416 5$: JSR PC,$TYPEC ;;GO TYPE THIS CHARACTER
10433 060340 123726 001156 6$: CMPB $FILLC,(SP)+ ;;IS IT TIME FOR FILLER CHARS.?
10434 060344 001350 BNE 2$ ;;IF NO GO GET NEXT CHAR.
10435 060346 013746 001154 MOV $NULL,-(SP) ;;GET # OF FILLER CHARS. NEEDED
10436 AND THE NULL CHAR.
10437 060352 105366 000001 7$: DECB 1(SP) ;;DOES A NULL NEED TO BE TYPED?
10438 060356 002770 BLT 6$ ;;BR IF NO--GO POP THE NULL OFF OF STACK
10439 060360 004737 060416 JSR PC,$TYPEC ;;GO TYPE A NULL
10440 060364 105337 060462 DECB $CHARCNT ;;DO NOT COUNT AS A COUNT
10441 060370 000770 BR 7$ ;;LOOP
10442
10443 ;HORIZONTAL TAB PROCESSOR
10444
10445 060372 112716 000040 8$: MOVB #' ,(SP) ;;REPLACE TAB WITH SPACE
10446 060376 004737 060416 9$: JSR PC,$TYPEC ;;TYPE A SPACE
10447 060402 132737 000007 060462 BITB #7,$CHARCNT ;;BRANCH IF NOT AT
10448 060410 001372 BNE 9$ ;;TAB STOP
10449 060412 005726 TST (SP)+ ;;POP SPACE OFF STACK

```

10450	060414	000724			BR	2\$	::GET NEXT CHARACTER
10451	060416	105777	120526		\$TYPEC: TSTB	@STPS	::WAIT UNTIL PRINTER IS READY
10452	060422	100375			BPL	\$TYPEC	
10453	060424	116677	000002	120520	MOVB	2(SP),@STPB	::LOAD CHAR TO BE TYPED INTO DATA REG.
10454	060432	122766	000015	000002	CMPB	#CR,2(SP)	::IS CHARACTER A CARRIAGE RETURN?
10455	060440	001003			BNE	1\$	::BRANCH IF NO
10456	060442	105037	060462		CLRB	\$CHARCNT	::YES--CLEAR CHARACTER COUNT
10457	060446	000406			BR	\$TYPEX	::EXIT
10458	060450	122766	000012	000002	1\$: CMPB	#LF,2(SP)	::IS CHARACTER A LINE FEED?
10459	060456	001402			BEQ	\$TYPEX	::BRANCH IF YES
10460	060460	105227			INCB	(PC)+	::COUNT THE CHARACTER
10461	060462	000000			\$CHARCNT: .WORD	0	::CHARACTER COUNT STORAGE
10462	060464	000207			\$TYPEX: RTS	PC	
10463							

```
10464 .SBTTL TTY INPUT ROUTINE
10465
10466 ;*****
10467 .ENABL LSB
10468 060466 000000 $TKCNT: .WORD 0 ;:NUMBER OF ITEMS IN QUEUE
10469 060470 000000 $TKQIN: .WORD 0 ;:INPUT POINTER
10470 060472 000000 $TKQOUT: .WORD 0 ;:OUTPUT POINTER
10471 060474 000011 $TKQSRT: .BLKB 9. ;:TTY KEYBOARD QUEUE
10472 060505 $TKQEND=.
10473 060506 .EVEN
10474
10475 ;*TK INITIALIZE ROUTINE
10476 ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
10477 ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
10478
10479 ;*CALL:
10480 ;* JSR PC,$TKINT
10481 ;* RETURN
10482
10483 060506 005037 060466 $TKINT: CLR $TKCNT ;:CLEAR COUNT OF ITEMS IN QUEUE
10484 060512 012737 060474 060470 MOV # $TKQSRT,$TKQIN ;:MOVE THE STARTING ADDRESS OF THE
10485 060520 013737 060470 060472 MOV $TKQIN,$TKQOUT ;:QUEUE INTO THE INPUT & OUTPUT POINTERS.
10486 060526 012737 060556 000060 MOV # $TKSRV,$TKVEC ;:INITIALIZE THE KEYBOARD VECTOR
10487 060534 012737 000200 000062 MOV #200,$TKVEC+2 ;: "BR" LEVEL 4
10488 060542 005777 120400 TST $TKB ;:CLEAR DONE FLAG
10489 060546 012777 000100 120370 MOV #100,$TKS ;:ENABLE TTY KEYBOARD INTERRUPT
10490 060554 000207 RTS PC ;:RETURN TO CALLER
10491
10492 ;*TK SERVICE ROUTINE
10493 ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
10494 ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
10495 ;*IT IN THE QUEUE.
10496 ;*IF THE CHARACTER IS A "CONTROL-C" (^C) $TKINT IS CALLED AND
10497 ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (OPERSCL)
10498
10499 060556 117746 120364 $TKSRV: MOVB $TKB,-(SP) ;:PICKUP THE CHARACTER
10500 060562 042716 177600 BIC #^C177,(SP) ;:STRIP THE JUNK
10501 060566 021627 000003 CMP (SP),#3 ;:IS IT A CONTROL C?
10502 060572 001007 BNE 1$ ;:BRANCH IF NO
10503 060574 104401 061545 TYPE ,SCNTLC ;:TYPE A CONTROL-C (^C)
10504 060600 004737 060506 JSR PC,$TKINT ;:INIT THE KEYBOARD
10505 060604 005726 TST (SP)+ ;:CLEAN UP STACK
10506 060606 000137 045036 JMP GPERSEL ;:CONTROL C RESTART
10507 060612 021627 000007 1$: CMP (SP),#7 ;:IS IT A CONTROL G?
10508 060616 001004 BNE 2$ ;:BRANCH IF NO
10509 060620 022737 000176 001140 CMP #SWREG,SWR ;:IS SOFT-SWR SELECTED?
10510 060626 001500 BEQ 6$ ;:GO TO SWR CHANGE
10511
10512 060630 2$: CMP #9,$TKCNT ;:IS THE QUEUE FULL?
10513 060630 022737 000011 060466 BNE 3$ ;:BRANCH IF NO
10514 060636 001004 TYPE ,SBELL ;:RING THE TTY BELL
10515 060640 104401 001216 TST (SP)+ ;:CLEAN CHARACTER OFF OF STACK
10516 060644 005726 BR 5$ ;:EXIT
10517 060646 000451 3$: CMP (SP),#23 ;:IS IT A CONTROL-S?
10518 060650 021627 000023 BNE 32$ ;:BRANCH IF NO
10519 060654 001021
```

```

10520 060656 005077 120262          CLR    @STKS          ;;DISABLE TTY KEYBOARD INTERRUPTS
10521 060662 005726          TST     (SP)+          ;;CLEAN CHAR OFF STACK
10522 060664 105777 120254      31$:  TSTB    @STKS          ;;WAIT FOR A CHAR
10523 060670 100375          BPL     31$          ;;LOOP UNTIL ITS THERE
10524 060672 117746 120250      MOVB    @STKB,-(SP)      ;;GET THE CHARACTER
10525 060676 042716 177600      BIC     #^C177,(SP)      ;;MAKE IT 7-BIT ASCII
10526 060702 022627 000021      CMP     (SP)+,#21      ;;IS IT A CONTROL-Q?
10527 060706 001366          BNE     31$          ;;BRANCH IF NO
10528 060710 012777 000100 120226  MOV     #100,@STKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
10529 060716 000002          RTI          ;;RETURN
10530 060720 005237 060466      32$:  INC     $TKCNT          ;;COUNT THIS CHARACTER
10531 060724 021627 000140      CMP     (SP),#140      ;;IS IT UPPER CASE?
10532 060730 002405          BLT     4$          ;;BRANCH IF YES
10533 060732 021627 000175      CMP     (SP),#175      ;;IS IT A SPECIAL CHAR?
10534 060736 003002          BGT     4$          ;;BRANCH IF YES
10535 060740 042716 000040      BIC     #40,(SP)          ;;MAKE IT UPPER CASE
10536 060744 112677 177520      4$:  MOVB    (SP)+,@STKQIN      ;;AND PUT IT IN QUEUE
10537 060750 005237 060470      INC     $TKQIN          ;;UPDATE THE POINTER
10538 060754 023727 060470 060505  CMP     $TKQIN,$STKQEND      ;;GO OFF THE END?
10539 060762 001003          BNE     5$          ;;BRANCH IF NO
10540 060764 012737 060474 060470  MOV     #$STKSRT,$TKQIN      ;;RESET THE POINTER
10541 060772 000002      5$:  RTI          ;;RETURN
10542
10543      ;*****
10544      ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
10545      ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
10546      ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
10547      ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
10548 060774 022737 000176 001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
10549 061002 001124          BNE     15$          ;;EXIT IF NOT
10550 061004 105777 120134      TSTB    @STKS          ;;IS A CHAR WAITING?
10551 061010 100121          BPL     15$          ;;IF NOT, EXIT
10552 061012 117746 120130      MOVB    @STKB,-(SP)      ;;YES
10553 061016 042716 177600      BIC     #^C177,(SP)      ;;MAKE IT 7-BIT ASCII
10554 061022 021627 000007      CMP     (SP),#7          ;;IS IT A CONTROL-G?
10555 061026 001300          BNE     2$          ;;IF NOT, PUT IT IN THE TTY QUEUE
10556
10557      ;;AND EXIT
10558
10559      ;*****
10560      ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
10561      ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
10562      ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
10562 061030 123727 001134 000001  6$:  CMPB    $AUTOB,#1      ;;ARE WE RUNNING IN AUTO-MODE?
10563 061036 001674          BEQ     2$          ;;BRANCH IF YES
10564 061040 005726          TST     (SP)+          ;;CLEAR CONTROL-G OFF STACK
10565 061042 004737 060506      JSR     PC,$TKINT      ;;FLUSH THE TTY INPUT QUEUE
10566 061046 005077 120072      CLR     @STKS          ;;DISABLE TTY KEYBOARD INTERRUPTS
10567 061052 112737 000001 001135  MOVB    #1,$INTAG      ;;SET INTERRUPT MODE INDICATOR
10568
10569 061060 104401 061557          TYPE    ,SCNTLG          ;;ECHO THE CONTROL-G (^G)
10570 061064 104401 061564      $GTSWR: TYPE    ,SMSWR          ;;TYPE CURRENT CONTENTS
10571 061070 013746 000176      MOV     SWREG,-(SP)      ;;SAVE SWREG FOR TYPEOUT
10572 061074 104402          TYPOC          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
10573 061076 104401 061575          TYPE    ,SMNEW          ;;PROMPT FOR NEW SWR
10574 061102 005046      19$:  CLR     -(SP)          ;;CLEAR COUNTER
10575 061104 005046          CLR     -(SP)          ;;THE NEW SWR

```

```
10576 061106 105777 120032      7$:  TSTB  @STKS      ;;CHAR THERE?
10577 061112 100375              BPL  7$      ;;IF NOT TRY AGAIN
10578
10579 061114 117746 120026      MOVB  @STKB,-(SP)  ;;PICK UP CHAR
10580 061120 042716 177600      BIC   #^C177,(SP)  ;;MAKE IT 7-BIT ASCII
10581
10582 061124 021627 000003      CMP   (SP),#3      ;;IS IT A CONTROL-C?
10583 061130 001015              BNE   9$      ;;BRANCH IF NOT
10584 061132 104401 061545      TYPE  ,SCNTLC      ;;YES, ECHO CONTROL-C (^C)
10585 061136 062706 000006      ADD   #6,SP      ;;CLEAN UP STACK
10586 061142 123727 001135 000001  CMPB  $INTAG,#1      ;;REENABLE TTY KEYBOARD INTERRUPTS?
10587 061150 001003              BNE   8$      ;;BRANCH IF NO
10588 061152 012777 000100 117764  MOV   #100,@STKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
10589 061160 000137 045036      8$:  JMP   OPERSEL      ;;CONTROL-C RESTART
10590
10591
10592 061164 021627 000025      9$:  CMP   (SP),#25      ;;IS IT A CONTROL-U?
10593 061170 001005              BNE   10$      ;;BRANCH IF NOT
10594 061172 104401 061552      TYPE  ,SCNTLU      ;;YES, ECHO CONTROL-U (^U)
10595 061176 062706 000006      20$:  ADD   #6,SP      ;;IGNORE PREVIOUS INPUT
10596 061202 000737              BR    19$      ;;LET'S TRY IT AGAIN
10597
10598
10599 061204 021627 000015      10$:  CMP   (SP),#15      ;;IS IT A <CR>?
10600 061210 001022              BNE   16$      ;;BRANCH IF NO
10601 061212 005766 000004      TST   4(SP)      ;;YES, IS IT THE FIRST CHAR?
10602 061216 001403              BEQ   11$      ;;BRANCH IF YES
10603 061220 016677 000002 117712  MOV   2(SP),@SWR      ;;SAVE NEW SWR
10604 061226 062706 000006      11$:  ADD   #6,SP      ;;CLEAR UP STACK
10605 061232 104401 001223      14$:  TYPE  ,SCRLF      ;;ECHO <CR> AND <LF>
10606 061236 123727 001135 000001  CMPB  $INTAG,#1      ;;RE-ENABLE TTY KBD INTERRUPTS?
10607 061244 001003              BNE   15$      ;;BRANCH IF NOT
10608 061246 012777 000100 117670  MOV   #100,@STKS      ;;RE-ENABLE TTY KBD INTERRUPTS
10609 061254 000002              15$:  RTI      ;;RETURN
10610 061256 004737 060416      16$:  JSR   PC,$TYPEC      ;;ECHO CHAR
10611 061262 021627 000060      CMP   (SP),#60      ;;CHAR < 0?
10612 061266 002420              BLT   18$      ;;BRANCH IF YES
10613 061270 021627 000067      CMP   (SP),#67      ;;CHAR > 7?
10614 061274 003015              BGT   18$      ;;BRANCH IF YES
10615 061276 042726 000060      BIC   #60,(SP)+      ;;STRIP-OFF ASCII
10616 061302 005766 000002      TST   2(SP)      ;;IS THIS THE FIRST CHAR
10617 061306 001403              BEQ   17$      ;;BRANCH IF YES
10618 061310 006316              ASL   (SP)      ;;NO, SHIFT PRESENT
10619 061312 006316              ASL   (SP)      ;;CHAR OVER TO MAKE
10620 061314 006316              ASL   (SP)      ;;ROOM FOR NEW ONE.
10621 061316 005266 000002      17$:  INC   2(SP)      ;;KEEP COUNT OF CHAR
10622 061322 056616 177776      BIS   -2(SP),(SP)  ;;SET IN NEW CHAR
10623 061326 000667              BR    7$      ;;GET THE NEXT ONE
10624 061330 104401 001222      18$:  TYPE  ,SQUES      ;;TYPE ?<CR><LF>
10625 061334 000720              BR    20$      ;;SIMULATE CONTROL-U
10626      .DSABL  LSB
10627
10628      .....
10629      ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
10630      ;*CALL:
10631
```

```
10632      ;*      RDCHR      ;:GET A CHARACTER FROM THE QUEUE
10633      ;*      RETURN HERE ;:CHARACTER IS ON THE STACK
10634      ;*      ;:WITH PARITY BIT STRIPPED OFF
10635      ;*
10636
10637 061336 011646      SRDCHR: MOV      (SP),-(SP) ;:PUSH DOWN THE PC AND
10638 061340 016666 000004 000002      MOV      4(SP),2(SP) ;:THE PS
10639 061346 005066 000004      CLR      4(SP) ;:GET READY FOR A CHARACTER
10640 061352 005046      CLR      -(SP) ;:PUT NEW PS ON STACK
10641 061354 012746 061362      MOV      #64$,-(SP) ;:PUT NEW PC ON STACK
10642 061360 000002      RTI      ;:POP NEW PC AND PS
10643 061362
10644 061362 005737 060466      64$: 1$: TST      $TKCNT ;:WAIT ON A CHARACTER
10645 061366 001775      BEQ      1$
10646 061370 005337 060466      DEC      $TKCNT ;:DECREMENT THE COUNTER
10647 061374 117766 177072 000004      MOVB   @ $TKQOUT,4(SP) ;:GET ONE CHARACTER
10648 061402 005237 060472      INC      $TKQOUT ;:UPDATE THE POINTER
10649 061406 023727 060472 060505      CMP      $TKQOUT,#$TKQEND ;:DID IT GO OFF OF THE END?
10650 061414 001003      BNE      2$ ;:BRANCH IF NO
10651 061416 012737 060474 060472      MOV      #$TKQSRT,$TKQOUT ;:RESET THE POINTER
10652 061424 000002      2$: RTI      ;:RETURN
10653      ;*****
10654      ;THIS ROUTINE WILL INPUT A STRING FROM THE TTY
10655      ;CALL:
10656      ;*      RDLIN      ;:INPUT A STRING FROM THE TTY
10657      ;*      RETURN HERE ;:ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
10658      ;*      ;:TERMINATOR WILL BE A BYTE OF ALL 0'S
10659
10660 061426 010346      SRDLIN: MOV      R3,-(SP) ;:SAVE R3
10661 061430 012703 061534      1$: MOV      #$TTYIN,R3 ;:GET ADDRESS
10662 061434 022703 061545      2$: CMP      #$TTYIN+9.,R3 ;:BUFFER FULL?
10663 061440 101405      BLOS      4$ ;:BR IF YES
10664 061442 104410      RDCHR      ;:GO READ ONE CHARACTER FROM THE TTY
10665 061444 112613      MOVB   (SP)+,(R3) ;:GET CHARACTER
10666 061446 122713 000177      10$: CMPB   #177,(R3) ;:IS IT A RUBOUT
10667 061452 001003      BNE      3$ ;:SKIP IF NOT
10668 061454 104401 001222      4$: TYPE   , $QUES ;:TYPE A '?'
10669 061460 000763      BR      1$ ;:CLEAR THE BUFFER AND LOOP
10670 061462 111337 061532      3$: MOVB   (R3),9$ ;:ECHO THE CHARACTER
10671 061466 104401 061532      TYPE   ,9$
10672 061472 122723 000015      CMPB   #15,(R3)+ ;:CHECK FOR RETURN
10673 061476 001356      BNE      2$ ;:LOOP IF NOT RETURN
10674 061500 105063 177777      CLRB   -1(R3) ;:CLEAR RETURN (THE 15)
10675 061504 104401 001224      TYPE   , $LF ;:TYPE A LINE FEED
10676 061510 012603      MOV      (SP)+,R3 ;:RESTORE R3
10677 061512 011646      MOV      (SP),-(SP) ;:ADJUST THE STACK AND PUT ADDRESS OF THE
10678 061514 016666 000004 000002      MOV      4(SP),2(SP) ;:FIRST ASCII CHARACTER ON IT
10679 061522 012766 061534 000004      MOV      #$TTYIN,4(SP)
10680 061530 000002      RTI      ;:RETURN
10681 061532 000      9$: .BYTE 0 ;:STORAGE FOR ASCII CHAR. TO TYPE
10682 061533 000      .BYTE 0 ;:TERMINATOR
10683 061534 000011      $TTYIN: .BLKB 9. ;:RESERVE 9. BYTES FOR TTY INPUT
10684 061545 136 006503 000012      $CNTLC: .ASCII / ^C / <15> <12> ;:CONTROL 'C'
10685 061552 052536 005015 000      $CNTLU: .ASCII / ^U / <15> <12> ;:CONTROL 'U'
10686 061557 136 006507 000012      $CNTLG: .ASCII / ^G / <15> <12> ;:CONTROL 'G'
10687 061564 005015 053523 020122      $MSWR: .ASCII <15> <12> / SWR /
```

CZRJHDO,RP04/5/6 DSKLS CTRLR2 MACY11 30A(1052) 24-MAY-79 15:07 ^{K 3} PAGE 245
CZRJHD.P12 01-MAR-79 09:10 TTY INPUT ROUTINE

SEQ 0243

10688	061572	020075	000	
10689	061575	040	047040	053505 SMNEW: .ASCIZ / NEW /
10690	061602	036440	000040	
10691				

.FROM THE TTY

```
10692 .SBTTL READ AN OCTAL NUMBER FROM THE TTY
10693
10694 *****
10695 *THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
10696 *CHANGE IT TO BINARY.
10697 *THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
10698 *OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
10699 *FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
10700 *THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
10701 *CALL:
10702 *      RDOCT          ;;READ AN OCTAL NUMBER
10703 *      RETURN HERE    ;;LOW ORDER BITS ARE ON TOP OF THE STACK
10704 *                    ;;HIGH ORDER BITS ARE IN $HIOCT
10705
10706 061606 011646          $RDOCT: MOV      (SP),-(SP)      ;;PROVIDE SPACE FOR THE
10707 061610 016666 000004 000002 MOV      4(SP),2(SP)      ;;INPUT NUMBER
10708 061616 010046          MOV      R0,-(SP)      ;;PUSH R0 ON STACK
10709 061620 010146          MOV      R1,-(SP)      ;;PUSH R1 ON STACK
10710 061622 010246          MOV      R2,-(SP)      ;;PUSH R2 ON STACK
10711 061624 104411 1$:      RDLIN          ;;READ AN ASCII LINE
10712 061626 012600          MOV      (SP)+,R0      ;;GET ADDRESS OF 1ST CHARACTER
10713 061630 010037 061734 MOV      R0,$$      ;;AND SAVE IT
10714 061634 005001          CLR      R1      ;;CLEAR DATA WORD
10715 061636 005002          CLR      R2
10716 061640 112046 2$:      MOVB      (R0)+,-(SP)      ;;PICKUP THIS CHARACTER
10717 061642 001420          BEQ      $$      ;;IF ZERO GET OUT
10718 061644 122716 000060 CMPB      #'0,(SP)      ;;MAKE SURE THIS CHARACTER
10719 061650 003026          BGT      $$      ;;IS AN OCTAL DIGIT
10720 061652 122716 000067 CMPB      #'7,(SP)
10721 061656 002423          BLT      $$
10722 061660 006301          ASL      R1      ;;*2
10723 061662 006102          ROL      R2
10724 061664 006301          ASL      R1      ;;*4
10725 061666 006102          ROL      R2
10726 061670 006301          ASL      R1      ;;*8
10727 061672 006102          ROL      R2
10728 061674 042716 177770 BIC      #'^C7,(SP)      ;;STRIP THE ASCII JUNK
10729 061700 062601          ADD      (SP)+,R1      ;;ADD IN THIS DIGIT
10730 061702 000756          BR       2$      ;;LOOP
10731 061704 005726 3$:      TST      (SP)+      ;;CLEAN TERMINATOR FROM STACK
10732 061706 010166 000012 MOV      R1,12(SP)      ;;SAVE THE RESULT
10733 061712 010237 061744 MOV      R2,$HIOCT
10734 061716 012602          MOV      (SP)+,R2      ;;POP STACK INTO R2
10735 061720 012601          MOV      (SP)+,R1      ;;POP STACK INTO R1
10736 061722 012600          MOV      (SP)+,R0      ;;POP STACK INTO R0
10737 061724 000002          RTI          ;;RETURN
10738 061726 005726 4$:      TST      (SP)+      ;;CLEAN PARTIAL FROM STACK
10739 061730 105010          CLRB      (R0)      ;;SET A TERMINATOR
10740 061732 104401          TYPE          ;;TYPE UP THRU THE BAD CHAR.
10741 061734 000000 5$:      .WORD      0
10742 061736 104401 001222 TYPE      ,SQUES      ;;'"' 'CR' & 'LF'
10743 061742 000730          BR       1$      ;;TRY AGAIN
10744 061744 000000 $HIOCT: .WORD      0      ;;HIGH ORDER BITS GO HERE
```

```
10745      .SBTTL  ERROR HANDLER ROUTINE
10746
10747      ;*****
10748      ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
10749      ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
10750      ;*AND GO TO $ERRTYP ON ERROR
10751      ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
10752      ;*SW15=1      HALT ON ERROR
10753      ;*SW13=1      INHIBIT ERROR TYPEOUTS
10754      ;*SW10=1      BELL ON ERROR
10755      ;*SW09=1      LOOP ON ERROR
10756      ;*CALL
10757      ;*      ERROR      N      ;;ERROR=EMT AND N=ERROR ITEM NUMBER
10758
10759      $ERROR:
10760      061746      CKSWR      ;;TEST FOR CHANGE IN SOFT-SWR
10761      061746      104407
10762      061750      012737      177777      015124
10763      061756      REGSA1:
10764
10765
10766      061756      105237      001103      7$:      INCB      $ERFLG      ;;SET THE ERROR FLAG
10767      061762      001775      BEQ      7$      ;;DON'T LET THE FLAG GO TO ZERO
10768      061764      013777      001102      117150      MOV      $STINM,$DISPLAY      ;;DISPLAY TEST NUMBER AND ERROR FLAG
10769      061772      032777      002000      117140      BIT      #BIT10,$SWR      ;;BELL ON ERROR?
10770      062000      001402      BEQ      1$      ;;NO - SKIP
10771      062002      104401      001216      TYPE      ,SBELL      ;;RING BELL
10772      062006      005237      001112      1$:      INC      $ERTTL      ;;COUNT THE NUMBER OF ERRORS
10773      062012      011637      001116      MOV      (SP),$ERRPC      ;;GET ADDRESS OF ERROR INSTRUCTION
10774      062016      162737      000002      001116      SUB      #2,$ERRPC
10775      062024      117737      117066      001114      MOVB      @ERRPC,$ITEMB      ;;STRIP AND SAVE THE ERROR ITEM CODE
10776      062032      032777      020000      117100      BIT      #BIT13,$SWR      ;;SKIP TYPEOUT IF SET
10777      062040      001004      BNE      20$      ;;SKIP TYPEOUTS
10778      062042      004737      062114      JSR      PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
10779      062046      104401      001223      TYPE      ,SCRLF
10780      062052
10781      062052      005777      117062      20$:
10782      062056      100002      2$:      TST      @SWR      ;;HALT ON ERROR
10783      062060      000000      BPL      3$      ;;SKIP IF CONTINUE
10784      062062      104407      HALT      ;;HALT ON ERROR!
10785      062064      032777      001000      117046      3$:      CKSWR      ;;TEST FOR CHANGE IN SOFT-SWR
10786      062072      001402      BIT      #BIT09,$SWR      ;;LOOP ON ERROR SWITCH SET?
10787      062074      013716      001110      BEQ      4$      ;;BR IF NO
10788      062100      005737      001214      4$:      MOV      $LPERR,(SP)      ;;FUDGE RETURN FOR LOOPING
10789      062104      001402      TST      $ESCAPE      ;;CHECK FOR AN ESCAPE ADDRESS
10790      062106      013716      001214      BEQ      5$      ;;BR IF NONE
10791      062112      MOV      $ESCAPE,(SP)      ;;FUDGE RETURN ADDRESS FOR ESCAPE
10792      062112      000002      5$:      RTI      ;;RETURN
```

10793
10794
10795
10796
10797
10798
10799
10800
10801
10802
10803
10804
10805
10806
10807
10808
10809
10810
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
10847
10848062114
062114 104401 001223
062120 010046
062122 005000
062124 153700 001114
062130 001004
062132 013746 001116
062136 104402
062140 000445
062142 005300
062144 006300
062146 006300
062150 006300
062152 062700 001226
062156 012037 062166
062162 001404
062164 104401
062166 000000
062170 104401 001223
062174 012037 062204
062200 001404
062202 104401
062204 000000
062206 104401 001223
062212 010146
062214 012001
062216 001415
062220 012000
062222 105720
062224 001003
062226 013146
062230 104402
062232 000402
062234
062234 013146
062236 104405
062240 005711
062242 001403
062244 104401 062264
062250 000764
062252 012601
062254 012600
062256 104401 001223
062262 000207
062264 020040 000
062270

.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"
2\$: .WORD 0 ; "ERROR MESSAGE" POINTER GOES HERE
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"
4\$: .WORD 0 ; "DATA HEADER" POINTER GOES HERE
TYPE , \$CRLF ; "CARRIAGE RETURN" & "LINE FEED"
5\$: MOV R1, -(SP) ; SAVE R1
MOV (RO)+, R1 ; PICKUP "DATA TABLE" POINTER
BEQ 9\$; BR IF NO DATA TO BE TYPED
MOV (RO)+, RO ; PICKUP "DATA FORMAT" POINTER
6\$: TSTB (RO)+ ; "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\$: .ASCII / / ; TWO(2) SPACES
.EVEN

.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.
*STYPOS---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

```

```

*STYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
*STYPOS OR STYPOC

```

```

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

```

```

*STYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER

```

```

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

```

```

STYPOS: MOV      @ (SP),-(SP)      ;;PICKUP THE MODE
        MOV      1(SP),%0FILL      ;;LOAD ZERO FILL SWITCH
        MOV      (SP)+,%0MODE+1    ;;NUMBER OF DIGITS TO TYPE
        ADD      #2,(SP)           ;;ADJUST RETURN ADDRESS
        BR       STYPON
STYPOC: MOV      #1,%0FILL          ;;SET THE ZERO FILL SWITCH
        MOV      #6,%0MODE+1        ;;SET FOR SIX(6) DIGITS
STYPON: MOV      #5,%0CNT           ;;SET THE ITERATION COUNT
        MOV      R3,-(SP)           ;;SAVE R3
        MOV      R4,-(SP)           ;;SAVE R4
        MOV      R5,-(SP)           ;;SAVE R5
        MOV      %0MODE+1,R4        ;;GET THE NUMBER OF DIGITS TO TYPE
        NEG      R4
        ADD      #6,R4              ;;SUBTRACT IT FOR MAX. ALLOWED
        MOV      R4,%0MODE          ;;SAVE IT FOR USE
        MOV      %0FILL,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     %0MODE            ;;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

```

```

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 062270 017646 000000
10875 062274 116637 000001 062513
10876 062302 112637 062515
10877 062306 062716 000002
10878 062312 000406
10879 062314 112737 000001 062513
10880 062322 112737 000006 062515
10881 062330 112737 000005 062512
10882 062336 010346
10883 062340 010446
10884 062342 010546
10885 062344 113704 062515
10886 062350 005404
10887 062352 062704 000006
10888 062356 110437 062514
10889 062362 113704 062513
10890 062366 016605 000012
10891 062372 005003
10892 062374 006105 1$:
10893 062376 000404 BR 3$
10894 062400 006105 2$:
10895 062402 006105 ROL R5
10896 062404 006105 ROL R5
10897 062406 010503 MOV R5,R3
10898 062410 006103 3$:
10899 062412 105337 062514 DECB %0MODE
10900 062416 100016 BPL 7$
10901 062420 042703 177770 BIC #177770,R3
10902 062424 001002 BNE 4$
10903 062426 005704 TST R4
10904 062430 001403 BEQ 5$

```

CZRJHDO,RP04/5/6 DSKLS CTRLR2
CZRJHD.P12 01-MAR-79 09:10

MACY11 30A(1052) 24-MAY-79 15:07 PAGE 250
BINARY TO OCTAL (ASCII) AND TYPE

SEQ 0248

10905	062432	005204		48:	INC	R4	::DON'T SUPPRESS ANYMORE 0'S
10906	062434	052703	000060		BIS	#'0,R3	::MAKE THIS DIGIT ASCII
10907	062440	052703	000040	58:	BIS	#',R3	::MAKE ASCII IF NOT ALREADY
10908	062444	110337	062510		MOVB	R3,88	::SAVE FOR TYPING
10909	062450	104401	062510		TYPE	,88	::GO TYPE THIS DIGIT
10910	062454	105337	062512	78:	DECB	\$OCNT	::COUNT BY 1
10911	062460	003347			BGT	28	::BR IF MORE TO DO
10912	062462	002402			BLT	68	::BR IF DONE
10913	062464	005204			INC	R4	::INSURE LAST DIGIT ISN'T A BLANK
10914	062466	000744			BR	28	::GO DO THE LAST DIGIT
10915	062470	012605		68:	MOV	(SP)+,R5	::RESTORE R5
10916	062472	012604			MOV	(SP)+,R4	::RESTORE R4
10917	062474	012603			MOV	(SP)+,R3	::RESTORE R3
10918	062476	016666	000002 000004		MOV	2(SP),4(SP)	::SET THE STACK FOR RETURNING
10919	062504	012616			MOV	(SP)+,(SP)	
10920	062506	000002			RTI		::RETURN
10921	062510	000		88:	.BYTE	0	::STORAGE FOR ASCII DIGIT
10922	062511	000			.BYTE	0	::TERMINATOR FOR TYPE ROUTINE
10923	062512	000		\$OCNT:	.BYTE	0	::OCTAL DIGIT COUNTER
10924	062513	000		\$OFILL:	.BYTE	0	::ZERO FILL SWITCH
10925	062514	000000		\$OMODE:	.WORD	0	::NUMBER OF DIGITS TO TYPE

```
10926 .SBTTL TRAP DECODER
10927
10928 ;*****
10929 ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
10930 ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
10931 ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
10932 ;*GO TO THAT ROUTINE.
10933
10934 062516 010046          STRAP: MOV      R0,-(SP)          ;;SAVE R0
10935 062520 016600 000002  MOV      2(SP),R0          ;;GET TRAP ADDRESS
10936 062524 005740          TST      -(R0)          ;;BACKUP BY 2
10937 062526 111000          MOVB    (R0),R0          ;;GET RIGHT BYTE OF TRAP
10938 062530 006300          ASL      R0              ;;POSITION FOR INDEXING
10939 062532 C16000 062552  MOV      $TRPAD(R0),R0      ;;INDEX TO TABLE
10940 062536 000200          RTS      R0              ;;GO TO ROUTINE
10941
10942
10943 ;.THIS IS USE TO HANDLE THE "GETPRI" MACRO
10944
10945 062540 011646          STRAP2: MOV      (SP),-(SP)      ;;MOVE THE PC DOWN
10946 062542 016666 000004 000002  MOV      4(SP),2(SP)      ;;MOVE THE PSW DOWN
10947 062550 C00002          RTI                      ;;RESTORE THE PSW
10948
10949 .SBTTL TRAP TABLE
10950
10951 ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
10952 ;*BY THE "TRAP" INSTRUCTION.
10953
10954 ; ROUTINE
10955 ; -----
10956 062552 062540          $TRPAD: .WORD    $STRAP2
10957 062554 060246          $TYPE      ;;CALL=TYPE          TRAP+1(104401) TTY TYPEOUT ROUTINE
10958 062556 062314          $TYPOC    ;;CALL=TYPOC         TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
10959 062560 062270          $TYPOS    ;;CALL=TYPOS         TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
10960 062562 062330          $TYPON    ;;CALL=TYPON         TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
10961 062564 060022          $TYPDS    ;;CALL=TYPDS         TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
10962
10963 062566 061064          $GTSWR     ;;CALL=GTSWR         TRAP+6(104406) GET SOFT-SWR SETTING
10964
10965 062570 060774          $CKSWR     ;;CALL=CKSWR         TRAP+7(104407) TEST FOR CHANGE IN SOFT-SWR
10966 062572 061336          $RDCHR     ;;CALL=RDCHR         TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
10967 062574 061426          $RDLIN     ;;CALL=RDLIN         TRAP+11(104411) TTY TYPEIN STRING ROUTINE
10968 062576 061606          $RDOCT     ;;CALL=RDOCT         TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
10969 062600 046100          T.SCOPI    ;;CALL=SCOPI         TRAP+13(104413) MY LOCAL SCOPES
10970 062602 046160          CHECKT     ;;CALL=CHECKD        TRAP+14(104414) CHECK DVA,RDY,DPR,DRY
10971 062604 046476          WAIT.T    ;;CALL=WAT          TRAP+15(104415) WAIT LOOP
```

```

10972      .SBTTL  POWER DOWN AND UP ROUTINES
10973
10974      ;*****
10975      :POWER DOWN ROUTINE
10976 062606 012737 062752 000024 $PWRDN: MOV    $ILLUP,@PWRVEC ;;SET FOR FAST UP
10977 062614 012737 000340 000026      MOV    #340,@PWRVEC+2 ;;PRIO:7
10978 062622 010046      MOV    R0,-(SP)    ;;PUSH R0 ON STACK
10979 062624 010146      MOV    R1,-(SP)    ;;PUSH R1 ON STACK
10980 062626 010246      MOV    R2,-(SP)    ;;PUSH R2 ON STACK
10981 062630 010346      MOV    R3,-(SP)    ;;PUSH R3 ON STACK
10982 062632 010446      MOV    R4,-(SP)    ;;PUSH R4 ON STACK
10983 062634 010546      MOV    R5,-(SP)    ;;PUSH R5 ON STACK
10984 062636 017746 116276      MOV    @SWR,-(SP) ;;PUSH @SWR ON STACK
10985 062642 010637 062756      MOV    SP,$SAVR6 ;;SAVE SP
10986 062646 012737 062660 000024      MOV    $PWRUP,@PWRVEC ;;SET UP VECTOR
10987 062654 000000      HALT
10988 062656 000776      BR      -2          ;;HANG UP
10989
10990      ;*****
10991      :POWER UP ROUTINE
10992 062660 012737 062752 000024 $PWRUP: MOV    $ILLUP,@PWRVEC ;;SET FOR FAST DOWN
10993 062666 013706 062756      MOV    $SAVR6,SP    ;;GET SP
10994 062672 005037 062756      CLR    $SAVR6      ;;WAIT LOOP FOR THE TTY
10995 062676 005237 062756      1$: INC    $SAVR6      ;;WAIT FOR THE INC
10996 062702 001375      BNE    1$          ;;OF WORD
10997 062704 012677 116230      MOV    (SP)+,@SWR    ;;POP STACK INTO @SWR
10998 062710 012605      MOV    (SP)+,R5      ;;POP STACK INTO R5
10999 062712 012604      MOV    (SP)+,R4      ;;POP STACK INTO R4
11000 062714 012603      MOV    (SP)+,R3      ;;POP STACK INTO R3
11001 062716 012602      MOV    (SP)+,R2      ;;POP STACK INTO R2
11002 062720 012601      MOV    (SP)+,R1      ;;POP STACK INTO R1
11003 062722 012600      MOV    (SP)+,R0      ;;POP STACK INTO R0
11004 062724 012737 062606 000024      MOV    $PWRDN,@PWRVEC ;;SET UP THE POWER DOWN VECTOR
11005 062732 012737 000340 000026      MOV    #340,@PWRVEC+2 ;;PRIO:7
11006 062740 104401      TYPE          ;;REPORT THE POWER FAILURE
11007 062742 062760      $PWRMG: .WORD    $POWER    ;;POWER FAIL MESSAGE POINTER
11008 062744 012716      MOV    (PC)+,(SP)    ;;RESTART AT BEGIN
11009 062746 017360      $PWRAD: .WORD    BEGIN    ;;RESTART ADDRESS
11010 062750 000002      RTI
11011 062752 000000      $ILLUP: HALT          ;;THE POWER UP SEQUENCE WAS STARTED
11012 062754 000776      BR      -2          ;; BEFORE THE POWER DOWN WAS COMPLETE
11013 062756 000000      $SAVR6: 0          ;;PUT THE SP HERE
11014 062760 005015 047520 042527 $POWER: .ASCIZ  <15><12>'POWER'
11015 062766 000122
11016      .EVEN
11017
11018      000001      .END

```

ACL	=	000040	2714#							
ACU	=	100000	2667#							
ADDMOD		000204	665#	3060						
ADTIMO		052534	9060	9115#						
AOE	=	001000	2583#							
AS		015050	2163	2166	2792#	4845*	4857	5037*	5048	5186*
ATA	=	100000	2571#	4842	5034	5183				
ATABLE		017340	2893#	3346						
ATTENT		015134	2827#	3306*	3346*	4845	5037	5186		
A10	=	000001	2605#							
A11	=	000002	2606#							
A12	=	000004	2607#							
A13	=	000010	2608#							
A14	=	000020	2609#							
A15	=	000040	2610#							
A16	=	000100	2611#							
A17	=	000200	2612#							
A16	=	000400	2546#	4682						
A17	=	001000	2547#	4682						
BA		015026	2159	2781#	4838*	5022*	5027*	5172*	5176*	
BA1	=	000010	2519#	4684	8652					
BASECH		051550	665	9045#	9120					
BEGIN		017360	663	2903#	9114	11009				
BEGIN2		017350	667	2901#						
B11ST		045664	3130	7933#						
B110	=	000001	623#	8389	8424	8426				
B1100	=	000001	613#	623						
B1101	=	000002	612#	622						
B1102	=	000004	611#	621						
B1103	=	000010	610#	620						
B1104	=	000020	609#	619						
B1105	=	000040	608#	618						
B1106	=	000100	607#	617						
B1107	=	000200	606#	616						
B1108	=	000400	605#	615	10294					
B1109	=	001000	604#	614	10302	10785				
B111	=	000002	622#	8403						
B1110	=	002000	603#	10769						
B1111	=	004000	602#	10309						
B1112	=	010000	601#							
B1113	=	020000	600#	8419	8421	10776				
B1114	=	040000	599#	8396	10280					
B1115	=	100000	598#	8391	8398	8405	8414	8416		
B112	=	000004	621#							
B113	=	000010	620#							
B114	=	000020	619#							
B115	=	000040	618#							
B116	=	000100	617#							
B117	=	000200	616#							
B118	=	000400	615#							
B119	=	001000	614#							
BL11		045712	7939#	7944						
BL12		045734	7939	7940	7946#					
BL13		045744	7947	7948#						
BPTVEC	=	000014	630#							
CA		015044	2790#							

CC	015070	2800#												
CHECKD=	104414	10970#												
CHECKE	046350	5355	5494	5675	5854	6040	6189	6371	6545	6733	6872	7047	7223	7410
		7602	8095#											
CHECKT	046160	4677	4763	4940	5111	5345	5459	5627	5810	5987	6154	6323	6502	6678
		6838	6999	7179	7357	7547	8038#	8235	8334	8507	10970			
CKSWR =	104407	10268	10760	10784	10965#									
CLAREA	046034	4649	5502	5509	6198	6205	6881	6888	7981#					
CLDISK	046116	3089	3106	3305	3427	3467	3525	4616	4670	4755	4932	5103	5236	5414
		5615	5798	5976	6109	6311	6491	6666	6793	6987	7167	7346	7535	7661
		7696	7730	8017#	8224	8322	8506	8693	9129					
CLR =	000040	2521#	3550	3592	3616	3646	3673	3704	3732	3775	3819	3889	3933	3972
		4032	4079	4138	4172	4206	4242	4273	4285	4306	4318	4347	4377	4416
		4426	4445	4455	4478	4509	4541	4580	8022					
COMMD	052650	4686	4788	4964	5634	5817	5993	6330	6508	6685	7006	7186	7363	7554
		8241	8339	8527	9224#									
COMPA	053722	9380*	9430	9511#	9657*									
COMPAR	047020	3501	4861	4878	5054	5201	5360	5520	5699	5882	6069	6216	6392	6571
		6759	6899	7067	7250	7440	7629	8258#	8548	8574				
COMWMD	055772	5135	5350	5469	6164	6848	9776#							
COUNTD	054370	9651#	9655*	9671										
CPHALT	005123	1701#	4678	4764	4941	5112	5346	5356	5460	5495	5628	5676	5811	5855
		5988	6041	6155	6190	6324	6372	6503	6546	6679	6734	6839	6873	7000
		7048	7180	7224	7358	7411	7548	7603	8236	8335	8508			
CR =	000015	538#	10454	10464										
CRC	047332	4663	4749	4926	5097	5289	5328	5438	5606	5780	5959	6133	6302	6473
		6648	6817	6978	7143	7327	7518	8218	8316	8374#	8477			
CRLF =	000200	539#	10425	10464										
CSF =	000002	2654#												
CSU =	000010	2656#												
CS1	015032	2148	2151	2159	2166	2173	2183	2186	2190	2206	2785#	4839*	5031*	5178*
		7679	8041	8045	8052	8098	8106							
CS2	015030	2148	2151	2159	2173	2206	2782#	5023*	5028*	5029*	5179*	5180*		
CYL	053010	4656*	4664	4742*	4750	4862	4919*	4927	5598*	5607	5772*	5781	5951*	5960
		6294*	6303	6465*	6474	6640*	6649	6970*	6979	7135*	7144	7319*	7328	7510*
		7519	8211*	8219	8308*	8317	8471*	8478	9254#					
DAREAD	053064	9303	9315#											
DATENV	050636	2202	5424*	5592*	5766*	5945*	6119*	6288*	6459*	6634*	6803*	6964*	7129*	7313*
		7504*	8768#	9539*	9556*	9606*	9622*							
DAWORD	053070	5604*	5778*	5957*	6300*	6471*	6646*	6976*	7141*	7325*	7516*	8216*	8313*	8476*
		9316#												
DB	015022	2159	2779#											
DCK =	100000	2589#	5835	6011	6031	6526	6703	6723	7204	7381	7401	7572	7592	8603
		8608	8624	8627										
DCL =	000100	2715#												
D,LEAR	015156	2854#												
DDISP =	177570	545#	744	2934										
DENVL =	000200	2599#	5033											
DE1 =	000040	2561#												
DEF20 =	000002	2557#												
DF1	012516	801	920	927	2213#									
DF11	012534	909	2220#											
DF14	012543	940	2223#											
DF15	012553	949	2226#											
DF16	012556	960	2227#											
DF17	012563	973	2229#											

[illegible]

DMD	=	000001	2593#	3469	3526	4617	4814	4990	5033	8659	8687	8955	9233	9385	9391
			9519	9532	9535	9549	9552	9593	9603	9619	9785	9946	9949	9958	9969
			10129												
DPR	=	000400	2564#	8066	8069	8118									
DRY	=	000200	2563#	8066	8074	8113									
DST		015036	2170	2787#	4844*	5036*	5185*								
DSWR	=	177570	544#	743	2933										
DS1		015054	2148	2151	2163	2166	2173	2183	2186	2190	2206	2794#	4842*	5034*	5183*
			8064	8069	8074	8113	8118								
DT		015056	2795#												
DTAGAP		055732	9738#												
DTE	=	010000	2586#	4840	5032	5181									
DTSY	=	001000	2601#	9410	9483	9487									
DT1		011764	800	919	926	2141#									
DT11		012026	908	2148#											
DT14		012046	939	2151#											
DT15		012070	948	2154#											
DT16		012100	959	2156#											
DT17		012116	972	2159#											
DT2		012000	2143#												
DT20		012136	985	2163#											
DT21		012156	996	2166#											
DT22		012172	1013	1027	2168#										
DT24		012204	1042	1055	2170#										
DT26		012224	1068	2173#											
DT27		012246	1083	2176#											
DT3		012012	888	895	2145#										
DT30		012264	1096	2179#											
DT31		012302	851	859	1226	1234	2183#								
DT32		012324	835	876	2186#										
DT33		012350	818	2190#											
DT34		012372	1113	2193#											
DT35		012412	1132	1204	2196#										
DT36		012434	1147	1164	2199#										
DT37		012452	1187	2202#											
DT40		012476	1251	2206#											
DT42		014672	1257	1264	1291	2485#									
DT44		014702	1271	1277	1284	1298	1305	1312	1319	1333	1340	1347	1354	1361	1368
			1375	1382	1389	1396	1417	2487#							
DT54		014716	1326	1403	1410	2489#									
DVA	=	004000	2549#	7679	8041	8045	8106								
ECDATA		050616	8749#	8797	8956*	9020*	9022*	9529*	9531*	9546*	9548*	9596*	9600*	9612*	9616*
ECM	=	000100	2580#	6031	6723	7401	7592								
EC1	=	004000	2680#	4760	5108	8231	8330	8524							
ECORR		051242	5848	6024	6539	6716	7217	7394	7585	8950#					
ECTEST		050654	8788#	8984	9023	9540	9557	9607	9623						
EC1		015062	2196	2199	2202	2797#									
EC2		015064	2196	2199	2202	2798#	5012*	5660	5844	6020	6356	6535	6712	7032	7213
			7390	7581											
EMTVEC	=	000030	633#	2918*	2919*	3041*	3082*								
EM1		002136	795	1074	1429#										
EM11		002307	900	913	1058	1448#									
EM14		002342	931	1453#											
EM15		002362	944	1456#											
EM16		002432	953	1463#											
EM17		002453	964	1466#											

SEQ 0255

[illegible]

CZRJHDO,RP04/5/6 DSKLS CTRLR2		MACY11 30A(1052) 24-MAY-79 15:07 PAGE 259													SEQ 0256	
CZRJHD.P12 01-MAR-79 09:10		CROSS REFERENCE TABLE -- USER SYMBOLS														
ER1	015034	2148	2151	2163	2173	2183	2186	2190	2196	2206	2786#	4840*	5032*	5181*		
		5652	5835	6011	6031	6348	6526	6703	6723	7024	7204	7381	7401	7572		
		7592														
ER2	015040	2163	2788#													
ER3	015046	2163	2791#													
EXT1	= 000001	2634#														
EXT10	= 000010	2637#														
EXT2	= 000002	2635#														
EXT20	= 000020	2638#														
EXT4	= 000004	2636#														
EXT40	= 000040	2639#														
E00	024330	3555	3568#													
E11	025736	3871	3879#													
E12	026064	3912	3919#													
E44	030570	4557	4565#													
FEN	= 000200	2660#														
FER	= 000020	2578#														
FILLEC	051414	5581	5754	5934	6277	6447	6623	6953	7117	7302	7493	9004#				
FIRST	017332	2880#	2951	3001*												
FMT22	= 010000	2681#	4656	4675	4742	4760	4919	4937	5091	5108	5267	5285	5322	5340		
		5432	5448	5467	5598	5621	5679	5772	5804	5858	5951	5982	6044	6127		
		6143	6162	6294	6317	6375	6465	6497	6549	6640	6672	6737	6811	6827		
		6846	6970	6993	7051	7135	7173	7227	7319	7352	7414	7510	7541	7606		
		8211	8231	8308	8330	8471	8524	8653								
FNWORD	056206	5096*	5327*	5437*	6132*	6816*	9901#									
FORMAT	054372	9652#	9670													
FOUT	056212	9898	9904#													
FSYNER	056162	9804*	9832#	10015*												
FUTABL	015150	2850#														
GCRC	056156	5099	5330	5440	6135	6819	9816#	10037	10042							
GECC1	050620	2193	2196	2202	5422*	5484	5507	5590*	5658	5764*	5842	5943*	6018	6117*		
		6179	6203	6286*	6354	6457*	6533	6632*	6710	6801*	6862	6886	6962*	7030		
		7127*	7211	7311*	7388	7502*	7579	8753#	8795	8881*	9012*	9028				
GECC2	050622	2193	2196	2202	5423*	5488	5508	5591*	5765*	5944*	6118*	6183	6204	6287*		
		6458*	6633*	6802*	6866	6887	6963*	7128*	7312*	7503*	8756#	8796	8882*	9013*		
		9029														
GNS	= ***** U	645	2960	2965	2970	2974	2978	2982	2986	2990	2994	2998	3021	3058		
		3073	3080	3171	3198	3202	3206	3210	3257	3263	3269	3349	3355	3361		
		3395	3401	3407	3433	3437	3441	3448	3833	3837	3841	3845	7735	7741		
		7845	7851	7858	7862	7866	7873	7877	9048	9054	9070	9076	9082	9088		
		9094	9098	9102	9106	9112	9118	10240	10957	10958	10959	10960	10961	10963		
		10965	10966	10967	10968	10969	10970	10971								
GO	= 000001	2543#	3480	8660	8688	9236	9788									
GRV	= 000010	2559#														
GTSWR	= 104406	3008	10963#													
HADTMP	050642	5421*	5589*	5763*	5942*	6116*	6285*	6456*	6631*	6800*	6961*	7126*	7310*	7501*		
		8778#	8971													
HARDER	050634	5421	5589	5763	5942	6116	6285	6456	6631	6800	6961	7126	7310	7501		
		8765#														
HCCRCE	047734	8504#														
HCE	= 000200	2581#	8593	8597	8603	8608										
HCI	= 002000	2679#														
HCRC	= 000400	2582#	8615	8618	8624	8627										
HDESYN	056172	9808*	9883#	10070*												
HDSYN	054724	9733#														
HEADER	054700	9730#														

[illegible]

[illegible]

NOSYNC	053124	7883*	9304	9368#	9453	9481	10269*
NOUNIT	015114	2809#	3226*	3276*	3283	3328*	7757*
NOWORD	053056	4661*	9311#				
NUNIT	015116	2811#	3283*	3284*			
N11	025724	3856	3873#				
N12	026052	3898	3914#				
N14	026300	3986	3989#				
N36	030012	4387	4401#				
OCYL =	100000	2717#					
OF	015042	2789#					
OFREV =	000200	2678#					
OFSETC	015202	2864#					
OF100 =	000004	2673#					
OF200 =	000010	2674#					
OF25 =	000001	2671#					
OF400 =	000020	2675#					
OF50 =	000002	2672#					
OF800 =	000040	2676#					
OPERSE	045036	7841#	10506	10589			
OP1 =	020000	2587#					
OR =	000200	2523#	5023	5029	5180		
OUT	053074	9300	9308	9313	9319#		
000	024332	3566	3570#				
010	025246	3789	3799#				
021	027022	4148	4157#				
043	030442	4517	4529#				
044	030546	4548	4559#				
PAR =	000010	2577#					
PAT =	000020	2520#					
PCJSR	015132	2173	2176	2179	2186	2190	2199
		8681*	8682*	8950*	8951*	9224*	9225*
		2526#				2825#	2825#
PGE =	002000	8716#	8865	8880		8038*	8038*
PIE1 =	100000	8725#				8039*	8039*
PIE10 =	000100	8726#	8832			8095*	8095*
PIE11 =	000040	8727#	8865			8096*	8096*
PIE12 =	000020	8728#				8504*	8504*
PIE13 =	000010	8729#				8505*	8505*
PIE14 =	000004	8730#					
PIE15 =	000002	8731#					
PIE16 =	000001	8732#					
PIE17 =	100000	8733#					
PIE18 =	040000	8734#					
PIE19 =	020000	8717#	8823				
PIE2 =	040000	8735#					
PIE20 =	010000	8736#	8840				
PIE21 =	004000	8737#	8871				
PIE22 =	002000	8738#	8848				
PIE23 =	001000	8739#	8871				
PIE24 =	000400	8740#					
PIE25 =	000200	8741#					
PIE26 =	000100	8742#					
PIE27 =	000040	8743#					
PIE28 =	000020	8744#					
PIE29 =	000010	8718#	8865				
PIE3 =	020000	8745#					
PIE30 =	000004						

SEQ 0260

[illegible]

REINTO	016264	2876#	3476	3483*	3496	3502	4757	4838	4863	5044	5048*	5056	5192	5196*
		5203	5503	5504	5507*	5508*	5510	5511	5521	5617	5701	5800	5884	5978
		6071	6199	6200	6203*	6204*	6206	6207	6217	6313	6394	6493	6573	6668
		6761	6882	6883	6886*	6887*	6889	6890	6900	6989	7069	7169	7252	7348
		7442	7537	7631	8227	8326	8550	8575						
RELEAS	015160	2855#												
RESVEC=	000010	627#												
RETCL	015204	2865#												
RHAS	014774	2759#	3148	3173	3944	4591	8025*							
RHBA	014752	2750#	3976	4036	4083	4672*	4757*	4934*	5105*	5335*	5446*	5617*	5800*	5978*
		6141*	6313*	6493*	6668*	6825*	6989*	7169*	7348*	7537*	8227*	8326*	8521*	8650*
		2772#	3069											
RHBAE	015020	2757#	4676*	4762*	4939*	5110*	5341*	5468*	5625*	5808*	5986*	6163*	6321*	6501*
RHCA	014770	6676*	6847*	6997*	7177*	7356*	7545*	8234*	8333*	8513*	8654*	8684*		
		2767#	7913	8694	8698									
RHCC	015014	2752#	3471*	3480*	3552	3861	3902	3940	4142	4174	4356	4390	4420	4449
RHCS1	014756	4482	4511	4543	4587	4791	4967	5138	5337*	8017	8647*	8660*	8686*	8688*
		9050	9061	9064	9084	9236*	9788*							
		2751#	3093	3129	3174	3232*	3550*	3592*	3616*	3646*	3673*	3704*	3732*	3775*
RHCS2	014754	3819*	3850	3889*	3893	3933*	3937	3949	3972*	4032*	4079*	4138*	4172*	4206*
		4209	4242*	4246	4273*	4277	4306*	4310	4347*	4351	4377*	4381	4416*	4426*
		4445*	4455*	4478*	4509*	4541*	4580*	4584	4597	4684*	8018	8652*		
		2771#	3779											
RHCS3	015016	2748#	3045	3091*	9058									
RHDB	014746	2754#	4674*	4758*	4935*	5106*	5336*	5457*	5620*	5803*	5981*	6152*	6316*	6496*
RHDS1	014762	6671*	6836*	6992*	7172*	7351*	7540*	8230*	8329*	8518*	8651*	8685*		
		2761#	8019											
RHDS1	015000	2762#	3233	3235	3240	3242	3245	3247	3265	3363	3371	3373	3376	3378
RHDT	015002	3381	3383	3412	3414									
		2764#	8905											
RHEC1	015006	2765#	3564	3782	5012	8899								
RHEC2	015010	2753#	3177	8020	8592	8596	8602	8607	8614	8617	8622	8628		
RHER1	014760	2755#												
RHER2	014764	2758#												
RHER3	014772	2766#												
RHLA	015012	2760#	3469*	3526*	4617*	4813	4989	5671*	5672*	6367*	6368*	7043*	7044*	8659*
RHMR	014776	8687*	8954	9233*	9234*	9235*	9383	9669	9785*	9786*	9787*	9945	10126	
		2756#	4675*	4760*	4937*	5108*	5340*	5467*	5621*	5804*	5982*	6162*	6317*	6497*
RHOF	014766	6672*	6846*	6993*	7173*	7352*	7541*	8231*	8330*	8524*	8653*			
		2763#	3357	3411	3413									
RHSN	015004	2749#	3475	3489	3510	3596	3620	3650	3677	3708	3736	4671*	4756*	4772
RHWC	014750	4852	4888	4933*	4949	5043	5064	5104*	5120	5191	5211	5334*	5444*	5616*
		5799*	5977*	6139*	6312*	6492*	6667*	6823*	6988*	7168*	7347*	7536*	7911	7913
		8226*	8325*	8519*	8649*									
		2883#	3015*	3068*	3070*	3124	3777	4202	4237	4342	4644	5017	5167	
RH70	017334	3013#												
RH70CK	020716	9340#	9378*	9421										
RKEY1	053120	9341#	9379*	9423										
RKEY2	053122	2576#												
RMR =	000004	9774#	9790*	9792*										
RNCTR1	055770	10207#	10212*											
RNO	057326	9790#												
RNWAT1	056050	7663	7677#											
RPTRP1	044346	7698	7709#											
RPTRP2	044452	2739#	2949*	4147	4148*	4157*	4516	4517*	4529*	4547	4548*	4567*	7662	7697
RPVEC	014744	9072	9079*	9090	10243*									

RPVECT	057442	2949	10237#	10243															
RSETR	053116	9339#	9377*	9419															
RSYNC	053112	5284	5300	5307	9337#	9408	9413	9451	9463	9479	9490	10009	10013	10064					
		10069																	
RTN	021360	3075	3082#																
RUNCTR	052646	9223#	9238*	9239*															
RUNWAT	052726	9238#																	
R11	025656	3858#																	
R15	026502	4048	4052#																
R16	026652	4101	4105#																
R44	030572	4563	4567#																
SAVDT	015126	2818#	3414*																
SAVER	046616	3474	3488	4771	4851	4948	5042	5119	5190	8184#									
SAVSN	015130	2821#	3413*																
SC =	100000	2552#	4792	4839	4968	5031	5139	5178											
SCOP1 =	104413	7960	10969#																
SCRC	056240	9934#	9941*																
SCYL	056230	9930#	9937*	10019															
SC1 =	000100	2640#																	
SC10 =	001000	2643#																	
SC2 =	000200	2641#																	
SC20 =	002000	2644#																	
SC4 =	000400	2642#																	
SEARCH	057056	8661	9245	9798	10119#														
SECGAP	054630	5256	5409	6104	6788	9728#	9944												
SECOTR	053012	4658*	4744*	4921*	5600*	5601*	5774*	5775*	5953*	5954*	6296*	6297*	6467*	6468*					
		6642*	6643*	6972*	6973*	7137*	7138*	7321*	7322*	7512*	7513*	8212*	8213*	8309*					
		8310*	8472*	8473*	9242	9255#													
SECTR	057054	10117#	10119*	10127															
SEECOM	015200	2863#	8686																
SEGPTR	056160	9803*	9821#	9981*	9992*														
SELECT	015120	2813#	2901*	2903*	3017	3287	3315	7747											
SELTST	022754	3192	3287#																
SERCH	015162	2856#																	
SETCK1	031466	4791#																	
SETCK2	032162	4967#																	
SETCK3	032706	5138#																	

CZRJHDO,RP04/5/6 DSKLS CTRLR2
CZRJHD.P12 01-MAR-79 09:10

MACY11 30A(1052) 24-MAY-79 15:07 PAGE 266
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0263

SS15	033540	5252*	5336#	5383*											
SS2	033724	5385#													
SS3	033314	5244*	5269#	5375*											
SS4	033320	5245*	5270#	5376*											
SS5	033324	5246*	5271#	5377*											
SS7	033352	5247*	5286#	5378*											
STACK =	001000	531#	2914	3039	3119	3464	3523	3557	3594	3618	3648	3675	3706	3734	
		3791	3848	3891	3935	3974	4034	4081	4140	4178	4208	4244	4275	4308	
		4349	4379	4418	4447	4480	4519	4550	4582	4614	4639	4737	4915	5086	
		5234	5406	5558	5731	5911	6101	6254	6424	6600	6785	6932	7096	7280	
		7470	7659	7694											
START	017364	2902	2906#												
STARTA	017606	2948#													
STK LMT=	177774	542#													
SWR	001140	743#	2912	2933*	2935	2941*	3006	3013	3054	3167	3190	4716	8001	8285	
		8890	8892	10280	10294	10296	10302	10309	10509	10548	10603*	10769	10776	10781	
		10785	10984	10997*											
SWREG	000176	647#	2941	3006	10509	10548	10571								
SW0	= 000001	595#													
SW00	= 000001	585#	595												
SW01	= 000002	584#	594												
SW02	= 000004	583#	593												
SW03	= 000010	582#	592												
SW04	= 000020	581#	591												
SW05	= 000040	580#	590												
SW06	= 000100	579#	589												
SW07	= 000200	578#	588	4718	8287										
SW08	= 000400	577#	587												
SW09	= 001000	576#	586	8001											
SW1	= 000002	594#													
SW10	= 002000	575#													
SW11	= 004000	574#													
SW12	= 010000	573#	3013												
SW13	= 020000	572#	3054	3167	3190										
SW14	= 040000	571#													
SW15	= 100000	570#													
SW2	= 000004	593#													
SW3	= 000010	592#													
SW4	= 000020	591#													
SW5	= 000040	590#													
SW6	= 000100	589#	8892												
SW7	= 000200	588#													
SW8	= 000400	587#	8890												

TOLGAP	055734	4651	9664	9739#												
TOTALA	015136	2828#	3183*	3185*	3821	3853	3895	4384								
TPVEC =	000064	636#														
TRAPVE=	000034	634#	2920*	2921*												
TRE =	040000	2551#	4352	4360	4382	4394	4839	5031	5178							
TRK =	052762	9244*	9247#													
TRK1 =	004000	2645#														
TRK10 =	040000	2648#														
TRK2 =	010000	2646#														
TRK20 =	100000	2649#														
TRK4 =	020000	2647#														
TRP =	021320	3067	3076#													
TRTVEC=	000014	629#														
TSECC	015142	2832#	5418*	5586*	5760*	5939*	6113*	6282*	- 6453*	6628*	6797*	6958*	7123*	7307*		
		7498*	7884*	9674	10215	10225	10270*									
TSECCG	050624	7888*	8759#	8883	9525	9537	9542	9554	9604	9620	9676*	9689*	10217*	10227*		
		10274*														
TSTNM	017330	2141	2143	2145	2148	2151	2154	2156	2159	2163	2166	2168	2170	2173		
		2176	2179	2183	2186	2190	2193	2196	2199	2202	2206	2485	2487	2489		
		2879#	3040*	3120*	3146*	3426*	3465*	3524*	3549*	3591*	3615*	3645*	3672*	3703*		
		3731*	3774*	3818*	3888*	3932*	3971*	4031*	4078*	4137*	4171*	4198*	4233*	4272*		
		4305*	4338*	4376*	4415*	4444*	4477*	4508*	4540*	4579*	4615*	4640*	4738*	4916*		
		5087*	5235*	5408*	5560*	5733*	5913*	6103*	6256*	6426*	6602*	6787*	6933*	7097*		
		7281*	7472*	7658*	7693*	7847										
TST1	021032	3018	3037#	7807												
TST10	024200	3522#														
TST11	024234	3547#														
TST12	024340	3589#														
TST13	024432	3606	3613#													
TST14	024530	3635	3643#													
TST15	024626	3662	3670#													
TST16	024730	3693	3701#													
TST17	025026	3721	3729#													
TST2	021524	3117#														
TST20	025130	3752	3772#													
TST21	025254	3816#														
TST22	025740	3876	3886#													
TST23	026066	3916	3930#													
TST24	026206	3956	3969#													
TST25	026406	3991	4029#													

TS144	030600	4577#												
TS145	030720	4604	4613#											
TS146	030754	4637#												
TS147	031306	4646	4697	4719	4736#									
TS15	022770	3288	3301#	7755	7765									
TS150	032000	4797	4914#											
TS151	032524	4974	5085#											
TS152	033164	5144	5233#											
TS153	033740	5353	5387	5405#										
TS154	034366	5479	5527	5557#										
TS155	035104	5650	5706	5730#										
TS156	035630	5833	5889	5910#										
TS157	036372	6009	6076	6100#										
TS16	023474	3425#												
TS160	037016	6174	6223	6253#										
TS161	037534	6346	6399	6423#										
TS162	040260	6524	6578	6599#										
TS163	041022	6701	6766	6784#										
TS164	041446	6858	6906	6931#										
TS165	042164	7022	7074	7095#										
TS166	042720	7202	7257	7279#										
TS167	043476	7379	7447	7469#										
TS17	024036	3429	3463#											
TS170	044254	7570	7636	7656#										
TS171	044366	7675	7680	7691#										
TS172	044462	7707	7728#											
TTNO = 000071		3040#	3120#	3146#	3426#	3465#	3524#	3549#	3591#	3615#	3645#	3672#	3703#	3731#
		3774#	3818#	3888#	3932#	3971#	4031#	4078#	4137#	4171#	4198#	4233#	4272#	4305#
		4338#	4376#	4415#	4444#	4477#	4508#	4540#	4579#	4615#	4640#	4738#	4916#	5087#
		5235#	5408#	5560#	5733#	5913#	6103#	6256#	6426#	6602#	6787#	6933#	7097#	7281#
		7472#	7658#	7693#										
		2659#												
TUF = 000100		9370#												
TY = 053126														
TYPDS = 104405		3260	3275	3352	7738	7744	7755	7796	10837	10961#				
TYPE = 104401		2958	2963	2968	2972	2976	2980	2984	2988	2992	2996	3019	3056	3071
		3078	3169	3196	3200	3204	3208	3255	3261	3267	3273	3347	3353	3359
		3393	3399	3405	3431	3435	3439	3446	3831	3835	3839	3843	4678	4764
		4941	5112	5346	5356	5460	5495	5628	5676	5811	5855	5988	6041	6155
		6190	6324	6372	6503	6546	6679	6734	6839	6873	7000	7048	7180	7224
		7358	7411	7548	7603	7733	7739	7751	7754	7794	7797	7843	7849	7855
		7856	7860	7864	7871	7875	8236	8335	8508	9046	9052	9068	9074	9080
		9086	9092	9096	9100	9104	9110	9116	10238	10385	10428	10503	10515	10569
		10570	10573	10584	10594	10605	10624	10668	10671	10675	10740	10742	10771	10779
		10801	10818	10820	10823	10825	10840	10845	10909	10957#	11006			
TYPOC = 104402		3063	3266	3358	3364	7848	7854	9051	9073	9085	9091	9109	10242	10572
		10809	10833	10958#										
		10960#												
TYPON = 104404		10959#												
TYPOS = 104403														
T.SCOP 046100		8001#	10969											
UN 021572		3129#	3131#											
UNIT 015112		2808#	3025#	3282#	3289#	3310	3327#	3329	3351	7737	7759	7764#	8023	
UNITS 015072		2807#	3220	3224	3282	3322	7760							
UNITSL 015122		2814#	3026#	3289										
UNLOAD 015152		2852#												
UNS 040000		2588#												
LPE = 020000		2529#	4248	4255	4281	4288	4354							

US1	=	000001	2516#	3868	3909	3938	3942	3953	4314	4321	4585	4589	4601		
US2	=	000002	2517#	3868	3909	3938	3942	3953	4314	4321	4585	4589	4601		
US4	=	000004	2518#	3868	3909	3938	3942	3953	4314	4321	4585	4589	4601		
IJWR	=	000010	2712#												
U11		025630	3822	3848#											
VAR1		032316	3097*	5021#											
VAR2		032324	3098*	3099*	3100*	5022#									
VAR3		032764	3101*	5171#											
VAR4		032772	3102*	3103*	3104*	5172#									
VAR5		033024	3105*	5179#											
VUF	=	000002	2711#												
VU30	=	010000	2665#												
VV	=	000100	2562#	3483	8065										
WAT.T		046476	8137#	10971											
WAT	=	104415	10971#												
WC		015024	2159	2780#	4773	4837*	4879	4950	5021*	5026*	5055	5121	5171*	5175*	5202
			7912												
WCF	=	040000	2530#												
WCF	=	000040	2579#												
WCRC		054710	4665	4751	4928	5608	5782	5961	6304	6475	6650	6980	7145	7329	7520
			8220	8318	8479	9425	9731#								
WCU	=	000001	2653#												
WCYL		056146	5091*	5098	5322*	5329	5432*	5439	6127*	6134	6811*	6818	9812#		
WECC1		055726	2193	5484	6179	6862	9736#								
WECC2		055730	2193	5488	6183	6866	7154	7157*	7158*	9737#					
WKEY1		056152	5094*	5325*	5435*	6130*	6814*	9814#							
WKEY2		056154	5095*	5326*	5436*	6131*	6815*	9815#							
WLE	=	004000	2585#												
WORD		053720	9403*	9408*	9417*	9419*	9421*	9423*	9425*	9475*	9479*	9510#	9520*	9533*	9550*
			10221*	10229*											
WRCHDA		047162	8308#												
WRCHDT		015166	2858#	8239											
WRCHK		015164	2857#	8338											
WRCHMD		046650	8211#												
WRDATA		054376	9310	9654#	9899										
WRFROM		015220	2875#	3098	3102	3490	3496*	3503	4853	4857*	4880	4934	5022	5027	5105
			5172	5176	5264	5290	5291	5335	5361	5445	5678	5700	5857	5872*	5883
			6043	6059*	6070	6140	6374	6393	6548	6564*	6572	6736	6752*	6760	6824
			7050	7068	7226	7251	7413	7429*	7431*	7441	7605	7621*	7622*	7630	8549
			8650												
WRHEAD		056242	9810	9937#											
WRIDAT		015170	2859#	4681	4944										
WRIFOR		015172	2860#	5115	5337	5464	6159	6843	8647						
WRITE		054156	9438	9450	9585#	9678	9685	9691	9695	9698	9986	10006	10021	10035	10049
			10062												
WRL	=	004000	2567#												
WRU	=	000400	2661#												
WSECTR		056150	5093*	5324*	5434*	6129*	6813*	9795	9813#						
WSSYNC		054676	9729#												
WSU	=	000004	2655#												
WTRK		056104	9797*	9801#											
WWORD		054154	9439	9443	9447	9451	9459	9464	9469	9580#	9629*	9679	9686	9692	9696
			9699	9987	9991	10008	10022	10036	10050	10063					
X		053020	4662*	4748*	4925*	5605*	5779*	5958*	6301*	6472*	6647*	6977*	7142*	7326*	7517*
			8217*	8314*	8522*	8539	9258#	9302							
XE2		022416	3188	3218#											

X11	025732	3828	3875#											
Y	053060	9312#												
Y11	025324	3827	3830#											
ZCODE	050640	2202	5425*	5593*	5767*	5946*	6120*	6289*	6460*	6635*	6804*	6965*	7130*	7314*
		7505*	8771#	8965*										
ZER =	000400	2600#	8974											
ZWORDS	054374	9653#	9672*											
%AUTOB	001134	740#	3010*	10562	10691									
%BDADR	001122	735#	2148	2151	2173	2206	8047*	8054*	8058*	8071*	8076*	8079*	8101*	8109*
		8115*	8120*											
%BDDAT	001126	737#	2141	2145	2151	2156	2168	2170	2176	2179	2183	2186	2487	3150*
		3151	3603*	3605	3632*	3634	3659*	3661	3690*	3692	3718*	3720	3749*	3751
		3785*	3867*	3868*	3870	3908*	3909*	3911	3952*	3953*	3955	3983*	3985	4045*
		4047	4097*	4100	4144*	4182*	4183*	4185	4215*	4217*	4219	4254*	4255*	4257
		4287*	4288*	4290	4320*	4321*	4323	4359*	4360*	4362	4393*	4394*	4396	4428*
		4429*	4431	4457*	4458*	4460	4492*	4493*	4495	4513*	4559*	4560*	4562	4600*
		4601*	4603	4705*	7953*	7955*	7957	8156*	8277*	8592*	8593	8602*	8603	8614*
		8615	8622*	8624	8694*	8695	9443*	9464*	9980*	9991*	10014*	10029*	10041*	10056*
		10068*												
%BELL	001216	769#	10515	10684	10771	10793								
%CHARC	060462	10430*	10440*	10447	10456*	10461#								
%CKSWR	060774	10548#	10965											
%CMTAG	001100	723#	2909	2910	2918	2924	2925							
%CM1 =	000006	755#	756#	757#	758#	759#	760#	761#						
%CM2 =	000014	755#	756#	757#	758#	759#	760#	761#						
%CM3 =	000006	753#	755											
%CM4 =	000006	761#	762#	763#	764#	765#	766#	767#						
%CNTLC	061545	10503	10584	10684#										
%CNTLG	061557	10569	10686#											
%CNTLU	061552	10594	10685#											
%CRLF	001223	771#	3273	7855	10429	10464	10605	10684	10745	10779	10793	10801	10820	10825
		10845												
%BLK	060236	10351	10385	10393#										
%DOAGN	045010	7790	7799	7805#										
%DTBL	060226	10354	10389#											
%ENDAD	045000	655	7801#											
%ENDCT	044746	7792#												
%ENDMG	045017	7751	7794	7809#										
%ENULL	045014	7754	7797	7808#										
%EOP	044712	3065	3213	7758	7782#									
%EOPCT	044740	7789#	7793											
%ERFLG	001103	726#	10258	10298	10300	10306*	10327	10766*	10793					
%ERMAX	001115	732#	2926*	10300	10322*	10327								
%ERROR	061746	2918	3082	10759#										
%ERRPC	001116	733#	2141	2143	2145	2148	2151	2154	2156	2159	2163	2166	2168	2170
		2173	2176	2179	2183	2186	2190	2193	2196	2199	2202	2206	2485	2487
		2489	10773*	10774*	10775	10793	10807							
		788#	10815											
%ERRTB	001226	10778	10800#											
%ERRTY	062114	730#	7743	7745*	10772*	10793								
%ERTTL	001112	768#	2925*	10321*	10788	10790	10793							
%ESCAP	001214	751#	10433	10464										
%FILLC	001156	750#	10464											
%FILLS	001155	734#												
%GDADR	001120	736#	2141	2143	2145	2168	2176	2179	2183	2186	2190	2487	2489	3153*
%GD DAT	001124	3598*	3605	3622*	3634	3652*	3661	3679*	3692	3710*	3720	3738*	3751	3786*

[illegible]

\$SETUP= 000117	2907#	2915	2916	2918	2920	2922	2924	2925	2927	3003	7784	10268	10507
	10512	10513	10543	10691	10760	10784	10792						
\$SS1 = 000000	2946#												
\$STUP = 177777	2907#												
\$SVLAD 057756	10289	10318#											
\$SVPC = 000200	653#	658											
\$SWR = 167770	474#	484	516	517	518	519	520	521	522	523	767	768	769
	2924	2925	2927	2928	3038	3118	3145	3163	3302	3426	3464	3523	3548
	3590	3614	3644	3671	3702	3730	3773	3817	3887	3931	3970	4030	4077
	4136	4170	4197	4232	4271	4304	4337	4375	4414	4443	4476	4507	4539
	4578	4614	4638	4737	4915	5086	5234	5406	5558	5731	5911	6101	6254
	6424	6600	6785	6932	7096	7280	7470	7657	7692	7729	7779	7785	7800
	7806	7808	10259	10260	10261	10262	10263	10280	10292	10294	10295	10298	10299
	10300	10307	10308	10309	10320	10323	10326	10751	10752	10753	10754	10755	10769
	10776	10781	10785	10793	11010								
\$SWRMK= 000000	523	524	10263	10264	10296								
\$TIMES 001212	767#	2924*	3038*	3118*	3145*	3163*	3302*	3548*	3590*	3614*	3644*	3671*	3702*
	3730*	3773*	3817*	3887*	3931*	3970*	4030*	4077*	4136*	4170*	4197*	4232*	4271*
	4304*	4337*	4375*	4414*	4443*	4476*	4507*	4539*	4578*	7729*	7785*	10307*	10314
	10317*	10326											
\$TKB 001146	746#	10467	10488	10499	10524	10552	10579						
\$TKCNT 060466	10468#	10483*	10513	10530*	10644	10646*							
\$TKINT 060506	2950	3165	9056	10483#	10504	10565							
\$TKQEN= 060505	10472#	10538	10649										
\$TKQIN 060470	10469#	10484*	10485	10536*	10537*	10538	10540*						
\$TKQOU 060472	10470#	10485*	10647	10648*	10649	10651*							
\$TKQSR 060474	10471#	10484	10540	10651									
\$TKS 001144	745#	4672	4673	10467	10489*	10520*	10522	10528*	10550	10566*	10576	10588*	10608*
\$TKSRV 060556	10486	10499#											
\$TMPO 001176	761#	2156	3228*	3229*	3554*	3570	3788*	3799	3865*	3906*	3946*	4147*	4157
	4490*	4516*	4529	4547*	4567	4791*	4792	4967*	4968	5138*	5139	8141*	8147
	8152	8156	8267*	8281	8383*	8429*							
\$TMP1 001200	762#	2154	2156	2170	3052*	7667*	7668*	7702*	7703*	8142*	8147	8152	8268*
	8283												
\$TMP2 001202	763#	2170	8145*	8149*	8151*	8154*	8385*	8427*					
\$TMP3 001204	764#	2156	2170	8139*	8140*	8384*	8386	8387*					
\$TMP4 001206	765#	8386*	8410*										
\$TMP5 001210	766#	8382*	8388*	8394	8511*	8590	8612						
\$TN = 000073	474#	484	3018	3031	3038#	3040	3110	3118#	3120	3126	3141	3145#	3146
	3152	3159	3163#	3288	3292	3302#	3417	3426#	3429	3454	3464#	3465	3519
	3523#	3524	3544	3548#	3586	3590#	3591	3606	3610	3614#	3615	3635	3640
	3644#	3645	3662	3667	3671#	3672	3693	3698	3702#	3703	3721	3726	3730#
	3731	3752	3769	3773#	3774	3813	3817#	3818	3875	3883	3887#	3888	3916
	3927	3931#	3932	3956	3966	3970#	3971	3991	4026	4030#	4031	4054	4073
	4077#	4078	4107	4132	4136#	4137	4166	4170#	4171	4186	4193	4197#	4198
	4204	4220	4228	4232#	4233	4239	4258	4267	4271#	4272	4291	4300	4304#
	4305	4324	4333	4337#	4338	4344	4363	4371	4375#	4376	4397	4399	4410
	4414#	4415	4432	4439	4443#	4444	4461	4472	4476#	4477	4496	4503	4507#
	4508	4535	4539#	4540	4574	4578#	4579	4604	4610	4614#	4615	4628	4638#
	4640	4646	4697	4719	4723	4737#	4738	4797	4898	4915#	4916	4974	5074
	5086#	5087	5144	5220	5234#	5235	5353	5387	5396	5406#	5408	5479	5526
	5546	5558#	5560	5650	5705	5719	5731#	5733	5833	5888	5899	5911#	5913
	6009	6075	6091	6101#	6103	6174	6222	6242	6254#	6256	6346	6398	6412
	6424#	6426	6524	6577	6588	6600#	6602	6701	6765	6775	6785#	6787	6858
	6905	6920	6932#	6933	7022	7073	7083	7096#	7097	7202	7256	7267	7280#
	7281	7379	7446	7457	7470#	7472	7570	7635	7648	7657#	7658	7675	7680

SEQ 0270

[illegible]

CZRJHDO,RP04/5/6 DSKLS CTRLR2
CZRJHD.P12 01-MAR-79 09:10MACY11 30A(1052) 24-MAY-79 15:07 PAGE 275
CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0271

CHECKA	511#	5355	5493	5675	5854	6039	6188	6371	6545	6732	6871	7047	7223	7409	7601
CHECKB	511#	4677	4763	4940	5111	5345	5459	5627	5810	5987	6154	6323	6502	6678	6838
	6999	7179	7357	7547	8235	8334	8507								
COMMEN	1#	486	638#												
ENDCOM	1#	493	638#												
ERROR	532#	3053	3133	3155	3512	3568	3607	3636	3663	3694	3722	3753	3797	3879	3919
	3957	3987	4050	4103	4155	4187	4221	4254	4292	4325	4364	4398	4433	4462	4497
	4527	4565	4605	4709	4712	4796	4868	4889	4972	5065	5142	5212	5367	5369	5486
	5490	5529	5531	5654	5662	5708	5710	5837	5846	5891	5893	6013	6022	6034	6078
	6080	6181	6185	6225	6227	6350	6358	6401	6403	6528	6537	6580	6582	6705	6714
	6726	6768	6770	6864	6868	6908	6910	7026	7034	7076	7078	7206	7215	7259	7261
	7383	7392	7404	7449	7451	7574	7583	7595	7638	7640	7673	7681	7711	8048	8055
	8059	8072	8077	8080	8102	8110	8116	8121	8157	8555	8557	8581	8583	8598	8609
	8619	8629	8699	8903	8909	8978									
ESCAPE	1#	638#													
GETPRI	1#	638#													
GETSWR	1#	474#	638#	3003											
HCOMPR	511#														
HCOMPW	511#														
MAKECL	511#	3519	4610												
MSG	3030#	3033	3109#	3112	3291#	3294	3417#	3419	3453#	3456	4628#	4630	4722#	4725	4897#
	4900	5073#	5076	5220#	5222	5395#	5398	5545#	5548	5718#	5721	5898#	5901	6090#	6093
	6241#	6244	6411#	6414	6587#	6590	6774#	6777	6919#	6922	7082#	7085	7266#	7269	7456#
	7459	7647#	7650	7684#	7687	7719#	7721								
MULT	1#	638#													
NEWTST	1#	638#	3031	3110	3141	3159	3292	3417	3454	3519	3544	3586	3610	3640	3667
	3698	3726	3769	3813	3883	3927	3966	4026	4073	4132	4166	4193	4228	4267	4300
	4333	4371	4410	4439	4472	4503	4535	4574	4610	4628	4723	4898	5074	5220	5396
	5546	5719	5899	6091	6242	6412	6588	6775	6920	7083	7267	7457	7648	7685	7719
POP	1#	638#	7917	7994	8194	8291	8432	8480	8663	8701	8928	8993	9030	9319	9492
	9563	9630	9702	9904	10072	10193	10233	10380	10734	10997	10998				
PUSH	1#	638#	7907	7981	8184	8258	8374	8377	8452	8644	8679	8788	8953	9004	9226
	9381	9516	9585	9658	9778	9942	10120	10214	10339	10708	10978	10984			
REPORT	1#	638#													
RFORGC	511#	5526	5705	5888	6075	6222	6398	6577	6765	6905	7073	7256	7446	7635	
RH7OCK	511#	3121	4199	4234	4339	4641									
SAVE	511#	10761													
SAVTST	511#	3040	3120	3146	3426	3465	3524	3549	3591	3615	3645	3672	3703	3731	3774
	3818	3888	3932	3971	4031	4078	4137	4171	4198	4233	4272	4305	4338	4376	4415
	4444	4477	4508	4540	4579	4615	4640	4738	4916	5087	5235	5408	5560	5733	5913
	6103	6256	6426	6602	6787	6933	7097	7281	7472	7658	7693				
SCOPE	533#	3037	3117	3144	3162	3301	3425	3463	3522	3547	3589	3613	3643	3670	3701
	3729	3772	3816	3886	3930	3969	4029	4076	4135	4169	4196	4231	4270	4303	4336
	4374	4413	4442	4475	4506	4538	4577	4613	4637	4736	4914	5085	5233	5405	5557
	5730	5910	6100	6253	6423	6599	6784	6931	7095	7279	7469	7656	7691	7728	7783
SETPRI	1#	638#	10640												
SETTRA	10949#	10958	10959	10960	10961	10963	10965	10966	10967	10968	10969	10970	10971		
SETUP	1#	638#	2908												
SKIP	1#	511#	638#	3018	3152	3288	3429	3606	3635	3662	3693	3721	3752	3875	3916
	3956	3991	4054	4107	4186	4220	4258	4291	4324	4363	4397	4399	4432	4461	4496
	4604	4697	4719	5353	5387	5479	5650	5833	6009	6174	6346	6524	6701	6858	7022
	7202	7379	7570	7675	7680	7707									
SLASH	1#	638#													
SMORE	511#	10269													
SPACE	638#														
STARS	1#	503	509	638#	651	718	773	2499	2501	2535	2537	3031	3036	3110	3116

	3141	3143	3159	3161	3239	3249	3292	3300	3333	3343	3370	3409	3417	3424	3454
	3462	3519	3521	3544	3546	3586	3588	3610	3612	3640	3642	3667	3669	3698	3700
	3726	3728	3769	3771	3813	3815	3883	3885	3927	3929	3966	3968	4026	4028	4073
	4075	4132	4134	4166	4168	4193	4195	4228	4230	4267	4269	4300	4302	4333	4335
	4371	4373	4410	4412	4439	4441	4472	4474	4503	4505	4535	4537	4574	4576	4610
	4612	4628	4636	4723	4735	4898	4913	5074	5084	5220	5232	5396	5404	5546	5556
	5719	5729	5899	5909	6091	6099	6242	6252	6412	6422	6588	6598	6775	6783	6920
	6930	7083	7094	7267	7278	7457	7468	7648	7655	7685	7690	7718	7719	7727	7776
	7817	7836	7897	7905	7924	7927	7966	7979	8031	8035	8089	8093	8128	8133	8174
	8182	8204	8207	8248	8256	8300	8303	8346	8372	8440	8450	8487	8502	8635	8642
	8669	8677	8708	8711	8938	8943	8998	9001	9040	9043	9137	9138	9197	9200	9329
	9353	9360	9362	9503	9505	9571	9573	9641	9643	9719	9720	9746	9748	9913	9915
	10083	10085	10203	10205	10236	10245	10255	10329	10396	10466	10543	10558	10629	10653	10694
	10747	10795	10851	10928	10974	10990									
SWRSU	1#	638#	2920#												
TJUMP	511#	3126	4204	4239	4344	4646	4797	4974	5144						
TRMTRP	10949#														
TSCLR	511#														
TSCLR1	511#														
TSCLR2	511#														
TSCLR3	511#														
TSCLR4	511#														
TSCLR5	511#														
TYPBIN	1#	638#													
TYPDEC	1#	638#	7795	10835											
TYPNAM	1#	638#													
TYPNUM	1#	638#													
TYPOCS	1#	638#													
TYPOCT	1#	638#	3357	3363	10571	10807	10832								
TYPTXT	1#	638#	2957	2963	2968	2972	2976	2980	2984	2988	2992	2996	3019	3056	3071
	3078	3169	3195	3200	3204	3208	3255	3261	3267	3347	3353	3359	3392	3398	3404
	3431	3455	3439	3446	3830	3835	3839	3843	7733	7739	7843	7849	7856	7860	7864
	7871	7875	9045	9052	9068	9074	9080	9086	9092	9096	9100	9104	9110	9116	10237
WDATAR	511#														
XREAD	511#														
SSCMRE	716#	755	756	757	758	759	760								
SSCMTM	716#	761	762	763	764	765	766								
SSESCA	1#	638#													
SSNEWI	1#	638#	3031	3110	3141	3159	3292	3417	3454	3519	3544	3586	3610	3640	3667
	3698	3726	3769	3813	3883	3927	3966	4026	4073	4132	4166	4193	4228	4267	4300
	4333	4371	4410	4439	4472	4503	4535	4574	4610	4628	4723	4898	5074	5220	5396
	5546	5719	5899	6091	6242	6412	6588	6775	6920	7083	7267	7457	7648	7685	7719
SSSET	10949#	10958	10959	10960	10961	10963	10965	10966	10967	10968	10969	10970	10971		
SSSKIP	1#	638#													
.EQUAT	1#	474#	528												
.FLOAT	511#														
.HEADE	1#	474#													
.KT11	1#	474#	679												
.SETUP	1#	474#	2906												
.SURMI	1#	474#	512												
.SURLO	474#	524#	525												
.SACT1	1#	474#	649												
.SAPT8	1#														
.SAPTH	1#														
.SAPTV	1#														
.SASTA	1#														

CZRJHDO,RP04/5/6 DSKLS CTRLR2
CZRJHD.P12 01-MAR-79 09:10

MACY11 30A(1052) 24-MAY-79 15:07 ^{B 6} PAGE 277
CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0273

.SCATC	1#	474#	639
.SCMTA	1#	474#	716
.SDB2D	1#		
.SDB2O	1#		
.SDIV	1#		
.SEOP	1#	474#	7774
.SERRO	1#	474#	10745
.SERRT	1#	474#	10793
.SMULT	1#		
.SPOWE	1#	474#	10972
.SRAND	1#		
.SRDDE	1#		
.SRDOC	1#	474#	10692
.SREAD	1#	474#	10464
.SR2AZ	1#		
.SSAVE	1#		
.SSB2D	1#		
.SSB2O	1#		
.SSCOP	1#	474#	10253
.SSIZE	1#		
.SSUPR	1#		
.STRAP	1#	474#	10926
.STYPB	1#		
.STYPD	1#	474#	10327
.STYPE	1#	474#	10394
.STYPO	1#	474#	10849
.S4OCA	1#		
.1170	1#		

. ABS. 062770 000

ERRORS DETECTED: 0

DSKZ:CZRJHD.BIN,DSKZ:CZRJHD.LST/CRF/SOL/NL:TOC:MD:MC:CND/LI:ME=CZRJHD.SML,CZRJHD.P11,CZRJHD.P12
RUN-TIME: 72 108 8 SECONDS
RUN-TIME RATIO: 582/189=3.0
CORE USED: 37k (73 PAGES)