

RP04/5/6

DISKLESS 1
CZRJGC0

AH-9210C-MC
COPYRIGHT 74-78
FICHE 1 OF 2

DEC 1978
digital
MADE IN USA

This image shows a microfiche card with a grid of frames. Each frame contains a small, high-contrast image of a document page, likely containing text or data. The frames are arranged in a regular grid pattern across the card. The text within the frames is too small to be legible, but it appears to be organized into columns and rows, possibly representing a table or a list of data points. The overall appearance is that of a standard microfiche card used for data storage and retrieval.

RP04/5/6

DISKLESS 1
CZRJGCO

AH-9210C-MC
COPYRIGHT '74-78
FICHE 2 OF 2

DEC 1978
digital
MADE IN USA

Table of microfiche frames containing data, organized in columns and rows. The frames contain small text and graphical elements, likely representing data points or system logs. The text is too small to be legible but appears to be structured data.

.REM @

IDENTIFICATION

PRODUCT CODE: AC-9208C-MC
PRODUCT NAME: CZRJGCO RP04/5/6 DISKLESS CONTROLLER TEST-PART I
DATE CREATED: MAY 1976
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) 1974,1978 DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION

DIGITAL	PDP	UNIBUS	MASSBUSS
DEC	DECUS	DECTAPE	

CONTENTS

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

1.0 ABSTRACT

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

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

THE DIAGNOSTIC DOES NOT DO ANY TESTING OF THE RH70 CONTROLLER WHEN IT IS USED ON AN RWPO4/5/6 SYSTEM TO TEST RP04/5/6 DISK DRIVES CONNECTED TO THAT TYPE OF CONTROLLER. IT IS ASSUMED THAT THE RH70 SPECIFIC CONTROLLER DIAGNOSTIC HAVE BEEN SUCCESSFULLY RUN TO COMPLETION BEFORE THIS DIAGNOSTIC IS RUN.

2.0 REQUIREMENTS

2.1 EQUIPMENT

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

2.2 STORAGE

THIS PROGRAM REQUIRES 16K WORDS OF MEMORY.

2.3 PRELIMINARY PROGRAMS

THIS CAN BE THE FIRST PROGRAM RUN ON AN RJP04/5/6 SYSTEM BUT THE CONTROLLER DIAGNOSTICS MUST BE RUN FIRST IN THE CASE OF AN RWPO4/5/6 SYSTEM.

3.0 LOADING PROCEDURE

USE STANDARD PROCEDURE FOR LOADING .ABS TAPES

4.0 STARTING PROCEDURE

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

4.1 CONTROL SWITCH SETTINGS

SEE SECTION 5.1

4.2 STARTING ADDRESS

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

200 START

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

204 RESTART

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

210 START

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

4.3 PROGRAM AND/OR OPERATOR ACTION

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

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

5.0 OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS

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

'SWR = NNNNNN NEW ='

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

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

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

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

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

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

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

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

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

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

NOTE: ALSO SEE SECTION 8.3

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

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

NOTE: ALSO SEE SECTION 8.3

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

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

5.2

SUB-ROUTINE ABSTRACTS

SEE SECTION 9 'SUBROUTINES'

6.0 ERRORS

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

6.1 'FATAL' ERRORS

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

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

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

7.0 RESTRICTIONS

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

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

8.0 MISCELLANEOUS

8.1 EXECUTION TIME

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

8.2 STACK POINTER

THE STACK IS INITIALLY SET TO 1000

8.3 OPERATOR SELECTABLE SCOPE LOOPS

HERE IS A DETAILED EXPLANATION OF HOW THE LOOP ON ERROR WORKS. FOR INSTRUCTIONS REGARDING USAGE OF THIS TECHNIQUE, HIT ^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.

@

486

;*DRIVE MUST BE LOCKED ON PORT A OR PORT B

490
491
492
493
494
495
496
497
498
499
500
501

;*INTERNAL PROGRAM MACROS BEGIN HERE
;*****

;*
;*NOTE: MACROS BEGINNING WITH ''.\$'' ARE SUPPLIED BY AN
;* EXTERNAL SYSMAC.SML SYSTEM MACRO PACKAGE WHICH
;* MUST BE MADE AVAILABLE TO THE SOURCE PROGRAM
;* AT ASSEMBLY TIME.
;*


```
503 000174 000000    DISPREG: .WORD 0      ;;SOFTWARE DISPLAY REGISTER
504 000176 000000    SWREG:  .WORD 0      ;;SOFTWARE SWITCH REGISTER
505
506 000046 041406    $ENDAD          ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
507 000052 020000    .WORD 20000     ;;2)SET LOC.52 TO 20000
508
509 .SBTTL  STARTING ADDRESS
510
511 .=200
512 000200 000137 004240    RA:  JMP  @#BEGIN      ;NORMAL START
513 000204 000137 046244    ADDMOD: JMP @#BASECH   ;MODIFY ADDRESSES
514 000210 000137 004230    JMP  @#BEGIN2        ;SELECT DRIVE START
515
516
517 ;**STARTING ADDRESS 200 FOR NORMAL STARTS
518 ;**THIS WILL TEST ALL DRIVES ON THE SYSTEM A SINGLE DRIVE AT A TIME
519
520 ;**STARTING ADDRESS 204 FOR PARAMETER MODIFICATION,
521 ;**RESTART AUTOMATICALLY FROM 200 AFTER PARAMETER MODIFICATION.
522
523 ;**STARTING ADDRESS 210 WILL TEST ONLY ONE SPECIFIED DRIVE
524
```


527	001100	000000	\$PASS:	.WORD	0	::CONTAINS	PASS COUNT
528	001102	000	\$TSTNM:	.BYTE	0	::CONTAINS	THE TEST NUMBER
529	001103	000	\$ERFLG:	.BYTE	0	::CONTAINS	ERROR FLAG
530	001104	000000	\$ICNT:	.WORD	0	::CONTAINS	SUBTEST ITERATION COUNT
531	001106	000000	\$LPADR:	.WORD	0	::CONTAINS	SCOPE LOOP ADDRESS
532	001110	000000	\$LPERR:	.WORD	0	::CONTAINS	SCOPE RETURN FOR ERRORS
533	001112	000000	\$ERTTL:	.WORD	0	::CONTAINS	TOTAL ERRORS DETECTED
534	001114	000	\$ITEMB:	.BYTE	0	::CONTAINS	ITEM CONTROL BYTE
535	001115	001	\$ERMAX:	.BYTE	1	::CONTAINS	MAX. ERRORS PER TEST
536	001116	000000	\$ERRPC:	.WORD	0	::CONTAINS	PC OF LAST ERROR INSTRUCTION
537	001120	000000	\$GDADR:	.WORD	0	::CONTAINS	ADDRESS OF 'GOOD' DATA
538	001122	000000	\$BDADR:	.WORD	0	::CONTAINS	ADDRESS OF 'BAD' DATA
539	001124	000000	\$GDDAT:	.WORD	0	::CONTAINS	'GOOD' DATA
540	001126	000000	\$BDDAT:	.WORD	0	::CONTAINS	'BAD' DATA
541	001130	000000		.WORD	0	::RESERVED--	NOT TO BE USED
542	001132	000000		.WORD	0		
543	001134	000	\$AUTOB:	.BYTE	0	::AUTOMATIC	MODE INDICATOR
544	001135	000	\$INTAG:	.BYTE	0	::INTERRUPT	MODE INDICATOR
545	001136	000000		.WORD	0		
546	001140	177570	\$SWR:	.WORD	DSWR	::ADDRESS	OF SWITCH REGISTER
547	001142	177570	\$DISPLAY:	.WORD	DDISF	::ADDRESS	OF DISPLAY REGISTER
548	001144	177560	\$TKS:	177560		::TTY KBD	STATUS
549	001146	177562	\$TKB:	177562		::TTY KBD	BUFFER
550	001150	177564	\$TPS:	177564		::TTY PRINTER	STATUS REG. ADDRESS
551	001152	177566	\$TPB:	177566		::TTY PRINTER	BUFFER REG. ADDRESS
552	001154	000	\$NULL:	.BYTE	0	::CONTAINS	NULL CHARACTER FOR FILLS
553	001155	002	\$FILLS:	.BYTE	2	::CONTAINS	# OF FILLER CHARACTERS REQUIRED
554	001156	012	\$FILLC:	.BYTE	12	::INSERT	FILL CHARS. AFTER A 'LINE FEED'
555	001157	000	\$TPFLG:	.BYTE	0	::'TERMINAL	AVAILABLE' FLAG (BIT<07>=0=YES)
556	001160	000000	\$REGAD:	.WORD	0	::CONTAINS	THE ADDRESS FROM
557	001162	000000	\$REG0:	.WORD	0	::CONTAINS	((REGAD)+0)
558	001164	000000	\$REG1:	.WORD	0	::CONTAINS	((REGAD)+2)
559	001166	000000	\$REG2:	.WORD	0	::CONTAINS	((REGAD)+4)
560	001170	000000	\$REG3:	.WORD	0	::CONTAINS	((REGAD)+6)
561	001172	000000	\$REG4:	.WORD	0	::CONTAINS	((REGAD)+10)
562	001174	000000	\$REG5:	.WORD	0	::CONTAINS	((REGAD)+12)
563	001176	000000	\$TMP0:	.WORD	0	::USER	DEFINED
564	001200	000000	\$TMP1:	.WORD	0	::USER	DEFINED
565	001202	000000	\$TMP2:	.WORD	0	::USER	DEFINED
566	001204	000000	\$TMP3:	.WORD	0	::USER	DEFINED
567	001206	000000	\$TMP4:	.WORD	0	::USER	DEFINED
568	001210	000000	\$TMP5:	.WORD	0	::USER	DEFINED
569	001212	000000	\$TIMES:	0		::MAX.	NUMBER OF ITERATIONS
570	001214	000000	\$ESCAPE:	0		::ESCAPE	ON ERROR ADDRESS
571	001216	177607	\$BELL:	.ASCIZ	<207><377><377>	::CODE	FOR BELL
572	001222	077	\$QUES:	.ASCII	/?/	::QUESTION	MARK
573	001223	015	\$CRLF:	.ASCII	<15>	::CARRIAGE	RETURN
574	001224	000012	\$LF:	.ASCIZ	<12>	::LINE	FEED

000377

12
12
12
12
12
12
12
12

575					
576					
577					
578			:*ITEM1		
579	001226	057460	EM1		:WRONG DATA IN READING OR WRITING HARDWARE REGISTER
580	001230	062733	DH1		:PC
581					:REG. ADDR.
582					:GOOD DATA
583					:RECEIVED DATA
584	001232	067312	DT1		:\$ERRPC,REGADR,\$GDDAT,\$BDDAT
585	001234	070030	DF1		:0,0,0,0,0
586					
587					
588					
589					
590					
591			:*ITEM2		
592	001236	057543	EM2		:ERROR ON DATA COMMAND
593					
594	001240	066070	DH33		:PC
595					:PC OF JSR
596					:TEST NO
597					:WORD NO.
598					:GOOD DATA
599					:CONTENTS OF RHCS1
600					:CONTENTS OF RHDS1
601					:CONTENTS OF RHER1
602	001242	067672	DT33		:\$ERRPC,PCJSR,\$TSTNM,ERWORD,\$GDDAT,CS1,DS1,ER1
603	001244	070200	DF33		:0,0,0,1,0,0,0,0
604					
605					
606			:*ITEM3		
607	001246	057543	EM2		:ERROR ON DATA COMMAND
608					
609	001250	065645	DH32		:PC
610					:PC OF JSR
611					:TEST NO
612					:WORD NO.
613					:GOOD DATA
614					:BAD DATA
615					:CONTENTS OF RHCS1
616					:CONTENTS OF RHDS1
617					:CONTENTS OF RHER1
618					
619	001252	067646	DT32		:\$ERRPC,PCJSR,\$TSTNM,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1
620	001254	070167	DF32		:0,0,0,1,0,0,0,0,0
621					
622					
623			:*ITEM4		
624	001256	057543	EM2		:ERROR ON DATA COMMAND
625					
626	001260	065442	DH31		:PC
627					:TEST NO
628					:WORD NO.
629					:GOOD DATA
630					:BAD DATA

Line	Code	Address	Item	Register	Description
799			*ITEM23		
800	001446	000000		0	:NO LONGER USED DUE TO SPECIAL 'NED'
801	001450	000000		0	:TEST TABLE TYPE OUT ROUTINE
802	001452	000000		0	
803	001454	000000		0	
804					
805					
806			*ITEM 24		
807	001456	060545		EM24	:LOOK AHEAD REGISTER AT THE
808					:BEGINNING OF A SECTOR IS IN
809					:ERROR
810	001460	064546		DH24	:PC
811					:RHDST
812					:BAD RHLA
813					:GOOD RHLA
814					:SECTOR NO
815					:SECTOR CLOCK
816	001462	067526		DT24	:\$ERRPC,DST,\$BDDAT,\$TMP1,\$TMP2,\$TMP3
817	001464	070124		DF24	:0,0,0,0,0
818					
819			*ITEM 25		
820	001466	060640		EM25	:LOOK AHEAD REGISTER IS
821					:IN ERROR
822					
823	001470	064546		DH24	:PC
824					:RHDST
825					:BAD RHLA
826					:GOOD RHLA
827					:SECTOR NO
828					:SECTOR CLOCK
829	001472	067526		DT24	:\$ERRPC,DST,\$BDDAT,\$TMP1,\$TMP2,\$TMP3
830	001474	070124		DF24	:0,0,0,0,0
831			*ITEM26		
832	001476	057631		EM11	:CONTROLLER OR DRIVE STATUS
833					
834	001500	064731		DH26	:PC
835					:PC OF JSR
836					:FAILING REGISTER ADDRESS
837					:CONTENTS OF RHCS1
838					:CONTENTS OF RHCS2
839					:CONTENTS OF RHDS1
840					:CONTENTS OF RHER1
841					
842	001502	067546		DT26	:\$ERRPC,PCJSR,\$BDADR,CS1,CS2,DS1,ER1
843	001504	070133		DF26	:0,0,0,0,0,0,
844					
845					
846					
847			*ITEM27		
848	001506	057460		EM1	:ERROR IN READING OR WRITING HARDWARE REGISTER
849					
850	001510	065134		DH27	:PC
851					:PC OF JSR
852					:TEST NUMBER
853					:FAILING REGISTER
854					:GOOD DATA

855				:RECEIVED DATA
856				
857	001512	067570	DT27	:\$ERRPC,PCJSR,TSTNM,REGADR,\$GDDAT,\$BDDAT
858	001514	070143	DF27	:0,0,0,0,0,0
859				
860				
861				
862			:*ITEM30	
863	001516	060700	EM30	:CURRENT CYLINDER DOES NOT REFLECT DESIRED CYLINDER REG.
864	001520	065277	DH30	:PC
865				:PC OF JSR
866				:REGISTER ADDRESS
867				:GOOD DATA
868				:BAD DATA
869				
870	001522	067606	DT30	:\$ERRPC,PCJSR,REGADR,\$GDDAT,\$BDDAT
871	001524	070151	DF30	:0,0,0,0,0
872				
873				
874				
875			:*ITEM31	
876	001526	061021	EM31	:ECC GENERATED IS INCORRECT
877				:EVERY WORD IN THIS SECTOR IS GIVEN IN 'DATA USED'
878				
879	001530	066273	DH34	:PC
880				:TEST NUMBER
881				:GOOD ECC1
882				:GOOD ECC2
883				:WRITTEN ECC1
884				:WRITTEN ECC2
885				:DATA USED
886				
887	001532	067714	DT34	:\$ERRPC,TSTNM,GECC1,GECC2,WECC1,WECC2,DISK
888				
889	001534	070210	DF34	:0,0,0,0,0,0,0
890				
891				
892			:*ITEM32	
893	001536	061144	EM32	:ON READ COMMAND AFTER DATA AND ECC HAVE BEEN READ
894				:ECC REGISTER OR RHER1 IS IN ERROR
895				:ONLY LOWER 11 BITS OF PATTERN REGISTER
896				:CAN BE READ
897				:THIS SHUOLD MATCH LOWER 11 BITS OF ECC1
898				
899	001540	066456	DH35	:PC
900				:TEST NUMBER
901				:GOOD ECC1
902				:GOOD ECC2
903				:PATTERN REGISTER
904				:RHER1
905				
906	001542	067734	DT35	:\$ERRPC,TSTNM,GECC1,GECC2,EC2,ER1
907				
908	001544	070217	DF35	:0,0,0,0,0,0
909				
910				

911					
912			:*ITEM33		
913	001546	061433	EM33		:HIGH COUNT BIT NOT HIGH AFTER 38859 CLOCKS
914	001550	066661	DH36		:PC
915					:PC OF JSR
916					:TEST NUMBER
917					:RHMR
918					:POSITION REG.
919					:PATTERN REGISTER
920					
921	001552	067756	DT36		:\$ERRPC,PCJSR,TSTNM,MR,EC1,EC2
922					
923	001554	070227	DF36		:0,0,0,0,0,0
924					
925			:*ITEM34		
926	001556	061505	EM34		:ZERO DETECT BIT NOT HIGH WHEN THE
927					:32 BIT ECC REGISTER HAS ITS 21 BITS
928					:OF ZEROS
929					:ERROR PRINTOUT WILL CONTINUE TILL
930					:ZERO DETECT BIT IS HIGH
931	001560	066661	DH36		:PC
932					:PC OF JSR
933					:TEST NUMBER
934					:RHMR
935					:POSITION REG.
936					:PATTERN REGISTER
937					
938	001562	067756	DT36		:\$ERRPC,PCJSR,TSTNM,MR,EC1,EC2
939					
940	001564	070227	DF36		:0,0,0,0,0,0
941					
942					
943					
944			:*ITEM35		
945	001566	061600	EM35		:POSITION REGISTER OR 11 BITS OF
946					:PATTERN REGISTER INCORRECT
947					:LOWER 11 BITS OF PATTERN REGISTER
948					:SHOULD MATCH LOWER 11 BITS OF GOOD ECC1
949					:DATA ENVELOPE AND N-CODE ZEROS ARE IN DECIMAL
950					
951	001570	067020	DH37		:PC
952					:TEST NUMBER
953					:ECC POSITION
954					:GOOD POSITION
955					:GOOD ECC1
956					:GOOD ECC2
957					:ECC PATTERN
958					:DATA ENVELOPE
959					:N-CODE ZEROS
960					
961	001572	067774	DT37		:\$ERRPC,TSTNM,EC1,POSITI,GECC1,GECC2,EC2,DATENV,ZCODE
962					
963	001574	070235	DF37		:0,0,0,0,0,0,0,0,0,0
964					
965					
966					

1023	000020	PAT=	20	:INVERT PARITY ON MASS BUS TO EVEN (BIT #4)
1024	000040	CLR=	40	:CLEAR (BIT #5)
1025	000100	IR=	100	:INPUT READY (BIT #6)
1026	000200	OR=	200	:OUTPUT READY (BIT #7)
1027	000400	MPE=	400	:MASS BUS PARITY ERROR (BIT #8)
1028	001000	MXF=	1000	:MISSED TRANSFER ERROR (BIT #9)
1029	002000	PGE=	2000	:PROGRAM ERROR (BIT #10)
1030	004000	NEM=	4000	:NON EXISTANT MEMORY (BIT #11)
1031	010000	NED=	10000	:NON EXISTANT DRIVE (BIT #12)
1032	020000	UPE=	20000	:UNIBUS PARITY ERROR (BIT #13)
1033	040000	WCE=	40000	:WRITE CHECK ERROR (BIT #14)
1034	100000	DLT=	100000	:DATA LATE (BIT #15)
1035				
1036		;*DATA BUFFER REGISTER (RHDB)		
1037		;*EACH BIT IS CALLED BY BIT NUMBER		
1038				
1039				
1040				
1041		;*RP04 REGISTERS		
1042				
1043				
1044				
1045		;*CONTROL AND STATUS 1 REGISTER. (#00)		
1046				
1047	000001	GO=	1	:GO (BIT #0)
1048	000100	IE=	100	:INTERRUPT ENABLE (BIT #6)
1049	000200	RDY=	200	:READY (BIT #7)
1050	000400	A16=	400	:HIGH ORDER UNIBUS BITS (BIT #8)
1051	001000	A17=	1000	:HIGH ORDER UNIBUS BITS (BIT #9)
1052	002000	PSEL=	2000	:PORT SELECT (BIT #10)
1053	004000	DVA=	4000	:DEVICE AVAILABLE (BIT #11)
1054	020000	MCPE=	20000	:MASSBUSS PARITY ERROR (BIT #13)
1055	040000	TRE=	40000	:TRANSFER ERROR (BIT #14)
1056	100000	SC=	100000	:SPECIAL CONDITION (BIT #15)
1057				
1058		;*STATUS REGISTER (RHDS1) (#01)		
1059				
1060	000001	DF5=	1	:DRIVE FORWARD 5"/SEC. (BIT #0)
1061	000002	DFF20=	2	:DRIVE FORWARD 20"/SEC. (BIT #1)
1062	000004	DIGB=	4	:DRIVE TO INNER GAVRD BAND (BIT #2)
1063	000010	GRV=	10	:GO REVERSE (BIT #3)
1064	000020	DL64=	20	:DIFFERENCE LESS THAN 64 (BIT #4)
1065	000040	DE1=	40	:DIFFERENCE EQUALS 1 (BIT #5)
1066	000100	VV=	100	:VOLUME VALID (BIT #6)
1067	000200	DRY=	200	:DRIVE READY (BIT #7)
1068	000400	DPR=	400	:DRIVE PRESENT (BIT #8)
1069	001000	PROG=	1000	:PROGRAMABLE (BIT #9)
1070	002000	LST=	2000	:LAST SECTOR TRANSFERRED (BIT #10)
1071	004000	WRL=	4000	:WRITE LOCK (BIT #11)
1072	010000	MOL=	10000	:MEDIUM ON-LINE (BIT #12)
1073	020000	PIP=	20000	:POSITIONING OPERATION IN PROGRESS (BIT #13)
1074	040000	ERR=	40000	:COMPOSIT ERROR. (BIT #14)
1075	100000	ATA=	100000	:ATTENTION ACTIVE (BIT #15)
1076				
1077		;*ERROR REGISTER #01 (RHER1) (#02)		
1078	000001	ILF=	1	:ILLEGAL FUNCTION (BIT #0)


```
1135
1136
1137
1138      000001
1139      000002
1140      000004
1141      000010
1142      000020
1143      000040
1144      000100
1145      000200
1146      000400
1147      001000
1148      002000
1149      004000
1150      010000
1151      020000
1152      040000
1153      100000
1154
1155      ;*LOOK-AHEAD REGISTER (RHLA) (#07)
1156
1157      000001
1158      000002
1159      000004
1160      000010
1161      000020
1162      000040
1163      000100
1164      000200
1165      000400
1166      001000
1167      002000
1168      004000
1169      010000
1170      020000
1171      100000
1172
1173      ;*RP04 ERROR REGISTER #2 (RHER2) (#10)
1174
1175      000001
1176      000002
1177      000004
1178      000010
1179      000020
1180      000040
1181      000100
1182      000200
1183      000400
1184      001000
1185      002000
1186      004000
1187      020000
1188
1189
1190
```

EXT1=	1	:EXTENSION 1 (BIT #0)
EXT2=	2	:EXTENSION 2 (BIT #1)
EXT4=	4	:EXTENSION 3 (BIT #2)
EXT10=	10	:EXTENSION 4 (BIT #3)
EXT20=	20	:EXTENSION 5 (BIT #4)
EXT40=	40	:EXTENSION 6 (BIT #5)
SC1=	100	:SECTOR COUNT FIELD 0 (BIT #6)
SC2=	200	:SECTOR COUNT FIELD 1 (BIT #7)
SC4=	400	:SECTOR COUNT FIELD 2 (BIT #8)
SC10=	1000	:SECTOR COUNT FIELD 3 (BIT #9)
SC20=	2000	:SECTOR COUNT FIELD 4 (BIT #10)
TRK1=	4000	:TRACK FIELD 1 (BIT #11)
TRK2=	10000	:TRACK FIELD 2 (BIT #12)
TRK4=	20000	:TRACK FIELD 3 (BIT #13)
TRK10=	40000	:TRACK FIELD 4 (BIT #14)
TRK20=	100000	:TRACK FIELD 5 (BIT #15)

WCU=	1	:WRITE CURRENT UNSAFE (BIT #0)
CSF=	2	:CURRENT SINK FAILURE (BIT #1)
WSU=	4	:WRITE SELECT UNSAFE (BIT #2)
CSU=	10	:CURRENT SWITCH UNSAFE (BIT #3)
MSE=	20	:MOTOR SEQUENCE ERROR (BIT #4)
TDF=	40	:TRANSITIONS DETECTOR FAILURE (BIT #5)
TUF=	100	:TRANSITIONS UNSAFE (BIT #6)
FEN=	200	:FAILSAFE ENABLED (BIT #7)
WRU=	400	:WRITE READY UNSAFE (BIT #8)
MHS=	1000	:MULTIPLE HEAD SELECT (BIT #9)
NHS=	2000	:NO HEAD SELECTION (BIT #10)
IXE=	4000	:INDEX ERROR (BIT #11)
VU30=	10000	:30VOLT UNSAFE (BIT #12)
PLU=	20000	:PLO UNSAFE (BIT #13)
ACU=	100000	:ACUNSAFE (BIT #15)

WCU=	1	:WRITE CURENT UNSAFE
CSF=	2	:CURRENT SINK FAILURE
WSU=	4	:CURENT SELECT UNSAFE
CSU=	10	:CURRENT SWITCH UNSAFE
RAW=	20	:READ AND WRITE
TDF=	40	:TRANSITIONS DETECTOR FAILURE
TUF=	100	:TRANSITIONS UNSAFE
ABS=	200	:ABNORMAL STOP
WRU=	400	:WRITE READY UNSAFE
MHS=	1000	:MULTIPLE HEAD SELECT
NHS=	2000	:NO HEAD SELECTION
IXE=	4000	:INDEX ERROR
PLU=	20000	:PLO UNSAFE


```
      ;*OFFSET REGISTER (RHOF) (#11)
```


1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305

.SBTTL REGISTER ADDRESSES

;*RP04 VECTOR ADDRESS

RPVEC: 254 ;RP04 VECTOR ADDRESS

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

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

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

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

;*ADDITIONAL REGISTERS LOCATED IN THE RH70 CONTROLLER LOGIC

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

1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334

001704 000000
001706 000000
001710 000000
001712 000000

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

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

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

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

```
1335
1336      ;*FLAGS & INTERNAL PROGRAM CONTROL WORDS
1337
1338 001754 000010 UNITS: .BLKW 8.      ;TABLE OF DRIVES PRESENT TO TEST
1339 001774 000000 UNIT:  .WORD 0      ;UNIT UNDER TEST
1340 001776 000000 NOUNIT: .WORD 0      ;NUMBER OF UNITS PRESENT
1341      ;USED TO KEEP TRACK OF UNIT UNDER TEST
1342 002000 000000 NUNIT:  .WORD 0      ;USED TO DETERMINE IF THERE IS MORE
1343      ;THAN ONE UNIT
1344 002002 000000 SELECT: .WORD 0      ;ALL ONES INDICATE UNIT TO BE SELECTED
1345 002004 000000 UNITS1: .WORD 0      ;UNIT NO. SELECTED
1346
1347 002006 000000 ERFLG$: 0      ;ERROR FLAG
1348
1349 002010 000000 SAVDT: 0      ;SAVE DRIVE TYPE REGISTER
1350      ;FOR COMPARISON IN DRIVE CLEAR TEST
1351      ;AND RH INIT TEST
1352 002012 000000 SAVSN: 0      ;SAVE SERIAL NUMBER REGISTER
1353      ;FOR COMPARISON IN DRIVE CLEAR TEST
1354      ;AND RH INIT TEST
1355
1356 002014 000000 PCJSR: 0      ;SAVE PC OF JSR WHICH GAVE THE ERROR
1357
1358 002016 000000 ATTENT: 0      ;ATTENTION BIT FOR PRESENT UNIT
1359 002020 000000 TOTALAT: 0      ;TATAL ATTENTION BITS
1360
1361 002022 000000 TMPILL: 0      ;TEMPORARY ILLEGAL FUNCTION
1362
1363 002024 000000 TSECC: 0      ;FLAG TO SAY IF ECC TEST OR NOT
1364      ;WHEN =177777 IT IS AN ECC TEST
1365      ;WHEN =0IT IS NOT AN ECC TEST
1366
1367 002026 000000 TESDTE: 0      ;FLAG TO SAY IF DRIVE TIMING ERROR OR NOT
1368      ;WHEN = 177777 IT IS A DTE TEST
1369      ;WHEN = 0 IT IS NOT A DTE TEST
1370
1371 002030 000000 TAGDTE: 0      ;TEMPORARY TAG USED IN DRIVE TIMING
1372      ;ERROR TEST
1373
1374 002032 000000 TSTNM: 0      ;TEST NUMBER
1375
1376 002034 000000 FIRST: 0      ;IF ZERO WILL TYPE HEADER
1377
1378 002036 000000 RP06: 0      ;IF 0 PROGRAM WILL TREAT DRIVE AS RP04
1379
1380 002040 000000 RH70: 0      ;IF 1 PROGRAM IS RUNNING ON RH70
1381      ;IF 0 PROGRAM IS ON AN RH11
```

```
1382
1383
1384           ;*FUNCTION EQUATES
1385
1386           ;*TABLE OF FUNCTIONS FOR RHCS1, THEN 'GO' BIT HAS TO BE SET
1387
1388 002042      FUTABL:
1389 002042      NOPERA: 0           ;NO OPERATION
1390 002044      UNLOAD: 2         ;UNLOAD (STAND BY)
1391 002046      RECALI: 6         ;RECALIBRATE
1392 002050      DCLEAR: 10        ;DRIVE CLEAR
1393 002052      RELEAS: 12        ;RELEASE (DUAL-PORT OPERATION)
1394 002054      SERCH: 30         ;SEARCH COMMAND
1395 002056      WRCHEK: 50        ;WRITE CHECK DATA
1396 002060      WRCHDT: 52        ;WRITE CHECK HEADER AND DATA
1397 002062      WRIDAT: 60        ;WRITE DATA
1398 002064      WRIFOR: 62        ;WRITE HEADER AND DATA (FORMAT)
1399 002066      READAT: 70        ;READ DATA
1400 002070      REFOR: 72         ;READ HEADER AND DATA
1401 002072      SEECOM: 4         ;SEEK COMMAND
1402 002074      OFSETC: 14        ;OFFSET COMMAND
1403 002076      RETCL: 16         ;RETURN TO CENTERLINE
1404 002100      PKACK: 22         ;PACK ACKNOWLEDGE
1405 002102      READIN: 20        ;READ IN
1406 002104      ILLEGL: .WORD     ;COMPUTED ILLEGAL FUNCTION
1407
1408
1409           ;*DATA BUFFERS FOR READ WRITE
1410
1411
1412 002110      000422      WRFROM: .BLKW 274.      ;WRITE FROM THIS BUFFER
1413 003154      000422      REINTO: .BLKW 274.      ;READ INTO THIS BUFFER
1414
1415
1416
1417           ;*TABLE FOR ATTENTION BITS
1418           ;*ATTENTION TABLE
1419
1420 004220      001      002      004      ATABLE: .BYTE 1,2,4,10,20,40,100,200
1421 004223      010      020      040
1422 004226      100      200
1423
```



```

1424
1425
1426          .SBTTL      ***DIAGNOSTIC CODE***
1427          .SBTTL
1428
1429          .SBTTL      SETUP TESTS
1430
1431 004230 012737 177777 002002 BEGIN2: MOV    #-1,@#SELECT ;SELECT UNIT
1432 004236 000402                BR      START
1433 004240 005037 002002          BEGIN: CLR    @#SELECT ;DO NOT SELECT UNIT
1434                                         ;NORMAL RUN
1435
1436          START:
1437 004244 000005                RESET
1438 004246 012706 001100          MOV    #SCMTAG,R6 ;:FIRST LOCATION TO BE CLEARED
1439 004252 005026                CLR    (R6)+ ;:CLEAR MEMORY LOCATION
1440 004254 022706 001140          CMP    #SWR,R6 ;:DONE?
1441 004260 001374                BNE    -6 ;:LOOP BACK IF NO
1442 004262 012706 001000          MOV    #STACK,SP ;:SETUP THE STACK POINTER
1443 004266 012737 054220 000020          MOV    #SCOPE,@#IOTVEC ;:IOT VECTOR FOR SCOPE ROUTINE
1444 004274 012737 000340 000022          MOV    #340,@#IOTVEC+2 ;:LEVEL 7
1445 004302 012737 056436 000030          MOV    #ERROR,@#EMTVEC ;:EMT VECTOR FOR ERROR ROUTINE
1446 004310 012737 000340 000032          MOV    #340,@#EMTVEC+2 ;:LEVEL 7
1447 004316 012737 057206 000034          MOV    #TRAP,@#TRAPVEC ;:TRAP VECTOR FOR TRAP CALLS
1448 004324 012737 000340 000036          MOV    #340,@#TRAPVEC+2 ;:LEVEL 7
1449 004332 012737 057276 000024          MOV    #PWRDN,@#PWRVEC ;:POWER FAILURE VECTOR
1450 004340 012737 000340 000026          MOV    #340,@#PWRVEC+2 ;:LEVEL 7
1451 004346 005037 001212                CLR    $TIMES ;:INITIALIZE NUMBER OF ITERATIONS
1452 004352 005037 001214                CLR    $ESCAPE ;:CLEAR THE ESCAPE ON ERROR ADDRESS
1453 004356 112737 000001 001115          MOV    #1,$ERMAX ;:ALLOW ONE ERROR PER TEST
1454 004364 012737 004364 001106          MOV    #,$LPADR ;:INITIALIZE THE LOOP ADDRESS FOR SCOPE
1455 004372 012737 004372 001110          MOV    #,$LPERR ;:SETUP THE ERROR LOOP ADDRESS
1456 004400 013746 000004                MOV    @#ERRVEC,-(SP) ;:SAVE ERROR VECTOR
1457 004404 012737 004440 000004          MOV    #64$,@#ERRVEC ;:SET UP ERROR VECTOR
1458 004412 012737 177570 001140          MOV    #DSWR,SWR ;:SETUP FOR A HARDWARE SWICH REGISTER
1459 004420 012737 177570 001142          MOV    #DDISP,DISPLAY ;:AND A HARDWARE DISPLAY REGISTER
1460 004426 022777 177777 174504          CMP    #-1,@SWR ;:TRY TO REFERENCE HARDWARE SWR
1461 004434 001012                BNE    66$ ;:BRANCH IF NO TIMEOUT TRAP OCCURRED
1462 004436 000403                BR     65$ ;:BRANCH IF NO TIMEOUT
1463 004440 012716 004446          64$: MOV    #65$,(SP) ;:SET UP FOR TRAP RETURN
1464 004444 000002                RTI
1465 004446 012737 000176 001140 65$: MOV    #SWREG,SWR ;:POINT TO SOFTWARE SWR
1466 004454 012737 000174 001142          MOV    #DISPREG,DISPLAY
1467 004462 012637 000004          66$: MOV    (SP)+,@#ERRVEC ;:RESTORE ERROR VECTOR
1468
1469
1470
1471 004466 012737 000000 177776 STARTA: MOV    #0,PS ;:SET PROCESSOR STATUS TO 0
1472 004474 012777 054136 175124          MOV    #RPVECT,@RPVEC ;:THIS IS FOR UNTIMELY DRIVE INTERRUPTS
1473 004502 004737 055176                JSR    PC,@#$TKINT ;:INITIALIZE THE TTY KEYBOARD
1474 004506 005737 002034                TST    @#FIRST ;:IS THIS FIRST TIME ROUND ?
1475 004512 001001                BNE    1$ ;:SKIP HEADER IF NOT
1476 004514 000402                BR     2$ ;:DO HEADER IF SO
1477
1478 004516 000137 005326          1$: JMP    @#SND1 ;:SKIP OVERALL PROGRAM HEADER
1479 004522          2$:

```

```

1480 004522 104401 004530          TYPE      ,65$          ;;TYPE ASCIZ STRING
1481 004526 000434          BR        64$          ;;GET OVER THE ASCIZ
1482
1483 004620 104401 004626          TYPE      ,67$          ;;TYPE ASCIZ STRING
1484 004624 000417          BR        66$          ;;GET OVER THE ASCIZ
1485
1486 004664 104401 004672          TYPE      ,69$          ;;TYPE ASCIZ STRING
1487 004670 000433          BR        68$          ;;GET OVER THE ASCIZ
1488 004760 104401 004766          TYPE      ,71$          ;;TYPE ASCIZ STRING
1489 004764 000427          BR        70$          ;;GET OVER THE ASCIZ
1490 005044 104401 005052          TYPE      ,73$          ;;TYPE ASCIZ STRING
1491 005050 000430          BR        72$          ;;GET OVER THE ASCIZ
1492 005132 104401 005140          TYPE      ,75$          ;;TYPE ASCIZ STRING
1493 005136 000415          BR        74$          ;;GET OVER THE ASCIZ
1494 005172 104401 005200          TYPE      ,77$          ;;TYPE ASCIZ STRING
1495 005176 000430          BR        76$          ;;GET OVER THE ASCIZ
1496 005260 104401 005266          TYPE      ,79$          ;;TYPE ASCIZ STRING
1497 005264 000420          BR        78$          ;;GET OVER THE ASCIZ
1498
1499 005326 012737 177777 002034 SND1:  MOV     #-1,@#FIRST    ;NEXT TIME DO NOT GIVE HEADER
1500
1501 005334 005737 000042          TST      @#42          ;;ARE WE RUNNING UNDER XXDP/ACT?
1502 005340 001006          BNE      64$          ;;BRANCH IF YES
1503 005342 023727 001140 000176          CMP      SWR,#SWREG    ;;SOFTWARE SWITCH REG SELECTED?
1504 005350 001005          BNE      65$          ;;BRANCH IF NO
1505 005352 104406          GTSWR                    ;;GET SOFT-SWR SETTINGS
1506 005354 000403          BR        65$
1507 005356 112737 000001 001134 64$:  MOV     #1,$AUTOB      ;;SET AUTO-MODE INDICATOR
1508
1509 005364 032777 010000 173546 RH70CK: BIT     #SW12,@SWR      ;LOOK TO SEE IF USING RH70
1510 005372 001403          BFC      3$          ;IF SW12 = 0, SKIP NEXT
1511 005374 012737 000001 002040          MOV     #1,@#RH70      ;IF SW12 = 1, CU IS AN RH70
1512
1513 005402 005737 002002          3$:  TST      @#SELECT      ;200 START?
1514 005406 001434          BEQ      TST1          ;SKIP NEXT IF STARTING FROM 200 -----)
1515 005410 104401 005416          TYPE      ,65$          ;;TYPE ASCIZ STRING
1516 005414 000422          BR        64$          ;;GET OVER THE ASCIZ
1517 005462 104412          RDOCT
1518 005464 042716 177770          BIC     #177770,(SP)   ;ONLY KEEP LAST 3 BITS
1519 005470 011637 001774          MOV     (SP),@#UNIT    ;SAVE UNIT TO BE TESTED
1520 005474 012637 002004          MOV     (SP)+,@#UNITSL ;SAVE UNIT TO BE TESTED
1521
1522

```



```
1664 007172 XE2:
1665
1666 007172 012700 000010 2$: MOV #8.,R0 ;LOAD 'UNITS' TABLE COUNTER
1667 007176 012703 001754 MOV #UNITS,R3 ;LOAD 'UNITS' TABLE POINTER
1668 007202 012723 177777 3$: MOV #-1,(R3)+ ;PRESET 1ST TABLE BLOCK TO ALL ONES
1669 007206 005300 DEC R0 ;COUNT DOWN
1670 007210 001374 BNE 3$ ;PRESET NEXT BLOCK IF 8 NOT DONE
1671
1672 007212 012703 001754 10$: MOV #UNITS,R3 ;RELOAD THE TABLE POINTER
1673 007216 005005 CLR R5 ;INITIALIZE UNIT NO. TO 0
1674 007220 005037 001776 CLR @#NOUNIT ;NO. OF UNITS PRESENT
1675 007224 012700 000010 MOV #8.,R0 ;RELOAD THE TABLE COUNTER
1676 007230 011137 001176 MOV @R1,@#$TMP0 ;ADDR OF RHAS INTO TEMPORARY STORAGE
1677 007234 006037 001176 4$: ROR @#$TMP0 ;SET CARRY IF 0 BIT = 1 (UNIT ATTEN.)
1678 007240 103120 BCC 5$ ;CHECK NEXT UNIT IF ONE NOT IN BIT 0
1679
1680 007242 010577 172370 11$: MOV R5,@RHCS2 ;INSERT UNIT NO. INTO RHCS2 UNIT ADDR.
1681 007246 022777 024020 172410 CMP #24020,@RHDT ;READ RHDT - IS IT A DUAL PORT RP04 ?
1682 007254 001503 BEQ 6$ ;YES...TYPE THE UNIT NO.
1683 007256 022777 020020 172400 CMP #20020,@RHDT ;READ RHDT - IS IT A SINGLE PORT RP04 ?
1684 007264 001477 BEQ 6$ ;YES...TYPE THE UNIT NO.
1685
1686
1687
1688 007266 022777 024021 172370 CMP #24021,@RHDT ;DUAL PORT RP05 ?
1689 007274 001473 BEQ 6$ ;TYPE UNIT NO. IF SO
1690 007276 022777 020021 172360 CMP #20021,@RHDT ;SINGLE PORT RP05 ?
1691 007304 001467 BEQ 6$ ;TYPE NO. IF SO
1692
1693 007306 022777 024022 172350 CMP #24022,@RHDT ;READ RHDT - IS IT A DUAL PORT RP06 ?
1694 007314 001463 BEQ 6$ ;YES...TYPE THE UNIT NO.
1695 007316 022777 020022 172340 CMP #20022,@RHDT ;READ RHDT - IS IT A SINGLE PORT RP06 ?
1696 007324 001457 BEQ 6$ ;YES...TYPE THE UNIT NO.
1697
1698
1699
1700 ;*NO...IT'S NOT AN RP04/RP05/RP06 DEVICE SO TYPE
1701 ;*OUT THE DEVICE TYPE
1702
1703 007326 104401 007334 TYPE ,65$ ;;TYPE ASCIZ STRING
1704 007332 000410 BR 64$ ;;GET OVER THE ASCIZ
1705
1706 007354 010546 MOV R5,-(SP) ;PUT THE UNIT NUMBER ON STACK
1707 007356 104405 TYPDS ;TYPE IT
1708 007360 104401 007366 TYPE ,67$ ;;TYPE ASCIZ STRING
1709 007364 000406 BR 66$ ;;GET OVER THE ASCIZ
1710 007402 017746 172256 MOV @RHDT,-(SP) ;PUT RHDT ON THE STACK
1711 007406 104402 TYPOC ;TYPE IT
1712 007410 104401 007416 TYPE ,69$ ;;TYPE ASCIZ STRING
1713 007414 000422 BR 68$ ;;GET OVER THE ASCIZ
1714 007462 000407 BR 5$ ;UNIT NOT AN RP04/RP05/RP06 SO TEST NEXT ONE
1715
1716 007464 010523 6$: MOV R5,(R3)+ ;LOAD TABLE POSITION AND INCR IT
1717 007466 104401 001223 TYPE ,$CRLF ;CRLF
1718 007472 010546 MOV R5,-(SP) ;PUT UNIT NO. ON THE STACK
1719 007474 104405 TYPDS ;TYPE THE UNIT NO.
```



```

1735
1736
1737 007544 000004          TST5:  SCOPE
1738 007546 012737 000001 001212    MOV     #1,$TIMES      ;;DO 1 ITERATION
1739 007554 012737 010272 001106    MOV     #1$,$LPADR     ;;SET SCOPE LOOP ADDRESS
1740
1741 007562 012737 000005 002032    MOV     #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
1742 007570 004737 042620          JSR     PC,@#CLDISK    ;FILL UNIT NO.
1743 007574 005037 002016          CLR     @#ATTENT       ;CLEAR
1744
1745                          ;*TEST FOR UNIT #0
1746
1747 007600 005737 001774          TST     @#UNIT         ;IS UNIT #0 NEXT IN THE UNITS TABLE ?
1748 007604 001022          BNE     10$           ;IF NOT, TEST THIS UNIT
1749 007606 012700 000041          MOV     #41,R0        ;IF SO, CHECK THE LOAD MEDIA LOCATION
1750 007612 122710 000011          CMPB    #11,(R0)      ;WAS IT AN RP04/5/6 ?
1751 007616 001015          BNE     10$           ;NO...GO AHEAD WITH TESTING UNIT #0
1752 007620 005737 002002          TST     @#SELECT      ;WAS UNIT #0 SELECTED ?
1753                          ;(IE. WAS IT A 210 START ?)
1754 007624 001012          BNE     10$           ;IF SO...TEST IT
1755
1756                          ;*INCREMENT THE UNITS TABLE TO NEXT DRIVE (IF ANY)
1757                          ;*& DECREMENT THE 'NOUNITS' PRESENT (TO BE TESTED)
1758
1759 007626 012700 001754          MOV     #UNITS,R0     ;LOAD THE UNITS TABLE POINTER
1760 007632 005720          TST     (R0)+         ;SELECT THE NEXT UNIT IN THE TABLE
1761                          ;(DOUBLE INCREMENT THE POINTER, R0)
1762 007634 022710 177777          CMP     #-1,(R0)     ;IS THERE ANOTHER TABLE ENTRY PRESENT ?
1763 007640 001404          BEQ     10$           ;IF NOT (LOC = -1)...MUST USE UNIT #0
1764 007642 011037 001774          MOV     (R0),@#UNIT   ;SET UP TO BE THE UNIT UNDER TEST
1765 007646 005337 001776          DEC     @#NOUNITS     ;DECREMENT BECAUSE UNIT # 0 WON'T BE TESTED
1766 007652 013700 001774          10$:  MOV     @#UNIT,R0    ;R0 CONTAINS UNIT NO.
1767
1768
1769                          ;*SET UP THE PROPER DEVICE TYPE FLAG
1770
1771 007656 010077 171754          MOV     R0,@#RHCS2    ;SET UP UNIT ADDRESS
1772 007662 005037 002036          CLR     @#RP06        ;CLEAR RP06 DEVICE TYPE FLAG
1773 007666 022777 024022 171770          CMP     #24022,@#RHDT ;DUAL PORT RP06 ?
1774 007674 001405          BEQ     2$           ;YES...SET THE FLAG
1775 007676 022777 020022 171760          CMP     #20022,@#RHDT ;SINGLE PORT RP06 ?
1776 007704 001401          BEQ     2$           ;YES...SET FLAG
1777 007706 000403          BR      3$           ;NO...DON'T SET RP06 FLAG
1778 007710 012737 177777 002036 2$:  MOV     #-1,@#RP06    ;SET IT
1779
1780                          3$:  ;ASSUME THE NEXT UNIT IS AN RP04
1781
1782 007716 116037 004220 002016          MOVB    ATABLE(R0),@#ATTENT ;SET APPROPRIATE ATTENTION BIT
1783
1784 007724 104401 007732          TYPE    ,65$         ;;TYPE ASCIZ STRING
1785 007730 000414          BR      64$         ;;GET OVER THE ASCIZ
1786 007762 0137:6 001774          MOV     @#UNIT,-(SP)  ;UNIT NO. TO STACK
1787 007766 104405          TYPDS   ;TYPE DRIVE NO.
1788 007770 104401 007776          TYPE    ,67$         ;;TYPE ASCIZ STRING
1789 007774 000410          BR      66$         ;;GET OVER THE ASCIZ
1790 010016 017746 171644          MOV     @#RHSN,-(SP)  ;;SAVE @#RHSN FOR TYPEOUT

```


1791	010022	104402			TYPOC		::GO TYPE--OCTAL ASCII(ALL DIGITS)
1792	010024	104401	010032		TYPE	,69\$::TYPE ASCIZ STRING
1793	010030	000410			BR	68\$::GET OVER THE ASCIZ
1794	010052	017746	171606		MOV	@RHDT,-(SP)	::SAVE @RHDT FOR TYPEOUT
1795	010056	104402			TYPOC		::GO TYPE--OCTAL ASCII(ALL DIGITS)
1796							
1797	010060	022777	024020	171576	CMP	#24020,@RHDT	:DUAL PORT RP04 ?
1798	010066	001425			BEQ	4\$:TYPE ASCII MSG OUT
1799	010070	022777	020020	171566	CMP	#20020,@RHDT	:SINGLE PORT RP04 ?
1800	010076	001421			BEQ	4\$:TYPE THE MESSAGE
1801							
1802	010100	022777	024021	171556	CMP	#24021,@RHDT	:DUAL PORT RP05 ?
1803	010106	001453			BEQ	6\$:TYPE MSG
1804	010110	022777	020021	171546	CMP	#20021,@RHDT	:SINGLE PORT RP05 ?
1805	010116	001447			BEQ	6\$:TYPE MSG
1806							
1807	010120	022777	024022	171536	CMP	#24022,@RHDT	:DUAL PORT RP06 ?
1808	010126	001424			BEQ	5\$:TYPE MSG
1809	010130	022777	020022	171526	CMP	#20022,@RHDT	:SINGLE PORT RP06 ?
1810	010136	001420			BEQ	5\$:TYPE MSG
1811	010140	000454			BR	1\$:DRIVE IS NOT AN RP04/RP05/RP06 - SO
1812							:DON'T TYPE THE ASCII MESSAGE
1813							
1814							
1815							:-SHOULD NEVER HAPPEN AT THIS POINT
1816							:UNLESS DRIVE GOT SICK WHILE TESTING
1817	010142			4\$:			:WAS IN PROGRESS
1818	010142	104401	010150		TYPE	,71\$::TYPE ASCIZ STRING
1819	010146	000413			BR	70\$::GET OVER THE ASCIZ
1820	010176	000435			BR	1\$:SKIP NEXT MESSAGE
1821	010200			5\$:			
1822	010200	104401	010206		TYPE	,73\$::TYPE ASCIZ STRING
1823	010204	000413			BR	72\$::GET OVER THE ASCIZ
1824	010234	000416			BR	1\$:SKIP NEXT
1825	010236			6\$:			
1826	010236	104401	010244		TYPE	,75\$::TYPE ASCIZ STRING
1827	010242	000413			BR	74\$::GET OVER THE ASCIZ
1828							
1829	010272	005777	171370	1\$:	TST	@RHSDN	:READ SERIAL NO. AND DRIVE TYPE
1830	010276	005777	171362		TST	@RHDT	:THESE TWO ARE TO HELP SCOPE LOOPS
1831	010302	017737	171360	002012	MOV	@RHSDN,@#SAVSN	:SAVE TO CHECK IF CLR RHCS2 BIT 5 CLEARS ANY BITS
1832	010310	017737	171350	002010	MOV	@RHDT,@#SAVDT	:SAVE TO CHECK IF CLR RHCS2 BIT 5 CLEARS ANY BITS
1833							
1834							


```
1851
1852
1853
1854 010662 000004
1855 010664 012737 000007 002032
1856 010672 005737 002040
1857 010676 001402
1858 010700 000137 010752
1859 010704 004737 042620
1860
1861 010710 012706 001000
1862 010714 012737 000007 002032
1863 010722 013737 001636 010744
1864 010730 013777 001774 170700
1865 010736 004537 042272
1866 010742 020017
1867 010744 176710
1868 010746 104001
1869 010750 000207
1870
1871
1872
1873
1874
1875
1876 010752 000004
1877 010754 012706 001000
1878 010760 012737 000010 002032
1879 010766 013737 001640 011010
1880 010774 013777 001774 170634
1881 011002 004537 042272
1882 011006 001476
1883 011010 176700
1884 011012 104001
1885 011014 000207
1886

.SBTTL REGISTER TESTS

TST7: SCOPE
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
TST @#RH70 ;TEST FLAG FOR RH70 CONTROLLER
BEQ 30$ ;IF FLAG = 1, THIS TEST IS SKIPPED
JMP TST10 ;JUMP TO NEXT TEST -----)
JSR PC,@#CLDISK ;GIVE INITIALIZE

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

TST10: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
MOV @#RHCS1,@#PRCS1+14 ;GET REGISTER ADDRESS
MOV @#UNIT,@#RHCS2 ;MOVE UNIT NO. UNDER TEST
JSR R5,BITST ;TEST BITS IN REGISTER
.WORD 1476 ;ONLY THESE BITS ARE TESTED FOR READ/WRITE
.WORD 176700 ;ADDRESS OF REG. BEING TESTED
ERROR 1 ;INCORRECT DATA RECEIVED
RTS PC ;RETURN TO BLT3 ROUTINE
```

```
1887
1888
1889 011016 000004          TST11: SCOPE
1890
1891 011020 012706 001000    MOV    #STACK,SP          ;RESET STACK
1892 011024 012737 000011 002032  MOV    #TTNO,@#TSTNM     ;THIS SAVES TEST NUMBER
1893 011032 004737 042620    JSR    PC,@#CLDISK       ;INIT AND SET UNIT NUMBER AND DEVICE -
1894                                     ;CPU REG. CORRESPONDENCE (R1-R4)
1895
1896                                     ;*SET FORCED PARITY ERROR 'PAT'
1897 011036 052777 000020 170572  BIS    #PAT,@RHCS2       ;SET 'PAT' TO INVERT PARITY
1898                                     ;GENERATED
1899 011044 005077 170572    CLR    @RHER1            ;WRITE DCL REGISTER USING BAD CONTROL PARITY
1900
1901                                     ;*WITH THIS PARITY ERROR NOTHING WILL BE READ TILL IT IS
1902                                     ;*CLEARED
1903
1904 011050 011137 001126    MOV    @R1,@#$BDDAT      ;RHCS1 ---> $BDDAT
1905 011054 022737 104200 001126  CMP    #SC!DVA!RDY,@#$BDDAT ;COMPARE RHCS1 AFTER PARITY
1906                                     ;ERROR
1907 011062 001406          BEQ    1$                ;BRANCH IF SC!DVA!RDY=1
1908 011064 012737 104200 001124  MOV    #SC!DVA!RDY,@#$GDDAT ;GOOD DATA
1909 011072 010137 042270    MOV    R1,@#REGADR       ;REGISTER ADDRESS RHCS1
1910 011076 104001          ERROR  1                ;SETTING PAT AND
1911                                     ;WRITING DCL REGISTER RHCS1
1912                                     ;DID NOT SET SC!DVA!RDY
1913                                     ;WITH 'PAT' BIT HIGH
1914 011100 013746 001774          1$:  MOV    @#UNIT,-(SP)     ;GET UNIT NUMBER
1915 011104 052716 000120    BIS    #PAT!IR,(SP)      ;INCLUDE PAT AND IR
1916 011110 012637 001124    MOV    (SP)+,@#$GDDAT    ;PUT ON STACK
1917 011114 011237 001126    MOV    @R2,@#$BDDAT      ;RHCS2 ---> $BDDAT
1918 011120 023737 001124 001126  CMP    @#$GDDAT,@#$BDDAT ;COMPARE RHCS2
1919 011126 001403          BEQ    2$                ;OK - SC!DVA!RDY ARE HIGH
1920 011130 010237 042270    MOV    R2,@#REGADR       ;REGISTER ADDRESS
1921 011134 104001          ERROR  1                ;READING DCL REGISTER RHCS2 DID NOT
1922                                     ;SHOW UNIT#!PAT!IR BITS HIGH
1923
1924 011136 011437 001126          2$:  MOV    @R4,@#$BDDAT      ;RHER1 ---> $BDDAT
1925 011142 022737 000010 001126  CMP    #PAR,@#$BDDAT     ;ERROR REGISTER RHER1 SHOULD
1926                                     ;HAVE 'PAR' SET
1927 011150 001406          BEQ    3$                ;A - OK, IT DOES
1928 011152 012737 000010 001124  MOV    #PAR,@#$GDDAT     ;GOOD DATA
1929 011160 010437 042270    MOV    R4,@#REGADR       ;FAILING REGISTER RHER1
1930 011164 104001          ERROR  1                ;PARITY ERROR DID NOT
1931                                     ;SET 'PAR'
1932
1933 011166          3$:
```

```
1934
1935
1936
1937 011166 000004          TST12: SCOPE
1938 011170 012706 001000  MOV      #STACK,SP      ;RESET STACK
1939 011174 012737 000012 002032  MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
1940 011202 013737 001632 011224  MOV      @#RHWC,@#PRWC+14 ;GET REGISTER ADDRESS
1941 011210 013777 001774 170420  PRWC:  MOV      @#UNIT,@RHCS2 ;MOVE UNIT NO. UNDER TEST
1942 011216 004537 042272  JSR      R5,BITST      ;TEST BITS IN REGISTER
1943 011222 177777          .WORD    177777        ;ONLY THESE BITS ARE TESTED FOR READ/WRITE
1944 011224 176702          .WORD    176702        ;ADDRESS OF REG. BEING TESTED
1945 011226 104001          ERROR    1             ;INCORRECT DATA RECEIVED
1946 011230 000207          RTS      PC            ;RETURN TO BLT3 ROUTINE
1947
1948
1949
1950
1951
1952 011232 000004          TST13: SCOPE
1953 011234 012706 001000  MOV      #STACK,SP      ;RESET STACK
1954 011240 012737 000013 002032  MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
1955 011246 013737 001634 011270  MOV      @#RHBA,@#PRBA+14 ;GET REGISTER ADDRESS
1956 011254 013777 001774 170354  PRBA:  MOV      @#UNIT,@RHCS2 ;MOVE UNIT NO. UNDER TEST
1957 011262 004537 042272  JSR      R5,BITST      ;TEST BITS IN REGISTER
1958 011266 177776          .WORD    177776        ;ONLY THESE BITS ARE TESTED FOR READ/WRITE
1959 011270 176704          .WORD    176704        ;ADDRESS OF REG. BEING TESTED
1960 011272 104001          ERROR    1             ;INCORRECT DATA RECEIVED
1961 011274 000207          RTS      PC            ;RETURN TO BLT3 ROUTINE
1962
1963
```

```
1964  
1965  
1966  
1967 011276 000004  
1968 011300 012706 001000  
1969 011304 012737 000014 002032  
1970 011312 013737 001642 011334  
1971 011320 013777 001774 170310 PRER1:  
1972 011326 004537 042272  
1973 011332 177777  
1974 011334 176714  
1975 011336 104001  
1976 011340 000207  
1977
```

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

```
1978
1979
1980
1981 011342 000004          TST15: SCOPE
1982
1983 011344 012737 000015 002032      MOV    #TTNO,@#TSTNM      ;THIS SAVES TEST NUMBER
1984 011352 004737 042620          JSR    PC,@#CLDISK       ;SET UNIT NUMBER AND INIT
1985 011356 013700 001660          MOV    @#RHMR,R0        ;R0 HAS MAINTENANCE REG. ADR.
1986 011362 012701 000001          MOV    #1,R1            ;R1 HAS DATA
1987 011366 012702 000005          MOV    #5,R2            ;R2 HAS COUNT OF NUMBER OF BITS
1988 011372 012710 000001          1$:   MOV    #DMD,@R0      ;SET DIAGNOSTIC MODE BIT
1989 011376 050110          BIS    R1,@R0           ;SET DATA IN RHMR
1990 011400 010146          MOV    R1,-(SP)         ;SAVE DATA FOR COMPARES
1991 011402 052716 000401          BIS    #DMD!400,(SP)    ;INCLUDE BIT 0
1992 011406 011637 001124          MOV    (SP),@#$GDDAT    ;SAVE FOR ERROR PRINTOUT
1993 011412 022610          CMP    (SP)+,@R0        ;COMPARE DATA
1994 011414 001405          BEQ    2$               ;BRANCH IF GOOD
1995 011416 011037 001126          MOV    @R0,$BDDAT       ;BAD DATA
1996 011422 010037 042270          MOV    R0,@#REGADR     ;FAILING REG. ADR.
1997 011426 104001          ERROR 1                 ;MAINTENANCE REGISTER
1998                                ;FAILED TO SET INDICATED
1999                                ;BITS
2000 011430 000241          2$:   CLC                ;CLEAR CARRY
2001 011432 006101          ROL    R1                ;GET NEXT DATA
2002 011434 052701 000400          BIS    #400,R1           ;SET UNUSED BITS
2003 011440 042701 001000          BIC    #BIT09,R1        ;CLEAR READ ONLY BIT
2004 011444 005302          DEC    R2                ;COUNT
2005 011446 001351          BNE    1$               ;BRANCH IF 5 BITS NOT DONE
2006
2007
2008                                ;*NOW FLOAT A 0
2009
2010 011450 012701 000435          MOV    #435,R1          ;R1 HAS DATA
2011 011454 012702 000005          MOV    #5,R2            ;R2 HAS COUNT BITS
2012 011460 012710 000001          3$:   MOV    #DMD,@R0      ;SET DIAGNOSTIC MODE BITS
2013 011464 050110          BIS    R1,@R0           ;SET DATA IN RHMR
2014 011466 020110          CMP    R1,@R0           ;COMPARE DATA
2015 011470 001407          BEQ    4$               ;BRANCH IF GOOD
2016 011472 010137 001124          MOV    R1,@#$GDDAT     ;GOOD DATA
2017 011476 011037 001126          MOV    @R0,@#$BDDAT    ;BAD DATA
2018 011502 010037 042270          MOV    R0,@#REGADR     ;FAILING REG. ADR. RHMR
2019 011506 104001          ERROR 1                 ;MAINTENANCE REGISTER
2020                                ;DOES NOT ALLOW WRITING
2021                                ;ZEROS
2022 011510 000261          4$:   SEC                ;SET CARRY
2023 011512 006101          ROL    R1                ;GET NEXT DATA
2024 011514 042701 001340          BIC    #BIT05!BIT06!BIT07!BIT09,R1 ;CLEAR READ ONLY BIT
2025 011520 052701 000400          BIS    #BIT08,R1        ;SET BIT ZEROED BY ROL
2026 011524 005302          DEC    R2                ;COUNT IF 5 BITS DONE
2027 011526 001354          BNE    3$               ;BRANCH IF INCOMPLETE
2028
2029
```

2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063

011530 000004
011532 012706 001000
011536 012737 000016 002032
011544 013737 001644 011566
011552 013777 001774 170056
011560 004537 042272
011564 017437
011566 176706
011570 104001
011572 000207

TST16: SCOPE

PRDST:

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

011574 000004
011576 012706 001000
011602 012737 000017 002032
011610 013737 001646 011632
011616 013777 001774 170012
011624 004537 042272
011630 177777
011632 176740
011634 104001
011636 000207

TST17: SCOPE

PRER2:

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



```
2116  
2117  
2118 012014 000004      TST23: SCOPE  
2119 012016 012706 001000      MOV   #STACK,SP      ;RESET STACK  
2120 012022 012737 000023 002032      MOV   #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER  
2121  
2122 012030 004737 042620      JSR   PC,@#CLDISK    ;HERE IT IS USED TO SETUP HARDWARE/  
2123                                ;CPU REGISTER CORRESPONDENCE  
2124                                ;R1=RHCS1  
2125                                ;R2=RHCS2  
2126                                ;R3=RHDS1  
2127                                ;R4=RHER1  
2128 012034 005037 002006      CLR   @#ERFLG$       ;CLEAR ERROR FLAG  
2129  
2130                                ;*SIMULATED DISK AREA WILL BE USED AS A TEMPORARY  
2131                                ;*STORAGE TABLE FOR DRIVES PRESENT DETERMINED FROM 'NED' = 0 IN RHCS2  
2132  
2133 012040 012701 051422      MOV   #DISK,R1        ;LOAD TABLE POINTER  
2134 012044 012700 000010      MOV   #8.,R0          ;LOAD TABLE LOCATION COUNTER  
2135 012050 012721 177777      1$:  MOV   #-1,(R1)+    ;FILL 8 LOCATIONS WITH -1  
2136 012054 005300      DEC   R0               ;COUNT DOWN ONE LOCATION  
2137 012056 001374      BNE   1$               ;BRANCH IF 8 NOT DONE  
2138  
2139 012060 005012      CLR   @R2              ;SELECT UNIT NO.0 (U2!U1!U0=0)  
2140 012062 012700 000010      MOV   #8.,R0          ;RELOAD TABLE LOCATION COUNTER  
2141 012066 012701 051422      MOV   #DISK,R1        ;RELOAD THE TABLE POINTER  
2142 012072 005714      2$:  TST   @R4             ;READ A DRIVE REGISTER (RHER1)  
2143 012074 032712 010000      BIT   #NED,@R2        ;NON EXISTENT DRIVE BIT = 0 ?  
2144 012100 001415      BEQ   3$               ;YES...DRIVE PRESENT, CHECK THE TYPE  
2145 012102 005300      7$:  DEC   R0               ;NO...DECREMENT DRIVE COUNT  
2146 012104 001454      BEQ   4$               ;CHECK RESULTS IF 8 DRIVES DONE  
2147  
2148 012106 011246      10$: MOV   @R2,-(SP)       ;PUT RHCS2 ON THE STACK  
2149 012110 042716 177770      BIC   #^C7,(SP)      ;MASK ALL BUT THE UNIT NUMBER  
2150 012114 005216      INC   (SP)             ;INCREMENT THE UNIT NUMBER  
2151 012116 013703 001640      MOV   @#RHCS1,R3      ;GET RHCS1 ADDRESS  
2152 012122 005203      INC   R3               ;ADDRESS UPPER BYTE OF RHCS1  
2153 012124 112713 000100      MOVB  #100,@R3        ;SET 'TRE' IN RHCS1  
2154                                ;WITHOUT ADDRESSING DRIVE  
2155 012130 012612      MOV   (SP)+,@R2       ;RHCS2 HAS THE INCREMENTED UNIT  
2156                                ;WITH 'NED' CLEARED  
2157 012132 000757      BR    2$               ;TEST FOR NEXT DRIVE  
2158  
2159  
2160  
2161                                ;*CHECK THE UNIT TYPE AND BUILD 'NED' DERIVED UNITS TABLE  
2162  
2163 012134 022777 024020 167522 3$:  CMP   #24020,@RHDT    ;IS THIS A DUAL PORT RP04 ?  
2164 012142 001425      BEQ   8$               ;ENTER IN TABLE IF SO  
2165 012144 022777 020020 167512      CMP   #20020,@RHDT    ;IS THIS A SINGLE PORT RP04 ?  
2166 012152 001421      BEQ   8$               ;ENTER IN TABLE IF SO  
2167  
2168 012154 022777 024022 167502      CMP   #24022,@RHDT    ;IS THIS A DUAL PORT RP06 ?  
2169 012162 001415      BEQ   8$               ;ENTER IN TABLE IF SO  
2170 012164 022777 020022 167472      CMP   #20022,@RHDT    ;IS THIS A SINGLE PORT RP06 ?  
2171 012172 001411      BEQ   8$               ;ENTER IN TABLE IF SO
```

```
2172
2173 012174 022777 024021 167462      CMP      #24021,@RHDT      ;IS THIS A DUAL PORT RP05 ?
2174 012202 001405                      BEQ      8$           ;ENTER IN TABLE IF SO
2175 012204 022777 020021 167452      CMP      #20021,@RHDT ;IS THIS A SINGLE PORT RP05 ?
2176 012212 001401                      BEQ      8$           ;ENTER IN TABLE IF SO
2177
2178 012214 000732                      BR       7$           ;NO RP04 FOUND SO CHECK NEXT UNIT
2179
2180 012216 012746 000010      8$:      MOV      #8.,-(SP)      ;LOAD MAX NO. OF DRIVES
2181 012222 160016                      SUB      R0,(SP)      ;(SP) NOW HAS THE PRESENT DRIVE NO.
2182 012224 012621                      MOV      (SP)+,(R1)+  ;LOAD TABLE, INCR TABLE LOCATION &
2183                                     ;RESTORE THE STACK TO WHERE IT WAS
2184 012226 005300                      DEC      R0           ;DECREMENT THE DRIVE COUNT
2185 012230 001402                      BEQ      4$           ;CHECK RESULTS IF 8 UNITS CHECKED
2186 012232 005212                      INC      @R2          ;SELECT NEXT UNIT
2187 012234 000716                      BR       2$           ;GO TEST IT
2188
2189
2190                                     ;*COMPARE 'NED' DERIVED UNITS TABLE WITH THAT DERIVED USING RHAS IN T4
2191
2192 012236 004037 043514      4$:      JSR      R0,@#COMPAR ;COMPARE RESULTS
2193 012242 001754                      UNITS                      ;PHER1/RHAS DERIVED DATA
2194 012244 051422                      DISK                      ;'NED' TEST DATA
2195 012246 000010                      8.                        ;NO.OF WORDS TO COMPARE
2196 012250 012256                      5$                        ;RETURN FOR ERROR HEADER
2197 012252 012304                      6$                        ;RETURN FOR ERROR DATA
2198 012254 012422                      13$                       ;RETURN FOR GOOD COMPARISON (NEXT TEST)
2199
2200
2201
2202                                     ;*SPECIAL 'NED'/'RHAS' TABLE TYPE OUT ROUTINE (BYPASSES .SERRTYP AND
2203                                     ;*HENCE IGNORES INHIBIT ERROR TYPEOUT SWITCH)
2204
2205 012256 104022      5$:      ERROR    22
2206 012260 012703      MOV      #8.,R3           ;LENGTH OF BOTH UNIT TABLES
2207 012264 012701      MOV      #UNITS,R1        ;ADDRESS OF RHAS/RHER1 UNITS TABLE
2208 012270 012702      MOV      #DISK,R2        ;ADDRESS OF 'NED' RHCS2 UNITS TABLE
2209 012274 012137      14$:     MOV      (R1)+,@#$GDDAT ;LOAD RHAS UNIT NO.INTO '$GDDAT' AND
2210                                     ;INCREMENT THE TABLE LOCATION
2211 012300 012237      MOV      (R2)+,@#$BDDAT ;LOAD 'NED' UNIT NO. INTO '$BDDAT'
2212                                     ;& INCR TABLE LOCATION
2213
2214 012304 032777 020000 166626 6$:      BIT      #SW13,@SWR      ;INHIBIT ERROR TYPE OUTS ?
2215 012312 001043      BNE     13$           ;YES...EXIT
2216 012314 022737 177777 001124      CMP      #-1,@#$GDDAT   ;DOES RHAS UNIT TABLE LOCATION = -1 ?
2217 012322 001413      BEQ     11$           ;YES...DON'T TYPE IT - CHECK 'NED' TABLE
2218 012324 104401 062722      TYPE    ,SPACE8        ;NO...TAB OVER PC COLUMN
2219 012330 104401 062722      TYPE    ,SPACE8        ;TAB OVER THE TEST NO. COLUMN
2220 012334 013746 001124      MOV     @#$GDDAT,-(SP) ;SAVE @#$GDDAT FOR TYPEOUT
2221 012340 104403      TYPOS   ;GO TYPE--OCTAL ASCII
2222 012342 006          .BYTE   6              ;TYPE 6 DIGITS
2223 012343 000          .BYTE   0              ;SUPPRESS LEADING ZEROS
2224 012344 104401 062730      TYPE    ,SPACE2        ;SPACE OVER TO THE NEXT COLUMN
2225 012350 000406      BR      12$           ;CHECK THE 'NED' UNIT TABLE
2226
2227 012352 104401 062722      11$:    TYPE    ,SPACE8        ;TAB OVER THE PC COMUMN
```



```
2323 012666 022737 000000 001126      CMP    #0,@#$BDDAT      ;COMPARE DATA
2324 012674 001402                BEQ    5$                ;BRANCH IF GOOD
2325 012676 004737 013434          JSR    PC,@#ERCS2C      ;JUMP TO ERROR FOR CLR (BIT 5)
2326
2327                                ;*CONTROL AND STATUS 2 REGISTER
2328
2329 012702 012746 000100          5$:  MOV    #100,-(SP)       ;INCLUDE IR
2330 012706 053716 001774          BIS    @#UNIT,(SP)     ;SET UNIT NO.
2331 012712 012637 001124          MOV    (SP)+,@#$GDDAT  ;GOOD DATA FOR TYPE OUT
2332 012716 013037 001126          MOV    @ (R0)+,@#$BDDAT ;TEST DATA
2333 012722 023737 001124 001126    CMP    @#$GDDAT,@#$BDDAT ;COMPARE DATA
2334 012730 001402                BEQ    6$                ;BRANCH IF GOOD
2335 012732 004737 013434          JSR    PC,@#ERCS2C      ;JUMP TO ERROR FOR CLR (BIT 5)
2336
2337                                ;*CONTROL AND STATUS 1 REGISTER
2338
2339 012736 012737 004276 001124 6$:  MOV    #4276,@#$GDDAT  ;GOOD DATA FOR ERROR TYPEOUT
2340 012744 013037 001126          MOV    @ (R0)+,@#$BDDAT ;TEST DATA
2341 012750 022737 004276 001126    CMP    #4276,@#$BDDAT  ;COMPARE DATA
2342 012756 001402                BEQ    7$                ;BRANCH IF GOOD
2343 012760 004737 013434          JSR    PC,@#ERCS2C      ;JUMP TO ERROR FOR CLR (BIT 5)
2344
2345                                ;*ERROR 1 REGISTER
2346
2347 012764 012737 000000 001124 7$:  MOV    #0,@#$GDDAT     ;GOOD DATA FOR ERROR TYPEOUT
2348 012772 013037 001126          MOV    @ (R0)+,@#$BDDAT ;TEST DATA
2349 012776 022737 000000 001126    CMP    #0,@#$BDDAT     ;COMPARE DATA
2350 013004 001402                BEQ    10$               ;BRANCH IF GOOD
2351 013006 004737 013434          JSR    PC,@#ERCS2C      ;JUMP TO ERROR FOR CLR (BIT 5)
2352
2353                                ;*DESIRED SECTOR/TRACK REGISTER
2354
2355 013012 012737 017437 001124 10$:  MOV    #17437,@#$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
2356 013020 013037 001126          MOV    @ (R0)+,@#$BDDAT ;TEST DATA
2357 013024 022737 017437 001126    CMP    #17437,@#$BDDAT ;COMPARE DATA
2358 013032 001402                BEQ    11$               ;BRANCH IF GOOD
2359 013034 004737 013434          JSR    PC,@#ERCS2C      ;JUMP TO ERROR FOR CLR (BIT 5)
2360
2361                                ;*ERROR 2 REGISTER
2362
2363 013040 012737 000000 001124 11$:  MOV    #0,@#$GDDAT     ;GOOD DATA FOR ERROR TYPEOUT
2364 013046 013037 001126          MOV    @ (R0)+,@#$BDDAT ;TEST DATA
2365 013052 022737 000000 001126    CMP    #0,@#$BDDAT     ;COMPARE DATA
2366 013060 001402                BEQ    12$               ;BRANCH IF GOOD
2367 013062 004737 013434          JSR    PC,@#ERCS2C      ;JUMP TO ERROR FOR CLR (BIT 5)
2368
2369                                ;*OFFSET REGISTER
2370
2371 013066 012737 116000 001124 12$:  MOV    #116000,@#$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
2372 013074 013037 001126          MOV    @ (R0)+,@#$BDDAT ;TEST DATA
2373 013100 022737 116000 001126    CMP    #116000,@#$BDDAT ;COMPARE DATA
2374 013106 001402                BEQ    13$               ;BRANCH IF GOOD
2375 013110 004737 013434          JSR    PC,@#ERCS2C      ;JUMP TO ERROR FOR CLR (BIT 5)
2376
2377                                ;*DESIRED CYLINDER ADDRESS REGISTER
2378
```

```

2379 013114 012737 001777 001124 13$: MOV #1777,@#$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
2380 013122 013037 001126 001126 MOV @ (R0)+,@#$BDDAT ;TEST DATA
2381 013126 022737 001777 001126 CMP #1777,@#$BDDAT ;COMPARE DATA
2382 013134 001402 BEQ 14$ ;BRANCH IF GOOD
2383 013135 004737 013434 JSR PC,@#ERCS2C ;JUMP TO ERROR FOR CLR (BIT 5)
2384
2385 ;*ERROR 3 REGISTER
2386
2387 013142 012737 000000 001124 14$: MOV #0,@#$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
2388 013150 013037 001126 001126 MOV @ (R0)+,@#$BDDAT ;TEST DATA
2389 013154 022737 000000 001126 CMP #0,@#$BDDAT ;COMPARE DATA
2390 013162 001402 BEQ 15$ ;BRANCH IF GOOD
2391 013164 004737 013434 JSR PC,@#ERCS2C ;JUMP TO ERROR FOR CLR (BIT 5)
2392
2393 ;*ATTENTION SUMMARY REGISTER
2394
2395 013170 013037 001126 001124 15$: MOV @ (R0)+,@#$BDDAT ;GET RHAS CONTENTS
2396 013174 012737 000000 001124 MOV #0,@#$GDDAT ;GOOD DATA FOR ERROR TYPE OUT
2397 013202 123737 001124 001126 CMPB @#$GDDAT,@#$BDDAT ;COMPARE FOR RHAS
2398 013210 001402 BEQ 16$ ;BRANCH IF GOOD
2399 013212 004737 013434 JSR PC,@#ERCS2C ;JUMP TO ERROR FOR CLR (BIT 5)
2400
2401 ;IN RHCS2
2402
2403 ;*MAINTAINABILITY REGISTER
2404
2405 013216 012737 000400 001124 16$: MOV #400,@#$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
2406 013224 013037 001126 001126 MOV @ (R0)+,@#$BDDAT ;TEST DATA
2407 013230 022737 000400 001126 CMP #400,@#$BDDAT ;COMPARE DATA
2408 013236 001402 BEQ 17$ ;BRANCH IF GOOD
2409 013240 004737 013434 JSR PC,@#ERCS2C ;JUMP TO ERROR FOR CLR (BIT 5)
2410
2411 ;*DRIVE STATUS REGISTER
2412
2413 013244 012737 000600 001124 17$: MOV #600,@#$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
2414 013252 013046 001126 001126 MOV @ (R0)+,-(SP) ;GET RHDS1
2415 013254 011637 001126 001126 MOV (SP),@#$BDDAT ;TEST DATA
2416 013260 042716 001100 BIC #VV!PROG,(SP) ;CLEAR VV AND PROG
2417 013264 022726 000600 CMP #600,(SP)+ ;COMPARE DATA
2418 013270 001402 BEQ 20$ ;BRANCH IF GOOD
2419 013272 004737 013434 JSR PC,@#ERCS2C ;JUMP TO ERROR FOR CLR (BIT 5)
2420
2421 ;IN RHCS2
2422
2423 ;*DRIVE TYPE
2424
2425 013276 013737 002010 001124 20$: MOV @#SAVDT,@#$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
2426 013304 013037 001126 001126 MOV @ (R0)+,@#$BDDAT ;TEST DATA
2427 013310 023737 002010 001126 CMP @#SAVDT,@#$BDDAT ;COMPARE DATA
2428 013316 001402 BEQ 21$ ;BRANCH IF GOOD
2429 013320 004737 013434 JSR PC,@#ERCS2C ;JUMP TO ERROR FOR CLR (BIT 5)
2430
2431 ;*SERIAL NUMBER REGISTER
2432
2433 013324 013737 002012 001124 21$: MOV @#SAVSN,@#$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
2434 013332 013037 001126 001126 MOV @ (R0)+,@#$BDDAT ;TEST DATA
2435 013336 023737 002012 001126 CMP @#SAVSN,@#$BDDAT ;COMPARE DATA
  
```



```
2435 013344 001402          BEQ    22$          ;BRANCH IF GOOD
2436 013346 004737 013434   JSR    PC,@#ERCS2C ;JUMP TO ERROR FOR CLR (BIT 5)
2437
2438
2439
2440 013352 012737 000000 001124 22$:   MOV    #0,@#$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
2441 013360 013037 001126          MOV    @ (R0)+,@#$BDDAT ;TEST DATA
2442 013364 022737 000000 001126   CMP    #0,@#$BDDAT ;COMPARE DATA
2443 013372 001402          BEQ    23$          ;BRANCH IF GOOD
2444 013374 004737 013434   JSR    PC,@#ERCS2C ;JUMP TO ERROR FOR CLR (BIT 5)
2445
2446
2447
2448
2449 013400 012737 000000 001124 23$:   MOV    #0,@#$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
2450 013406 013037 001126          MOV    @ (R0)+,@#$BDDAT ;TEST DATA
2451 013412 022737 000000 001126   CMP    #0,@#$BDDAT ;COMPARE DATA
2452 013420 001402          BEQ    24$          ;BRANCH IF GOOD
2453 013422 004737 013434   JSR    PC,@#ERCS2C ;JUMP TO ERROR FOR CLR (BIT 5)
2454
2455
2456
2457
2458 013426 005720          24$:   TST    (R0)+          ;AS THE LOOK-AHEAD REG. CANNOT BE PREDICTED
2459
2460
2461
2462
2463 013430 005720          25$:   TST    (R0)+          ;AS THE CURRENT REG. CANNOT BE PREDICTED
2464
2465
2466 013432          26$:
2467 013432 000405          BR     TST25          ;BRANCH OVER JSR
2468
2469
2470 013434 014037 042270   ERCS2C: MOV    -(R0), @#REGADR ;FAILING REGISTER ADDRESS
2471 013440 104001          ERROR 1              ;CLR (BIT 5) IN RHCS2 DID
2472
2473
2474 013442 005720          TST    (R0)+          ;NOT CLEAR APPROPRIATE BITS
2475 013444 000207          RTS    PC              ;OR CLEARED EXTRA BITS
                          ;UNDO -(R0) FOR BAD DATA
                          ;RETURN TO TEST ABOVE
```



```
2535
2536
2537
2538 013610 000004          TST26: SCOPE
2539 013612 012706 001000  MOV      #STACK,SP          ;RESET STACK
2540 013616 012737 000026 002032  MOV      #TTNO,@#TSTNM      ;THIS SAVES TEST NUMBER
2541 013624 004737 042620  JSR      PC,@#CLDISK        ;INIT AND SET UP GENERAL CPU/DEVICE
2542                                     ;REGISTER CORRESPONDENCE
2543
2544                                     ;*FILL ALL POSSIBLE REGISTER BITS WITH ONES
2545
2546 013630 012777 177777 165774  MOV      #177777,@RHW        ;WORD COUNT REGISTER GETS 177777
2547 013636 012777 177777 165770  MOV      #177777,@RHBA       ;BUS ADDRESS REGISTER GETS 177777
2548 013644 052777 157010 165764  BIS      #157010,@RHCS2      ;CONTROL AND STATUS 2 GETS 177430
2549 013652 012777 001476 165760  MOV      #1476,@RHCS1        ;CONTROL AND STATUS REGISTER 1 GETS 21476
2550 013660 012777 177777 165754  MOV      #177777,@RHER1      ;ERROR REGISTER1 GETS 177777
2551 013666 012777 017437 165750  MOV      #17437,@RHDST       ;DESIRED SECTOR TRACK
2552 013674 012777 177777 165744  MOV      #177777,@RHER2      ;ERROR REGISTER 2
2553 013702 012777 016277 165740  MOV      #16277,@RHOF        ;OFFSET REGISTER
2554 013710 012777 000777 165734  MOV      #777,@RHCA          ;DESIRED CYLINDER
2555 013716 012777 177777 165730  MOV      #177777,@RHER3      ;ERROR REGISTER 3
2556 013724 012777 000001 165726  MOV      #DMD,@RHMR          ;MAINTENANCE REGISTER
2557 013732 012777 177777 165720  MOV      #177777,@RHMR       ;MAINTENANCE REGISTER
2558
2559                                     ;*BEFORE RESET SAVE REGISTERS IN READ INTO BUFFER
2560 013740 004037 043312  JSR      RO,@#SAVER          ;SAVE
2561 013744 001632          RHWC          ;FROM
2562 013746 003154          REINTO        ;TO
2563 013750 000021          17.          ;NUMBER
2564
2565                                     ;*GIVE RESET AND REINSTATE UNIT NUMBER
2566 013752 000005          RESET
2567 013754 004737 055176  JSR      PC,@#$TKINT         ;INITIALIZE TK
2568 013760 053777 001774 165650  BIS      @#UNIT,@RHCS2
2569
2570                                     ;*CHANGE ORIGINAL SAVED REGISTERS TO EXPECTED VALUES AFTER RESET
2571 013766 005037 003156  CLR      @#REINTO+2          ;CLEAR SAVED RHBA
2572 013772 013746 001774  MOV      @#UNIT,-(SP)        ;GET UNIT NUMBER FRO SAVED RHCS2
2573 013776 052716 000100  BIS      #IR,(SP)           ;INCLUDE IR
2574 014002 012637 003160  MOV      (SP)+,@#REINTO+4    ;SAVED RHCS2
2575 014006 012737 004276 003162  MOV      #DVA!RDY!76,@#REINTO+6 ;SAVED RHCS1
2576 014014 005037 003164  CLR      @#REINTO+10         ;SAVED RHER1
2577 014020 005037 003170  CLR      @#REINTO+14         ;SAVED RHER2
2578 014024 012737 116000 003172  MOV      #116000,@#REINTO+16 ;SAVED RHOF
2579 014032 005037 003176  CLR      @#REINTO+22         ;SAVED RHER3
2580 014036 105037 003200  CLR      @#REINTO+24         ;SAVED RHAS
2581 014042 012737 000400 003202  MOV      #400,@#REINTO+26 ;SAVED RHMR
2582
2583                                     ;*CHANGE RHDS1 WITHOUT CHANGING PROG BIT
2584 014050 013746 003204  MOV      @#REINTO+30,-(SP)   ;GET RHDS1
2585 014054 042716 176777  BIC      #^CPRG,(SP)        ;CLEAR EVERYTHING EXCEPT PROG
2586 014060 052716 000700  BIS      #700,(SP)          ;SET EXPECTED BITS - 'DPR', 'DRY' & 'VV'
2587
2588 014064 012637 003204          4$: MOV      (SP)+,@#REINTO+30 ;SAVED RHDS1
2589 014070 005037 003212  CLR      @#REINTO+36         ;SAVED RHEC1
2590 014074 005037 003214  CLR      @#REINTO+40         ;SAVED RHEC2
```

```
2591
2592
2593 014100 004037 043312      ;*AFTER RESET, SAVE REGISTERS FOR COMPARISONS TO BE DONE
2594 014104 001632      JSR      R0,@#SAVER      ;SAVE
2595 014106 002110      RHW C      ;FROM
2596 014110 000021      WRFROM      ;TO
2597                          17.      ;NUMBER
2598
2599      ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
2600      ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
2601 014112 113737 003201 002135 ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
2602      MOV B @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS
2603
2604 014120 004037 043514      ;*COMPARE REGISTERS BEFORE RESET WITH REGISTERS AFTER RESET
2605 014124 003154      JSR      R0,@#COMPAR      ;COMPARE
2606 014126 002110      REINTO      ;GOOD BUFFER
2607 014130 000021      WRFROM      ;TEST BUFFER
2608 014132 014140      17.      ;NUMBER
2609 014134 014140      1$      ;RETURN FOR ERROR
2610 014136 014160      1$      ;SAME
2611                          2$      ;RETURN FOR GOOD COMPARISON
2612 014140 013705 047624      1$:      MOV @#ERWORD,R5      ;GETTING READY TO INDEX
2613 014144 060505      ADD R5,R5      ;DOUBLE ERROR WORD
2614 014146 016537 001630 042270 ;MOV RHW C-2(R5),@#REGADR ;FAILING REGISTER ADDRESS
2615 014154 104001      ERROR 1      ;REGISTER CONTENTS AFTER
2616                          ;A RESET THAT IS AN
2617                          ;UNIBUS INITIALIZE CAUSED
2618                          ;AN IMPROPER REGISTER CHANGE
2619 014156 000207      RTS PC      ;RETURN TO COMPARISON
2620 014160      2$:      ;RETURN TO POINT ON GOOD COMPARISON
```



```
2723
2724
2725 014564 000004
2726 014566 012737 000031 002032 TST31: SCOPE
2727 014574 005737 002040      MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
2728 014600 001402      TST @#RH70 ;TEST FLAG FOR RH70 CONTROLLER
2729 014602 000137 015024      BEQ 30$ ;IF FLAG = 1, THIS TEST IS SKIPPED
2730      JMP TST32 ;JUMP TO NEXT TEST -----)
2731 014606 012700 051422      MOV #SILOTB,R0 ;TABLE POINTER
2732 014612 012705 000103      MOV #67.,R5 ;COUNTER
2733 014616 005020 1$: CLR (R0)+ ;CLEAR TOTAL TABLE
2734 014620 005305      DEC R5 ;COUNT
2735 014622 001375      BNE 1$ ;BRANCH IF NOT COMPLETELY CLEAR
2736 014624 004737 042620      JSR PC,@#CLDISK ;CLEAR ALL REG.
2737 014630 005000      CLR R0
2738 014632 012705 000102      MOV #66.,R5 ;COUNT
2739 014636 010077 164766 2$: MOV R0,@RHDB ;LOAD SILO WITH COUNT FROM 0 TO 65
2740 014642 005200      INC R0 ;NEXT COUNT
2741 014644 005305      DEC R5 ;IS 66 LOADS DONE?
2742 014646 001373      BNE 2$ ;BRANCH IF NOT.
2743 014650 013746 001774      MOV @#UNIT,-(SP)
2744 014654 052716 000200      BIS #OR,(SP)
2745 014660 004737 042224      JSR PC,@#PUTREG ;SAVE REGISTERS
2746 014664 022637 001712      CMP (SP)+,@#CS2 ;'DR' SHOULD BE SET IR RESET
2747 014670 001405      BEQ 3$ ;BRANCH IF YES
2748 014672 010237 001122      MOV R2,@#$BDADR ;SAVE RHCS2 ADR. FAILING REG.
2749 014676 104011      ERROR 11 ;'DR' WAS NOT SET, IR WAS NOT
2750 014700 005037 002006      CLR @#ERFLG$ ;RESET AFTER SILO WAS FULL
2751 014704 012700 051422 3$: MOV #SILOTB,R0 ;POINTER
2752 014710 012705 000102      MOV #66.,R5 ;COUNTER
2753 014714 017720 164710 4$: MOV @RHDB,(R0)+ ;READ SILO
2754 014720 005305      DEC R5 ;COUNT
2755 014722 001374      BNE 4$ ;BRANCH IF 66 NOT DONE
2756 014724 012700 051422      MOV #SILOTB,R0 ;POINTER
2757 014730 012705 000102      MOV #66.,R5
2758 014734 005046      CLR -(SP)
2759 014736 021620 5$: CMP (SP),(R0)+
2760 014740 001425      BEQ 7$ ;BRANCH IF GOOD
2761 014742 014037 001126      MOV -(R0),@#$BDDAT ;BAD DATA
2762 014746 011637 001124      MOV (SP),@#$GDDAT ;GOOD DATA
2763 014752 013737 001630 042270      MOV @#RHDB,@#REGADR ;FAILING REG. RHDB
2764 014760 005737 002006      TST @#ERFLG$ ;IS THIS FIRST ERROR?
2765 014764 001002      BNE 6$ ;IF NOT BRANCH
2766 014766 104012      ERROR 12 ;THESE TWO ERROR CALLS ARE FOR
2767 014770 000401      BR 64$ ;BRANCH TO AVOID PRINTING NEXT ERROR
2768 014772 104013 6$: ERROR 13 ;THE SAME TYPEOUT. SILO
2769 ;HAD A COUNT WRITTEN IN.
2770 ;ON READ OUT AN ERROR WAS
2771 ;DETECTED. THE TOTAL SILO
2772 ;READOUT IS IN LOCATION
2773 ;'SILOTB' TO THE NEXT 65
2774 ;WORDS.
2775 014774 005720 64$: TST (R0)+ ;INCREMENT (R0)
2776 ;ARE FURTHER COMPARES TO
2777 014776 017746 164136      MOV @SWR,-(SP) ;BE DONE
2778 015002 042716 177577      BIC #^CSW07!SW08,(SP) ;ONLY KEEP SW7 AND SW8
```


2870	015374	001407		BEQ	6\$:BRANCH IF GOOD
2871	015376	010037	042270	MOV	R0,@#REGADR		:SILO ADDRESS
2872	015402	005037	001124	CLR	@#\$GDDAT		:GOOD DATA
2873	015406	010337	001126	MOV	R3,@#\$BDDAT		:BAD DATA
2874	015412	104001		ERROR	1		:SILO SHOULD BE ZERO
2875							:AFTER THE ONE WORD PUT IN
2876							:HAS BEEN TAKEN OUT AS
2877							:SILO IS A DESTRUCTIVE READ
2878	015414	032704	100000	6\$: BIT	#DLT,R4		:
2879	015420	001013		BNE	TST34		:BRANCH IF DLT SET
2880	015422	013746	001774	MOV	@#UNIT,-(SP)		:GET UNIT NO
2881	015426	052716	100300	BIS	#DLT!OR!IR,(SP)		:
2882	015432	012637	001124	MOV	(SP)+,@#\$GDDAT		:GOOD DATA
2883	015436	010437	001126	MOV	R4,@#\$BDDAT		:BAD DATA
2884	015442	010237	042270	MOV	R2,@#REGADR		:RHCS2 ADDRESS
2885	015446	104001		ERROR	1		:DATA LATE ERROR

```
2886
2887
2888           .SBTTL  MORE REGISTER TESTS
2889
2890
2891 015450 000004           TST34:  SCOPE
2892
2893 015452 012737 000034 002032      MOV  #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
2894 015460 005737 002040           TST  @#RH70 ;TEST FLAG FOR RH70 CONTROLLER
2895 015464 001402           BEQ  30$ ;IF FLAG = 1, THIS TEST IS SKIPPED
2896 015466 000137 015606           JMP  TST35 ;JUMP TO NEXT TEST -----)
2897 015472 012706 001000           MOV  #STACK,SP ;RESET STACK
2898 015476 004737 042620           JSR  PC,CLDISK ;CLEAR DISK REG.
2899
2900 015502 012711 003566           MOV  #3566,@R1 ;LOAD RHCS1 WITH ANY NUMBER
2901 015506 010146           MOV  R1,-(SP) ;GETTING READY TO FORM ODD BYTE
2902 015510 005216           INC  (SP) ;SP NOW HAS ODD BYTE FOR RHCS1
2903 015512 112736 000005           MOVB #5,@(SP)+ ;MOVE 5 INTO ODD BYTE FOR RHCS1
2904 015516 011137 001126           MOV  @R1,@#$BDDAT ;TEST DATA
2905 015522 022737 006766 001126      CMP  #2566!DVA!RDY,@#$BDDAT ;RHCS1 SHOULD HAVE 6766
2906 015530 001406           BEQ  1$ ;BRANCH IF GOOD
2907 015532 012737 006766 001124      MOV  #2566!DVA!RDY,@#$GDDAT ;GOOD DATA
2908 015540 010137 042270           MOV  R1,@#REGADR ;FAILING REGISTER RHCS1
2909 015544 104001           ERROR 1 ;MOVING A NUMBER INTO
2910 ;ODD BYTE OF RHCS1 GAVE
2911 ;WRONG RESULTS
2912
2913 015546 112711 000032           1$: MOVB #32,@R1 ;MOVE INTO EVEN BYTE
2914 015552 011137 001126           MOV  @R1,@#$BDDAT ;TEST DATA
2915 015556 022737 006632 001126      CMP  #2432!DVA!RDY,@#$BDDAT ;RHCS1 SHOULD HAVE 6632
2916 015564 001460           BEQ  TST36 ;DO NEXT RH11 TEST IF GOOD -----)
2917 015566 012737 006632 001124      MOV  #2432!DVA!RDY,@#$GDDAT ;GOOD DATA
2918 015574 010137 042270           MOV  R1,@#REGADR ;FAILING REGISTER RHCS1
2919 015600 104001           ERROR 1 ;MOVING A NUMBER INTO EVEN
2920 ;BYTE OF RHCS1 GAVE WRONG
2921 ;RESULT
2922
2923 015602 000137 015726           JMP  TST36 ;SKIP RH70 TEST -----)
2924
```

```
2925
2926
2927 015606 000004          TST35: SCOPE
2928
2929 015610 012737 000035 002032      MOV    #TTNO,@#TSTNM    ;THIS SAVES TEST NUMBER
2930 015616 012706 001000          MOV    #STACK,SP      ;RESET STACK
2931 015622 004737 042620          JSR    PC,CLDISK      ;CLEAR DISK REG.
2932
2933 015626 012711 003566          MOV    #3566,@R1      ;LOAD RHCS1 WITH ANY NUMBER
2934 015632 010146          MOV    R1,-(SP)       ;GETTING READY TO FORM ODD BYTE
2935 015634 005216          INC    (SP)           ;SP NOW HAS ODD BYTE FOR RHCS1
2936 015636 112736 000005          MOVB   #5,@(SP)+      ;MOVE 5 INTO ODD BYTE FOR RHCS1
2937 015642 011137 001126          MOV    @R1,@#$BDDAT   ;TEST DATA
2938
2939 015646 022737 004766 001126      CMP    #566!DVA!RDY,@#$BDDAT ;RHCS1 SHOULD HAVE 4766
2940 015654 001406          BEQ    1$             ;BRANCH IF GOOD
2941 015656 012737 004766 001124      MOV    #566!DVA!RDY,@#$GDDAT ;GOOD DATA
2942 015664 010137 042270          MOV    R1,@#REGADR    ;FAILING REGISTER RHCS1
2943 015670 104001          ERROR  1             ;MOVING A NUMBER INTO
2944                                     ;ODD BYTE OF RHCS1 GAVE
2945                                     ;WRONG RESULTS
2946
2947 015672 112711 000032          1$:  MOVB   #32,@R1      ;MOVE INTO EVEN BYTE
2948 015676 011137 001126          MOV    @R1,@#$BDDAT   ;TEST DATA
2949 015702 022737 004632 001126      CMP    #432!DVA!RDY,@#$BDDAT ;RHCS1 SHOULD HAVE 4632
2950 015710 001406          BEQ    TST36          ;:BRANCH IF GOOD
2951 015712 012737 004632 001124      MOV    #432!DVA!RDY,@#$GDDAT ;GOOD DATA
2952 015720 010137 042270          MOV    R1,@#REGADR    ;FAILING REGISTER RHCS1
2953 015724 104001          ERROR  1             ;MOVING A NUMBER INTO EVEN
2954                                     ;BYTE OF RHCS1 GAVE WRONG
2955                                     ;RESULTS
2956
```

```
2957
2958
2959 015726 000004          TST36: SCOPE
2960
2961 015730 012737 000036 002032      MOV  #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
2962 015736 005737 002040          TST  @#RH70        ;TEST FLAG FOR RH70 CONTROLLER
2963 015742 001402          BEQ  30$           ;IF FLAG = 1, THIS TEST IS SKIPPED
2964 015744 000137 016104          JMP  TST37        ;JUMP TO NEXT TEST -----)
2965 015750 004737 042620          JSR  PC,@#CLDISK  ;GIVE INIT & SETUP REGISTER CORRES
2966
2967 015754 052712 177000          BIS  #177000,(R2)  ;LOAD RHCS2
2968 015760 010246          MOV  R2,-(SP)     ;GETTING READY FOR ODD BYTE
2969 015762 005216          INC  (SP)         ;SP NOW HAS ODD BYTE FOR RHCS2
2970 015764 105036          CLRB @#(SP)+      ;CLERR RHCS2 ODD BYTE
2971 015766 013746 001774          MOV  @#UNIT,-(SP) ;GET UNIT NO.
2972 015772 052716 000100          BIS  #IR,(SP)     ;INPUT READY AS IT IS SET
2973 015776 011237 001126          MOV  @R2,@#$BDDAT ;TEST DATA
2974 016002 022637 001126          CMP  (SP)+,@#$BDDAT ;COMPARE TO SEE THAT
2975                                     ;'CLRB' DID CLEAR
2976 016006 001411          BEQ  1$           ;
2977 016010 013737 001774 001124      MOV  @#UNIT,@#$GDDAT ;
2978 016016 052737 000100 001124      BIS  #IR,@#$GDDAT  ;GOOD DATA
2979 016024 010237 042270          MOV  R2,@#REGADR ;FAILING REGISTER RHCS2
2980 016030 104001          ERROR 1          ;CLEARING ODD BYTE OF RHCS2
2981                                     ;GAVE WRONG RESULTS
2982 016032 013746 001774          1$: MOV  @#UNIT,-(SP) ;
2983 016036 052716 000010          BIS  #BAI,(SP)    ;
2984 016042 052712 020000          BIS  #UPE,@R2     ;HAVE UPE AND MPE IN RHCS2
2985                                     ;BESIDES UNIT SELECT
2986 016046 112612          MOV  (SP)+,@R2    ;MOVE INTO EVEN BYTE OF RHCS2
2987 016050 013746 001774          MOV  @#UNIT,-(SP) ;
2988 016054 052716 020110          BIS  #UPE!IR!BAI,(SP) ;
2989 016060 011637 001124          MOV  (SP),@#$GDDAT ;GOOD DATA
2990 016064 011237 001126          MOV  @R2,@#$BDDAT ;TEST DATA
2991 016070 022637 001126          CMP  (SP)+,@#$BDDAT ;COMPARE TO SEE THAT MOV B DID
2992                                     ;MOVE EVEN BYTE ONLY
2993 016074 001403          BEQ  TST37        ;;BRANCH IF GOOD
2994 016076 010237 042270          MOV  R2,@#REGADR ;FAILING REGISTER RHCS2
2995 016102 104001          ERROR 1          ;MOVING A NUMBER INTO EVEN
2996                                     ;BYTE OF RHCS2 GAVE WRONG
2997                                     ;RESULTS
2998
```

```
2999
3000
3001 016104 000004          TST37: SCOPE
3002
3003 016106 012737 000037 002032      MOV    #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
3004 016114 012706 001000              MOV    #STACK,SP    ;RESET STACK
3005 016120 004737 042620              JSR    PC,CLDISK    ;CLEAR DISK REGISTERS
3006 016124 013704 001632              MOV    @#RHWC,R4    ;R4 NOW IS WORD COUNT REGISTER
3007 016130 012714 025252              MOV    #25252,@R4   ;LOAD RHWC
3008 016134 010446                    MOV    R4,-(SP)     ;GETTING READY TO FORM ODD BYTE
3009 016136 005216                    INC    (SP)         ;SP NOW HAS ODD BYTE FOR RHWC
3010 016140 112736 000377              MOV    #377,@(SP)+ ;MOVE 377 INTO ODD BYTE OF RHWC
3011 016144 011437 001126              MOV    @R4,@#SBDDAT ;TEST DATA
3012 016150 022737 177652 001126      CMP    #177652,@#SBDDAT;COMPARE TO SEE IF MOV B DID OK
3013 016156 001406                    BEQ    1$           ;BRANCH IF GOOD
3014 016160 012737 177652 001124      MOV    #177652,@#SGDDAT ;GOOD DATA
3015 016166 010437 042270              MOV    R4,@#REGADR  ;REGISTER FAILING RHWC
3016 016172 104001                    ERROR  1           ;MOVING INTO ODD BYTE OF RHWC
3017                                ;GAVE WRONG RESULTS
3018 016174 112714 000123          1$:  MOV    #123,@R4    ;MOVE INTO EVEN BYTE OF RHWC
3019 016200 011437 001126              MOV    @R4,@#SBDDAT ;TEST DATA
3020 016204 022737 177523 001126      CMP    #177523,@#SBDDAT
3021 016212 001406                    BEQ    TST40        ;:BRANCH IF GOOD
3022 016214 012737 177523 001124      MOV    #177523,@#SGDDAT ;GOOD DATA
3023 016222 010437 042270              MOV    R4,@#REGADR  ;REGISTER FAILING RHWC
3024 016226 104001                    ERROR  1
```



```

3053
3054          .SBTTL  DCL COMMAND TESTS
3055
3056          **:      FOUR GENERAL REGISTERS WILL BE RESERVED FOR HARDWARE
3057          **:      R1=RHCS1 CONTROL AND STATUS1
3058          **:      R2=RHCS2 CONTROL AND STATUS2
3059          **:      R3=RHDS1 DRIVE STATUS 1
3060          **:      R4=RHER1 ERROR REGISTER1
3061
3062          **:      WHENEVER ANY OTHER USE IS MADE OF THESE REGISTERS
3063          **:      APPROPRIATE SAVING MUST BE DONE
3064
3065
3066
3067
3068
3069
3070
3071          **:      ERROR REGISTER #01 (RHER1) TEST
3072          **:      BIT #1 (ILLEGAL REGISTER) CANNOT BE TESTED ON PDP11 THIS BIT
3073          **:      IS FOR PDP10 USE ONLY
3074
3075
3076
3077 016354 000004          TST41:  SCOPE
3078
3079 016356 012706 001000          MOV      #STACK,SP          ;RESET STACK
3080 016362 012737 000041 002032          MOV      #ITNO,@#TSTNM      ;THIS SAVES TEST NUMBER
3081 016370 004737 042620          JSR      PC,@#CLDISK        ;CLEAR REGISTERS
3082 016374 012777 000001 163256          MOV      #DMD,@RHMR        ;SET DIAGNOSTIC MODE
3083 016402 005037 002022          CLR      @#TMPILL          ;GET READY TO MAKE ILLEGAL FUNCTION
3084 016406 012700 002042          1$:     MOV      #FUTABL,R0    ;LOAD FUNCTION CODE TABLE START
3085 016412 012705 000021          MOV      #17,R5            ;COUNTER (16 GOOD FUNCTIONS)
3086 016416 023720 002022          2$:     CMP      @#TMPILL,(R0)+ ;IS THIS A LEGAL FUNCTION CODE?
3087 016422 001004          BNE      3$                ;NO - DECR. FUNCT. CODE CTR
3088 016424 062737 000002 002022          ADD      #2,@#TMPILL        ;YES MAKE NEXT FUNCTION CODE
3089 016432 000765          BR       1$                ;TEST NEXT FUNCTION CODE
3090 016434 005305          3$:     DEC      R5          ;MAKE NEXT CODE IF 1ST 16
3091                                     ;LEGAL FUNCTIONS NOT DONE
3092                                     ;BRANCH IF 16 NOT COMPLETE
3093 016436 001367          BNE      2$                ;BRANCH IF 16 NOT COMPLETE
3094 016440 032737 000100 002022          BIT      #100,@#TMPILL      ;ALL BITS UP TO BIT #5 COMPARED?
3095 016446 001077          BNE      12$               ;YES - EXIT ----->
3096 016450 013737 002022 002104          MOV      @#TMPILL,@#ILLEGL ;NO - TEST THE ILLEGAL FUNCTION
3097 016456 062737 000002 002022          ADD      #2,@#TMPILL        ;TEST NEW FUNCTION CODE NEXT TIME
3098
3099 016464 004737 042620          4$:     JSR      PC,@#CLDISK        ;SET DIAGNOSTIC MODE
3100 016470 012777 000001 163162          MOV      #DMD,@RHMR        ;ILLEGAL FUNCTION ----> RHCS1
3101 016476 013711 002104          MOV      @#ILLEGL,@R1      ;ERROR RETURN POINT
3102 016502 012737 016464 001110          MOV      #4$,@#SLPERR      ;SAVE REGISTERS
3103 016510 004737 042224          JSR      PC,@#PUTREG        ;THERE SHOULD NOT BE ANY ERROR YET
3104 016514 005737 001716          TST      @#RER1            ;CONTINUE IF RHER1 STILL = 0
3105 016520 001403          BEQ      5$                ;FAILING REGISTER ADDRESS RHER1
3106 016522 010437 001122          MOV      R4,@#$BDADR        ;ALTHOUGH AN ILLEGAL FUNCTION
3107 016526 104011          ERROR   11                 ;HAS BEEN MOVED INTO RHCS1
3108                                     ;NO ERRORS SHOULD SHOW TILL

```



```
3164
3165
3166 016650 000004          TST42: SCOPE
3167 016652 012706 001000  MOV      #STACK,SP      ;RESET STACK
3168 016656 012737 000042 002032  MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
3169
3170 016664 004737 042620      JSR      PC,@#CLDISK    ;INIT AND SET GENERAL REGISTERS
3171
3172                          ;*FILL ALL POSSIBLE BITS WITH ONES
3173
3174 016670 012777 177777 162734  MOV      #177777,@RHWC  ;WORD COUNT REGISTER GETS 177777
3175 016676 012777 177777 162730  MOV      #177777,@RHBA  ;BUS ADDRESS REGISTER GETS 177777
3176 016704 012777 017437 162732  MOV      #17437,@RHDST  ;DESIRED SECTOR TRACK GETS 17437
3177 016712 012777 016377 162730  MOV      #16377,@RHOF   ;OFFSET REGISTER GETS 16277
3178 016720 012777 000777 162724  MOV      #777,@RHCA    ;DESIRED CYLINDER GETS 777
3179 016726 012746 001400      MOV      #A16!A17,-(SP) ;GET BIT 9 AND 8
3180 016732 053716 002102      BIS      @#READIN,(SP)
3181 016736 012677 162676      MOV      (SP)+,@RHCS1   ;FILL READ IN PRESET IN RHCS1
3182 016742 012777 000001 162710  MOV      #DMD,@RHMR    ;SET DIAGNOSTIC MODE
3183
3184                          ;*THE REGISTERS WILL BE SAVED IN REINTO BUFFER
3185 016750 004037 043312      JSR      R0,@#SAVER    ;SAVE
3186 016754 001632              RHWC              ;FROM
3187 016756 003154              REINTO           ;TO
3188 016760 000021              17.              ;NUMBER SAVED
3189
3190                          ;*GIVE READ IN PRESET COMMAND
3191 016762 052777 000001 162650  BIS      #GO,@RHCS1    ;INCLUDE GO TO READ IN PRESET
3192
3193                          ;*NOW SAVED REGISTERS WILL BE CHANGED TO EXPECTED VALUE
3194 016770 005037 003166      CLR      @#REINTO+12   ;CLEAR SAVED RHDST
3195 016774 042737 016000 003172  BIC      #FMT22!HCI!ECI,@#REINTO+16 ;CLEAR FMT22,HCI,ECI IN
3196                          ;SAVED RHOF
3197 017002 052737 000100 003172  BIS      #VV,@#REINTO+16 ;SET VV IN SAVED RHOF
3198 017010 005037 003174      CLR      @#REINTO+20   ;CLEAR SAVED RHCA
3199
3200                          ;*AFTER A READ IN PRESET COMMAND
3201                          ;*SAVE REGISTERS AGAIN SO THAT COMPARES CAN BE DONE
3202
3203 017014 004037 043312      JSR      R0,@#SAVER    ;SAVE
3204 017020 001632              RHWC              ;FROM
3205 017022 002110              WRFROM           ;TO
3206 017024 000021              17.              ;NUMBER OF REGISTERS SAVED
3207
3208                          ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
3209                          ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
3210                          ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
3211 017026 113737 003201 002135  MOVB     @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS
3212
3213                          ;*COMPARE REGISTERS BEFORE READ IN PRESET COMMAND
3214                          ;*WITH AFTER COMMAND
3215
3216 017034 004037 043514      JSR      R0,@#COMPAR   ;COMPARE
3217 017040 003154              REINTO           ;GOOD BUFFER
3218 017042 002110              WRFROM           ;TEST BUFFER
3219 017044 000021              17.              ;NUMBER OF REGISTERS
```

3220	017046	017054			1\$:RETURN FOR ERROR
3221	017050	017054			1\$:SAME
3222	017052	017074			2\$:RETURN FOR GOOD COMPARISON
3223								
3224	017054	013705	047624		1\$:	MOV	@#ERWORD,R5	:GETTING READY TO INDEX
3225	017060	060505				ADD	R5,R5	:DOUBLE ERROR WORD
3226	017062	016537	001630	042270		MOV	RHWC-2(R5),@#REGADR	:FAILING REG. ADDRESS
3227	017070	104001				ERROR	1	:READ IN PRESET CAUSED IMPROPER
3228								:REGISTER CHANGE
3229	017072	000207				RTS	PC	:RETURN FOR FURTHER COMPARISONS
3230								
3231	017074				2\$:			:NO ERRORS
3232								

```
3233
3234
3235 017074 000004
3236 017076 012737 000043 002032 TST43: SCOPE
3237
3238 ;*START WITH CLR IN RHCS2 (BIT5)
3239 017104 004737 042620 JSR PC,@#CLDISK ;CLEAR ALL POSSIBLE BITS
3240 017110 012777 000001 162542 MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE
3241 017116 013711 002042 MOV @#NOPERA,@R1 ;PUT NOP OPERATION=0 IN RHCS1
3242 017122 012700 001632 MOV #RHC,R0 ;STARTING ADDRESS OF REG
3243 017126 012703 001706 MOV #WC,R3 ;STARTING ADDRESS OF WHERE SAVED
3244 017132 012702 000021 MOV #RHC2-RHC+2/2,R2 ;NUMBER OF REGISTERS
3245 017136 013023 1$: MOV @(R0)+,(R3)+ ;SAVE HARDWARE REG
3246 017140 005302 DEC R2 ;COUNT
3247 017142 001375 BNE 1$ ;BRANCH IF NOT COMPLETE
3248 017144 013737 001662 017164 MOV @RHDS1,@#2$ ;GET ADDRESS OF DRIVE STATUS
3249 017152 010137 017172 MOV R1,@#3$ ;GET ADDRESS OF RHCS1
3250 017156 052711 000001 BIS #GO,@R1 ;GO TO RHCS1
3251 017162 104415 WAT ;WAIT FOR DRY IN RHDS1
3252 017164 000000 2$: .WORD 0 ;ADDRESS OF DRIVE STATUS RHDS1
3253 017166 000200 DRY ;DRY WILL BE WAITED ON
3254 017170 104415 WAT ;WAIT FOR RDY IN RHCS1
3255 017172 000000 3$: .WORD 0 ;ADDRESS OF RHCS1 PUT HERE BY AN
3256 ;EARLIER MOV
3257 017174 000200 RDY ;RDY WILL BE WAITED ON
3258
3259 ;*AFTER A NO OP COMMAND
3260 ;*SAVE REGISTERS AGAIN SO THAT COMPARES CAN BE DONE
3261
3262 017176 004037 043312 JSR R0,@#SAVER ;SAVE
3263 017202 001632 RHWC ;FROM
3264 017204 002110 WRFROM ;TO
3265 017206 000021 17. ;NUMBER OF REGISTERS SAVED
3266
3267 ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
3268 ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
3269 ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
3270 017210 113737 001733 002135 MOVB @#AS+1,@#WRFROM+25;SAVE UPPER RHAS
3271
3272
3273 ;*COMPARE REGISTERS BEFORE NO OP COMMAND
3274 ;*WITH AFTER COMMAND
3275
3276 017216 004037 043514 JSR R0,@#COMPAR ;COMPARE
3277 017222 001706 WC ;GOOD BUFFER
3278 017224 002110 WRFROM ;TEST BUFFER
3279 017226 000021 17. ;NUMBER OF REGISTERS
3280 017230 017236 4$ ;RETURN FOR ERROR
3281 017232 017236 4$ ;SAME
3282 017234 017256 5$ ;RETURN FOR GOOD COMPARISON
3283
3284 017236 013705 047624 4$: MOV @#ERWORD,R5 ;GETTING READY TO INDEX
3285 017242 060505 ADD R5,R5 ;DOUBLE ERROR WORD
3286 017244 016537 001630 042270 MOV RHC-2(R5),@#REGADR ;FAILING REG. ADDRESS
3287 017252 104001 ERROR 1 ;NO OP COMMAND CAUSED IMPROPER
3288 ;REGISTER CHANGE
```

```
3289 017254 000207          RTS      PC          ;RETURN FOR FURTHER COMPARISONS
3290
3291 017256          5$:          ;NO ERRORS
3292
3293
3294
3295 017256 012737 017264 001110      MOV      #14$,@#$LPERR ;SET SCOPE LOOP TO 14$
3296 017264 004737 042620      JSR      PC,@#CLDISK  ;INIT LAST ALL ZERO TEST
3297 017270 012777 000001 162362      MOV      #DMD,@RHMR   ;SET DIAGNOSTIC MODE
3298
3299
3300          ;*NOW START WITH ALL ONES IN ALL POSSIBLE REGISTERS
3301
3302 017276 012700 001632      MOV      #RHC2,R0     ;ADDRESS OF FIRST REGISTER
3303 017302 012705 000021      MOV      #RHC2-RHC+2/2,R5 ;NO. OF REGISTERS
3304 017306 012730 177676      MOV      #177676,@(R0)+ ;FILL WITH ALL ONES
3305 017312 013777 001774 162316      MOV      @#UNIT,@RHCS2 ;REINSTATE UNIT NUMBER UNDER TEST
3306          ;KEEP INTERRUPT DISABLED
3307 017320 005305          DEC      R5          ;COUNT
3308 017322 001371          BNE     6$          ;BRANCH IF INCOMPLETE
3309 017324 013711 002042      MOV      @#NOPERA,@R1 ;PUT NOP OPERATION =0 IN RHCS1
3310 017330 012700 001632      MOV      #RHC2,R0     ;STARTING ADDRESS OF REG
3311 017334 012703 001706      MOV      #WC,R3       ;STARTING ADDRESS OF WHERE SAVED
3312 017340 012702 000021      MOV      #RHC2-RHC+2/2,R2 ;NUMBER OF REGISTERS
3313 017344 013023          MOV      @(R0)+,(R3)+ ;SAVE HARDWARE REG
3314 017346 005302          DEC      R2          ;COUNT
3315 017350 001375          BNE     7$          ;BRANCH IF NOT COMPLETE
3316 017352 013737 001662 017372      MOV      @#RHDS1,@#10$ ;GET ADDRESS OF DRIVE STATUS
3317 017360 010137 017400      MOV      R1,@#11$    ;GET ADDRESS OF RHCS1
3318 017364 052711 000001      BIS      #GO,@R1     ;GO TO RHCS1
3319 017370 104415          WAT          ;WAIT FOR DRY IN RHDS1
3320 017372 000000          .WORD    0          ;ADDRESS OF DRIVE STATUS RHDS1
3321 017374 000200          DRY        ;DRY WILL BE WAITED ON
3322 017376 104415          WAT        ;WAIT FOR RDY IN RHCS1
3323 017400 000000          .WORD    0          ;ADDRESS OF RHCS1 PUT HERE BY AN
3324          ;EARLIER MOV.
3325 017402 000200          RDY        ;RDY WILL BE WAITED ON
3326
3327          ;*AFTER A NO OP COMMAND
3328          ;*SAVE REGISTERS AGAIN SO THAT COMPARES CAN BE DONE
3329
3330 017404 004037 043312      JSR      R0,@#SAVER   ;SAVE
3331 017410 001632          RHC      ;FROM
3332 017412 002110          WRFROM   ;TO
3333 017414 000021          17.     ;NUMBER OF REGISTERS SAVED
3334
3335          ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
3336          ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
3337          ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
3338 017416 113737 001733 002135      MOV     @#AS+1,@#WRFROM+25;SAVE UPPER RHAS
3339
3340          ;*COMPARE REGISTERS BEFORE NO OP COMMAND
3341          ;*WITH AFTER COMMAND
3342
3343
3344 017424 004037 043514      JSR      R0,@#COMPAR ;COMPARE
```

3345	017430	001706			WC			:GOOD BUFFER
3346	017432	002110			WRFROM			:TEST BUFFER
3347	017434	000021			17.			:NUMBER OF REGISTERS
3348	017436	017444			12\$:RETURN FOR ERROR
3349	017440	017444			12\$:SAME
3350	017442	017464			13\$:RETURN FOR GOOD COMPARISON
3351								
3352	017444	013705	047624	12\$:	MOV	@#ERWORD,R5		:GETTING READY TO INDEX
3353	017450	060505			ADD	R5,R5		:DOUBLE ERROR WORD
3354	017452	016537	001630 042270		MOV	RHWC-2(R5),@#REGADR		:FAILING REG. ADDRESS
3355	017460	104001			ERROR	1		:NO OP COMMAND CAUSED IMPROPER
3356								:REGISTER CHANGE
3357	017462	000207			RTS	PC		:RETURN FOR FURTHER COMPARISONS
3358								
3359	017464			13\$:				:NO ERRORS


```
3360
3361
3362 017464 000004          TST44: SCOPE
3363 017466 012706 001000    MOV    #STACK,SP      ;RESET STACK
3364 017472 012737 000044 002032  MOV    #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
3365 017500 004737 042620    JSR    PC,@#CLDISK   ;SET REGISTERS AND CLEAR
3366
3367          ;*FILL ALL POSSIBLE BITS WITH ONES
3368
3369 017504 012777 177777 162116  MOV    #177777,@RHDB  ;BUS ADDRESS REGISTER GETS 177777
3370 017512 012777 177777 162112  MOV    #177777,@RHWC  ;WORD COUNT REGISTER GETS 177777
3371 017520 012777 177777 162106  MOV    #177777,@RHBA  ;BUS ADDRESS REGISTER GETS 177777
3372 017526 052777 157010 162102  BIS    #157010,@RHCS2 ;CONTROL AND STATUS 2 GETS 157010
3373 017534 012777 001476 162076  MOV    #1476,@RHCS1  ;CONTROL AND STATUS REGISTER/GETS 1476
3374 017542 012777 177777 162072  MOV    #177777,@RHER1 ;ERROR REGISTER1 GETS 177777
3375 017550 012777 017437 162066  MOV    #17437,@RHDST  ;DESIRED SECTOR TRACK
3376 017556 012777 177777 162062  MOV    #177777,@RHER2 ;ERROR REGISTER 2
3377 017564 012777 016277 162056  MOV    #16277,@RHOF   ;OFFSET REGISTER
3378 017572 012777 177777 162052  MOV    #177777,@RHCA  ;DESIRED CYLINDER
3379 017600 012777 177777 162046  MOV    #177777,@RHER3 ;ERROR REGISTER 3
3380 017606 012777 000001 162044  MOV    #DMD,@RHMR    ;MAINTENANCE REGISTER
3381 017614 012777 177777 162036  MOV    #177777,@RHMR ;MAINTENANCE REGISTER
3382
3383
3384          ;*THIS SETS BITS FOR ALL PRESENT DRIVES
3385
3386 017622 013700 002020    MOV    @#TOTALAT,R0  ;GET DRIVE PRESENT
3387 017626 005012          CLR    @R2           ;CLEAR RHCS2 AND CARRY BIT
3388 017630 012705 000010    MOV    #8.,R5       ;COUNTER
3389 017634 006000          ROR    R0           ;GET BIT INTO CARRY
3390 017636 103002          BCC   31$          ;BRANCH IF NO UNIT ON THIS BIT
3391 017640 012714 177777    MOV    #-1,@R4      ;MOVE INTO ERROR REGISTER TO SET ATA
3392 017644 005212          INC    @R2          ;INCREMENT RHCS2 - UNIT NO.
3393 017646 005305          DEC    R5           ;COUNT
3394 017650 001401          BEQ   27$          ;BRANCH IF 8 DONE
3395 017652 000770          BR    30$          ;CONTINUE THIS ROUTINE
3396 017654 013746 001774 27$:  MOV    @#UNIT,-(SP) ;REINSTATE SET BITS
3397 017660 052716 157010    BIS    #157010,(SP) ;
3398 017664 012612          MOV    (SP)+,@R2   ;
3399
3400
3401 017666 012777 000001 161764  MOV    #DMD,@RHMR   ;SET DMD
3402 017674 013711 002050    MOV    @#DCLEAR,@R1 ;DRIVE CLEAR = 10 INTO RHCS1
3403 017700 052711 000001    BIS    #GO,@R1     ;GO
3404 017704 012700 001630    MOV    #RHDB, R0   ;R0 CONTAINS ADDR. OF ADDR. OF REG.
3405
3406
3407          ;*DATA BUFFER REGISTER
3408
3409 017710 012737 177777 001124 28$:  MOV    #177777,@#$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
3410 017716 013037 001126    MOV    @R0+,@#$BDDAT ;TEST DATA
3411 017722 022737 177777 001126    CMP    #177777,@#$BDDAT ;COMPARE DATA
3412 017730 001402          BEQ   3$           ;BRANCH IF GOOD
3413 017732 004737 020570    JSR    PC,@#ERCLFC ;JUMP TO ERROR FOR CLR (BIT 5)
3414
3415
```

```
3416 ;*WORD COUNT REGISTER
3417
3418 017736 012737 177777 001124 3$: MOV #177777,@#$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
3419 017744 013037 001126 001126 MOV @ (R0)+,@#$BDDAT ;TEST DATA
3420 017750 022737 177777 001126 CMP #177777,@#$BDDAT ;COMPARE DATA
3421 017756 001402 BEQ 4$ ;BRANCH IF GOOD
3422 017760 004737 020570 JSR PC,@#ERCLFC ;JUMP TO ERROR FOR CLR (BIT 5)
3423
3424 ;*BUS ADDRESS REGISTER
3425
3426
3427 017764 012737 177776 001124 4$: MOV #177776,@#$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
3428 017772 013037 001126 001126 MOV @ (R0)+,@#$BDDAT ;TEST DATA
3429 017776 022737 177776 001126 CMP #177776,@#$BDDAT ;COMPARE DATA
3430 020004 001402 BEQ 5$ ;BRANCH IF GOOD
3431 020006 004737 020570 JSR PC,@#ERCLFC ;JUMP TO ERROR FOR CLR (BIT 5)
3432
3433 ;*CONTROL AND STATUS 2 REGISTER
3434
3435
3436
3437 020012 012746 000110 5$: MOV #110,-(SP) ;INCLUDE IR
3438 020016 053716 001774 BIS @#UNIT,(SP) ;SET UNIT NO.
3439 020022 012637 001124 MOV (SP)+,@#$GDDAT ;GOOD DATA FOR TYPE OUT
3440 020026 013037 001126 MOV @ (R0)+,@#$BDDAT ;TEST DATA
3441 020032 023737 001124 001126 CMP @#$GDDAT,@#$BDDAT ;COMPARE DATA
3442 020040 001402 BEQ 6$ ;BRANCH IF GOOD
3443 020042 004737 020570 JSR PC,@#ERCLFC ;JUMP TO ERROR FOR CLR (BIT 5)
3444
3445 ;*CONTROL AND STATUS 1 REGISTER
3446
3447
3448 020046 005737 002000 6$: TST @#NUNIT ;ARE THERE MORE THAN ONE UNIT
3449 020052 001404 BEQ 32$ ;BRANCH IF ONLY ONE UNIT
3450 020054 012737 104210 001124 MOV #104210,@#$GDDAT ;GOOD DATA
3451 020062 000403 BR 33$
3452 020064 012737 004210 001124 32$: MOV #4210,@#$GDDAT ;GOOD DATA
3453 020072 013037 001126 33$: MOV @ (R0)+,@#$BDDAT ;TEST DATA
3454
3455 020076 023737 001124 001126 CMP @#$GDDAT,@#$BDDAT ;COMPARE DATA
3456 020104 001402 BEQ 7$ ;BRANCH IF GOOD
3457 020106 004737 020570 JSR PC,@#ERCLFC ;JUMP TO ERROR FOR CLR BIT 5
3458 ;IN RHCS2
3459
3460 ;*ERROR 1 REGISTER
3461
3462 020112 012737 000000 001124 7$: MOV #0,@#$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
3463 020120 013037 001126 MOV @ (R0)+,@#$BDDAT ;TEST DATA
3464 020124 022737 000000 001126 CMP #0,@#$BDDAT ;COMPARE DATA
3465 020132 001402 BEQ 10$ ;BRANCH IF GOOD
3466 020134 004737 020570 JSR PC,@#ERCLFC ;JUMP TO ERROR FOR CLR (BIT 5)
3467
3468 ;*DESIRED SECTOR/TRACK REGISTER
3469
3470 020140 012737 017437 001124 10$: MOV #17437,@#$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
3471 020146 013037 001126 MOV @ (R0)+,@#$BDDAT ;TEST DATA
```

```

3472 020152 022737 017437 001126      CMP    #17437,@#$BDDAT ;COMPARE DATA
3473 020160 001402                BEQ    11$              ;BRANCH IF GOOD
3474 020162 004737 020570      JSR    PC,@#ERCLFC    ;JUMP TO ERROR FOR CLR (BIT 5)
3475
3476                                ;*ERROR 2 REGISTER
3477
3478 020166 012737 000000 001124 11$:  MOV    #0,@#$GDDAT    ;GOOD DATA FOR ERROR TYPEOUT
3479 020174 013037 001126      MOV    @ (R0)+,@#$BDDAT ;TEST DATA
3480 020200 022737 000000 001126      CMP    #0,@#$BDDAT    ;COMPARE DATA
3481 020206 001402                BEQ    12$              ;BRANCH IF GOOD
3482 020210 004737 020570      JSR    PC,@#ERCLFC    ;JUMP TO ERROR FOR CLR (BIT 5)
3483
3484                                ;*OFFSET REGISTER
3485
3486 020214 012737 116000 001124 12$:  MOV    #116000,@#$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
3487 020222 013037 001126      MOV    @ (R0)+,@#$BDDAT ;TEST DATA
3488 020226 022737 116000 001126      CMP    #116000,@#$BDDAT ;COMPARE DATA
3489 020234 001402                BEQ    13$              ;BRANCH IF GOOD
3490 020236 004737 020570      JSR    PC,@#ERCLFC    ;JUMP TO ERROR FOR CLR (BIT 5)
3491
3492                                ;*DESIRED CYLINDER ADDRESS REGISTER
3493
3494 020242 012737 001777 001124 13$:  MOV    #1777,@#$GDDAT  ;GOOD DATA FOR ERROR TYPEOUT
3495 020250 013037 001126      MOV    @ (R0)+,@#$BDDAT ;TEST DATA
3496 020254 022737 001777 001126      CMP    #1777,@#$BDDAT  ;COMPARE DATA
3497 020262 001402                BEQ    14$              ;BRANCH IF GOOD
3498 020264 004737 020570      JSR    PC,@#ERCLFC    ;JUMP TO ERROR FOR CLR (BIT 5)
3499
3500                                ;*ERROR 3 REGISTER
3501
3502 020270 012737 000000 001124 14$:  MOV    #0,@#$GDDAT    ;GOOD DATA FOR ERROR TYPEOUT
3503 020276 013037 001126      MOV    @ (R0)+,@#$BDDAT ;TEST DATA
3504 020302 022737 000000 001126      CMP    #0,@#$BDDAT    ;COMPARE DATA
3505 020310 001402                BEQ    15$              ;BRANCH IF GOOD
3506 020312 004737 020570      JSR    PC,@#ERCLFC    ;JUMP TO ERROR FOR CLR (BIT 5)
3507
3508                                ;*ATTENTION SUMMARY REGISTER
3509
3510 020316 013737 002020 001124 15$:  MOV    @#TOTALAT,@#$GDDAT ;SET ALL BITS OF DRIVE PRESENT IN RHAS
3511 020324 043737 002016 001124      BIC    @#ATTENT,@#$GDDAT ;CLEAR ONLY WORKING DRIVE BIT
3512 020332 013037 001126      MOV    @ (R0)+,@#$BDDAT ;GET RHAS
3513 020336 123737 001124 001126      CMPB   @#$GDDAT,@#$BDDAT ;COMPARE DATA
3514 020344 001402                BEQ    16$              ;BRANCH IF GOOD
3515 020346 004737 020570      JSR    PC,@#ERCLFC    ;JUMP TO ERROR FOR CLR (BIT 5) IN RHCS2
3516
3517                                ;*MAINTAINABILITY REGISTER
3518
3519 020352 012737 000400 001124 16$:  MOV    #400,@#$GDDAT   ;GOOD DATA FOR ERROR TYPEOUT
3520 020360 013037 001126      MOV    @ (R0)+,@#$BDDAT ;TEST DATA
3521 020364 022737 000400 001126      CMP    #400,@#$BDDAT   ;COMPARE DATA
3522 020372 001402                BEQ    17$              ;BRANCH IF GOOD
3523 020374 004737 020570      JSR    PC,@#ERCLFC    ;JUMP TO ERROR FOR CLR (BIT 5)
3524
3525                                ;*DRIVE STATUS REGISTER
3526
3527 020400 012737 000700 001124 17$:  MOV    #700,@#$GDDAT   ;GOOD DATA FOR PRINTOUT
  
```

```
3528 020406 013046          MOV    @ (R0)+, -(SP)      ;GET RHDS1
3529 020410 011637 001126    MOV    (SP), @#$BDDAT     ;TEST DATA
3530 020414 042716 001000    BIC    #PROG, (SP)        ;CLEAR PROG BIT
3531 020420 022726 000700    CMP    #700, (SP)+        ;COMPARE DATA
3532 020424 001402          BEQ    20$                 ;BRANCH IF GOOD
3533 020426 004737 020570    JSR    PC, @#ERCLFC       ;JUMP TO ERROR FOR DRIVE CLEAR
3534
3535          ;*DRIVE TYPE
3536
3537 020432 013737 002010 001124 20$:  MOV    @#SAVDT, @#$GDDAT   ;GOOD DATA FOR ERROR TYPEOUT
3538 020440 013037 001126    MOV    @ (R0)+, @#$BDDAT  ;TEST DATA
3539 020444 023737 002010 001126    CMP    @#SAVDT, @#$BDDAT  ;COMPARE DATA
3540 020452 001402          BEQ    21$                 ;BRANCH IF GOOD
3541 020454 004737 020570    JSR    PC, @#ERCLFC       ;JUMP TO ERROR FOR CLR (BIT 5)
3542
3543          ;*SERIAL NUMBER REGISTER
3544
3545
3546 020460 013737 002012 001124 21$:  MOV    @#SAVSN, @#$GDDAT   ;GOOD DATA FOR ERROR TYPEOUT
3547 020466 013037 001126    MOV    @ (R0)+, @#$BDDAT  ;TEST DATA
3548 020472 023737 002012 001126    CMP    @#SAVSN, @#$BDDAT  ;COMPARE DATA
3549 020500 001402          BEQ    22$                 ;BRANCH IF GOOD
3550 020502 004737 020570    JSR    PC, @#ERCLFC       ;JUMP TO ERROR FOR CLR (BIT 5)
3551
3552          ;*ECC1 POSITION
3553
3554 020506 012737 000000 001124 22$:  MOV    #0, @#$GDDAT        ;GOOD DATA FOR ERROR TYPEOUT
3555 020514 013037 001126    MOV    @ (R0)+, @#$BDDAT  ;TEST DATA
3556 020520 022737 000000 001126    CMP    #0, @#$BDDAT       ;COMPARE DATA
3557 020526 001402          BEQ    23$                 ;BRANCH IF GOOD
3558 020530 004737 020570    JSR    PC, @#ERCLFC       ;JUMP TO ERROR FOR CLR (BIT 5)
3559
3560          ;*ECC2 PATTERN
3561
3562 020534 012737 000000 001124 23$:  MOV    #0, @#$GDDAT        ;GOOD DATA FOR ERROR TYPEOUT
3563 020542 013037 001126    MOV    @ (R0)+, @#$BDDAT  ;TEST DATA
3564 020546 022737 000000 001126    CMP    #0, @#$BDDAT       ;COMPARE DATA
3565 020554 001402          BEQ    24$                 ;BRANCH IF GOOD
3566 020556 004737 020570    JSR    PC, @#ERCLFC       ;JUMP TO ERROR FOR CLR (BIT 5)
3567
3568          ;*LOOK-AHEAD REGISTER
3569
3570
3571 020562 005720          24$:  TST    (R0)+              ;AS THE LOOK-AHEAD REG. CANNOT BE PREDICTED
3572          ;IT IS NOT CHECKED AFTER AN INIT
3573
3574          ;*CURRENT CYLINDER ADDRESS REGISTER
3575
3576 020564 005720          25$:  TST    (R0)+              ;AS THE CURRENT CYL REG. CANNOT BE PREDICTED
3577          ;AFTER AN INIT IT IS NOT CHECKED
3578
3579          26$:
3580 020566 000413          BR     TST45               ;BRANCH OVER JSR
3581
3582 020570 014037 042270    ERCLFC: MOV    -(R0), @#REGADR ;FAILING REGISTER ADDRESS
3583 020574 104001          ERROR 1                   ;CLR FUNCTION = 10 IN RHCS1 DID
```



```

3594
3595
3596
3597 020616 000004          TST45: SCOPE
3598 J20620 012706 001000    MOV      #STACK,SP          ;RESET STACK
3599 020624 012737 000045 002032    MOV      #TTNO,@#TSTNM     ;THIS SAVES TEST NUMBER
3600 020632 004737 042620          JSR      PC,@#CLDISK       ;INIT AND SET UP GENERAL REG. CORRES.
3601                                ;AND UNIT NUMBER
3602 020636 012777 000001 161014    MOV      #DMD,@RHMR        ;SET DIAGNOSTIC MODE BIT
3603                                ;THIS ENABLES COMMANDS WITHOUT MOL
3604                                ;AND HOLDS RHLA FROM MOVING
3605 020644 005077 160774          CLR      @RHDST            ;MAKE DESIRED SECTOR TRACK LEGAL
3606 020650 013777 002072 160762    MOV      @#SEECOM,@RHCS1   ;LOAD SEEK COMMAND INTO CONTROLLER
3607 020656 017746 161014          MOV      @RHCC,-(SP)       ;GET CURRENT CYLINDER
3608
3609
3610                                ;*FOLLOWING ARE TWO BLOCKS OF CODE TO LOAD RHCA WITH THE PROPER
3611                                ;*ADDRESS DEPENDING UPON WHETHER THE DRIVE IS AN RP06 OR RP04
3612
3613 020662 005737 002036          TST      @#RP06           ;MOVE DRIVE TYPE FLAG TO ITSELF TO TEST
3614 020666 001416          BEQ      11$              ;TREAT THE DRIVE AS AN RP04
3615
3616                                ;TREAT THE DRIVE AS AN RP06
3617 020670 022726 001456          CMP      #814.,(SP)+      ;IS CURRENT CYLINDER SAME AS 814. ?
3618 020674 001404          BEQ      9$                ;BRANCH IF YES TO MAKE RHCA = 813.
3619 020676 012737 001456 001210    MOV      #814.,@#STMP5    ;GET READY TO MAKE RHCA = 814.
3620 020704 000403          BR       10$              ;FILL RHCA
3621 020706 012737 001455 001210 9$:   MOV      #813.,@#STMP5    ;GET READY TO MAKE RHCA = 813.
3622 020714 013777 001210 160730 10$:  MOV      @#STMP5,@RHCA    ;MAKE DESIRED CYLINDER 814., OR 813.
3623 020722 000415          BR       14$              ;SAVE REGISTERS
3624
3625                                ;TREAT THE DRIVE AS AN RP04
3626 020724 022726 000632          11$:   CMP      #410.,(SP)+      ;IS CURRENT CYLINDER SAME AS 410. ?
3627 020730 001404          BEQ      12$              ;BRANCH IF YES TO MAKE RHCA = 409.
3628 020732 012737 000632 001210    MOV      #410.,@#STMP5    ;GET READY TO MAKE RHCA = 410.
3629 020740 000403          BR       13$              ;FILL RHCA
3630 020742 012737 000631 001210 12$:  MOV      #409.,@#STMP5    ;GET READY TO MAKE RHCA = 409.
3631 020750 013777 001210 160674 13$:  MOV      @#STMP5,@RHCA    ;MAKE DESIRED CYLINDER 410., OR 409.
3632
3633                                ;*SAVE REGISTERS FOR COMPARISON AFTER GO
3634
3635 020756 004077 043312          14$:   JSR      R0,@#SAVER        ;SAVE
3636 020762 001632          RHW     ;FROM
3637 020764 003154          REINTO  ;TO
3638 020766 000023          19.     ;NUMBER OF REGISTERS SAVED
3639
3640                                ;*GIVE GO TO COMMAND
3641 020770 052777 000001 160642    BIS      #GO,@RHCS1       ;GO TO COMMAND
3642
3643                                ;*CHANGE SAVED REGISTERS TO EXPECTED VALUES
3644
3645 020776 052737 000001 003162    BIS      #GO,@#REINTO+6   ;SAVED RHCS1
3646 021004 052737 020000 003204    BIS      #PIP,@#REINTO+30 ;SAVED RHDS1
3647 021012 042737 000200 003204    BIC      #DRY,@#REINTO+30 ;SAVED RHDS1
3648
3649

```

```
3650 ;*AFTER GO HAS BEEN GIVEN FOR SEEK COMMAND
3651 ;*SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN
3652 ;*BE DONE
3653
3654 021020 004037 043312 JSR RO,@#SAVER ;SAVE
3655 021024 001632 RHWC ;FROM
3656 021026 002110 WRFROM ;TO
3657 021030 000023 19. ;NUMBER OF REGISTERS SAVED
3658
3659 ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
3660 ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
3661 ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
3662
3663 021032 113737 003201 002135 MOVB @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS
3664
3665 ;*COMPARE REGISTERS BEFORE SEEK COMMAND
3666 ;*WITH CONTENTS AFTER GO IS ISSUED
3667
3668 021040 004037 043514 JSR RO,@#COMPAR ;COMPARE
3669 021044 003154 REINTO ;GOOD BUFFER
3670 021046 002110 WRFROM ;TEST BUFFER
3671 021050 000023 19. ;NUMBER
3672 021052 021060 1$ ;RETURN FOR ERROR
3673 021054 021060 1$ ;SAME
3674 021056 021100 2$ ;RETURN FOR GOOD COMPARISON
3675
3676 021060 013705 047624 1$: MOV @#ERWORD,R5 ;GETTING READY TO INDEX
3677 021064 060505 ADD R5,R5 ;DOUBLE ERROR WORD
3678 021066 016537 001630 042270 MOV RHWC-2(R5),@#REGADR ;FAILING REGISTER ADDRESS
3679
3680 021074 104001 ERROR 1 ;IMPROPER REGISTER CHANGE
3681 ;AFTER SEEK COMMAND
3682 ;WITH GO IS GIVEN
3683 021076 000207 RTS PC ;RETURN TO COMPARISON
3684
3685 ;*NOW GIVE INIT AND GET GO AND PIP DOWN
3686
3687
3688 021100 052712 000040 2$: BIS #CLR,@R2 ;RH INITILIZE
3689 021104 013712 001774 MOV @#UNIT,@R2 ;REINSTATE UNIT NUMBER
3690 021110 012777 000001 160542 MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE BIT
3691 ;THIS ENABLES COMMANDS WITHOUT MOL
3692 ;AND HOLDS RHLA FROM MOVING
3693
3694 ;*CHANGE REGISTERS TO EXPECTED VALUE
3695
3696 021116 042737 000001 003162 BIC #GO,@#REINTO+6 ;SAVED RHCS1
3697 021124 042737 020000 003204 BIC #PIP,@#REINTO+30 ;SAVED RHDS1
3698 021132 052737 000200 003204 BIS #DRY,@#REINTO+30 ;SAVED RHDS1
3699 021140 017737 160530 003216 MOV @RHLA,@#REINTO+42;SAVED RHLA
3700 021146 013737 001210 003220 MOV @#$TMP5,@#REINTO+44 ;SAVED RHCC
3701
3702
3703 ;*AFTER INITILIZE SAVE REGISTERS SO THAT
3704 ;*COMPARES CAN BE DONE
3705
```

```
3706 021154 004037 043312 JSR RO,@#SAVER ;SAVE
3707 021160 001632 RHCW ;FROM
3708 021162 002110 WRFROM ;TO
3709 021164 000023 19. ;NUMBER OF REGISTERS SAVED
3710
3711
3712 ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
3713 ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
3714 ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
3715
3716 021166 113737 003201 002135 MOVB @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS
3717
3718
3719 ;*COMPARE REGISTERS AFTER INITIALIZE
3720
3721 021174 004037 043514 JSR RO,@#COMPAR ;COMPARE
3722 021200 003154 REINTO ;GOOD BUFFER
3723 021202 002110 WRFROM ;TEST BUFFER
3724 021204 000023 19. ;NUMBER OF REGISTERS TO BE
3725 ;COMPARED
3726 021206 021214 3$ ;RETURN POINT FOR ERROR
3727 021210 021214 3$ ;SAME
3728 021212 021234 4$ ;RETURN POINT FOR GOOD COMPARISON
3729
3730 021214 013705 047624 3$: MOV @#ERWORD,R5 ;GETTING READY TO INDEX
3731 021220 060505 ADD R5,R5 ;DOUBLE ERROR WORD
3732 021222 016537 001630 042270 MOV RHCW-2(R5),@#REGADR ;FAILING REGISTER ADDRESS
3733 021230 104001 ERROR 1 ;IMPROPER REGISTER
3734 ;CONTENTS AFTER GIVING AN
3735 ;INITILIZE FOLLOWING A
3736 ;SEEK COMMAND
3737 021232 000207 RTS PC ;RETURN TO COMPARISON
3738
3739 021234 4$: ;GOOD COMPARISON
3740
3741 021234 004737 042620 JSR PC,@#CLDISK ;INIT AND SET UP GENERAL REG.
3742 ;AND UNIT NUMBER
3743 021240 012777 000001 160412 MOV #DMD,@#RHMR ;SET DIAGNOSTIC MODE BIT
3744 ;THIS ENABLES COMMANDS WITHOUT MOL
3745 ;AND HOLDS RHLA FROM MOVING
3746
3747 021246 013777 002072 160364 MOV @#SEECOM,@#RHCS1 ;LOAD SEEK COMMAND INTO RHC
3748 021254 005077 160372 CLR @#RHCA ;DESIRED CYLINDER ADDRESS
3749
3750 ;*SAVE REGISTERS FOR COMPARISON AFTER GO
3751 021260 004037 043312 JSR RO,@#SAVER ;SAVE
3752 021264 001632 RHCW ;FROM
3753 021266 003154 REINTO ;TO
3754 021270 000023 19. ;NUMBER OF REGISTERS SAVED
3755
3756 ;*GIVE GO TO SEEK COMMAND
3757 021272 052777 000001 160340 BIS #GO,@#RHCS1 ;GO TO SEEK COMMAND
3758
3759 ;*CHANGE SAVED REGISTERS TO EXPECTED VALUES
3760
3761 021300 052737 000001 003162 BIS #GO,@#REINTO+6 ;SAVED RHCS1
```



```

3762 021306 052737 020000 003204 BIS #PIP,@#REINTO+30 ;SAVED RHDS1
3763 021314 042737 000200 003204 BIC #DRY,@#REINTO+30 ;SAVED RHDS1
3764
3765
3766 ;*AFTER GO HAS BEEN GIVEN TO SEEK COMMAND
3767 ;*SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN
3768 ;*BE DONE
3769 021322 004037 043312 JSR RO,@#SAVER ;SAVE
3770 021326 001632 RHWC ;FROM
3771 021330 002110 WRFROM ;TO
3772 021332 000023 19. ;NUMBER OF REGISTERS SAVED
3773
3774
3775 ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
3776 ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
3777 ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
3778 021334 113737 003201 002135 MOVB @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS
3779
3780
3781 ;*COMPARE REGISTERS BEFORE COMMAND
3782 ;*WITH CONTENTS AFTER GO IS GIVEN
3783
3784 021342 004037 043514 JSR RO,@#COMPAR ;COMPARE
3785 021346 003154 REINTO ;GOOD BUFFER
3786 021350 002110 WRFROM ;TEST BUFFER
3787 021352 000023 19. ;NUMBER
3788 021354 021362 5$ ;RETURN FOR ERROR
3789 021356 021362 5$ ;SAME
3790 021360 021402 6$ ;RETURN FOR GOOD COMPARISON
3791 021362 013705 047624 5$: MOV @#ERWORD,R5 ;GETTING READY TO INDEX
3792 021366 060505 ADD R5,R5 ;DOUBLG ERROR WORD
3793 021370 016537 001630 042270 MOV RHWC-2(R5),@#REGADR ;FAILING REGISTER ADDRESS
3794 021376 104001 ERROR 1 ;IMPROPER REGISTER CHANGE
3795 ;AFTER COMMAND
3796 ;WITH GO IS GIVEN
3797 021400 000207 RTS PC ;RETURN TO COMPARISON
3798
3799 ;*NOW GIVE INIT AND GET GO AND PIP DOWN
3800
3801 021402 052712 000040 6$: BIS #CLR,@R2 ;RH INITILIZE
3802 021406 013712 001774 MOV @#UNIT,@R2 ;REINSTATE UNIT NUMBER
3803 021412 012777 000001 160240 MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE BIT
3804 ;THIS ENABLES COMMANDS WITHOUT MOL
3805 ;AND HOLDS RHLA FROM MOVING
3806
3807 ;*CHANGE REGISTERS TO EXPECTED VALUE
3808
3809 021420 042737 000001 003162 BIC #GO,@#REINTO+6 ;SAVED RHCS1
3810 021426 042737 020000 003204 BIC #PIP,@#REINTO+30 ;SAVED RHDS1
3811 021434 052737 000200 003204 BIS #DRY,@#REINTO+30 ;SAVED RHDS1
3812 021442 017737 160226 003216 MOV @RHLA,@#REINTO+42;SAVED RHLA
3813 021450 005037 003220 CLR @#REINTO+44 ;SAVED RHCC
3814
3815
3816 ;*AFTER INITIALIZE SAVE REGISTERS SO THAT
3817 ;*COMPARES CAN BE DONE
    
```

```
3818  
3819 021454 004037 043312 JSR R0,@#SAVER ;SAVE  
3820 021460 001632 RHCW ;FROM  
3821 021462 002110 WRFROM ;TO  
3822 021464 000023 19. ;NUMBER OF REGISTERS SAVED  
3823  
3824 ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT  
3825 ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS  
3826 ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE  
3827 021466 113737 003201 002135 MOVB @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS  
3828  
3829  
3830 ;*COMPARE REGISTERS AFTER INITIALIZE  
3831  
3832 021474 004037 043514 JSR R0,@#COMPAR ;COMPARE  
3833 021500 003154 REINTO ;GOOD BUFFER  
3834 021502 002110 WRFROM ;TEST BUFFER  
3835 021504 000023 19. ;NUMBER OF REGISTERS TO BE  
3836 ;COMPARED  
3837 021506 021514 7$ ;RETURN POINT FOR ERROR  
3838 021510 021514 7$ ;SAME  
3839 021512 021534 8$ ;RETURN POINT FOR GOOD COMPARISON  
3840  
3841 021514 013705 047624 7$: MOV @#ERWORD,R5 ;GETTING READY TO INDEX  
3842 021520 060505 ADD R5,R5 ;DOUBLE ERROR WORD  
3843 021522 016537 001630 042270 MOV RHCW-2(R5),@#REGADR ;FAILING REGISTER ADDRESS  
3844 021530 104001 ERROR 1 ;# CONTENTS AFTER GIVING AN  
3845 ; COMMAND  
3846 021532 000207 RTS PC ;RETURN TO COMPARISON  
3847 021534 8$: ;GOOD COMPARISON
```

```
3848
3849
3850 021534 000004          TST46: SCOPE
3851 021536 012706 001000  MOV      #STACK,SP      ;RESET STACK
3852 021542 012737 000046 002032  MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
3853 021550 004737 042620          JSR      PC,@#CLDISK    ;INIT AND SET UP GENERAL REG.
3854                                ;AND UNIT NUMBER
3855 021554 012777 000001 160076  MOV      #DMD,@RHMR    ;SET DIAGNOSTIC MODE BIT
3856                                ;THIS ENABLES COMMANDS WITHOUT MOL
3857                                ;AND HOLDS RHLA FROM MOVING
3858
3859 021562 013777 002044 160050  MOV      @#UNLOAD,@RHCS1 ;LOAD UNLOAD COMMAND INTO RH
3860
3861                                ;*SAVE REGISTERS FOR COMPARISON AFTER GO
3862
3863 021570 004037 043312          JSR      R0,@#SAVER     ;SAVE
3864 021574 001632          RHWC          ;FROM
3865 021576 003154          REINTO        ;TO
3866 021600 000023          19.          ;NUMBER OF REGISTERS SAVED
3867
3868                                ;*GIVE GO TO UNLOAD COMMAND
3869 021602 052777 000001 160030  BIS      #GO,@RHCS1    ;GO TO UNLOAD COMMAND
3870
3871                                ;*CHANGE SAVED REGISTERS TO EXPECTED VALUES
3872 021610 052737 000001 003162  BIS      #GO,@#REINTO+6 ;SAVED RHCS1
3873 021616 052737 020000 003204  BIS      #PIP,@#REINTO+30 ;SAVED RHDS1
3874 021624 042737 000200 003204  BIC      #DRY,@#REINTO+30 ;SAVED RHDS1
3875
3876 021632 005737 001100          TST      @#$PASS      ;IS THIS FIRST PASS
3877 021636 001053          BNE      5$          ;BRANCH IF NOT FIRST PASS
3878 021640 032777 020000 157272  BIT      #SW13,@SWR    ;INHIBIT ERROR PRINT HIGH?
3879 021646 001047          BNE      5$          ;BRANCH IF SW13 HIGH
3880
3881 021650 104401 021656          TYPE     ,65$         ;;TYPE ASCIZ STRING
3882 021654 000441          BR       64$         ;;GET OVER THE ASCIZ
3883
3884 021760 013746 001774          MOV      @#UNIT,-(SP) ;UNIT UNDER TEST
3885 021764 104405          TYPDS
3886
3887                                ;*AFTER GO HAS BEEN GIVEN TO UNLOAD COMMAND
3888                                ;*SAVED REGISTERS AGAIN SO THAT COMPARISONS CAN
3889                                ;*BE DONE
3890
3891 021766 004037 043312          5$: JSR      R0,@#SAVER     ;SAVE
3892 021772 001632          RHWC          ;FROM
3893 021774 002110          WRFROM        ;TO
3894 021776 000023          19.          ;NUMBER OF REGISTERS SAVED
3895
3896                                ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
3897                                ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
3898                                ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
3899 022000 113737 003201 002135  MOVVB   @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS
3900
3901
3902                                ;*COMPARE REGISTERS BEFORE UNLOAD COMMAND
3903                                ;*WITH AFTER GO
```



```

3967
3968 022174 032777 000020 156736 BIT #SW4,@SWR ;TEST FOR NO OFFSET OR RTC
3969 022202 001402 BEQ 6$ ;IF = 0, DO THE NEXT TWO TESTS
3970 022204 000137 023162 JMP TST51 ;SKIP THE NEXT TWO TESTS -----)
3971 022210 6$: ;CONTINUE WITH NEXT TWO TESTS
3972
3973
3974
3975 022210 000004 TST47: SCOPE
3976 022212 012706 001000 MOV #STACK,SP ;RESET STACK
3977 022216 012737 000047 002032 MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
3978 022224 004737 042620 JSR PC,@#CLDISK ;INIT AND SET UP GENERAL REG.
3979 ;AND UNIT NUMBER
3980 022230 012777 000001 157422 MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE BIT
3981 ;THIS ENABLES COMMANDS WITHOUT MOL
3982 ;AND HOLDS RHLA FROM MOVING
3983
3984 ;*GIVE ONE INDEX PULSE TO CLEAR RHLA BEFORE THE START OF THIS TEST
3985 022236 052777 000004 157414 BIS #MINX,@RHMR ;SET INDEX PULSE
3986 022244 042777 000004 157406 BIC #MINX,@RHMR ;CLEAR INDEX
3987
3988 ;*TO ENABLE LOOP ON THIS TEST THE POSITIONER HAS TO
3989 ;*BE BROUGHT TO CENTER LINE
3990
3991 022252 017777 157420 157372 MOV @RHCC,@RHCA ;SET DESIRED CYLINDER TO RHCC
3992 022260 013711 002072 MOV @#SEECOM,@R1 ;SEEK COMMAND TO RHCS1
3993 022264 005211 INC @R1 ;GO TO SEEK COMMAND
3994
3995 ;*FOUR SECTOR CLOCKS ARE GIVEN TO TAKE POSITIONER OFF OFFSET POSITION
3996 022266 012700 000004 MOV #4,R0 ;COUNTER
3997 022272 012777 000011 157360 5$: MOV #MSTCK!DMD,@RHMR ;SET SECTOR CLOCK
3998 022300 012777 000001 157352 MOV #DMD,@RHMR ;RESET SECTOR CLOCK
3999 022306 005300 DEC R0 ;COUNT
4000 022310 001370 BNE 5$ ;BRANCH IF NOT COMPLETE
4001
4002 022312 004737 042620 JSR PC,@#CLDISK ;INIT AND SET UP GENERAL REG.
4003 ;AND UNIT NUMBER
4004 022316 012777 000001 157334 MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE BIT
4005 ;THIS ENABLES COMMANDS WITHOUT MOL
4006 ;AND HOLDS RHLA FROM MOVING
4007
4008 022324 013777 002074 157306 MOV @#OFSETC,@RHCS1 ;LOAD AN OFFSET BIT
4009 022332 012777 000001 157310 MOV #OF25,@RHOF ;SET AN OFFSET BIT
4010
4011 ;*SAVE REGISTERS FOR COMPARISON AFTER GO
4012 022340 004037 043312 JSR R0,@#SAVER ;SAVE
4013 022344 001632 RHWC ;FROM
4014 022346 003154 REINTO ;TO
4015 022350 000023 19. ;NUMBER OF REGISTERS SAVED
4016
4017 ;*GIVE GO TO OFFSET COMMAND
4018 022352 052777 000001 157260 BIS #GO,@RHCS1 ;GO TO OFFSET COMMAND
4019
4020 ;*CHANGE SAVED REGISTERS TO EXPECTED VALUES
4021 022360 052737 000001 003162 BIS #GO,@#REINTO+6 ;SAVED RHCS1
4022 022366 052737 020000 003204 BIS #PIP,@#REINTO+30 ;SAVED RHDS1
  
```

```

4023 022374 042737 000200 003204 BIC #DRY,@#REINTO+30 ;SAVED RHDS1
4024
4025 ;*AFTER GO HAS BEEN GIVEN TO OFFSET COMMAND
4026 ;*SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN
4027 ;*BE DONE
4028
4029 022402 004037 043312 JSR RO,@#SAVER ;SAVE
4030 022406 001632 RHWC ;FROM
4031 022410 002110 WRFROM ;TO
4032 022412 000023 19. ;NUMBER OF REGISTERS SAVED
4033
4034 ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
4035 ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
4036 ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
4037 022414 113737 003201 002135 MOVB @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS
4038
4039
4040 ;*COMPARE REGISTERS BEFORE OFFSET COMMAND
4041 ;*WITH AFTER GO
4042
4043 022422 004037 043514 JSR RO,@#COMPAR ;COMPARE
4044 022426 003154 REINTO ;GOOD BUFFER
4045 022430 002110 WRFROM ;TEST BUFFER
4046 022432 000023 19. ;NUMBER
4047 022434 022442 1$ ;RETURN FOR ERROR
4048 022436 022442 1$ ;SAME
4049 022440 022462 2$ ;RETURN FOR GOOD COMPARISON
4050
4051 022442 013705 047624 1$: MOV @#ERWORD,R5 ;GETTING READY TO INDEX
4052 022446 060505 ADD R5,R5 ;DOUBLE ERROR WORD
4053 022450 016537 001630 042270 MOV RHWC-2(R5),@#REGADR ;FAILING REGISTER ADDRESS
4054 022456 104001 ERROR 1 ;IMPROPER REGISTER CHANGE
4055 ;AFTER OFFSET COMMAND
4056 ;WITH GO IS GIVEN
4057 022460 000207 RTS PC ;RETURN TO COMPARISON
4058
4059 ;*NOW GIVE INIT AND GET ALL GO AND PIP DOWN
4060
4061 022462 052712 000040 2$: BIS #CLR,@R2 ;RH INITILIZE
4062 022466 013712 001774 MOV @#UNIT,@R2 ;REINSTATE UNIT NUMBER
4063 022472 012777 000001 157160 MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE BIT
4064 ;THIS ENABLES COMMANDS WITHOUT MOL
4065 ;AND HOLDS RHLA FROM MOVING
4066
4067 ;*CHANGE REGISTERS TO EXPECTED VALUE
4068 022500 042737 000001 003162 BIC #GO,@#REINTO+6 ;SAVED RHCS1
4069 022506 042737 000001 003172 BIC #OF25,@#REINTO+16;SAVED RHOF
4070 022514 042737 020000 003204 BIC #PIP,@#REINTO+30 ;SAVED RHDS1
4071 022522 052737 000200 003204 BIS #DRY,@#REINTO+30 ;SAVED RHDS1
4072 022530 017737 157140 003216 MOV @RHLA,@#REINTO+42;SAVED RHLA
4073
4074 ;*AFTER INITIALIZE SAVE REGISTERS SO THAT
4075 ;*COMPARES CAN BE DONE
4076
4077 022536 004037 043312 JSP RO,@#SAVER ;SAVE
4078 022542 001632 RHWC ;FROM
    
```

```
4079 022544 002110 WRFROM ;TO
4080 022546 000023 19. ;NUMBER OF REGISTERS SAVED
4081
4082 ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
4083 ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
4084 ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
4085 022550 113737 003201 002135 MOVB @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS
4086
4087 ;*COMPARE REGISTERS AFTER INITIALIZE
4088 022556 004037 043514 JSR R0,@#COMPAR ;COMPARE
4089 022562 003154 REINTO ;GOOD BUFFER
4090 022564 002110 WRFROM ;TEST BUFFER
4091 022566 000023 19. ;NUMBER OF REGISTERS TO BE
4092 ;COMPARED
4093 022570 022576 3$ ;RETURN POINT FOR ERROR
4094 022572 022576 3$ ;SAME
4095 022574 022616 4$ ;RETURN POINT FOR GOOD COMPARISON
4096
4097 022576 013705 047624 3$: MOV @#ERWORD,R5 ;GETTING READY TO INDEX
4098 022602 060505 ADD R5,R5 ;DOUBLE ERROR WORD
4099 022604 016537 001630 042270 MOV RHWC-2(R5),@#REGADR ;FAILING REGISTER ADDRESS
4100 022612 104001 ERROR 1 ;IMPROPER REGISTER
4101 ;CONTENTS AFTER GIVING AN
4102 ;INITIALIZE FOLLOWING A
4103 ;OFFSET COMMAND
4104 022614 000207 RTS PC ;RETURN TO COMPARISON
4105
4106 022616 4$: ;GOOD COMPARISON
```



```
4107
4108
4109 022616 000004          TST50: SCOPE
4110 022620 012706 001000  MOV      #STACK,SP      ;RESET STACK
4111 022624 012737 000050 002032  MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
4112 022632 004737 042620          JSR      PC,@#CLDISK    ;INIT AND SET UP GENERAL REG.
4113                                ;AND UNIT NUMBER
4114 022636 012777 000001 157014  MOV      #DMD,@RHMR     ;SET DIAGNOSTIC MODE BIT
4115                                ;THIS ENABLES COMMANDS WITHOUT MOL
4116                                ;AND HOLDS RHLA FROM MOVING
4117
4118                                ;*GIVE ONE INDEX PULSE TO CLEAR RHLA BEFORE THE START OF THIS TEST
4119 022644 052777 000004 157006  BIS      #MINX,@RHMR    ;SET INDEX PULSE
4120 022652 042777 000004 157000  BIC      #MINX,@RHMR    ;CLEAR INDEX
4121
4122
4123 022660 013777 002076 156752  MOV      @#RETCL,@RHCS1 ;LOAD RETURN TO CENTER LINE COMMAND INTO RHCS1
4124
4125                                ;*SAVE REGISTERS FOR COMPARISON AFTER GO
4126 022666 004037 043312          JSR      R0,@#SAVER     ;SAVE
4127 022672 001632          RHCW                     ;FROM
4128 022674 003154          REINTO                    ;TO
4129 022676 000023          19.                      ;NUMBER OF REGISTERS SAVED
4130
4131                                ;*GIVE GO TO RETURN TO CENTER LINE COMMAND
4132 022700 052777 000001 156732  BIS      #GO,@RHCS1    ;GO TO RETURN TO CENTER COMMAND
4133
4134
4135                                ;*FOUR SECTOR CLOCKS ARE GIVEN TO TAKE POSITIONER TO CENTER LINE
4136 022706 012700 000004          MOV      #4,R0          ;COUNTER
4137 022712 012777 000011 156740 5$: MOV      #MSTCK!DMD,@RHMR ;SET SECTOR CLOCK
4138 022720 012777 000001 156732  MOV      #DMD,@RHMR    ;RESET SECTOR CLOCK
4139 022726 005300          DEC      R0             ;COUNT
4140 022730 001370          BNE     5$              ;BRANCH IF NOT COMPLETE
4141
4142
4143                                ;*CHANGE SAVED REGISTERS TO EXPECTED VALUES
4144 022732 052737 000001 003162  BIS      #GO,@#REINTO+6 ;SAVED RHCS1
4145 022740 052737 020000 003204  BIS      #PIP,@#REINTO+30 ;SAVED RHDS1
4146 022746 042737 000200 003204  BIC      #DRY,@#REINTO+30 ;SAVED RHDS1
4147
4148                                ;*AFTER GO HAS BEEN GIVEN TO RETURN TO CENTER LINE COMMAND
4149                                ;*SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN
4150                                ;*BE DONE
4151 022754 004037 043312          JSR      R0,@#SAVER     ;SAVE
4152 022760 001632          RHCW                     ;FROM
4153 022762 002110          WRFROM                    ;TO
4154 022764 000023          19.                      ;NUMBER OF REGISTERS SAVED
4155
4156                                ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
4157                                ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
4158                                ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
4159 022766 113737 003201 002135  MOVWB   @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS
4160
4161
4162                                ;*COMPARE REGISTERS BEFORE RETURN TO CENTER LINE COMMAND
```

```
4163                                     ;*WITH AFTER GO
4164 022774 004037 043514                JSR    R0,@#COMPAR      ;COMPARE
4165 023000 003154                        REINTO                       ;GOOD BUFFER
4166 023002 002110                        WRFROM                       ;TEST BUFFER
4167 023004 000023                        19.                          ;NUMBER
4168 023006 023014                        1$                            ;RETURN FOR ERROR
4169 023010 023014                        1$                            ;SAME
4170 023012 023034                        2$                            ;RETURN FOR GOOD COMPARISON
4171
4172 023014 013705 047624                1$:  MOV    @#ERWORD,R5      ;GETTING READY TO INDEX
4173 023020 060505                        ADD    R5,R5                ;DOUBLE ERROR WORD
4174 023022 016537 001630 042270        MOV    RHC-2(R5),@#REGADR ;FAILING REGISTER ADDRESS
4175 023030 104001                        ERROR 1                      ;IMPROPER REGISTER CHANGE
4176                                     ;AFTER RETURN TO CENTER LINE COMMAND
4177                                     ;WITH GO IS GIVEN
4178 023032 000207                        RTS    PC                    ;RETURN TO COMPARISON
4179
4180                                     ;*NOW GIVE INIT AND GET ALL GO AND PIP DOWN
4181
4182 023034 052712 000040                2$:  BIS    #CLR,@R2        ;RH INITILIZE
4183 023040 013712 001774                MOV    @#UNIT,@R2          ;REINSTATE UNIT NUMBER
4184 023044 012777 000001 156606        MOV    #DMD,@RHMR         ;SET DIAGNOSTIC MODE BIT
4185                                     ;THIS ENABLES COMMANDS WITHOUT MOL
4186                                     ;AND HOLDS RHLA FROM MOVING
4187
4188                                     ;*CHANGE REGISTERS TO EXPECTED VALUE
4189 023052 042737 000001 003162          BIC    #GO,@#REINTO+6     ;SAVED RHCS1
4190 023060 042737 020000 003204          BIC    #PIP,@#REINTO+30  ;SAVED RHDS1
4191 023066 052737 000200 003204          BIS    #DRY,@#REINTO+30 ;SAVED RHDS1
4192 023074 017737 156574 003216          MOV    @RHLA,@#REINTO+42;SAVED RHLA
4193
4194                                     ;*AFTER INITIALIZE SAVE REGISTERS SO THAT
4195                                     ;*COMPARES CAN BE DONE
4196 023102 004037 043312                JSR    R0,@#SAVER         ;SAVE
4197 023106 001632                        RHWC                          ;FROM
4198 023110 002110                        WRFROM                       ;TO
4199 023112 000023                        19.                          ;NUMBER OF REGISTERS SAVED
4200
4201                                     ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
4202                                     ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
4203                                     ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
4204 023114 113737 003201 002135        MOV    @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS
4205
4206
4207                                     ;*COMPARE REGISTERS AFTER INITIALIZE
4208 023122 004037 043514                JSR    R0,@#COMPAR      ;COMPARE
4209 023126 003154                        REINTO                       ;GOOD BUFFER
4210 023130 002110                        WRFROM                       ;TEST BUFFER
4211 023132 000023                        19.                          ;NUMBER OF REGISTERS TO BE
4212                                     ;COMPARED
4213 023134 023142                        3$                            ;RETURN POINT FOR ERROR
4214 023136 023142                        3$                            ;SAME
4215 023140 023162                        4$                            ;RETURN POINT FOR GOOD COMPARISON
4216
4217
4218 023142 013705 047624                3$:  MOV    @#ERWORD,R5      ;GETTING READY TO INDEX
```

```
4219 023146 060505 ADD R5,R5 ;DOUBLE ERROR WORD
4220 023150 016537 001630 042270 MOV RHW-C-2(R5),@#REGADR ;FAILING REGISTER ADDRESS
4221 023156 104001 ERROR 1 ;IMPROPER REGISTER
4222 ;CONTENTS AFTER GIVING AN
4223 ;INITIALIZE FOLLOWING A RETURN TO
4224 ;CENTER LINE COMMAND
4225 023160 000207 RTS PC ;RETURN TO COMPARISON
4226
4227 023162 4$: ;GOOD COMPARISON
4228
```

```
4229
4230
4231 023162 000004          TST51: SCOPE
4232 023164 012706 001000  MOV      #STACK,SP      :RESET STACK
4233 023170 012737 000051 002032  MOV      #TTNO,@#TSTNM  :THIS SAVES TEST NUMBER
4234 023176 004737 042620      JSR      PC,@#CLDISK    :INIT AND SET UP GENERAL REG.
4235                                :AND UNIT NUMBER
4236 023202 012777 000001 156450  MOV      #DMD,@RHMR     :SET DIAGNOSTIC MODE BIT
4237                                :THIS ENABLES COMMANDS WITHOUT MOL
4238                                :AND HOLDS RHLA FROM MOVING
4239
4240
4241                                ;*GIVE ONE INDEX PULSE TO CLEAR RHLA BEFORE THE START OF THIS TEST
4242
4243 023210 052777 000004 156442  BIS      #MINX,@RHMR    :SET INDEX PULSE
4244 023216 042777 000004 156434  BIC      #MINX,@RHMR    :CLEAR INDEX
4245
4246 023224 013777 002046 156406  MOV      @#RECALI,@RHCS1 ;LOAD RECALIBRATE COMMAND INTO RHCS1
4247
4248                                ;*SAVE REGISTERS FOR COMPARISON AFTER GO
4249 023232 004037 043312      JSR      RO,@#SAVER     :SAVE
4250 023236 001632              RHCW              :FROM
4251 023240 003154              REINTO            :TO
4252 023242 000023              19.              :NUMBER OF REGISTERS SAVED
4253
4254                                ;*GIVE GO TO RECALIBRATE COMMAND
4255 023244 052777 000001 156366  BIS      #GO,@RHCS1    :GO TO RECALIBRATE COMMAND
4256
4257
4258                                ;*FOUR SECTOR CLOCKS ARE GIVEN TO TAKE POSITIONER TO CYLINDER 0
4259 023252 012700 000004      MOV      #4,RO          :COUNTER
4260 023256 012777 000011 156374 5$:  MOV      #MSTCK!DMD,@RHMR ;SET SECTOR CLOCK
4261 023264 012777 000001 156366      MOV      #DMD,@RHMR    :RESET SECTOR CLOCK
4262 023272 005300              DEC      RO            :COUNT
4263 023274 001370              BNE      5$           :BRANCH IF NOT COMPLETE
4264
4265
4266                                ;*CHANGE SAVED REGISTERS TO EXPECTED VALUES
4267 023276 052737 000001 003162  BIS      #GO,@#REINTO+6 ;SAVED RHCS1
4268 023304 052737 020000 003204  BIS      #PIP,@#REINTO+30 ;SAVED RHDS1
4269 023312 042737 000200 003204  BIC      #DRY,@#REINTO+30 ;SAVED RHDS1
4270
4271                                ;*AFTER GO HAS BEEN GIVEN TO RECALIBRATE COMMAND
4272                                ;*SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN
4273                                ;*BE DONE
4274 023320 004037 043312      JSR      RO,@#SAVER     :SAVE
4275 023324 001632              RHCW              :FROM
4276 023326 002110              WRFROM            :TO
4277 023330 000023              19.              :NUMBER OF REGISTERS SAVED
4278
4279                                ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
4280                                ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
4281                                ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
4282 023332 113737 003201 002135  MOVB     @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS
4283
4284
```

```
4285 ;*COMPARE REGISTERS BEFORE RECALIBRATE COMMAND
4286 ;*WITH AFTER GO
4287 023340 004037 043514 JSR RO,@#COMPAR ;COMPARE
4288 023344 003154 REINTO ;GOOD BUFFER
4289 023346 002110 WRFROM ;TEST BUFFER
4290 023350 000023 19. ;NUMBER
4291 023352 023360 1$ ;RETURN FOR ERROR
4292 023354 023360 1$ ;SAME
4293 023356 023400 2$ ;RETURN FOR GOOD COMPARISON
4294
4295 023360 013705 047624 1$: MOV @#ERWORD,R5 ;GETTING READY TO INDEX
4296 023364 060505 ADD R5,R5 ;DOUBLE ERROR WORD
4297 023366 016537 001630 042270 MOV RHWC-2(R5),@#REGADR ;FAILING REGISTER ADDRESS
4298 023374 104001 ERROR 1 ;IMPROPER REGISTER CHANGE
4299 ;AFTER RECALIBRATE COMMAND
4300 ;WITH GO IS GIVEN
4301 023376 000207 RTS PC ;RETURN TO COMPARISON
4302
4303 ;*NOW GIVE INIT AND GET ALL GO AND PIP DOWN
4304
4305 023400 052712 000040 2$: BIS #CLR,@R2 ;RH INITILIZE
4306 023404 013712 001774 MOV @#UNIT,@R2 ;REINSTATE UNIT NUMBER
4307 023410 012777 000001 156242 MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE BIT
4308 ;THIS ENABLES COMMANDS WITHOUT MOL
4309 ;AND HOLDS RHLA FROM MOVING
4310
4311 ;*CHANGE REGISTERS TO EXPECTED VALUE
4312 023416 042737 000001 003162 BIC #GO,@#REINTO+6 ;SAVED RHCS1
4313 023424 042737 020000 003204 BIC #PIP,@#REINTO+30 ;SAVED RHDS1
4314 023432 052737 000200 003204 BIS #DRY,@#REINTO+30 ;SAVED RHDS1
4315 023440 017737 156230 003216 MOV @RHLA,@#REINTO+42;SAVED RHLA
4316
4317 ;*AFTER INITIALIZE SAVE REGISTERS SO THAT
4318 ;*COMPARES CAN BE DONE
4319 023446 004037 043312 JSR RO,@#SAVER ;SAVE
4320 023452 001632 RHWC ;FROM
4321 023454 002110 WRFROM ;TO
4322 023456 000023 19. ;NUMBER OF REGISTERS SAVED
4323
4324 ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
4325 ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
4326 ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
4327 023460 113737 003201 002135 MOVB @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS
4328
4329 ;*COMPARE REGISTERS AFTER INITIALIZE
4330
4331 023466 004037 043514 JSR RO,@#COMPAR ;COMPARE
4332 023472 003154 REINTO ;GOOD BUFFER
4333 023474 002110 WRFROM ;TEST BUFFER
4334 023476 000023 19. ;NUMBER OF REGISTERS TO BE
4335 ;COMPARED
4336 023500 023506 3$ ;RETURN POINT FOR ERROR
4337 023502 023506 3$ ;SAME
4338 023504 023526 4$ ;RETURN POINT FOR GOOD COMPARISON
4339
4340 023506 013705 047624 3$: MOV @#ERWORD,R5 ;GETTING READY TO INDEX
```



```

4350
4351
4352 023526 000004          TST52: SCOPE
4353 023530 012706 001000  MOV      #STACK,SP      ;RESET STACK
4354 023534 012737 000052 002032  MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
4355 023542 004737 042620          JSR      PC,@#CLDISK    ;INIT AND SET UP GENERAL REG.
4356                                ;AND UNIT NUMBER
4357 023546 012777 000001 156104  MOV      #DMD,@RHMR     ;SET DIAGNOSTIC MODE BIT
4358                                ;THIS ENABLES COMMANDS WITHOUT MOL
4359                                ;AND HOLDS RHLA FROM MOVING
4360
4361 023554 013777 002052 156056  MOV      @#RELEASE,@RHCS1 ;LOAD RELEASE COMMAND INTO RHCS1
4362
4363 ;*SAVE REGISTERS FOR COMPARISON AFTER GO
4364 023562 004037 043312  JSR      RO,@#SAVER     ;SAVE
4365 023566 001632          RHW      ;FROM
4366 023570 003154          REINTO   ;TO
4367 023572 000023          19.      ;NUMBER OF REGISTERS SAVED
4368
4369 ;*GIVE GO TO RELEASE COMMAND
4370 023574 052777 000001 156036  BIS      #GO,@RHCS1    ;GO TO RELEASE COMMAND
4371 023602 052777 000001 156050  BIS      #DMD,@RHMR    ;SET DMD TO HOLD RHLA
4372
4373 ;*CHANGE SAVED REGISTERS TO EXPECTED VALUES
4374 ;*AFTER GO HAS BEEN GI'EN TO RELEASE COMMAND
4375 023610 052737 000001 003202  BIS      #DMD,@#REINTO+26;SAVED RHMR
4376 023616 017737 156052 003216  MOV      @RHLA,@#REINTO+42;SAVED RHLA
4377
4378
4379 ;*SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN
4380 ;*BE DONE
4381 023624 004037 043312  JSR      RO,@#SAVER     ;SAVE
4382 023630 001632          RHW      ;FROM
4383 023632 002110          WRF      ;TO
4384 023634 000023          19.      ;NUMBER OF REGISTERS SAVED
4385
4386 ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
4387 ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
4388 ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
4389 023636 113737 003201 002135  MOV      @#REINTO+25,@#WRF+25;SAVE UPPER RHAS
4390
4391
4392 ;*COMPARE REGISTERS BEFORE RELEASE COMMAND
4393 ;*WITH AFTER GO
4394 023644 004037 043514  JSR      RO,@#COMPAR    ;COMPARE
4395 023650 003154          REINTO   ;GOOD BUFFER
4396 023652 002110          WRF      ;TEST BUFFER
4397 023654 000023          19.      ;NUMBER
4398 023656 023664          1$      ;RETURN FOR ERROR
4399 023660 023664          1$      ;SAME
4400 023662 023704          2$      ;RETURN FOR GOOD COMPARISON
4401
4402 023664 013705 047624          1$:  MOV      @#ERWORD,R5    ;GETTING REAYD TO INDEX
4403 023670 060505          ADD      R5,R5          ;DOUBLE ERROR WORD
4404 023672 016537 001630 042270  MOV      RHW-2(R5),@#REGADR ;FAILING REGISTER ADDRESS
4405 023700 104001          ERROR  1              ;IMPROPER REGISTER CHANGE
  
```



```
4411
4412
4413
4414 023704 000004          TST53: SCOPE
4415 023706 012706 001000    MOV      #STACK,SP      ;RESET STACK
4416 023712 012737 000053 002032    MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
4417 023720 004737 042620    JSR      PC,@#CLDISK    ;INIT DRIVE
4418 023724 012777 000001 155726    MOV      #DMD,@RHMR     ;SET DIAGNOSTIC MODE
4419 023732 004037 045172    JSR      RO,@#MAKECYL   ;SUBROUTINE TO GIVE A SEEK
4420 023736 000000          0                          ;CHANGE RHCC TO 0
4421
4422
4423 023740 000004          TST54: SCOPE
4424 023742 012706 001000    MOV      #STACK,SP      ;RESET STACK
4425 023746 012737 000054 002032    MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
4426 023754 004737 042620    JSR      PC,@#CLDISK    ;INIT AND SET UP GENERAL REGISTERS
4427
4428          ;*THESE ARE REGULAR SET UPS FOR SEARCH COMMAND
4429 023760 012777 000025 155656    MOV      #21.,@RHDS1    ;DESIRED SECTOR/TRACK REGISTER
4430          ;TRACK 0 SECTOR 21
4431 023766 005077 155660    CLR      @RHCA          ;DESIRED CYLINDER =0
4432 023772 012777 010000 155650    MOV      #FMT22,@RHOF   ;FORMAT BIT=1 (16 BITS PER WORD)
4433 024000 013711 002054    MOV      @#SERCH,@R1    ;FILL SEARCH COMMAND IN RHCS1
4434
4435          ;*NOW SAVE REGISTERS STARTING FROM RHWC IN WRITE FROM BUFFER
4436 024004 004037 043312    JSR      RO,@#SAVER     ;SAVE REGISTERS FOR COMPARISON
4437          ;AT THE END OF THE SEARCH
4438 024010 001632          RHWC                     ;START SAVING FROM RHWC
4439 024012 003154          REINTO                  ;SAVE INTO REINTO
4440 024014 000023          19.                     ;NUMBER OF REGISTERS SAVED
4441
4442 024016 004737 042654    JSR      PC,@#CHECKT    ;CHECK THAT DVA,RDY,DPR,DRY = 1
4443 024022 104401 062450    TYPE     ,CPHALT        ;AND THAT NO OTHERS = 1. CANNOT CON-
4444 024026 000000          HALT                    ;STOP THE TEST
4445
4446          ;*NOW THE DIAGNOSTIC MODE BIT WILL BE SET
4447          ;*AND THE SEARCH OPERATION STARTED
4448
4449 024030 005037 001200    CLR      @#STMP1        ;THIS WILL HAVE THE EXPECTED
4450          ;VALUE OF RHLA REGISTER
4451
4452 024034 013700 001660    MOV      @#RHMR,RO      ;NOW RO HAS MAINTENANCE REG. ADDR.
4453 024040 017703 155600    MOV      @RHDS1,R3      ;GET DESIRED SECTOR/TRACK REG.
4454 024044 042703 177400    BIC      #177400,R3     ;GET SECTOR ONLY
4455 024050 010337 053550    MOV      R3,@#SECTR     ;DUPLICATE SECTOR
4456 024054 012710 000001    MOV      #DMD,@RO       ;S
4457 024060 052777 000001 155552    BIS      #GO,@RHCS1     ;GO
4458 024066 052710 000010    BIS      #MSTCK,@RO     ;SET SECTOR CLOCK
4459 024072 042710 000010    BIC      #MSTCK,@RO     ;CLEAR SECTOR CLOCK
4460 024076 000240          NOP                      ;ALLOW TIME BETWEEN SECTOR CLOCKS
4461 024100 052710 000010    BIS      #MSTCK,@RO     ;SET SECTOR CLOCK
4462 024104 042710 000010    BIC      #MSTCK,@RO     ;CLEAR SECTOR CLOCK
4463 024110 000240          NOP                      ;ALLOW TIME BETWEEN SECTOR CLOCKS
4464 024112 052710 000014    BIS      #MINX!MSTCK,@RO ;SET INDEX AND SECTOR CLOCK
4465 024116 012710 000001    MOV      #DMD,@RO       ;RESET INDEX AND SECTOR CLOCK
4466 024122 005703          TST      R3              ;IF SECTOR REQUIRED JUMP OUT
```



```
4579 024504 042710 000012 BIC #MSTCK!MCLK,@R0 ;CLEAR SECTOR AND CLOCK
4580
4581 ;*NOW ALL REGISTERS WILL BE COMPARED
4582 ;*SO FILL EXPECTED VALUE INTO SAVED LOCATIONS
4583
4584 024510 052737 100000 003162 BIS #SC,@#REINTO+6 ;INCLUDE SC IN SAVED RHCS1
4585 024516 053737 002016 003200 BIS @#ATTENT,@#REINTO+24 ;FILL APPROPRIATE ATTENTION
4586 ;IN SAVED RHAS
4587 024524 052737 000001 003202 BIS #DMD,@#REINTO+26 ;SET DMD IN RHMR SAVED
4588 024532 052737 100000 003204 BIS #ATA,@#REINTO+30 ;SET ATA IN RHDS1 SAVED
4589 024540 013737 001200 003216 MOV @#$TMP1,@#REINTO+42 ;MOVE EXPECTED VALUE
4590 ;INTO RHLA SAVED
4591
4592 ;*AFTER SEARCH COMMAND
4593 ;*SAVE REGISTERS AGAIN SO THAT COMPARES CAN BE DONE
4594
4595 024546 004037 043312 JSR R0,@#SAVER ;SAVE
4596 024552 001632 RHCW ;FROM
4597 024554 002110 WRFROM ;TO
4598 024556 000023 19. ;NUMBER
4599
4600 ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
4601 ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
4602 ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
4603 024560 113737 003201 002135 MOVB @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS
4604
4605
4606 ;*COMPARE REGISTERS BEFORE SEARCH WITH AFTER
4607
4608 024566 004037 043514 JSR R0,@#COMPAR ;COMPAR
4609 024572 003154 REINTO ;GO BUFFER
4610
4611 024574 002110 WRFROM ;TEST BUFFER
4612 024576 000022 18. ;NUMBER
4613 024600 024606 13$ ;RETURN FOR ERROR
4614 024602 024606 13$ ;SAME
4615 024604 024626 14$ ;RETURN FOR GOOD COMPARISON
4616 024606 013705 047624 13$: MOV @#ERWORD,R5 ;GETTING READY TO INDEX
4617 024612 060505 ADD R5,R5 ;DOUBLE ERROR WORD
4618 024614 016537 001630 042270 MOV RHCW-2(R5),@#REGADR ;FAILING REG. ADDRESS
4619 024622 104001 ERROR 1 ;CONTENTS OF REGISTER
4620 024624 000207 RTS PC ;CHANGED AT END OF
4621 024626 14$: ;SEARCH
4622
```

4623								
4624	024626	000004			TST55:	SCOPE		
4625	024630	012706	001000		MOV	#STACK,SP		:RESET STACK
4626	024634	012737	000055	002032	MOV	#TTNO,@#TSTNM		:THIS SAVES TEST NUMBER
4627	024642	004737	042620		JSR	PC,@#CLDISK		:INIT DRIVE
4628	024646	012777	000001	155004	MOV	#DMD,@RHMR		:SET DIAGNOSTIC MODE
4629	024654	004037	045172		JSR	RO,@#MAKECYL		:SUBROUTINE TO GIVE A SEEK
4630	024660	000000			0			:CHANGE RHCC TO 0

```
4631
4632
4633
4634          .SBTTL  READ/WRITE ADDRESSING VIA RHMR
4635
4636
4637 024662 000004          TST56: SCOPE
4638 024664 012706 001000      MOV      #STACK,SP          ;RESET STACK
4639 024670 012737 000056 002032  MOV      #TTNO,@#TSTNM      ;THIS SAVES TEST NUMBER
4640
4641 024676 012700 051324          MOV      #SECGAP,RO          ;POINTER
4642 024702 012701 000460          MOV      #304.,R1           ;COUNTER
4643 024706 012720 177777          1$:  MOV      #-1,(RO)+        ;CLEAR 'DISK' AREA TO ALL ONES
4644 024712 005301          DEC      R1                 ;
4645 024714 001374          BNE     1$                 ;
4646 024716 004737 042620          JSR     PC,CLDISK          ;THIS IS USED TO SET UP GENERAL
4647                                     ;REGISTER CORRESPONDENCE
4648
4649                                     ;*THESE ARE TO SET UP FOR DISKLESS USE ONLY
4650
4651 024722 012737 010000 052642      MOV      #FMT22,@#WCYL      ;FORMAT 22=16 BIT WORDS AND
4652                                     ;CYLINDER 0
4653 024730 005037 052644          CLR     @#WSECTR           ;TRACK=0, SECTOR=0
4654 024734 005037 052646          CLR     @#WKEY1           ;KEY1=0
4655 024740 005037 052650          CLR     @#WKEY2           ;KEY2=0
4656 024744 012737 000400 052702      MOV      #256.,@#FNWORD    ;256 DATAWORDS
4657 024752 004537 044026          JSR     R5,@#CRC          ;GO TO CALCULATE CRC
4658 024756 052642          WCYL
4659 024760 052652          GCRC
4660
4661                                     ;*THESE ARE REGULAR SETUPS FOR RH11 & 'WRFROM' OUTPUT BUFFER
4662
4663 024762 012777 177374 154642      MOV      #-260.,@RHWC      ;256 DATA WORDS 4 HEADER WORDS
4664 024770 012700 002110          MOV      #WRFROM,RO        ;BUS ADDRESS TO BE
4665 024774 010077 154634          MOV      RO,@RHBA         ;BUFFER 'WRFROM'
4666 025000 012705 000403          MOV      #259.,R5         ;COUNTER
4667 025004 012720 010000          MOV      #FMT22,(RO)+     ;FORMAT =16 BIT WORD
4668
4669 025010 005020          2$:  CLR     (RO)+             ;SECTOR=0, TRACK=0,KEYS=0, ALL DATA=0
4670 025012 005305          DEC     R5                 ;& CYLINDER=0....SO CLEAR ALL 'WRFROM'
4671 025014 001375          BNE     2$                 ;CONTINUE IF ALL 259 NOT COMPLETE
4672 025016 005077 154622          CLR     @RH DST           ;TRACK=0, SECTOR=0
4673
4674
4675 025022 004737 042654          JSR     PC,@#CHECKT       ;CHECK THAT DVA,RDY,DPR,DRY = 1
4676 025026 104401 062450          TYPE   ,CPHALT           ;AND THAT NO OTHERS = 1. CANNOT CON-
4677 025032 000000          HALT                    ;STOP THE TEST
4678
4679 025034 013711 002064          MOV     @#WRIFOR,@R1      ;GET READY FOR WRITE HEADER
4680                                     ;AND DATA WITH 62 IN RHCS1
4681 025040 005037 002006          CLR     @#ERFLG$         ;CLEAR ERROR FLAG
4682 025044 012777 010000 154576      MOV     #FMT22,@RHOF      ;FORMAT BIT=1 16 BIT WORDS
4683 025052 005077 154574          CLR     @RHCA            ;CYLINDER 0
4684 025056 004737 052466          JSR     PC,@#COMWHD       ;WRITE HEADER AND DATA FROM 'WRFROM'
4685                                     ;INTO THE RHMR REGISTER AND BACK INTO
4686                                     ;CORE 'DISK' AREA
```

```
4687
4688
4689
4690
4691
4692
4693
4694
4695
4696
4697 025062 017737 154544 001126
4698 025070 001401
4699 025072 104040
4700
4701
4702 025074 005737 002006
4703 025100 001034
4704 025102 004737 043044
4705 025106 104401 062450
4706 025112 000000
4707 025114 005037 052422
4708 025120 005037 052424
4709
4710
4711
4712 025124 004037 042536
4713 025130 003154
4714 025132 004214
4715 025134 000000
4716 025136 005037 002006
4717
4718
4719
4720 025142 004037 043514
4721 025146 003154
4722 025150 051422
4723 025152 000421
4724 025154 025162
4725 025156 025166
4726 025160 025172
4727 025162 104007
4728 025164 000207
4729 025166 104010
4730
4731
4732
4733
4734 025170 000207
4735
```

5\$:

3\$:

4\$:

```

;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR
;*PRINT OUTS FROM THE 'COMWHD' ROUTINE THAT MEANS
;*ALL HEADER ON DISK IS GOOD IE. ONLY DATA IS
;*TO BE CHECKED TO SEE IF IT IS ZERO

;*AND WRITE DATA GAP AND TOLERANCE GAP TO SEE IF THEY
;*ARE ALL ZEROS (ECC1 AND ECC2 MAY NOT BE 0) AS WELL AS
;*CHECKING RHC FOR ZERO

MOV @RHWC,$BDDAT ;MOVE WORD COUNTER INTO BAD DATA
BEQ 5$ ;SHOULD HAVE COUNTED UP TO ZERO
ERROR 40 ;RHC DID NOT = 0 AFTER A WRITE
;HEADER AND DATA

TST @#ERFLG$ ;HAVE ANY ERRORS OCCURED?
BNE TST57 ;:BRANCH IF YES
JSR PC,@#CHECKE ;CHECK THAT BITS = 1
TYPE ,CPHALT ;CANNOT CONTINUE TESTING IF THEY DON'T
HALT ;STOP THE TEST
CLR @#WECC1 ;CLEAR ECC
CLR @#WECC2

;*REINTO BUFFER IS FILLED WITH EXPECTED DATA OF ALL 0'S

JSR R0,@#CLAREA ;CLEAR 'REINTO'
REINTO ;FROM
REINTO+<272.*2> ;TO
.WORD 0 ;FILL WITH ZEROS
CLR @#ERFLG$ ;CLEAR ERROR FLAG

;*COMPARE 'REINTO' BUFFER WITH 'DISK' BUFFER
JSR R0,@#COMPAR ;CHECK
REINTO ;GOOD BUFFER
DISK ;TEST BUFFER
273. ;NUMBER OF WORDS CHECKED
3$ ;RETURN POINT FOR ERROR HEADER
4$ ;RETURN POINT FOR ERROR DATA
TST57 ;RETURN FOR GOOD COMPARISON
ERROR 7 ;READ ERROR10 NEXT
RTS PC ;RETURN TO 'COMPAR'
ERROR 10 ;WORD NOS 1 TO 256 ARE
;DATA WORDS
;257 AND 258 ARE ECC
;ZEROED OUT
;259 TO 273 TOLERANCE GAP

RTS PC
```

```

4736
4737 025172 000004          TST57: SCOPE
4738 025174 012706 001000  MOV      #STACK,SP      ;RESET STACK
4739 025200 012737 000057 002032  MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
4740
4741 025206 012700 051324          MOV      #SECGAP,R0     ;POINTER
4742 025212 012701 000460          MOV      #304.,R1      ;COUNTER
4743 025216 005020          1$: CLR      (R0)+         ;CLEAR SIMULATED 'DISK' AREA IN CORE
4744 025220 005301          DEC      R1             ;
4745 025222 001375          BNE     1$             ;
4746 025224 004737 042620          JSR     PC,@#CLDISK    ;THIS IS USED TO SET GENERAL REGISTERS
4747
4748          ;*THESE ARE TO BE SETUP FOR DISKLESS USE ONLY
4749
4750 025230 012737 010000 052642  MOV      #FMT22,@#WCYL  ;FORMAT 22 = 16 BIT WORDS AND
4751          ;CYLINDER 0
4752 025236 012737 000001 052644  MOV      #1,@#WSECTR    ;TRACK=0, SECTOR=1
4753 025244 005037 052646          CLR     @#WKEY1        ;KEY1=0
4754 025250 005037 052650          CLR     @#WKEY2        ;KEY2=0
4755 025254 012737 000400 052702  MOV      #256.,@#FNWORD ;256 DATA WORDS
4756 025262 004537 044026          JSR     R5,@#CRC       ;GO TO CALCULATE CRC
4757 025266 052642          WCYL
4758 025270 052652          GCRC
4759
4760          ;*THESE ARE REGULAR SETUPS FOR THE RH11 AND 'WRFROM' BUFFER
4761
4762 025272 012777 177374 154332  MOV      #-260.,@RHWC   ;256 DATA WORDS 4 HEADER WORDS
4763 025300 012700 002110          MOV     #WRFROM,R0     ;THESE TWO INSTRUCTIONS GETS
4764 025304 010077 154324          MOV     R0,@RHBA       ;ADDR. OF WRFROM INTO R0 AND
4765          ;BUS ADDRESS REGISTER
4766 025310 012720 010000          MOV     #FMT22,(R0)+   ;FORMAT=16 BIT WORDS
4767          ;CYLINDER=0
4768 025314 012720 000001          2$: MOV     #1,(R0)+     ;TRACK=0, SECTOR=1, KEYS=0
4769 025320 005020          CLR     (R0)+         ;KEY1=0
4770 025322 005020          CLR     (R0)+         ;KEY2=0
4771 025324 012705 000400          MOV     #256.,R5      ;COUNTER
4772 025330 012720 177777          3$: MOV     #-1,(R0)+   ;MOVE ALL ONES FOR DATA
4773 025334 005305          DEC     R5             ;
4774 025336 001374          BNE     3$            ;BRANCH IF DATA NOT COMPLETE
4775 025340 012777 000001 154276  MOV     #1,@RHDST      ;TRACK=0 SECTOR=1
4776
4777 025346 004737 042654          JSR     PC,@#CHECKT    ;CHECK THAT DVA,RDY,DPR,DRY = 1
4778 025352 104401 062450          TYPE   ,CPHALT        ;AND THAT NO OTHERS = 1. CANNOT CON-
4779 025356 000000          HALT                   ;STOP THE TEST
4780
4781 025360 013711 002064          MOV     @#WRIFOR,@R1   ;GET READY FOR WRITE HEADER AND
4782          ;DATA WITH 62 IN RHCS1
4783 025364 005037 002006          CLR     @#ERFLG$      ;CLEAR ERROR FLAG
4784 025370 012777 010000 154252  MOV     #FMT22,@RHOF   ;FORMAT BIT=1 (16 BIT WORDS)
4785 025376 005077 154250          CLR     @RHCA         ;CYLINDER =0
4786 025402 004737 052466          JSR     PC,@#COMWHD    ;WRITE HEADER AND DATA INTO 'DISK' AREA
4787
4788          ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
4789          ;*FROM THE 'COMWHD' ROUTINE THAT MEANS ALL THE HEADER ON 'DISK'
4790          ;*IS GOOD IE. ONLY THE DATA IS TO BE CHECKED TO SEE IF IT IS
4791          ;*ALL ONES AND WRITE DATA GAP AND TOLERANCE GAP TO SEE IF
    
```



```
4792 ;*THEY ARE ALL ZEROS, - ECC1 AND ECC2 ARE NOT CHECKED.
4793
4794 ;*RHC IS CHECKED TO BE = 0
4795
4796 025406 017737 154220 001126 MOV @RHC,$BDDAT ;LOAD WORD COUNTER JUST IN CASE
4797 025414 001401 BEQ 6$ ;SHOULD BE = 0
4798 025416 104040 ERROR 40 ;RHC DOES NOT = 0 AFTER A WRITE
4799 ;HEADER AND DATA IS COMPLETED
4800
4801 025420 005737 002006 6$: TST @#ERFLG$ ;HAVE ANY ERRORS OCCURRED?
4802 025424 001041 BNE TST60 ;:BRANCH IF YES
4803 025426 004737 043044 JSR PC,@#CHECKE ;CHECK THAT BITS = 1
4804 025432 104401 062450 TYPE ,CPHALT ;CANNOT CONTINUE TESTING IF THEY DON'T
4805 025436 000000 HALT ;STOP THE TEST
4806 025440 005037 052422 CLR @#WECC1 ;CLEAR ECC
4807 025444 005037 052424 CLR @#WECC2 ;CLEAR ECC
4808
4809 ;*FILL 'REINTO' BUFFER WITH EXPECTED DATA OF ALL 1'S
4810
4811 025450 004037 042536 JSR R0,@#CLAREA ;FILL REINTO BUFFER
4812 025454 003154 REINTO ;FROM
4813 025456 004152 REINTO+<255.*2> ;TO
4814 025460 177777 .WORD -1 ;DATA
4815 025462 004037 042536 JSR R0,@#CLAREA ;FILL REST
4816 025466 004154 REINTO+<256.*2> ;FROM
4817 025470 004214 REINTO+<272.*2> ;TO
4818 025472 000000 0 ;DATA
4819
4820 025474 005037 002006 CLR @#ERFLG$ ;CLEAR ERROR FLAG
4821
4822 ;*NOW COMPARE 'DISK' BUFFER WITH 'REINTO' BUFFER IN CORE
4823
4824 025500 004037 043514 JSR R0,@#COMPAR ;CHECK
4825 025504 003154 REINTO ;GOOD BUFFER
4826 025506 051422 DISK ;TEST BUFFER
4827 025510 000421 273. ;NUMBER OF WORDS CHECKED
4828 025512 025520 4$ ;RETURN POINT FOR ERROR HEADER
4829 025514 025524 5$ ;RETURN POINT FOR ERROR DATA
4830 025516 025530 TST60 ;RETURN FOR GOOD COMPARISON
4831 025520 104007 4$: ERROR 7 ;READ ERROR 10 NEXT
4832 025522 000207 RTS PC ;RETURN TO COMPARE
4833 025524 104010 5$: ERROR 10 ;WORD NOS 1 TO 256 ARE
4834 ;DATA WORDS
4835 ;WORD NOS 257 AND 258
4836 ;ARE ECC WHICH HAVE BEEN
4837 ;ZEROED
4838 ;WORD NOS 259
4839 ;IS DATA GAP
4840 ;WORD NOS 260 TO 273
4841 ;ARE TOLERANCE GAP
4842 025526 000207 RTS PC ;RETURN TO COMPARE
4843
4844
4845
```

```
4846
4847 025530 000004
4848 025532 012706 001000
4849 025536 012737 000060 002032
4850
4851 025544 012700 051324
4852 025550 012701 000460
4853 025554 012720 000377
4854 025560 005301
4855 025562 001374
4856 025564 004737 042620
4857
4858
4859
4860
4861 025570 012737 010000 052642
4862
4863 025576 012737 000401 052644
4864 025604 005037 052646
4865 025610 005037 052650
4866 025614 012737 000400 052702
4867 025622 004537 044026
4868 025626 052642
4869 025630 052652
4870
4871
4872
4873 025632 012777 177374 153772
4874 025640 012700 002110
4875
4876 025644 010077 153764
4877
4878 025650 012720 010000
4879
4880 025654 012720 000401
4881 025660 005020
4882 025662 005020
4883 025664 012705 000400
4884 025670 012720 052525
4885 025674 005305
4886 025676 001374
4887 025700 012777 000401 153736
4888
4889 025706 004737 042654
4890 025712 104401 062450
4891 025716 000000
4892
4893 025720 013711 002064
4894
4895 025724 005037 002006
4896 025730 012777 010000 153712
4897 025736 005077 153710
4898 025742 004737 052466
4899
4900
4901
```

TST60: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
MOV #SECGAP,R0 ;POINTER
MOV #304,R1 ;COUNTER
1\$: MOV #377,(R0)+ ;LOAD SIMULATED 'DISK' AREA WITH 377
DEC R1 ;
BNE 1\$;
JSR PC,CLDISK ;THIS IS USED TO SET UP GENERAL
;REGISTER CORRESPONDENCE
;*THESE ARE TO BE SETUP FOR DISKLESS USE ONLY
MOV #FMT22,@#WCYL ;FORMAT 22=16 BIT WORDS AND
;CYLINDER 0
MOV #401,@#WSECTR ;TRACK=1, SECTOR=1
CLR @#WKEY1 ;KEY1=0
CLR @#WKEY2 ;KEY2=0
MOV #256,@#FNWORD ;256 DATA WORDS
JSR R5,@#CRC ;GO TO CALCULATE CRC
WCYL
GCRC
;*THESE ARE REGULAR SETUPS FOR RH11 AND 'WRFROM' BUFFER
MOV #-260,@#RHWC ;256 DATA WORDS 4 HEADER WORDS
MOV #WRFROM,R0 ;THESE TWO INSTRUCTIONS GET
;ADDR. OF WRFROM INTO R0
;AND BUS ADDRESS REGISTER
MOV R0,@#RHBA
MOV #FMT22,(R0)+ ;FORMAT=16 BIT WORDS
;CYLINDER=0
2\$: MOV #401,(R0)+ ;TRACK=1, SECTOR=1, KEYS=0
CLR (R0)+ ;KEY1=0
CLR (R0)+ ;KEY2=0
MOV #256,R5 ;COUNTER
3\$: MOV #052525,(R0)+ ;MOVE ALTERNATE ONES FOR DATA
DEC R5 ;COUNT
BNE 3\$;BRANCH IF DATA NOT COMPLETE
MOV #401,@#RHDST ;TRACK=1 SECTOR=1
JSR PC,@#CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-
HALT ;STOP THE TEST
MOV @#WRIFOR,@R1 ;GET READY FOR WRITE HEADER
;AND DATA WITH 62 IN RHCS1
CLR @#ERFLG\$;CLEAR ERROR FLAG
MOV #FMT22,@#RHOF ;FORMAT BIT=1(16 BIT WORDS
CLR @#RHCA ;CYLINDER=0
JSR PC,@#COMWHD ;WRITE HEADER AND DATA INTO 'DISK' CORE
;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
;*FROM THE 'COMWHD' ROUTINE THAT MEANS ALL HEADER IN

```
4902      ;*'DISK' IS GOOD. DATA IS TO BE CHECKED TO SEE
4903      ;*IF IT IS ALL 052525 AND WRITE DATA GAP AND
4904      ;*TOLERANCE GAP TO SEE IF THEY ARE ALL ZEROS.
4905
4906      ;*RHCW IS CHECKED TO BE = 0
4907
4908 025746 017737 153660 001126  MOV @RHCW,$BDDAT ;LOAD AND TEST FOR ZERO
4909 025754 001401  BEQ 6$ ;RHCW SHOULD = 0
4910 025756 104040  ERROR 40 ;RHCW DID NOT = 0 AFTER A WRITE
4911      ;HEADER AND DATA WAS COMPLETED
4912
4913      ;*ONLY ECC1 AND ECC2 ARE NOT CHECKED
4914
4915 025760 005737 002006      6$: TST @#ERFLG$ ;HAVE ANY ERRORS OCCURED?
4916 025764 001041  BNE TST61 ;:BRANCH IF YES
4917 025766 004737 043044  JSR PC,@#CHECKE ;CHECK THAT BITS = 1
4918 025772 104401 062450  TYPE ,CPHALT ;CANNOT CONTINUE TESTING IF THEY DON'T
4919 025776 000000  HALT ;STOP THE TEST
4920 026000 005037 052422  CLR @#WECC1 ;CLEAR ECC
4921 026004 005037 052424  CLR @#WECC2 ;CLEAR ECC
4922
4923      ;*FILL 'REINTO' BUFFER WITH EXPECTED DATA
4924
4925 026010 004037 042536  JSR R0,@#CLAREA ;FILL REINTO BUFFER
4926 026014 003154  REINTO ;FROM
4927 026016 004152  REINTO+<255.*2> ;TO
4928 026020 052525  .WORD 52525 ;DATA
4929 026022 004037 042536  JSR R0,@#CLAREA ;FILL REST
4930 026026 004154  REINTO+<256.*2> ;FROM
4931 026030 004214  REINTO+<272.*2> ;TO
4932 026032 000000  .WORD 0 ;DATA
4933 026034 005037 002006  CLR @#ERFLG$ ;CLEAR ERROR FLAG
4934
4935      ;*NOW COMPARE 'DISK' BUFFER WITH 'REINTO' BUFFER IN CORE
4936
4937 026040 004037 043514  JSR R0,@#COMPAR ;CHECK
4938 026044 003154  REINTO ;GOOD BUFFER
4939 026046 051422  DISK ;TEST BUFFER
4940 026050 000421  273. ;NUMBER OF WORDS CHECKED
4941 026052 026060  4$ ;RETURN POINT FOR ERROR HEADER
4942 026054 026064  5$ ;RETURN POINT FOR ERROR DATA
4943 026056 026070  TST61 ;RETURN FOR GOOD COMPARISON
4944 026060 104007  ERROR 7 ;READ ERROR 10 NEXT
4945 026062 000207  RTS PC ;RETURN TO COMPARE
4946 026064 104010  ERROR 10 ;WORD NOS 1 TO 256 ARE
4947      ;DATA WORDS
4948      ;WORD NOS 257 AND 258
4949      ;ARE ECC WHICH HAVE BEEN
4950      ;ZEROED
4951      ;WORD NOS 259
4952      ;IS DATA GAP
4953      ;WORD NOS 260 TO 273
4954      ;ARE TOLERANCE GAP
4955 026066 000207  RTS PC ;RETURN TO COMPARE
4956
4957
```



```
4960
4961 026070 000004
4962 026072 012706 001000
4963 026076 012737 000061 002032
4964
4965 026104 012700 051324
4966 026110 012701 000460
4967 026114 012720 177777
4968 026120 005301
4969 026122 001374
4970 026124 004737 042620
4971
4972
4973
4974
4975 026130 012737 010000 052642
4976
4977 026136 005037 052644
4978 026142 005037 052646
4979 026146 005037 052650
4980 026152 012737 000400 052702
4981 026160 004537 044026
4982 026164 052642
4983 026166 052652
4984
4985
4986
4987 026170 012777 177374 153434
4988 026176 012700 002110
4989 026202 010077 153426
4990 026206 012705 000403
4991 026210 012720 010000
4992
4993 026216 005020
4994 026220 005305
4995 026222 001375
4996 026224 005077 153414
4997
4998 026230 004737 042654
4999 026234 104401 062450
5000 026240 000000
5001
5002 026242 013711 002064
5003
5004 026246 005037 002006
5005 026252 012777 010000 153370
5006 026260 005077 153366
5007 026264 012777 000001 153366
5008 026272 052777 000001 153340
5009 026300 000240
5010 026302 052777 000001 153330
5011
5012 026310 004737 042224
5013 026314 032737 002000 001714
5014 026322 001404
5015 026324 013737 001636 001122

TST61: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER

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

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

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

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

MOV @#WRIFOR,@R1 ;GET READY FOR WRITE HEADER
;AND DATA WITH 62 IN RHCS1
CLR @#ERFLG$ ;CLEAR ERROR FLAG
MOV #FMT22,@RHOF ;FORMAT BIT=1 16 BIT WORDS
CLR @RHCA ;CYLINDER 0
MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE
BIS #GO,@RHCS1 ;GO
NOP
BIS #GO,@RHCS1 ;THIS GO SHOULD SET PGE

JSR PC,@#PUTREG ;SAVE REGISTERS
BIT #PGE,@#CS1 ;IS PGE SET
BEQ 3$ ;BRANCH IF GOOD
MOV @#RHCS2,@#$BDADR
```


5019
5020 :** THESE TESTS ARE THROUGH THE MAINTAINABILITY REGISTER - RHMR
5021
5022 :** THE SECTOR GAP AND SYNC BYTE ARE ALWAYS READ AS
5023 :** ZEROS AND 144000 NO MATTER WHAT IS IN THE SIMULATED DISK AREA
5024 :** TAGGED SECGAP: AND WSSYNC:
5025
5026 :** THE HEADER CONSISTING OF CYLINDER ADDRESS, SECTOR,
5027 :** TRACK AND THE KEYS ARE READ FROM LOCATION
5028 :** CYL:, SECTOR:, KEY1:, AND KEY2 AND NOT FROM
5029 :** HEADER: ON SIMULATED DISK
5030
5031 :** CRC IS READ FROM SIMULATED DISK LOCATION WCRC:
5032 :** HEADER GAP IS ALWAYS READ AS ZEROS NO MATTER
5033 :** WHAT IS ON THE SIMULATED DISK AREA
5034
5035 :** THE DATA SYNC IS READ FROM HDWSYN:
5036 :** ON SIMULATED DISK
5037
5038 :** ALL DATA IS READ FROM SIMULATED DISK DISK:

55
55
55
55
55
55
55
55
55
55
55

```
5039
5040
5041
5042
5043 026334 000004          TST62: SCOPE
5044 026336 012706 001000    MOV      #STACK,SP      ;RESET STACK
5045 026342 012737 000062 002032  MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
5046
5047 026350 012746 000000          MOV      #0,-(SP)       ;DATA TO BE READ
5048 026354 012705 000400          MOV      #256.,R5      ;COUNTER
5049 026360 012700 051422          MOV      #DISK,R0      ;START OF SIMULATED DISK DATA
5050 026364 011620          1$: MOV      (SP),(R0)+    ;MOVE IN DATA ON TO SIMULATED DISK
5051 026366 005305          DEC      R5             ;COUNT
5052 026370 001375          BNE     1$             ;BRANCH IF 256 NOT COMPLETE
5053 026372 005726          TST     (SP)+         ;UNDO -(SP)
5054 026374 012705 000021          MOV      #17.,R5      ;2 ECC WORDS
5055 026400 005020          2$: CLR      (R0)+      ;CLEAR ECC, DATA GAP, AND
5056 026402 005305          DEC      R5             ;TOLERANCE GAP
5057 026404 001375          BNE     2$             ;BRANCH IF NOT COMPLETE
5058 026406 012737 010000 047504  MOV      #0!FMT22,@#CYL ;16 BITS PER WORD
5059 026414 112737 000000 047507  MOVVB   #0,@#SECOTR+1  ;TRACK 0
5060 026422 112737 000000 047506  MOVVB   #0,@#SECOTR   ;SECTOR 0
5061 026430 012737 000000 047510  MOV      #0,@#KEY1    ;KEY1=0
5062 026436 012737 000000 047512  MOV      #0,@#KEY2    ;KEY2=0
5063 026444 012737 000400 047564  MOV      #256.,@#DAWORD ;NO. OF DATA WORDS
5064 026452 005037 047514          CLR     @#X           ;THIS IS A READ COMMAND
5065 026456 004537 044026          JSR     R5,@#CRC      ;GO TO CALCULATE CRC
5066 026462 047504          CYL
5067 026464 051404          WCRC
5068 026466 004737 042620          JSR     PC,@#CLDISK   ;SETUP GENERAL REGISTERS
5069 026472 012777 177374 153132  MOV      #-256.-4.,@RHWC ;256. DATA 4 HEADER WORDS
5070 026500 012777 003154 153126  MOV      #REINTO,@RHBA ;STARTING ADDRESS OF READ BUFFER
5071 026506 112746 000000          MOVVB   #0,-(SP)     ;IN LOWER BYTE GET SECTOR
5072 026512 112766 000000 000001  MOVVB   #0,1(SP)     ;GET TRACK IN HIGHER BYTE
5073 026520 012677 153120          MOV      (SP)+,@RHDST ;TRACK/SECTOR IN RHDST
5074 026524 012777 014000 153116  MOV      #FMT22!ECI,@RHOF ;16 BITS PER WORD
5075 026532 005077 153114          CLR     @RHCA        ;CYLINDER 0
5076 026536 004737 042654          JSR     PC,@#CHECKT   ;CHECK THAT DVA,RDY,DPR,DRY = 1
5077 026542 104401 062450          TYPE   ,CPHALT      ;AND THAT NO OTHERS = 1. CANNOT CON-
5078 026546 000000          HALT                ;STOP THE TEST
5079 026550 013711 002070          MOV      @#REFOR,@R1  ;READ HEADER AND DATA=72
5080 026554 005037 002006          CLR     @#ERFLG$     ;CLEAR ERROR FLAG
5081 026560 004737 047344          JSR     PC,@#COMHD   ;READ HEADER AND DATA
5082 026564 017737 153042 001126  MOV      @RHWC,$BDDAT ;LOAD AND TEST RHWC
5083 026572 001401          BEQ     20$         ;SHOULD = 0
5084 026574 104040          ERROR  40           ;RHWC DOES NOT = 0 AFTER A READ
5085 026576 005737 002006          20$: TST     @#ERFLG$   ;ANY ERRORS ALREADY THERE
5086 026602 001046          BNE     TST63       ;BRANCH IF YES
5087 026604 004737 043044          JSR     PC,@#CHECKE  ;CHECK THAT BITS = 1
5088 026610 104401 062450          TYPE   ,CPHALT      ;CANNOT CONTINUE TESTING IF THEY DON'T
5089 026614 000000          HALT                ;STOP THE TEST
5090 026616 012700 002110          MOV      #WRFROM,R0  ;GETTING READY TO FILL EXPECTED DATA
5091 026622 012720 010000          MOV      #0!FMT22,(R0)+ ;CYLINDER 0
5092 026626 112746 000000          MOVVB   #0,-(SP)     ;IN LOWER BYTE GET SECTOR
5093 026632 112766 000000 000001  MOVVB   #0,1(SP)     ;GET TRACK IN HIGHER BYTE
5094 026640 012620          MOV      (SP)+,(R0)+ ;GET TRACK/SECTOR IN BUFFER
```


5095	026642	012720	000000		MOV	#0,(R0)+		;KEY1 IN BUFFER
5096	026646	012720	000000		MOV	#0,(R0)+		;KEY2 IN BUFFER
5097	026652	012701	000400		MOV	#256.,R1		;DATA WORD COUNTER
5098	026656	012702	000000		MOV	#0,R2		;DATA
5099	026662	010220		3\$:	MOV	R2,(R0)+		;DATA INTO BUFFER
5100	026664	005301			DEC	R1		;COUNT
5101	026666	001375			BNE	3\$;BRANCH IF 256 NOT DONE
5102	026670	004037	043514		JSR	R0,@#COMPAR		;CHECK
5103	026674	002110			WRFROM			;GOOD BUFFER
5104	026676	003154			REINTO			;TEST BUFFER
5105	026700	000404			4+256.			;NUMBER OF WORDS CHECKED
5106	026702	026710			4\$;RETURN POINT FOR ERROR HEADER
5107	026704	026714			5\$;RETURN POINT FOR ERROR DATA
5108	026706	026720			TST63			;RETURN FOR GOOD COMPARISON
5109	026710	104004		4\$:	ERROR	4		;READ NEXT ERROR
5110	026712	000207			RTS	PC		;RETURN TO "COMPAR"
5111	026714	104005		5\$:	ERROR	5		;WORD NOS 1 TO 4 ARE
5112	026716	000207			RTS	PC		;RETURN TO "COMPAR"
5113								
5114								
5115								
5116								

```

5117
5118 026720 000004          TST63: SCOPE
5119 026722 012706 001000  MOV      #STACK,SP      ;RESET STACK
5120 026726 012737 000063 002032  MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
5121
5122 026734 012746 177777          MOV      #-1,-(SP)      ;DATA TO BE READ
5123 026740 012705 000400          MOV      #256.,R5      ;COUNTER
5124 026744 012700 051422          MOV      #DISK,R0      ;START OF SIMULATED DISK DATA
5125 026750 011620          1$: MOV      (SP),(R0)+    ;MOVE IN DATA ON TO SIMULATED DISK
5126 026752 005305          DEC      R5            ;COUNT
5127 026754 001375          BNE      1$           ;BRANCH IF 256 NOT COMPLETE
5128 026756 005726          TST      (SP)+        ;UNDO -(SP)
5129 026760 012705 000021          MOV      #17.,R5      ;2 ECC WORDS
5130 026764 005020          2$: CLR      (R0)+      ;CLEAR ECC, DATA GAP, AND
5131 026766 005305          DEC      R5            ;TOLERANCE GAP
5132 026770 001375          BNE      2$           ;BRANCH IF NOT COMPLETE
5133 026772 012737 010000 047504  MOV      #0!FMT22,@#CYL ;16 BITS PER WORD
5134 027000 112737 000000 047507  MOVVB   #0,@#SECOTR+1  ;TRACK 0
5135 027006 112737 000001 047506  MOVVB   #1,@#SECOTR   ;SECTOR 1
5136 027014 012737 000000 047510  MOV      #0,@#KEY1    ;KEY1=0
5137 027022 012737 000000 047512  MOV      #0,@#KEY2    ;KEY2=0
5138 027030 012737 000400 047564  MOV      #256.,@#DAWORD ;NO. OF DATA WORDS
5139 027036 005037 047514          CLR      @#X          ;THIS IS A READ COMMAND
5140 027042 004537 044026          JSR      R5,@#CRC     ;GO TO CALCULATE CRC
5141 027046 047504          CYL
5142 027050 051404          WCRC
5143 027052 004737 042620          JSR      PC,@#CLDISK  ;SETUP GENERAL REGISTERS
5144 027056 012777 177374 152546  MOV      #-256.-4.,@RHWC ;256. DATA 4 HEADER WORDS
5145 027064 012777 003154 152542  MOV      #REINTO,@RHBA ;STARTING ADDRESS OF READ BUFFER
5146 027072 112746 000001          MOVVB   #1,-(SP)      ;IN LOWER BYTE GET SECTOR
5147 027076 112766 000000 000001  MOVVB   #0,1(SP)      ;GET TRACK IN HIGHER BYTE
5148 027104 012677 152534          MOV      (SP)+,@RHDST ;TRACK/SECTOR IN RHDST
5149 027110 012777 014000 152532  MOV      #FMT22!ECI,@RHOF ;16 BITS PER WORD
5150 027116 005077 152530          CLR      @RHCA        ;CYLINDER 0
5151 027122 004737 042654          JSR      PC,@#CHECKT  ;CHECK THAT DVA,RDY,DPR,DRY = 1
5152 027126 104401 062450          TYPE    ,CPHALT      ;AND THAT NO OTHERS = 1. CANNOT CON-
5153 027132 000000          HALT                ;STOP THE TEST
5154 027134 013711 002070          MOV      @#REFOR,@R1  ;READ HEADER AND DATA=72
5155 027140 005037 002006          CLR      @#ERFLG$    ;CLEAR ERROR FLAG
5156 027144 004737 047344          JSR      PC,@#COMHD   ;READ HEADER AND DATA
5157 027150 017737 152456 001126  MOV      @RHWC,$BDDAT ;LOAD AND TEST RHWC
5158 027156 001401          BEQ     20$          ;SHOULD = 0
5159 027160 104040          ERROR  40           ;RHWC DOES NOT = 0 AFTER A READ
5160 027162 005737 002006          20$: TST      @#ERFLG$   ;ANY ERRORS ALREADY THERE
5161 027166 001046          BNE     TST64        ;BRANCH IF YES
5162 027170 004737 043044          JSR      PC,@#CHECKE  ;CHECK THAT BITS = 1
5163 027174 104401 062450          TYPE    ,CPHALT      ;CANNOT CONTINUE TESTING IF THEY DON'T
5164 027200 000000          HALT                ;STOP THE TEST
5165 027202 012700 002110          MOV      #WRFROM,R0   ;GETTING READY TO FILL EXPECTED DATA
5166 027206 012720 010000          MOV      #0!FMT22,(R0)+ ;CYLINDER 0
5167 027212 112746 000001          MOVVB   #1,-(SP)      ;IN LOWER BYTE GET SECTOR
5168 027216 112766 000000 000001  MOVVB   #0,1(SP)      ;GET TRACK IN HIGHER BYTE
5169 027224 012620          MOV      (SP)+,(R0)+  ;GET TRACK/SECTOR IN BUFFER
5170 027226 012720 000000          MOV      #0,(R0)+    ;KEY1 IN BUFFER
5171 027232 012720 000000          MOV      #0,(R0)+    ;KEY2 IN BUFFER
5172 027236 012701 000400          MOV      #256.,R1    ;DATA WORD COUNTER
    
```

5173	027242	012702	177777		MOV	#-1,R2		:DATA
5174	027246	010220		3\$:	MOV	R2,(R0)+		:DATA INTO BUFFER
5175	027250	005301			DEC	R1		:COUNT
5176	027252	001375			BNE	3\$:BRANCH IF 256 NOT DONE
5177	027254	004037	043514		JSR	R0,@#COMPAR		:CHECK
5178	027260	002110			WRFROM			:GOOD BUFFER
5179	027262	003154			REINTO			:TEST BUFFER
5180	027264	000404			4+256.			:NUMBER OF WORDS CHECKED
5181	027266	027274			4\$:RETURN POINT FOR ERROR HEADER
5182	027270	027300			5\$:RETURN POINT FOR ERROR DATA
5183	027272	027304			TST64			:RETURN FOR GOOD COMPARISON
5184	027274	104004		4\$:	ERROR	4		:READ NEXT ERROR
5185	027276	000207			RTS	PC		:RETURN TO 'COMPAR'
5186	027300	104005		5\$:	ERROR	5		:WORD NOS 1 TO 4 ARE
5187	027302	000207			RTS	PC		:RETURN TO 'COMPAR'
5188								
5189								
5190								

```

5191
5192 027304 000004          TST64: SCOPE
5193 027306 012706 001000  MOV      #STACK,SP      ;RESET STACK
5194 027312 012737 000064 002032  MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
5195
5196 027320 012746 052525          MOV      #052525,-(SP)  ;DATA TO BE READ
5197 027324 012705 000400          MOV      #256.,R5      ;COUNTER
5198 027330 012700 051422          MOV      #DISK,R0      ;START OF SIMULATED DISK DATA
5199 027334 011620          1$: MOV      (SP),(R0)+    ;MOVE IN DATA ON TO SIMULATED DISK
5200 027336 005305          DEC      R5            ;COUNT
5201 027340 001375          BNE     1$            ;BRANCH IF 256 NOT COMPLETE
5202 027342 005726          TST     (SP)+        ;UNDO -(SP)
5203 027344 012705 000021          MOV      #17.,R5      ;2 ECC WORDS
5204 027350 005020          2$: CLR      (R0)+      ;CLEAR ECC, DATA GAP, AND
5205 027352 005305          DEC      R5            ;TOLERANCE GAP
5206 027354 001375          BNE     2$            ;BRANCH IF NOT COMPLETE
5207 027356 012737 010000 047504  MOV      #0!FMT22,@#CYL ;16 BITS PER WORD
5208 027364 112737 000001 047507  MOVB    #1,@#SECOTR+1  ;TRACK 1
5209 027372 112737 000001 047506  MOVB    #1,@#SECOTR    ;SECTOR 1
5210 027400 012737 000000 047510  MOV      #0,@#KEY1     ;KEY1=0
5211 027406 012737 000000 047512  MOV      #0,@#KEY2     ;KEY2=0
5212 027414 012737 000400 047564  MOV      #256.,@#DAWORD ;NO. OF DATA WORDS
5213 027422 005037 047514          CLR     @#X           ;THIS IS A READ COMMAND
5214 027426 004537 044026          JSR     R5,@#CRC      ;GO TO CALCULATE CRC
5215 027432 047504          CYL
5216 027434 051404          WCRC
5217 027436 004737 042620          JSR     PC,@#CLDISK   ;SETUP GENERAL REGISTERS
5218 027442 012777 177374 152162  MOV      #-256.-4.,@RHWC ;256. DATA 4 HEADER WORDS
5219 027450 012777 003154 152156  MOV      #REINTO,@RHBA ;STARTING ADDRESS OF READ BUFFER
5220 027456 112746 000001          MOVB    #1,-(SP)      ;IN LOWER BYTE GET SECTOR
5221 027462 112766 000001 000001  MOVB    #1,1(SP)      ;GET TRACK IN HIGHER BYTE
5222 027470 012677 152150          MOV      (SP)+,@RHDST  ;TRACK/SECTOR IN RHDST
5223 027474 012777 014000 152146  MOV      #FMT22!ECI,@RHOF ;16 BITS PER WORD
5224 027502 005077 152144          CLR     @RHCA        ;CYLINDER 0
5225 027506 004737 042654          JSR     PC,@#CHECKT   ;CHECK THAT DVA,RDY,DPR,DRY = 1
5226 027512 104401 062450          TYPE   ,CPHALT      ;AND THAT NO OTHERS = 1. CANNOT CON-
5227 027516 000000          HALT                ;STOP THE TEST
5228 027520 013711 002070          MOV      @#REFOR,@R1  ;READ HEADER AND DATA=72
5229 027524 005037 002006          CLR     @#ERFLG$     ;CLEAR ERROR FLAG
5230 027530 004737 047344          JSR     PC,@#COMHD    ;READ HEADER AND DATA
5231 027534 017737 152072 001126  MOV      @RHWC,$BDDAT ;LOAD AND TEST RHWC
5232 027542 001401          BEQ     20$          ;SHOULD = 0
5233 027544 104040          ERROR  40           ;RHWC DOES NOT = 0 AFTER A READ
5234 027546 005737 002006          20$: TST     @#ERFLG$    ;ANY ERRORS ALREADY THERE
5235 027552 001046          BNE     TST65        ;BRANCH IF YES
5236 027554 004737 043044          JSR     PC,@#CHECKE   ;CHECK THAT BITS = 1
5237 027560 104401 062450          TYPE   ,CPHALT      ;CANNOT CONTINUE TESTING IF THEY DON'T
5238 027564 000000          HALT                ;STOP THE TEST
5239 027566 012700 002110          MOV      #WRFROM,R0   ;GETTING READY TO FILL EXPECTED DATA
5240 027572 012720 010000          MOV      #0!FMT22,(R0)+ ;CYLINDER 0
5241 027576 112746 000001          MOVB    #1,-(SP)      ;IN LOWER BYTE GET SECTOR
5242 027602 112766 000001 000001  MOVB    #1,1(SP)      ;GET TRACK IN HIGHER BYTE
5243 027610 012620          MOV      (SP)+,(R0)+  ;GET TRACK/SECTOR IN BUFFER
5244 027612 012720 000000          MOV      #0,(R0)+    ;KEY1 IN BUFFER
5245 027616 012720 000000          MOV      #0,(R0)+    ;KEY2 IN BUFFER
5246 027622 012701 000400          MOV      #256.,R1     ;DATA WORD COUNTER
    
```

5247	027626	012702	052525		MOV	#052525,R2		:DATA
5248	027632	010220		3\$:	MOV	R2,(R0)+		:DATA INTO BUFFER
5249	027634	005301			DEC	R1		:COUNT
5250	027636	001375			BNE	3\$:BRANCH IF 256 NOT DONE
5251	027640	004037	043514		JSR	RO,@#COMPAR		:CHECK
5252	027644	002110			WRFROM			:GOOD BUFFER
5253	027646	003154			REINTO			:TEST BUFFER
5254	027650	000404			4+256.			:NUMBER OF WORDS CHECKED
5255	027652	027660			4\$:RETURN POINT FOR ERROR HEADER
5256	027654	027664			5\$:RETURN POINT FOR ERROR DATA
5257	027656	027670			TST65			:RETURN FOR GOOD COMPARISON
5258	027660	104004		4\$:	ERROR	4		:READ NEXT ERROR
5259	027662	000207			RTS	PC		:RETURN TO 'COMPAR'
5260	027664	104005		5\$:	ERROR	5		:WORD NOS 1 TO 4 ARE
5261	027666	000207			RTS	PC		:RETURN TO 'COMPAR'
5262								
5263								
5264								

```

5265
5266 027670 000004          TST65: SCOPE
5267
5268 027672 012737 000065 002032      MOV      #TTNO,@#TSTNM      ;THIS SAVES TEST NUMBER
5269
5270 027700 012706 001000          MOV      #STACK,SP          ;RESET STACK
5271 027704 012737 000065 002032      MOV      #TTNO,@#TSTNM      ;THIS SAVES TEST NUMBER
5272 027712 004037 042536          JSR      RO,@#CLAREA        ;CLEAR SIMULATED DISK
5273 027716 051422          .WORD   DISK                ;FROM
5274 027720 052446          .WORD   TOLGAP+16          ;TO
5275 027722 000000          .WORD   0                  ;DATA
5276 027724 012737 010000 047504      MOV      #0!FMT22,@#CYL;CYLINDER 0
5277 027732 112737 000000 047507      MOVVB   #0,@#SECOTR+1;TRACK 0
5278 027740 112737 000000 047506      MOVVB   #0,@#SECOTR        ;SECTOR 0
5279 027746 005037 047510          CLR     @#KEY1              ;KEY1 0
5280 027752 005037 047512          CLR     @#KEY2              ;KEY2 0
5281 027756 012737 000400 047552      MOV      #256.,@#NOWORD     ;NO OF DATA WORDS
5282 027764 012737 000001 047514      MOV      #1,@#X             ;WRITE DATA
5283 027772 004537 044026          JSR      R5,@#CRC           ;GO TO CALCULATE CRC
5284 027776 047504          CYL
5285 030000 051404          WCRC
5286 030002 004037 042536          JSR      RO,@#CLAREA        ;FILL WRITE BUFFER WITH 377
5287 030006 002110          WRFROM          ;FROM LOCATION
5288 030010 003110          WRFROM+<256.*2>          ;TO LOCATION
5289 030012 000377          377                      ;DATA
5290 030014 004737 042620          JSR      PC,@#CLDISK        ;SETUP GENERAL REGISTERS
5291 030020 012777 177400 151604      MOV      #-256.,@RHWC       ;256. DATA WORDS
5292 030026 012777 002110 151600      MOV      #WRFROM,@RHBA     ;STARTING ADDRESS OF WRITE BUFFER
5293 030034 012746 000000          MOV      #0,-(SP)          ;SECTOR 0
5294 030040 112766 000000 000001      MOVVB   #0,1(SP)          ;TRACK 0
5295 030046 012677 151572          MOV      (SP)+,@RHDST       ;SECTOR 0 TRACK 0
5296 030052 012777 010000 151570      MOV      #FMT22,@RHOF      ;16 BITS PER WORD FORMAT
5297 030060 012777 000000 151564      MOV      #0,@RHCA          ;CYLINDER 0
5298 030066 004737 042654          JSR      PC,@#CHECKT        ;CHECK THAT DVA,RDY,DPR,DRY = 1
5299 030072 104401 062450          TYPE   ,CPHALT            ;AND THAT NO OTHERS = 1. CANNOT CON-
5300 030076 000000          HALT                      ;STOP THE TEST
5301 030100 013711 002062          MOV      @#WRIDAT,@R1      ;WRITE DATA=60
5302 030104 005037 002006          CLR     @#ERFLG$          ;CLEAR ERROR FLAG
5303 030110 004737 047344          JSR      PC,@#COMHD        ;WRITE DATA
5304 030114 004737 042224          JSR      PC,@#PUTREG       ;SAVE REGISTERS
5305 030120 005737 002006          TST     @#ERFLG$          ;HAVE ANY ERRORS OCCURED?
5306 030124 001041          BNE     TST66              ;BRANCH IF YES
5307 030126 012700 000377          MOV      #377,R0           ;GOOD DATA
5308 030132 012701 051422          MOV      #DISK,R1          ;DATA WRITTEN INTO 'DISK'
5309 030136 012702 000400          MOV      #256.,R2          ;COUNTER
5310 030142 012737 000401 047624 1$:      MOV      #256.+1,@#ERWORD  ;FOR ERROR WORD
5311 030150 020021          CMP     R0,(R1)+          ;COMPARE GOOD DATA WITH DATA ON DISK
5312 030152 001424          BEQ     3$                 ;BRANCH IF GOOD
5313 030154 010037 001124          MOV      R0,@#$GDDAT       ;GOOD DATA
5314 030160 014137 001126          MOV      -(R1),@#$BDDAT    ;BAD DATA
5315 030164 160237 047624          SUB     R2,@#ERWORD        ;ERROR WORD NO
5316 030170 005737 002006          TST     @#ERFLG$          ;ANY ERRORS ALREADY THERE?
5317 030174 001002          BNE     2$                 ;BRANCH IF YES
5318 030176 104004          ERROR   4                  ;ERROR ON WRITE DATA COMMAND
5319 030200 000401          BR      64$                ;BRANCH TO AVOID PRINTING NEXT ERROR
5320 030202 104005          2$:      ERROR   5                  ;WORD NO GIVES WORD IN ERROR
    
```

5321	030204	005721		64\$:	TST	(R1)+	:UNDO -(R1) FOR BAD DATA
5322	030206	017746	150726		MOV	@SWR, -(SP)	:GET SWITCH SETTING
5323	030212	042716	177177		BIC	#177177, (SP)	:KEEP ONLY SWITCH 7 AND 8
5324	030216	022726	000200		CMP	#SW07, (SP)+	:IS 7 SET AND 8 RESET
5325	030222	001402			BEQ	TST66	:BRANCH OUT IF YES
5326	030224	005302		3\$:	DEC	R2	:IF NOT COUNT 256 WORDS
5327	030226	001345			BNE	1\$:BRANCH IF 256. NOT DONE
5328							
5329							
5330							

```

5331
5332 030230 000004          TST66: SCOPE
5333 030232 012706 001000  MOV      #STACK,SP      ;RESET STACK
5334
5335 030236 012737 000066 002032  MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
5336 030244 004037 042536          JSR      RO,@#CLAREA    ;CLEAR SIMULATED DISK
5337 030250 051422          .WORD   DISK            ;FROM
5338 030252 052420          .WORD   DISK+776       ;TO
5339 030254 177400          .WORD   177400         ;DATA
5340
5341 030256 004037 042536          JSR      RO,@#CLAREA    ;CLEAR READ INTO BUFFER
5342 030262 003154          .WORD   REINTO         ;FROM
5343 030264 004152          .WORD   REINTO+776     ;TO
5344 030266 000000          .WORD   0              ;DATA
5345
5346          ;*THESE ARE TO SETUP FOR DISKLESS USE ONLY
5347
5348 030270 012737 010000 047504  MOV      #FMT22,@#CYL  ;CYLINDER 0 16 BITS PER WORD FORMAT
5349 030276 105037 047507          CLR     @#SECOTR+1     ;TRACK 0
5350 030302 112737 000001 047506  MOV     #1,@#SECOTR    ;SECTOR 1
5351 030310 005037 047510          CLR     @#KEY1         ;KEY1=0
5352 030314 005037 047512          CLR     @#KEY2         ;KEY2=0
5353 030320 012737 000012 047564  MOV     #10.,@#DAWORD  ;NO. OF DATA WORDS
5354 030326 005037 047514          CLR     @#X            ;THIS IS A READ COMMAND
5355 030332 004537 044026          JSR     R5,@#CRC       ;GO TO CALCULATE CRC
5356 030336 047504          CYL
5357 030340 051404          WCRC
5358
5359          ;*THESE ARE REGULAR SETUPS
5360
5361 030342 004737 042620          JSR     PC,@#CLDISK    ;SETUP GENERAL REGISTERS
5362 030346 013711 002066          MOV     @#READAT,@R1  ;READ DATA INTO RHCS1=70
5363 030352 012777 177766 151252  MOV     #-10.,@RHWC    ;10 DATA WORDS
5364 030360 012777 003154 151246  MOV     #REINTO,@RHBA  ;STARTING ADDRESS OF READ BUFFER
5365 030366 112746 000001          MOV     #1,-(SP)       ;IN LOWER BYTE GET SECTOR 1
5366 030372 112766 000000 000001  MOV     #0,1(SP)       ;GET TRACK0 IN UPPER BYTE
5367 030400 012677 151240          MOV     (SP)+,@RHDST   ;TRACK/SECTOR IN RHDST
5368 030404 012777 014000 151236  MOV     #FMT22!ECI,@RHOF ;16 BITS PER WORD
5369          ;ECC CORRECTION INHIBIT BECAUSE
5370          ;ECC IS NOT CHECKED HERE
5371 030412 005077 151234          CLR     @RHCA         ;CYLINDER 0
5372 030416 004737 042654          JSR     PC,@#CHECKT    ;CHECK THAT DVA,RDY,DPR,DRY = 1
5373 030422 104401 062450          TYPE   ,CPHALT        ;AND THAT NO OTHERS = 1. CANNOT CON-
5374 030426 000000          HALT                    ;STOP THE TEST
5375 030430 005037 002006          CLR     @#ERFLG$      ;CLEAR ERROR FLAG
5376 030434 004737 047344          JSR     PC,@#COMHD     ;READ DATA
5377
5378          ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUT
5379          ;*FROM 'COMHD' ROUTINE IN MEANS DATA IS TO BE CHECKED
5380
5381          ;*NOW THE DATA READ INTO 'REINTO' BUFFER WILL
5382          ;*BE CHECKED, ONLY 10 WORDS SHOULD BE CHANGED
5383          ;*ALL OTHER WORDS SHOULD REMAIN UNCHANGED
5384          ;*THE 'WRFROM' BUFFER IS FILLED WITH EXPECTED DATA AND CHECKED
5385
5386 030440 005737 002006          TST     @#ERFLG$      ;HAVE ANY ERRORS OCCURED?

```



```
5387 030444 001053          BNE    TST67          ;:BRANCH IF YES
5388 030446 004037 042536    JSR    R0,@#CLAREA   ;:CLEAR BUFFER
5389 030452 002110          WRFROM          ;:FROM
5390 030454 003106          WRFROM+776      ;:TO
5391 030456 000000          0              ;:DATA
5392
5393 030460 004037 042536    JSR    R0,@#CLAREA   ;:FILL EXPECTED DATA
5394 030464 002110          WRFROM          ;:FROM
5395 030466 002132          WRFROM+22      ;:TO
5396 030470 177400          177400        ;:DATA
5397
5398          ;*NOW READ DATA BUFFER IS CHECKED
5399
5400 030472 012700 002110    MOV    #WRFROM,R0    ;:GOOD DATA
5401 030476 012701 003154    MOV    #REINTO,R1    ;:DATA READ
5402 030502 012702 000400    MOV    #256.,R2      ;:COUNTER
5403 030506 012737 000401 047624 1$:  MOV    #257.,@#ERWORD ;:FOR ERROR WORD NO
5404 030514 022021          CMP    (R0)+,(R1)+   ;:COMPARE GOOD WITH READ BUFFER
5405 030516 001424          BEQ    2$           ;:BRANCH IF GOOD
5406 030520 014037 001124    MOV    -(R0),@#$GDDAT ;:GOOD DATA
5407 030524 014137 001126    MOV    -(R1),@#$BDDAT ;:BAD DATA
5408 030530 160237 047624    SUB    R2,@#ERWORD   ;:ERROR WORD NO
5409 030534 005737 002006    TST    @#ERFLG$     ;:ANY ERRORS ALREADY THERE
5410 030540 001002          BNE    3$           ;:IF YES BRANCH DO NOT TYPE HEADER
5411 030542 104004          ERROR    4          ;:ERROR ON READ DATA
5412 030544 000401          BR     4$           ;:BRANCH TO AVOID PRINTING NEXT ERROR
5413 030546 104005          3$:  ERROR    5          ;:WORD NO 1-10 ARE DATA
5414          ;:WORDS
5415          ;:WORD NOS 11-256 HAVE NOT BEEN
5416          ;:READ AND BUFFER SHOULD BE
5417          ;:ZERO IF OTHER THAN ZERO
5418          ;:WRONG NUMBER OF WORDS HAVE
5419          ;:BEEN READ IN THE DISK NOW
5420          ;:CONTAINS 177400 ALL 256
5421          ;:WORDS BUT ONLY 10 WORDS
5422          ;:SHOULD BE READ IN
5423
5424 030550 022021          4$:  CMP    (R0)+,(R1)+   ;:UNDO -(R0) AND -(R1) FOR ERROR
5425 030552 017746 150362    MOV    @SWR,-(SP)    ;:GET SWITCH SETTING
5426 030556 042716 177177    BIC    #177177,(SP)  ;:KEEP ONLY SWITCH 7 AND 8
5427 030562 022726 000200    CMP    #SW07,(SP)+  ;:IS 7 SET AND 8 RESET
5428 030566 001402          BEQ    TST67        ;:BRANCH OUT IF YES
5429 030570 005302          2$:  DEC    R2           ;:COUNT
5430 030572 001345          BNE    1$           ;:BRANCH IF NOT COMPLETE
5431
5432
5433
```

```
5434
5435      ;*
5436
5437 030574 000004      TST67: SCOPE
5438
5439      ;*      DATA TABLE
5440      ;*      20 WORDS OF 070707
5441      ;*      THEN 16 WORDS WITH ZERO FLOATING FROM RIGHT
5442      ;*      TO LEFT (EG. 177776,177775,177773 ETC)
5443
5444 030576 012706 001000      MOV      #STACK,SP      ;RESET STACK
5445 030602 012737 000067 002032      MOV      #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
5446
5447      ;*SET UP 'REINTO' FOR WHAT IS TO BE READ
5448 030610 012701 003154      MOV      #REINTO,R1      ;STARTING ADDRESS
5449 030614 012721 010000      MOV      #FMT22,(R1)+    ;CYLINDER 0 FORMAT 16 BIT WORDS
5450 030620 012721 000401      MOV      #401,(R1)+      ;TRACK=1, SECTOR=1
5451 030624 005021      CLR      (R1)+           ;KEY1=0
5452 030626 005021      CLR      (R1)+           ;KEY2=0
5453 030630 004037 042536      JSR      RO,@#CLAREA     ;FILL 'REINTO' BUFFER
5454 030634 003164      .WORD   REINTO+<4*2>    ;FROM
5455 030636 003232      .WORD   REINTO+<23.*2> ;TO
5456 030640 070707      .WORD   070707          ;DATA
5457
5458 030642 012700 177776      MOV      #177776,RO      ;GETTING READY TO FLOAT 0
5459 030646 012701 003234      MOV      #REINTO+<24.*2>,R1 ;STARTING ADDRESS WHERE 177776 GOES
5460 030652 010021      1$:     MOV      RO,(R1)+      ;MOVE IN FLOATING 0
5461 030654 000261      SEC      ;SET CARRY
5462 030656 006100      ROL      RO              ;GET 0 ONE BIT LEFT
5463 030660 103774      BCS     1$              ;BRANCH IF 16 NOT DONE
5464
5465 030662 004037 042536      JSR      RO,@#CLAREA     ;FILL THE REST OF BUFFER WITH 0
5466 030666 003274      .WORD   REINTO+<40.*2>  ;FROM
5467 030670 004152      .WORD   REINTO+776      ;TO
5468 030672 000000      .WORD   0               ;DATA
5469
5470      ;*SET UP SIMULATED DISK WITH WHAT IS TO BE READ
5471
5472 030674 004037 042536      JSR      RO,@#CLAREA     ;FILL 'DISK' BUFFER
5473 030700 051422      .WORD   DISK             ;FROM
5474 030702 051470      .WORD   DISK+<19.*2>    ;TO
5475 030704 070707      .WORD   070707          ;DATA
5476
5477 030706 012700 177776      MOV      #177776,RO      ;GETTING READY TO FLOAT ZEROS
5478 030712 012701 051472      MOV      #DISK+<20.*2>,R1 ;STARTING ADDRESS WHERE 177776 GOES
5479 030716 010021      2$:     MOV      RO,(R1)+      ;MOVE IN FLOATING 0
5480 030720 000261      SEC      ;SET CARRY
5481 030722 006100      ROL      RO              ;GET 0 ONE BIT LEFT
5482 030724 103774      BCS     2$              ;BRANCH IF 16 NOT DONE
5483
5484 030726 004037 042536      JSR      RO,@#CLAREA     ;FILL THE REST OF BUFFER WITH 177777
5485 030732 051532      .WORD   DISK+<36.*2>    ;FROM
5486 030734 052420      .WORD   DISK+776        ;TO
5487 030736 177777      .WORD   177777          ;DATA
5488
5489 030740 004737 043344      JSR      PC,@#WRCHHD     ;WRITE CHECK HEADER AND DATA
```

```
5490                                     ;CYLINDER 0, TRACK 1, SECTOR 1
5491
5492                                     ;*IF THE PROGRAM COMES BACK HERE THEN WRITE CHECK
5493                                     ;*HAS BEEN COMPLETED NOW WRITE CHECK ERROR BIT IS TO BE TESTED
5494
5495 030744 013746 001774      MOV      @#UNIT,-(SP)      ;GET UNIT NUMBER
5496 030750 052716 000100      BIS      #IR,(SP)        ;ONLY BIT 6 SHOULD BE SET
5497 030754 004737 042224      JSR      PC,@#PUTREG     ;SAVE REGISTERS
5498 030760 022637 001712      CMP      (SP)+,@#CS2    ;COMPARE RHCS2
5499 030764 001406              BEQ      4$              ;BRANCH IF GOOD
5500 030766 032712 040000      BIT      #WCE,@R2       ;WRITE CHECK ERROR HIGH?
5501 030772 001402              BEQ      3$              ;BRANCH IF ERROR NOT DUE TO 'WCE'
5502 030774 104017              ERROR    17              ;RHDB CONTAINS FAILING WORD
5503 030776 000401              BR       4$              ;RHBA CONTAINS ADDRESS+2
5504                                     ;OF THE WORD IN MEMORY FROM
5505                                     ;THE DISK THAT DID NOT COMPARE
5506                                     ;TRE AND SC WILL BE SET DUE TO
5507                                     ;WCE
5508 031000 104017              3$:      ERROR    17       ;WCE CORRECTLY WAS NOT SET BUT SOME
5509                                     ;BITS OTHER THAN IR
5510                                     ;AND UNIT NO. WAS SET
5511
5512                                     ;*NOW CHECK MEMORY TO SEE IF NOTHING GOT DESTROYED
5513                                     ;*FILL 'WRFROM' WITH WHAT SHOULD BE IN 'REINTO' THEN CHECK
5514
5515 031002 012700 002110      4$:      MOV      #WRFROM,R0    ;STARTING ADDRESS
5516 031006 012720 010000      MOV      #FMT22,(R0)+   ;CYLINDER
5517 031012 012720 000401      MOV      #401,(R0)+    ;TRACK=1, SECTOR=1
5518 031016 005020              CLR      (R0)+          ;KEY1=0
5519 031020 005020              CLR      (R0)+          ;KEY2=0
5520
5521 031022 004037 042536      JSR      R0,@#CLAREA    ;FILL 'WRFROM' BUFFER
5522 031026 002120              .WORD   WRFROM+<4*2>   ;FROM
5523 031030 002166              .WORD   WRFROM+<23.*2> ;TO
5524 031032 070707              .WORD   070707         ;DATA
5525
5526 031034 012700 177776      MOV      #177776,R0     ;GETTING READY TO FLOAT 0
5527 031040 012701 002170      MOV      #WRFROM+<24.*2>,R1 ;STARTING ADDRESS WHERE 177776 GOES
5528 031044 010021              MOV      R0,(R1)+      ;MOVE IN FLOATING 0
5529 031046 000261              SEC              ;SET CARRY
5530 031050 006100              ROL      R0            ;GET 0 ONE BIT LEFT
5531 031052 103774              BCS      5$            ;BRANCH IF 16 NOT DONE
5532
5533 031054 004037 042536      JSR      R0,@#CLAREA    ;FILL THE REST OF BUFFER WITH 0
5534 031060 002230              .WORD   WRFROM+<40.*2> ;FROM
5535 031062 003106              .WORD   WRFROM+776    ;TO
5536 031064 000000              .WORD   0              ;DATA
5537
5538                                     ;*NOW THE READ BUFFER WILL BE CHECKED
5539 031066 005037 002006      CLR      @#ERFLG$      ;CLEAR ERROR FLAG
5540
5541 031072 004037 043514      JSR      R0,@#COMPAR    ;CHECK
5542 031076 002110              WRFROM              ;GOOD BUFFER
5543 031100 003154              REINTO              ;TEST BUFFER
5544 031102 000400              256.                ;NUMBER OF WORDS CHECKED
5545 031104 031112              6$                  ;RETURN POINT FOR ERROR HEADER
```



```
5558
5559 031124 000004          TST70: SCOPE
5560
5561                      ;*DATA TABLE
5562                      ;*TOTAL OF 32 WORDS CONSISTING OF
5563                      ;*16 WORDS OF FLOATING ONES (EG. 1, 2, 4, 10)
5564                      ;*16 WORDS OF FLOATING ZEROS (EG. 177776, 177775)
5565
5566
5567 031126 012706 001000    MOV    #STACK,SP      ;RESET STACK
5568 031132 012737 000070 002032  MOV    #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
5569
5570                      ;*SET UP 'REINTO' FOR WHAT IS TO BE READ
5571
5572 031140 012700 000001    MOV    #1,R0          ;GETTING READY TO FLOAT 1
5573 031144 012701 003154    MOV    #REINTO,R1     ;STARTING ADDRESS WHERE 1 GOES
5574 031150 010021          1$:  MOV    R0,(R1)+       ;MOVE FLOATING 1
5575 031152 006100          ROL    R0              ;GET 1 ONE BIT LEFT
5576 031154 103375          BCC    1$              ;BRANCH IF 16 NOT DONE
5577 031156 012700 177776    MOV    #177776,R0     ;GETTING READY TO FLOAT 0
5578 031162 012701 003214    MOV    #REINTO+<16.*2>,R1 ;STARTING ADDRESS WHERE 177776 GOES
5579 031166 010021          2$:  MOV    R0,(R1)+       ;MOVE IN FLOATING 0
5580 031170 000261          SEC                      ;SET CARRY
5581 031172 006100          ROL    R0              ;GET 0 ONE BIT LEFT
5582 031174 103774          BCS    2$              ;BRANCH IF 16 NOT DONE
5583
5584 031176 004037 042536    JSR    R0,@#CLAREA    ;FILL REST OF BUFFER WITH 1
5585 031202 003254          .WORD REINTO+<32.*2> ;FROM
5586 031204 004152          .WORD REINTO+776     ;TO
5587 031206 000001          .WORD 1               ;WITH DATA
5588
5589                      ;*SET UP SIMULATED DISK WITH WHAT IS TO BE READ
5590
5591 031210 012700 000001    MOV    #1,R0          ;GETTING READY TO FLOAT 1
5592 031214 012701 051422    MOV    #DISK,R1       ;STARTING ADDRESS WHERE 1 GOES
5593 031220 010021          3$:  MOV    R0,(R1)+       ;MOVE FLOATING 1
5594 031222 006100          ROL    R0              ;GET 1 ONE BIT LEFT
5595 031224 103375          BCC    3$              ;BRANCH IF 16 NOT DONE
5596
5597 031226 012700 177776    MOV    #177776,R0     ;GETTING READY TO FLOAT 0
5598 031232 012701 051462    MOV    #DISK+<16.*2>,R1 ;STARTING ADDRESS WHERE 177776 GOES
5599 031236 010021          4$:  MOV    R0,(R1)+       ;MOVE FLOATING 0
5600 031240 000261          SEC                      ;SET CARRY
5601 031242 006100          ROL    R0              ;GET 0 ONE BIT LEFT
5602 031244 103774          BCS    4$              ;BRANCH IF 16 NOT DONE
5603
5604 031246 004037 042536    JSR    R0,@#CLAREA    ;FILL REST OF BUFFER WITH 0
5605 031252 051522          .WORD DISK+<32.*2> ;FROM
5606 031254 052420          .WORD DISK+776     ;TO
5607 031256 000000          .WORD 0             ;WITH DATA
5608
5609 031260 004737 043656    JSR    PC,@#WRCHDA    ;WRITE CHECK DATA
5610                      ;CYLINDER 0, TRACK 1, SECTOR 1
5611                      ;KEYS 0, 32 WORDS.
5612
5613                      ;*IF THE PROGRAM COMES BACK HERE THEN WRITE CHECK
```


5670
5671
5672 031426 000207
5673
5674 031430 000240
5675
5676

13\$: NOP

RTS PC

:WORD NO CONTAINS THE WORD
:NUMBER THAT GOT CHANGED
:RETURN TO COMPARISON SUBROUTINE
:ONLY A BRANCH POINT


```

5733
5734 031630 012777 000001 150002 4$: MOV #GO,@R1HCS1 ;GIVE NO-OP
5735 031636 004737 042224 JSR PC,@#PUTREG ;SAVE REGISTERS
5736 031642 020112 CMP R1,@R2 ;GO SHOULD NOT CLEAR ERROR
5737 031644 001410 BEQ 5$ ;FURTHER CHECK OF 'GO' FUNCTIONALITY
5738 031646 010237 042270 MOV R2,@#REGADR ;FAILING REGISTER
5739 031652 010137 001124 MOV R1,@#$GDDAT ;GOOD DATA
5740 031656 013737 071712 001126 MOV @#CS2,@#$BDDAT ;BAD DATA
5741 031664 104001 ERROR 1 ;'GO' WITH NO-OP CHANGED
5742 ;ERROR REGISTER
5743
5744 031666 013746 001736 5$: MOV @#DS1,-(SP) ;GET RHDS1
5745 031672 042716 001100 BIC #VV!PROG,(SP) ;CLEAR VV AND PROG
5746 031676 022726 140600 CMP #ATA!ERR!DPR!DRY,(SP)+ ;GO SHOULD NOT CLEAR ANY BITS
5747 031702 001401 BEQ 7$ ;CHECK NEXT ERROR BIT IF A-OK
5748 031704 104020 ERROR 20 ;'GO' WITH NO-OP SHOULD NOT CLEAR
5749 ;ATA AND/OR ERR
5750
5751
5752 ;*THIS IS THE MAIN BIT TESTING CONTROL LOGIC
5753
5754
5755 031706 006301 7$: ASL R1 ;GET NEXT BIT TO THE LEFT
5756 031710 103403 BCS 10$ ;GO ON TO NEXT REGISTER IF DONE
5757 031712 031001 BIT (R0),R1 ;IS THIS BIT TO BE TESTED ?
5758 031714 001374 BNE 7$ ;IF NOT, GET NEXT ONE
5759 031716 000717 BR 2$ ;IF TO BE TESTED, GO DO IT !
5760
5761 031720 005720 10$: TST (R0)+ ;ADVANCE R0 TO NEXT ERROR REG.
5762 031722 005337 001200 DEC @#$TMP1 ;REGISTER COUNTER
5763 031726 001310 BNE 1$ ;DO NEXT ONE, IF 3 NOT COMPLETE
5764
5765
5766 ;*NOW AFTER SETTING ATA IN RHDS1 '1' IN RHAS AT THE
5767 ;*DRIVE POSITION SHOULD CLEAR ATA IN RHDS1
5768
5769 031730 004737 042620 001110 11$: JSR PC,@#CLDISK ;CLEAR
5770 031734 012737 031730 MOV #11$, $LPERR ;ERROR RETURN
5771 031742 012714 177777 MOV #-1,@R4 ;SET BIT IN RHAS AND ATA IN RHDS1
5772 031746 013777 002016 147702 MOV @#ATTENT,@RHAS ;WRITE 1 INTO DRIVE BIT POSITION
5773 031754 004737 042224 JSR PC,@#PUTREG ;SAVE REGISTERS
5774 031760 105737 001732 TSTB @#AS ;THIS SHOULD BE ZERO
5775 031764 001401 BEQ 12$
5776 031766 104020 ERROR 20 ;MOVING A '1' INTO RHAS
5777 ;AT THE DRIVE BIT POSITION
5778 ;DID NOT CLEAR IT
5779
5780 031770 013746 001736 12$: MOV @#DS1,-(SP) ;GET RHDS1
5781 031774 042716 001000 BIC #PROG,(SP) ;MASK PROGRAMABLE
5782 032000 022726 040700 CMP #ERR!VV!DPR!DRY,(SP)+ ;RHDS1 SHOULD HAVE THESE BITS
5783 ;BUT ATA SHOULD BE CLEARED
5784
5785 032004 001401 BEQ 13$ ;CHECK RHER1 IF GOOD
5786 032006 104020 ERROR 20 ;MOVING '1' INTO RHAS AT THE
5787 ;DRIVE BIT POSITION DID NOT
5788 ;CLEAR ATA IN RHDS1
    
```



```
5801
5802 032022 000004          TST72: SCOPE
5803 032024 012706 001000  MOV      #STACK,SP      ;RESET STACK
5804
5805 032030 012737 000072 002032  MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
5806 032036 004037 042536      JSR      R0,@#CLAREA    ;CLEAR SIMULATED DISK
5807 032042 051422          .WORD   DISK            ;FROM
5808 032044 052420          .WORD   DISK+776       ;TO
5809 032046 177400          .WORD   177400         ;DATA
5810
5811 032050 004037 042536      JSR      R0,@#CLAREA    ;CLEAR READ INTO BUFFER
5812 032054 003154          .WORD   REINTO         ;FROM
5813 032056 004152          .WORD   REINTO+776    ;TO
5814 032060 000000          .WORD   0              ;DATA
5815
5816                      ;*THESE ARE TO SETUP FOR DISKLESS USE ONLY
5817
5818 032062 012737 010000 047504  MOV      #FMT22,@#CYL   ;CYLINDER 0 16 BITS PER WORD FORMAT
5819 032070 105037 047507      CLRB    @#SECOTR+1     ;TRACK 0
5820 032074 112737 000001 047506  MOVB    #1,@#SECOTR    ;SECTOR 1
5821 032102 005037 047510      CLR     @#KEY1         ;KEY1=0
5822 032106 005037 047512      CLR     @#KEY2         ;KEY2=0
5823 032112 012737 000012 047564  MOV      #10,@#DAWORD  ;NO. OF DATA WORDS
5824 032120 005037 047514      CLR     @#X            ;THIS IS A READ COMMAND
5825 032124 004537 044026      JSR      R5,@#CRC      ;GO TO CALCULATE CRC
5826 032130 047504          CYL
5827 032132 051404          WCRC
5828
5829                      ;*THESE ARE REGULAR SETUPS
5830
5831 032134 004737 042620      JSR      PC,@#CLDISK   ;SETUP GENERAL REGISTERS
5832 032140 013711 002066      MOV      @#READAT,@R1  ;READ DATA INTO RHCS1=70
5833 032144 012777 177766 147460  MOV      #-10,@#RHWC   ;10 DATA WORDS
5834 032152 012777 003154 147454  MOV      #REINTO,@#RHB ;STARTING ADDRESS OF READ BUFFER
5835 032160 112746 000001      MOVB    #1,-(SP)       ;IN LOWER BYTE GET SECTOR 1
5836 032164 112766 000000 000001  MOVB    #0,1(SP)       ;GET TRACK0 IN UPPER BYTE
5837 032172 012677 147446      MOV      (SP)+,@#RHDST ;TRACK/SECTOR IN RHDST
5838 032176 012777 014000 147444  MOV      #FMT22!ECI,@#RHOF ;16 BITS PER WORD
5839                      ;ECC CORRECTION INHIBIT BECAUSE
5840                      ;ECC IS NOT CHECKED HERE
5841 032204 005077 147442      CLR     @#RHCA        ;CYLINDER 0
5842 032210 004737 042654      JSR      PC,@#CHECKT   ;CHECK THAT DVA,RDY,DPR,DRY = 1
5843 032214 104401 062450      TYPE    ,CPHALT       ;AND THAT NO OTHERS = 1. CANNOT CON-
5844 032220 000000          HALT                   ;STOP THE TEST
5845 032222 052777 000010 147406  BIS     #BAI,@#RHCS2   ;SET BUS ADDRESS INHIBIT
5846 032230 005037 002006      CLR     @#ERFLG$      ;CLEAR ERROR FLAG
5847 032234 004737 047344      JSR      PC,@#COMHD    ;READ DATA
5848
5849                      ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUT
5850                      ;*FROM 'COMHD' ROUTINE IN MEANS DATA IS TO BE CHECKED
5851
5852                      ;*NOW THE DATA READ INTO 'REINTO' BUFFER WILL
5853                      ;*BE CHECKED, ONLY ONE WORD SHOULD BE CHANGED
5854                      ;*ALL OTHER WORDS SHOULD REMAIN UNCHANGED
5855                      ;*THE 'WRFROM' BUFFER IS FILLED WITH EXPECTED DATA AND CHECKED
5856
```



```
5904
5905 032366 000004          TST73: SCOPE
5906 032370 012706 001000  MOV      #STACK,SP          ;RESET STACK
5907 032374 012737 000073 002032  MOV      #TTNO,@#TSTNM      ;THIS SAVES TEST NUMBER
5908 032402 005737 002040  TST      @#RH70             ;TEST FLAG FOR RH70 CONTROLLER
5909 032406 001402          BEQ      30$                ;IF FLAG = 1, THIS TEST IS SKIPPED
5910 032410 000137 032732  JMP      TST74              ;JUMP TO NEXT TEST -----)
5911
5912 032414 004037 042536  JSR      R0,@#CLAREA        ;CLEAR SIMULATED DISK
5913 032420 051422          .WORD   DISK                ;FROM
5914 032422 052420          .WORD   DISK+776            ;TO
5915 032424 177400          .WORD   177400              ;DATA
5916
5917
5918 ;*THESE ARE TO SETUP FOR DISKLESS USE ONLY
5919
5920 032426 012737 010000 047504  MOV      #FMT22,@#CYL      ;CYLINDER 0, 16 BITS PER WORD FORMAT
5921 032434 105037 047507          CLR     @#SECOTR+1         ;TRACK 0
5922 032440 112737 000001 047506  MOVB    #1,@#SECOTR        ;SECTOR 1
5923 032446 005037 047510          CLR     @#KEY1             ;KEY1=0
5924 032452 005037 047512          CLR     @#KEY2             ;KEY2=0
5925 032456 012737 000001 047564  MOV      #1,@#DAWORD       ;NO. OF DATA WORDS
5926 032464 005037 047514          CLR     @#X                 ;THIS IS A READ COMMAND
5927 032470 004537 044026  JSR      R5,@#CRC           ;GO TO CALCULATE CRC
5928 032474 047504          CYL
5929 032476 051404          WCRC
5930
5931 ;*THESE ARE REGULAR SETUPS
5932
5933 032500 004737 042620          JSR     PC,@#CLDISK        ;SETUP GENERAL REGISTERS
5934 032504 013711 002066          MOV     @#READAT,@R1       ;READ DATA INTO RHCS1=70
5935 032510 012777 177777 147114  MOV     #-1,@#RHWC         ;10 DATA WORDS
5936 032516 012777 160000 147110  MOV     #160000,@#RHBA     ;STARTING ADDRESS OF READ BUFFER
5937 032524 052711 001400          BIS     #A16!A17,@R1      ;IS 760000
5938 032530 112746 000001          MOVB   #1,-(SP)           ;IN LOWER BYTE GET SECTOR 1
5939 032534 112766 000000 000001  MOVB   #0,1(SP)           ;GET TRACK0 IN UPPER BYTE
5940 032542 012677 147076          MOV     (SP)+,@#RHDST     ;TRACK/SECTOR IN RHDST
5941 032546 012777 014000 147074  MOV     #FMT22!ECI,@#RHOF ;16 BITS PER WORD
5942
5943 ;ECC CORRECTION INHIBIT BECAUSE
5944 ;ECC IS NOT CHECKED HERE
5944 032554 005077 147072          CLR     @#RHCA             ;CYLINDER 0
5945 032560 004737 042654          JSR     PC,@#CHECKT        ;CHECK THAT DVA,RDY,DPR,DRY = 1
5946 032564 104401 062450          TYPE   ,CPHALT            ;AND THAT NO OTHERS = 1. CANNOT CON-
5947 032570 000000          HALT
5948 032572 052777 000010 147036  BIS     #BAI,@#RHCS2      ;SET BUS ADDRESS INHIBIT
5949 032600 005037 002006          CLR     @#ERFLG$          ;CLEAR ERROR FLAG
5950 032604 004737 047344          JSR     PC,@#COMHD        ;READ DATA
5951
5952
5953
5954 032610 011137 001126          1$:  MOV     @R1,@#$BDDAT      ;TEST DATA
5955
5956 032614 022737 145670 001126  CMP     #SC!TRE!DVA!A16!A17!RDY!70,@#$BDDAT ;COMPARE RHCS1
5957 032622 001406          BEQ     2$                 ;BRANCH IF GOOD
5958 032624 012737 144270 001124  MOV     #SC!TRE!DVA!RDY!70,@#$GDDAT ;GOOD DATA
5959 032632 010137 042270          MOV     R1,@#REGADR       ;REGISTER RHCS1
```



```
5988
5989 032732 000004      TST74: SCOPE
5990
5991      :*DATA TABLE
5992      :*TOTAL OF 32 WORDS CONSISTING OF
5993      :*16 WORDS OF FLOATING ONES (EG. 1, 2, 4, 10)
5994      :*16 WORDS OF FLOATING ZEROS (EG. 177776, 177775)
5995
5996
5997 032734 012706 001000      MOV      #STACK,SP      :RESET STACK
5998 032740 012737 000074 002032      MOV      #TTNO,@#TSTNM  :THIS SAVES TEST NUMBER
5999 032746 004737 042620      JSR      PC,@#CLDISK    :INIT AND SET UP GENERAL REGISTERS
6000
6001      :*SET UP 'REINTO' FOR WHAT IS TO BE READ
6002
6003 032752 012700 000001      MOV      #1,R0          :GETTING READY TO FLOAT 1
6004 032756 012701 003154      MOV      #REINTO,R1     :STARTING ADDRESS WHERE 1 GOES
6005 032762 010021      1$:      MOV      R0,(R1)+       :MOVE FLOATING 1
6006 032764 006100      ROL      R0              :GET 1 ONE BIT LEFT
6007 032766 103375      BCC      1$              :BRANCH IF 16 NOT DONE
6008 032770 012700 177776      MOV      #177776,R0     :GETTING READY TO FLOAT 0
6009 032774 012701 003214      MOV      #REINTO+<16.*2>,R1 :STARTING ADDRESS WHERE 177776 GOES
6010 033000 010021      2$:      MOV      R0,(R1)+       :MOVE IN FLOATING 0
6011 033002 000261      SEC                      :SET CARRY
6012 033004 006100      ROL      R0              :GET 0 ONE BIT LEFT
6013 033006 103774      BCS      2$              :BRANCH IF 16 NOT DONE
6014
6015 033010 004037 042536      JSR      R0,@#CLAREA    :FILL REST OF BUFFER WITH 1
6016 033014 003254      .WORD   REINTO+<32.*2>  :FROM
6017 033016 004152      .WORD   REINTO+776     :TO
6018 033020 000001      .WORD   1               :WITH DATA
6019
6020      :*SET UP SIMULATED DISK WITH WHAT IS TO BE READ
6021
6022 033022 012700 000001      MOV      #1,R0          :GETTING READY TO FLOAT 1
6023 033026 012701 051422      MOV      #DISK,R1       :STARTING ADDRESS WHERE 1 GOES
6024 033032 010021      3$:      MOV      R0,(R1)+       :MOVE FLOATING 1
6025 033034 006100      ROL      R0              :GET 1 ONE BIT LEFT
6026 033036 103375      BCC      3$              :BRANCH IF 16 NOT DONE
6027
6028 033040 012700 177776      MOV      #177776,R0     :GETTING READY TO FLOAT 0
6029 033044 012701 051462      MOV      #DISK+<16.*2>,R1 :STARTING ADDRESS WHERE 177776 GOES
6030 033050 010021      4$:      MOV      R0,(R1)+       :MOVE FLOATING 0
6031 033052 000261      SEC                      :SET CARRY
6032 033054 006100      ROL      R0              :GET 0 ONE BIT LEFT
6033 033056 103774      BCS      4$              :BRANCH IF 16 NOT DONE
6034
6035 033060 004037 042536      JSR      R0,@#CLAREA    :FILL REST OF BUFFER WITH 0
6036 033064 051522      .WORD   DISK+<32.*2>   :FROM
6037 033066 052420      .WORD   DISK+776      :TO
6038 033070 000000      .WORD   0              :WITH DATA
6039
6040      :*CHANGE FIFTH WORD TO 0 ON DISK
6041
6042 033072 005037 051432      CLR      @#DISK+10      :CLEAR FIFTH WORD ON DISK
6043 033076 005037 002006      CLR      @#ERFLG$      :CLEAR ERROR FLAG
```

```
6044 033102 004737 043656 JSR PC,@#WRCHDA ;WRITE CHECK DATA
6045 ;CYLINDER 0, TRACK 1, SECTOR 1
6046 ;KEYS 0, 32 WORDS.
6047
6048 ;*IF THE PROGRAM COMES BACK HERE THEN WRITE CHECK
6049 ;*HAS BEEN COMPLETED, NOW WRITE CHECK ERROR BIT IS TESTED
6050 ;*ALONG WITH RHWC FOR PROPER WORD COUNT AND RHBA FOR ADDRESS
6051
6052 033106 013746 001774 MOV @#UNIT,-(SP) ;GET UNIT NUMBER
6053 033112 052716 040300 BIS #IR!OR!WCE,(SP) ;ONLY BIT 6 SHOULD BE SET
6054 033116 004737 042224 JSR PC,@#PUTREG ;SAVE REGISTERS
6055 033122 022637 001712 CMP (SP)+,@#CS2 ;COMPARE RHCS2
6056 033126 001407 BEQ 6$ ;BRANCH IF GOOD
6057 033130 032737 040000 001712 BIT #WCE,@#CS2 ;WRITE CHECK ERROR HIGH?
6058 033136 001002 BNE 5$ ;BRANCH IF ERROR NOT DUE TO 'WCE'
6059 033140 104017 ERROR 17 ;RHDB CONTAINS FAILING WORD
6060 033142 000401 BR 6$ ;RHBA CONTAINS ADDRESS+2
6061 ;OF THE WORD IN MEMORY FROM
6062 ;THE DISK THAT DID NOT COMPARE
6063 ;TRE AND SC WILL BE SET DUE TO WCE
6064 033144 104017 5$: ERROR 17 ;WCE WAS CORRECTLY NOT SET
6065 ;BUT SOME BITS OTHER THAN
6066 ;IR AND UNIT NO. WERE SET
6067
6068 033146 005737 002040 6$: TST @#RH70 ;TEST FOR RH70 CONTROLLER
6069 033152 001414 BEQ 16$ ;SKIP RH70 CODE AND DO RH11 IF NOT
6070
6071 033154 022737 177750 001706 CMP #-24.,@#WC ;COMPARE RHWC AFTER A FORCED
6072 ;WRITE CHECK ERROR
6073 033162 001402 BEQ 17$ ;CHECK RHBA IF GOOD
6074 033164 104017 ERROR 17 ;WORD COUNT REGISTER IN ERROR AFTER A
6075 ;FORCED WRITE CHECK ERROR ON FIFTH WORD
6076 033166 000421 BR 15$ ;BRANCH TO CONTINUE TEST
6077
6078 033170 022737 003174 001710 17$: CMP #REINTO+<8.*2>,@#BA ;COMPARE RHBA AFTER A FORCED
6079 ;WRITE CHECK ERROR IN FIFTH WORD
6080 033176 001415 BEQ 15$ ;CONTINUE IF GOOD
6081 033200 104017 ERROR 17 ;BUS ADDRESS REGISTER IN ERROR AFTER
6082 ;FORCED WRITE CHECK ERROR ON FIFTH WORD
6083 033202 000413 BR 15$ ;SKIP RH11 CODE AND CONTINUE WITH TEST
6084
6085 033204 022737 177745 001706 16$: CMP #-27.,@#WC ;COMPARE RHWC AFTER A FORCED
6086 ;WRITE CHECK ERROR
6087 033212 001402 BEQ 14$ ;CHECK RHBA IF GOOD
6088 033214 104017 ERROR 17 ;WORD COUNT REGISTER IN ERROR AFTER A
6089 ;FORCED WRITE CHECK ERROR ON FIFTH WORD
6090 033216 000405 BR 15$ ;BRANCH TO CONTINUE TEST
6091
6092 033220 022737 003166 001710 14$: CMP #REINTO+<5.*2>,@#BA ;COMPARE RHBA AFTER FORCED
6093 ;WRITE CHECK ERROR IN FIFTH WORD
6094 033226 001401 BEQ 15$ ;CONTINUE IF GOOD
6095 033230 104017 ERROR 17 ;BUS ADDRESS REGISTER IN ERROR AFTER
6096 ;FORCED WRITE CHECK ERROR ON FIFTH WORD
6097
6098 ;*NOW CHECK MEMORY TO SEE IF ANYTHING GOT DESTROYED
6099 ;*FILL 'WRFROM' WITH WHAT SHOULD BE IN REINTO THEN CHECK
```



```
6100
6101 033232 005037 002006      15$: CLR @#ERFLG$ ;CLEAR ERROR FLAG
6102 033236 012700 000001      MOV #1,R0 ;GETTING READY TO FLOAT 1
6103 033242 012701 002110      MOV #WRFROM,R1 ;START ADDRESS WHERE 1 GOES
6104 033246 010021              7$: MOV R0,(R1)+ ;MOVE FLOATING 1
6105 033250 006100              ROL R0 ;GET 1 ONE BIT LEFT
6106 033252 103375              BCC 7$ ;BRANCH IF 16 NOT DONE
6107
6108 033254 012700 177776      MOV #177776,R0 ;GETTING READY TO FLOAT 0
6109 033260 012701 002150      MOV #WRFROM+<16.*2>,R1 ;STARTING ADDRESS WHERE 177776 GOES
6110 033264 010021              10$: MOV R0,(R1)+ ;MOVE IN FLOATING 0
6111 033266 000261              SEC ;SET CARRY
6112 033270 006100              ROL R0 ;GET 0 ONE BIT LEFT
6113 033272 103774              BCS 10$ ;BRANCH IF CARRY SET
6114
6115 033274 004037 042536      JSR R0,@#CLAREA ;FILL REST OF BUFFER WITH 1
6116 033300 002210              .WORD WRFROM+<32.*2> ;FROM
6117 033302 003106              .WORD WRFROM+776 ;TO
6118 033304 000001              .WORD 1 ;WITH DATA
6119
6120 ;*NOW THE READ BUFFER WILL BE CHECKED
6121
6122 033306 004037 043514      JSR R0,@#COMPAR ;CHECK
6123 033312 002110              WRFROM ;GOOD BUFFER
6124 033314 003154              REINTO ;TEST BUFFER
6125 033316 000400              256. ;NUMBER OF WORDS CHECKED
6126 033320 033326              11$ ;RETURN POINT FOR ERROR HEADER
6127 033322 033332              12$ ;RETURN POINT FOR ERROR DATA
6128
6129 033324 033340              TST75 ;RETURN FOR GOOD COMPARISON
6130
6131 033326 104004              11$: ERROR 4 ;READ NEXT ERROR 5
6132 033330 000207              RTS PC ;RETURN TO COMPARISON SUBROUTINE
6133 033332 104005              12$: ERROR 5 ;DATA IN REINTO BUFFER GOT
6134 ;CHANGED AFTER A WRITE
6135 ;CHECK DATA COMMAND
6136 ;WORD NO CONTAINS THE WORD
6137 ;NUMBER THAT GOT CHANGED
6138 033334 000207              RTS PC ;RETURN TO COMPARISON SUBROUTINE
6139
6140 033336 000240              13$: NOP ;ONLY A BRANCH POINT
6141
6142
6143
```

```
6144
6145 033340 000004          TST75: SCOPE
6146
6147
6148 033342 012706 001000          MOV    #STACK,SP          ;RESET STACK
6149
6150 033346 012737 000075 002032          MOV    #TTNO,@#TSTNM      ;THIS SAVES TEST NUMBER
6151 033354 012746 125252          MOV    #125252,-(SP)      ;DATA TO BE READ
6152 033360 012705 000400          MOV    #256.,R5           ;COUNTER
6153 033364 012700 051422          MOV    #DISK,R0           ;START OF SIMULATED DISK DATA
6154 033370 011620          1$: MOV    (SP),(R0)+         ;MOVE IN DATA ON TO SIMULATED DISK
6155 033372 005305          DEC    R5                 ;COUNT
6156 033374 001375          BNE    1$                 ;BRANCH IF 256 NOT COMPLETE
6157 033376 005726          TST    (SP)+              ;UNDO -(SP)
6158 033400 012705 000021          MOV    #17.,R5           ;2 ECC WORDS
6159 033404 005020          2$: CLR    (R0)+             ;CLEAR ECC, DATA GAP, AND
6160 033406 005305          DEC    R5                 ;TOLERANCE GAP
6161 033410 001375          BNE    2$                 ;BRANCH IF NOT COMPLETE
6162 033412 012737 000000 047504          MOV    #0!,@#CYL ;16 BITS PER WORD
6163 033420 112737 000001 047507          MOV    #1,@#SECTR+1 ;TRACK 1
6164 033426 112737 000000 047506          MOV    #0,@#SECTR        ;SECTOR 0
6165 033434 012737 000000 047510          MOV    #0,@#KEY1         ;KEY1=0
6166 033442 012737 000000 047512          MOV    #0,@#KEY2         ;KEY2=0
6167 033450 012737 000004 047564          MOV    #4.,@#DAWORD      ;NO. OF DATA WORDS
6168 033456 005037 047514          CLR    @#X                ;THIS IS A READ COMMAND
6169 033462 004537 044026          JSR    R5,@#CRC           ;GO TO CALCULATE CRC
6170 033466 047504          CYL
6171 033470 051404          WCRC
6172 033472 004737 042620          JSR    PC,@#CLDISK        ;SETUP GENERAL REGISTERS
6173 033476 012777 177770 146126          MOV    #-4.-4.,@RHWC      ;4. DATA 4 HEADER WORDS
6174 033504 012777 003154 146122          MOV    #REINTO,@RHBA     ;STARTING ADDRESS OF READ BUFFER
6175 033512 112746 000000          MOV    #0,-(SP)          ;IN LOWER BYTE GET SECTOR
6176 033516 112766 000001 000001          MOV    #1,1(SP)          ;GET TRACK IN HIGHER BYTE
6177 033524 012677 146114          MOV    (SP)+,@RHDST       ;TRACK/SECTOR IN RHDST
6178 033530 012777 014000 146112          MOV    #FMT22!ECI,@RHOF ;16 BITS PER WORD
6179 033536 005077 146110          CLR    @RHCA              ;CYLINDER 0
6180 033542 004737 042654          JSR    PC,@#CHECKT        ;CHECK THAT DVA,RDY,DPR,DRY = 1
6181 033546 104401 062450          TYPE  ,CPHALT            ;AND THAT NO OTHERS = 1. CANNOT CON-
6182 033552 000000          HALT                      ;STOP THE TEST
6183 033554 013711 002070          MOV    @#REFOR,@R1        ;READ HEADER AND DATA=72
6184 033560 005037 002006          CLR    @#ERFLG$          ;CLEAR ERROR FLAG
6185 033564 004737 047344          JSR    PC,@#COMHD         ;READ HEADER AND DATA
6186 033570 017737 146036 001126          MOV    @RHWC,$BDDAT      ;LOAD AND TEST RHWC
6187 033576 001401          BEQ    20$                ;SHOULD = 0
6188 033600 104040          ERROR 40                 ;RHWC DOES NOT = 0 AFTER A READ
6189 033602 005737 002006          20$: TST    @#ERFLG$        ;ANY ERRORS ALREADY THERE
6190 033606 001055          BNE    TST76              ;BRANCH IF YES
6191 033610 004737 043044          JSR    PC,@#CHECKE        ;CHECK THAT BITS = 1
6192 033614 104401 062450          TYPE  ,CPHALT            ;CANNOT CONTINUE TESTING IF THEY DON'T
6193 033620 000000          HALT                      ;STOP THE TEST
6194 033622 012700 002110          MOV    #WRFROM,R0         ;GETTING READY TO FILL EXPECTED DATA
6195 033626 012720 000000          MOV    #0,(R0)+          ;CYLINDER 0
6196 033632 112746 000000          MOV    #0,-(SP)          ;IN LOWER BYTE GET SECTOR
6197 033636 112766 000001 000001          MOV    #1,1(SP)          ;GET TRACK IN HIGHER BYTE
6198 033644 012620          MOV    (SP)+,(R0)+       ;GET TRACK/SECTOR IN BUFFER
6199 033646 012720 000000          MOV    #0,(R0)+          ;KEY1 IN BUFFER
```

```
6200 033652 012720 000000      MOV    #0,(R0)+      ;KEY2 IN BUFFER
6201 033656 012701 000400      MOV    #256,R1      ;DATA WORD COUNTER
6202 033662 012702 125252      MOV    #125252,R2   ;DATA
6203 033666 010220          3$:  MOV    R2,(R0)+     ;DATA INTO BUFFER
6204 033670 005301          DEC    R1            ;COUNT
6205 033672 001375          BNE    3$           ;BRANCH IF 256 NOT DONE
6206 033674 004037 043514      JSR    R0,@#COMPAR  ;CHECK
6207 033700 002110          WRFROM             ;GOOD BUFFER
6208 033702 003154          REINTO            ;TEST BUFFER
6209 033704 000010          4+4.             ;NUMBER OF WORDS CHECKED
6210 033706 033714          4$               ;RETURN POINT FOR ERROR HEADER
6211 033710 033720          5$               ;RETURN POINT FOR ERROR DATA
6212 033712 033724          6$               ;RETURN FOR GOOD COMPARISON
6213 033714 104004          4$:  ERROR 4          ;READ NEXT ERROR
6214 033716 000207          RTS    PC          ;RETURN TO 'COMPAR'
6215 033720 104005          5$:  ERROR 5          ;WORD NOS 1 TO 4 ARE
6216 033722 000207          RTS    PC          ;RETURN TO 'COMPAR'
6217
6218
6219
6220 033724 004737 042224          6$:  JSR    PC,@#PUTREG ;SAVE REGISTERS
6221
6222 033730 022737 100020 001716    CMP    #FER!DCK,@#ER1 ;FORMAT ERROR SHOULD BE SET
6223 033736 001401          BEQ    TST76       ;BRANCH IF GOOD
6224 033740 104020          ERROR 20          ;A 16 BIT PER WORD READ WAS ATTEMPTED
6225
6226
6227
6228
6229
6230
6231
6232
6233
```

```
6234
6235 033742 000004          TST76: SCOPE
6236                          : *NOW A WRITE DATA WILL BE ATTEMPTED WITH
6237                          : *WRONG FORMAT BIT
6238
6239 033744 012706 001000    MOV      #STACK,SP          ;RESET STACK
6240
6241 033750 012737 000076 002032  MOV      #TTNO,@#TSTNM      ;THIS SAVES TEST NUMBER
6242
6243 033756 012737 177777 047620  MOV      #-1,@#NOSYNC       ;SET FLAG SO THAT DATA SYNC
6244                          :AND DATA IS NOT READ
6245 033764 004037 042536    FRMAT1: JSR      R0,@#CLAREA   ;CLEAR SIMULATED DISK
6246 033770 051422          .WORD   DISK                ;FROM
6247 033772 052446          .WORD   TOLGAP+16          ;TO
6248 033774 000000          .WORD   0                  ;DATA
6249                          : *THESE ARE SETUP FOR DISKLESS USE ONLY
6250 033776 005037 047504    CLR      @#CYL              ;CYLINDER 0, FORMAT 18 BIT WORDS
6251 034002 105037 047507    CLR     @#SECOTR+1         ;TRACK 0
6252 034006 105037 047506    CLR     @#SECOTR          ;SECTOR 0
6253 034012 005037 047510    CLR     @#KEY1            ;KEY1 0
6254 034016 005037 047512    CLR     @#KEY2            ;KEY2 0
6255 034022 012737 000004 047552  MOV      #4,@#NOWORD       ;NO OF DATA WORDS
6256 034030 012737 000001 047514  MOV      #1,@#X            ;WRITE DATA
6257 034036 004537 044026    JSR     R5,@#CRC           ;GO TO CALCULATE CRC
6258 034042 052642          WCYL
6259 034044 052652          GCRC
6260
6261                          : *THESE AER REGULAR SETUPS
6262
6263 034046 004037 042536    JSR     R0,@#CLAREA        ;FILL WRITE FROM BUFFER WITH 125252
6264 034052 002110          WRFROM                      ;FROM
6265 034054 002116          WRFROM+6                    ;TO
6266 034056 125252          125252                      ;DATA
6267 034060 004737 042620    JSR     PC,@#CLDISK        ;SETUP GENERAL REGISTERS
6268 034064 012777 177774 145540  MOV      #-4,@#RHWC         ;256 DATA WORDS
6269 034072 012777 002110 145534  MOV      #WRFROM,@#RHBA    ;STARTING ADDRESS OF WRITE BUFFER
6270 034100 005077 145540    CLR     @#RHDST            ;TRACK=0 SECTOR=0
6271 034104 012777 010000 145536  MOV      #FMT22,@#RHOF     ;16 BITS PER WORD FORMAT
6272 034112 005077 145534    CLR     @#RHCA             ;CYLINDER 0
6273 034116 004737 042654    JSR     PC,@#CHECKT        ;CHECK THAT DVA,RDY,DPR,DRY = 1
6274 034122 104401 062450    TYPE   ,CPHALT            ;AND THAT NO OTHERS = 1. CANNOT CON-
6275 034126 000000          HALT                        ;STOP THE TEST
6276 034130 013711 002062    MOV     @#WRIDAT,@R1       ;WRITE DATA=60
6277 034134 005037 002006    CLR     @#ERFLG$          ;CLEAR ERROR FLAG
6278 034140 004737 047344    JSR     PC,@#COMHD         ;WRITE DATA
6279
6280                          : *IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
6281                          : *FROM THE 'COMHD' ROUTINE IT MEANS SECTOR GAP, SYNC BYTE
6282                          : *HEADER, HEADER CRC, HEADER GAP AND SYNC BYTE HAVE GONE BY
6283                          : *AND SYNC'S WERE CORRECTLY DETECTED
6284                          : *DATA IS TO BE CHECKED
6284 034144 004737 042224    JSR     PC,@#PUTREG        ;SAVE REGISTERS
6285 034150 005737 002006    TST     @#ERFLG$          ;HAS ANY ERRORS OCCURED?
6286 034154 001041          BNE     4$                 ;BRANCH IF YES
6287 034156 012700 000000    MOV     #0,R0              ;GOOD DATA
6288 034162 012701 051422    MOV     #DISK,R1           ;DATA WRITTEN INTO 'DISK'
6289 034166 012702 000004    MOV     #4,R2              ;COUNTER
```

```
6290 034172 012737 000005 047624 1$: MOV #5,@#ERWORD ;FOR ERROR WORD
6291 034200 020021 CMP R0,(R1)+ ;COMPARE GOOD DATA WITH DATA ON DISK
6292 034202 001424 BEQ 3$ ;BRANCH IF GOOD
6293 034204 010037 001124 MOV R0,@#$GDDAT ;GOOD DATA
6294 034210 014137 001126 MOV -(R1),@#$BDDAT ;BAD DATA
6295 034214 160237 047624 SUB R2,@#ERWORD ;ERROR WORD NO
6296 034220 005737 002006 TST @#ERFLG$ ;ANY ERRORS ALREADY THERE?
6297 034224 001002 BNE 2$ ;BRANCH IF YES
6298 034226 104004 ERROR 4 ;ERROR ON WRITE DATA COMMAND
6299 ;ON A WRITE DATA WITH
6300 ;WRONG FORMAT NO DATA
6301 ;SHOULD BE WRITTEN
6302 ;WORD NO GIVES WORD IN ERROR
6303 034230 000401 BR 5$ ;BRANCH TO AVOID PRINTING NEXT ERROR
6304 034232 104005 2$: ERROR 5
6305 034234 005721 5$: TST (R1)+ ;UNDO -(R1) FOR BAD DATA
6306 034236 017746 144676 MOV @SWR,-(SP) ;GET SWITCH SETTING
6307 034242 042716 177177 BIC #177177,(SP) ;KEEP ONLY SWITCH 7 AND 8
6308 034246 022726 000200 CMP #SW07,(SP)+ ;IS 7 SET AND 8 RESET.
6309 034252 001402 BEQ 4$ ;BRANCH IF YES
6310 034254 005302 3$: DEC R2 ;IF NOT COUNT 256 WORDS
6311 034256 001345 BNE 1$ ;BRANCH IF 256 NOT DONE
6312
6313 ;*NOW CHECK TO SEE THAT FORMAT ERROR BIT GOT SET
6314
6315 034260 022737 000020 001716 4$: CMP #FER,@#ER1 ;FORMAT ERROR SHOULD BE SET
6316 034266 001401 BEQ TST77 ;BRANCH IF GOOD
6317 034270 104020 ERROR 20 ;A 16 BIT PER WORD WRITE DATA
6318 ;WAS ATTEMPTED WHEN THE DISK
6319 ;HAD THE FORMAT BIT =0=18
6320 ;BITS PER WORD THE WRITE
6321 ;WAS CORRECTLY ABORTED
6322 ;BUT ERROR REG. 1 WAS WRONG
6323
```

```
6324
6325 034272 000004          TST77: SCOPE
6326
6327 034274 012706 001000    MOV    #STACK,SP          ;RESET STACK
6328 034300 012737 000077 002032  MOV    #TTNO,@#TSTNM      ;THIS SAVES TEST NUMBER
6329 034306 004737 042620    JSR    PC,@#CLDISK        ;CLEAR DISK
6330 034312 004737 042654    JSR    PC,@#CHECKT        ;CHECK THAT DVA,RDY,DPR,DRY = 1
6331 034316 104401 062450    TYPE   ,CPHALT           ;AND THAT NO OTHERS = 1.  CANNOT CON-
6332 034322 000000           HALT                       ;STOP THE TEST
6333 034324 012700 001652    MOV    #RHCA,R0
6334 034330 012005           MOV    (R0)+,R5           ;R5 HAS ADDRESS OF REG. UNDER TEST
6335 034332 052777 000040 145276 1$:  BIS    #CLR,@RHCS2
6336 034340 013777 001774 145270 2$:  MOV    @#UNIT,@RHCS2      ;REINSTATE UNIT NO.
6337
6338 ;*SET UP FOR AN OPERATION (WRITE HEADER AND DATA)
6339
6340 034346 013777 002064 145264    MOV    @#WRIFOR,@RHCS1    ;WRITE HEADER AND DATA=62
6341 ;IN RHCS1
6342 034354 012777 177766 145250    MOV    #-10,@RHWC         ;10 WORDS
6343 034362 012777 002110 145244    MOV    #WRFROM,@RHBA     ;BUS ADDRESS = WRFROM
6344 034370 012777 000010 145246    MOV    #10,@RHDS1        ;DESIRED TRACK=0, SECTOR=10
6345 034376 052777 000010 145232    BIS    #BAI,@RHCS2       ;BUS ADDRESS INCREMENT INHIBIT
6346 034404 012777 010000 145236    MOV    #FMT22,@RHOF      ;FORMAT 16 BIT WORDS
6347 034412 005077 145234    CLR    @RHCA              ;CYLINDER =0
6348
6349 ;*SAVE REGISTERS
6350
6351 034416 004037 043312    JSR    R0,@#SAVER         ;SAVE
6352 034422 001640           RHCS1                     ;FROM
6353 034424 003154           REINTO                    ;TO
6354 034426 000016           14.                       ;NUMBER OF REGISTERS SAVED
6355
6356 ;*NOW THE COMMAND IS GIVES TO
6357 ;*WRITE HEADER AND DATA FOR CYL=0, SECTOR=10
6358 ;*TRACK=0 IT COMES BACK AFTER ONE SECTOR
6359 ;*HAS PASSED
6360
6361 034430 012777 000001 145222    MOV    #DMD,@RHMR        ;SET DIAGNOSTIC MODE
6362 034436 005277 145176    INC    @RHCS1            ;GO TO RHCS1 WITH 62
6363 034442 012715 177672    MOV    #177672,@R5       ;TRY WRITING ALL BITS EXCEPT
6364 ;GO, RMR, IE
6365 034446 052737 000001 003174    BIS    #DMD,@#REINTO+20  ;SET DMD IN SAVED REGISTER RHMR
6366 034454 052737 000004 003156    BIS    #RMR,@#REINTO+2  ;SET RMR IN SAVED REG. RHER1
6367 034462 042737 000200 003176    BIC    #DRY,@#REINTO+22  ;CLEAR DRY IN RHDS1
6368 034470 052737 040000 003176    BIS    #ERR,@#REINTO+22  ;SET ERR IN RHDS1
6369 034476 052737 000001 003154    BIS    #GO,@#REINTO      ;SET GO IN SAVED REG. RHCS1
6370 034504 042737 000200 003154    BIC    #RDY,@#REINTO     ;CLEAR RDY BIT
6371
6372 ;*AFTER AN ATTEMPT TO WRITE INTO A REGISTER
6373 ;*SAVE REGISTERS AGAIN SO THAT COMPARES CAN BE DONE
6374
6375 034512 004037 043312    JSR    R0,@#SAVER         ;SAVE
6376 034516 001640           RHCS1                     ;FROM
6377 034520 002110           WRFROM                    ;TO
6378 034522 000016           14.                       ;NUMBER
6379
```

```
6380
6381
6382
6383 034524 113737 003173 002127
6384
6385
6386
6387
6388 034532 004037 043514
6389 034536 003154
6390 034540 002110
6391 034542 000016
6392 034544 034552
6393 034546 034552
6394 034550 034572
6395 034552 013705 047624 4$:
6396 034556 060505
6397 034560 016537 001636 042270
6398 034566 104001
6399 034570 000207
6400
6401
6402
6403
6404 034572 004737 042620 5$:
6405
6406
```

```
;*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+17,@#WRFROM+17;SAVE UPPER RHAS

;*COMPARE REGISTERS BEFORE ATTEMPTED WRITE WITH AFTER

JSR RO,@#COMPAR ;COMPAR
REINTO ;GO BUFFER
WRFROM ;TEST BUFFER
14. ;NUMBER
4$ ;RETURN FOR ERROR
4$ ;SAME
5$ ;RETURN FOR GOOD COMPARISON
MOV @#ERWORD,R5 ;GETTING READY TO INDEX
ADD R5,R5 ;DOUBLE ERROR WORD
MOV RHCS1-2(R5),@#REGADR ;FAILING REG. ADDRESS
ERROR 1 ;CONTENTS OF REGISTER
RTS PC ;CHANGED WITH
;AN ATTEMPT TO WRITE
;DURING AN OPERATION

;*THE FOLLOWING CLEAR MAY SET THE ATA BIT BECAUSE GO IS HIGH
;*
JSR PC,@#CLDISK ;CLEAR DISK
```

```
6407
6408
6409 034576 000004          TST100: SCOPE
6410 034600 012706 001000    MOV      #STACK,SP      ;RESET STACK
6411 034604 012737 000100 002032  MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
6412 034612 004737 042620    JSR      PC,@#CLDISK    ;INIT DRIVE
6413 034616 012777 000001 145034  MOV      #DMD,@RHMR     ;SET DIAGNOSTIC MODE
6414 034624 004037 045172    JSR      RO,@#MAKECYL   ;SUBROUTINE TO GIVE A SEEK
6415 034630 000001          1                          ;CHANGE RHCC TO 1
6416
6417
6418 034632 000004          TST101: SCOPE
6419
6420 034634 012706 001000    MOV      #STACK,SP      ;RESET STACK
6421 034640 012737 000101 002032  MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
6422 034646 005037 047620    CLR      @#NOSYNC       ;SET FLAG SO THAT DATA SYNC
6423 034652 004737 044302    JSR      PC,@#SETDSK    ;SET UP SIMULATED DISK
6424 034656 004037 042536    JSR      RO,@#CLAREA    ;FILL REINTO BUFFER
6425 034662 003154          REINTO                    ;FROM LOCATION
6426 034664 004154          REINTO+<256.*2>         ;TO LOCATION
6427 034666 000000          0                          ;DATA
6428 034670 012700 002110    MOV      #WRFROM,RO     ;
6429 034674 012720 010000    MOV      #FMT22,(RO)+   ;10000 INTO WRFROM
6430 034700 012720 000401    MOV      #401,(RO)+     ;401=TRACK1,SECTOR1
6431 034704 012720 000001    MOV      #1,(RO)+       ;1 INTO WRFROM+
6432 034710 012720 000001    MOV      #1,(RO)+       ;1 INTO WRFROM+6
6433 034714 004037 042536    JSR      RO,@#CLAREA    ;FILL WRFROM
6434 034720 002120          WRFROM+10                ;FROM
6435 034722 003110          WRFROM+<256.*2>         ;TO
6436 034724 177400          177400                   ;DATA
6437 034726 004037 044430    JSR      RO,@#HCCRCE    ;
6438 034732 000072          72                        ;READ HEADER AND DATA
6439 034734 000001          1                          ;CYLINDER
6440 034736 000001          1                          ;SECTOR
6441 034740 000001          1                          ;TRACK
6442 034742 177400          -256.                    ;WORD COUNT
6443 034744 003154          REINTO                    ;RHBA BUFFER
6444 034746 000000          0                          ;READ
6445 034750 000001          1                          ;HEADER COMPARE
6446 034752 000240          1$:  NOP                 ;RETURN POINT FROM HCCRCE
6447
```



```
6448
6449 034754 000004          TST102: SCOPE
6450 034756 012706 001000    MOV      #STACK,SP      ;RESET STACK
6451 034762 012737 000102 002032    MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
6452 034770 004737 042620    JSR      PC,@#CLDISK    ;INIT DRIVE
6453 034774 012777 000001 144656    MOV      #DMD,@RHMR     ;SET DIAGNOSTIC MODE
6454 035002 004037 045172    JSR      RO,@#MAKECYL   ;SUBROUTINE TO GIVE A SEEK
6455 035006 000000          0                       ;CHANGE RHCC TO 0
6456
6457
6458 035010 000004          TST103: SCOPE
6459
6460 035012 012706 001000    MOV      #STACK,SP      ;RESET STACK
6461 035016 012737 000103 002032    MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
6462 035024 005037 047620    CLR      @#NOSYNC       ;SET FLAG SO THAT DATA SYNC
6463 035030 004737 044302    JSR      PC,@#SETDSK    ;SET UP SIMULATED DISK
6464 035034 004037 042536    JSR      RO,@#CLAREA    ;FILL REINTO BUFFER
6465 035040 003154          REINTO                 ;FROM LOCATION
6466 035042 004154          REINTO+<256.*2>      ;TO LOCATION
6467 035044 000000          0                       ;DATA
6468 035046 012700 002110    MOV      #WRFROM,RO     ;
6469 035052 012720 010000    MOV      #FMT22,(RO)+   ;10000 INTO WRFROM
6470 035056 012720 000401    MOV      #401,(RO)+     ;401=TRACK1,SECTOR1
6471 035062 012720 000001    MOV      #1,(RO)+       ;1 INTO WRFROM+
6472 035066 012720 000001    MOV      #1,(RO)+       ;1 INTO WRFROM+6
6473 035072 004037 042536    JSR      RO,@#CLAREA    ;FILL WRFROM
6474 035076 002120          WRFROM+10              ;FROM
6475 035100 003110          WRFROM+<256.*2>      ;TO
6476 035102 177400          177400                 ;DATA
6477 035104 004037 044430    JSR      RO,@#HCCRCE    ;
6478 035110 000072          72                     ;READ HEADER AND DATA
6479 035112 000000          0                       ;CYLINDER
6480 035114 000001          1                       ;SECTOR
6481 035116 000000          0                       ;TRACK
6482 035120 177400          -256.                  ;WORD COUNT
6483 035122 003154          REINTO                 ;RHBA BUFFER
6484 035124 000000          0                       ;READ
6485 035126 000001          1                       ;HEADER COMPARE
6486 035130 000240          1$: NOP                ;RETURN POINT FROM HCCRCE
6487
```

```
6488
6489 035132 000004          TST104: SCOPE
6490 035134 012706 001000      MOV      #STACK,SP          ;RESET STACK
6491 035140 012737 000104 002032  MOV      #TTNO,@#TSTNM      ;THIS SAVES TEST NUMBER
6492 035146 004737 042620      JSR      PC,@#CLDISK        ;INIT DRIVE
6493 035152 012777 000001 144500  MOV      #DMD,@RHMR         ;SET DIAGNOSTIC MODE
6494 035160 004037 045172      JSR      RO,@#MAKECYL       ;SUBROUTINE TO GIVE A SEEK
6495 035164 000001          1          ;CHANGE RHCC TO 1
6496
6497
6498 035166 000004          TST105: SCOPE
6499
6500 035170 012706 001000      MOV      #STACK,SP          ;RESET STACK
6501 035174 012737 000105 002032  MOV      #TTNO,@#TSTNM      ;THIS SAVES TEST NUMBER
6502 035202 012737 177777 047620  MOV      #-1,@#NOSYNC       ;SET FLAG SO THAT DATA SYNC
6503 035210 004737 044302      JSR      PC,@#SETDSK        ;SETUP SIMULATED DISK
6504 035214 004037 042536      JSR      RO,@#CLAREA        ;FILL WRFROM BUFFER
6505 035220 002110          WRFROM      ;FROM LOCATION
6506 035222 003110          WRFROM+<256.*2>          ;TO LOCATION
6507 035224 125252          125252          ;DATA
6508 035226 004037 042536      JSR      RO,@#CLAREA        ;FILL REINTO BUFFER
6509 035232 003154          REINTO      ;FROM LOCATION
6510 035234 004154          REINTO+<256.*2>          ;TO
6511 035236 177400          177400
6512 035240 004037 044430      JSR      RO,@#HCCRCE
6513 035244 000060          60          ;WRITE DATA
6514 035246 000001          1          ;CYLINDER
6515 035250 000001          1          ;SECTOR
6516 035252 000001          1          ;TRACK
6517 035254 177400          -256.        ;WORD COUNT
6518 035256 002110          WRFROM      ;RHBA BUFFER
6519 035260 000001          1          ;WRITE
6520 035262 000001          1          ;HEADER COMPARE
6521 035264 000240          1$:      NOP             ;RETURN POINT FROM HCCRCE
```



```
6558
6559
6560 035422 000004          TST110: SCOPE
6561
6562 035424 012706 001000    MOV      #STACK,SP      ;RESET STACK
6563 035430 012737 000110    MOV      #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
6564 035436 005037 047620    CLR      @#NOSYNC       ;SET FLAG SO THAT DATA SYNC
6565 035442 004737 044302    JSR      PC,@#SETDSK    ;SET UP SIMULATED DISK
6566 035446 005137 051404    COM      @#WCRC         ;CHANCE CRC TO GIVE HCRC
6567 035452 004037 042536    JSR      RO,@#CLAREA    ;FILL REINTO BUFFER
6568 035456 003154          REINTO                ;FROM LOCATION
6569 035460 004154          REINTO+<256.*2>      ;TO LOCATION
6570 035462 000000          0                     ;DATA
6571 035464 012700 002110    MOV      #WRFROM,RO     ;10000 INTO WRFROM
6572 035470 012720 010000    MOV      #FMT22,(RO)+  ;401=TRACK1,SECTOR1
6573 035474 012720 000401    MOV      #401,(RO)+    ;401=TRACK1,SECTOR1
6574 035500 012720 000001    MOV      #1,(RO)+      ;1 INTO WRFROM+
6575 035504 012720 000001    MOV      #1,(RO)+      ;1 INTO WRFROM+6
6576 035510 004037 042536    JSR      RO,@#CLAREA    ;FILL WRFROM
6577 035514 002120          WRFROM+10            ;FROM
6578 035516 003110          WRFROM+<256.*2>      ;TO
6579 035520 177400          177400              ;DATA
6580 035522 004037 044430    JSR      RO,@#HCCRCE
6581 035526 000072          72                  ;READ HEADER AND DATA
6582 035530 000000          0                   ;CYLINDER
6583 035532 000001          1                   ;SECTOR
6584 035534 000001          1                   ;TRACK
6585 035536 177400          -256.              ;WORD COUNT
6586 035540 003154          REINTO              ;RHBA BUFFER
6587 035542 000000          0                   ;READ
6588 035544 000000          0                   ;CRC ERROR
6589 035546 000240          1$: NOP            ;RETURN POINT FROM HCCRCE
6590
```


6619
6620
6621
6622 035654 005737 002036
6623 035660 001401
6624 035662 000402
6625 035664 000137 036340
6626 035670
6627
6628
6629
6630 035670 000004
6631 035672 012706 001000
6632 035676 012737 000112 002032
6633 035704 004737 042620
6634 035710 012777 000001 143742
6635 035716 004037 045172
6636 035722 001456
6637
6638
6639
6640
6641
6642 035724 000004
6643
6644 035726 012706 001000
6645 035732 012737 000113 002032
6646 035740 004037 042536
6647 035744 051422
6648 035746 052446
6649 035750 000000
6650 035752 012737 011456 047504
6651 035760 112737 000022 047507
6652 035766 112737 000025 047506
6653 035774 005037 047510
6654 036000 005037 047512
6655 036004 012737 000400 047552
6656 036012 012737 000001 047514
6657 036020 004537 044026
6658 036024 047504
6659 036026 051404
6660 036030 004037 042536
6661 036034 002110
6662 036036 003110
6663 036040 000377
6664 036042 004737 042620
6665 036046 012777 177400 143556
6666 036054 012777 002110 143552
6667 036062 012746 000025
6668 036066 112766 000022 000001
6669 036074 012677 143544
6670 036100 012777 010000 143542
6671 036106 012777 001456 143536
6672 036114 004737 042654
6673 036120 104401 062450
6674 036124 000000

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

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

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

TST113: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
JSR RO,@#CLAREA ;CLEAR SIMULATED DISK
.WORD DISK ;FROM
.WORD TOLGAP+16 ;TO
.WORD 0 ;DATA
MOV #814,!FMT22,@#CYL ;CYLINDER 814.
MOVB #18,@#SECOTR+1 ;TRACK 18.
MOVB #21,@#SECOTR ;SECTOR 21.
CLR @#KEY1 ;KEY1 0
CLR @#KEY2 ;KEY2 0
MOV #256,@#NOWORD ;NO OF DATA WORDS
MOV #1,@#X ;WRITE DATA
JSR R5,@#CRC ;GO TO CALCULATE CRC
CYL
WCRC
JSR RO,@#CLAREA ;FILL WRITE BUFFER WITH 377
WRFROM ;FROM LOCATION
WRFROM+<256.*2> ;TO LOCATION
377 ;DATA
JSR PC,@#CLDISK ;SETUP GENERAL REGISTERS
MOV #-256,@RHWC ;256. DATA WORDS
MOV #WRFROM,@RHBA ;STARTING ADDRESS OF WRITE BUFFER
MOV #21,-(SP) ;SECTOR 21.
MOVB #18,1(SP) ;TRACK 18.
MOV (SP)+,@RHDS ;SECTOR 21. TRACK 18.
MOV #FMT22,@RHOF ;16 BITS PER WORD FORMAT
MOV #814,@RHCA ;CYLINDER 814.
JSR PC,@#CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-
HALT ;STOP THE TEST

6675	036126	013711	002062		MOV	@#WRIDAT,@R1	:WRITE DATA=60	
6676	036132	005037	002006		CLR	@#ERFLG\$:CLEAR ERROR FLAG	
6677	036136	004737	047344		JSR	PC,@#COMHD	:WRITE DATA	
6678	036142	004737	042224		JSR	PC,@#PUTREG	:SAVE REGISTERS	
6679	036146	005737	002006		TST	@#ERFLG\$:HAVE ANY ERRORS OCCURED?	
6680	036152	001062			BNE	5\$:BRANCH IF YES	
6681	036154	012700	000377		MOV	#377,R0	:GOOD DATA	
6682	036160	012701	051422		MOV	#DISK,R1	:DATA WRITTEN INTO 'DISK'	
6683	036164	012702	000400		MOV	#256.,R2	:COUNTER	
6684	036170	012737	000401	047624	MOV	#256.+1,@#ERWORD	:FOR ERROR WORD	
6685	036176	020021			CMP	R0,(R1)+	:COMPARE GOOD DATA WITH DATA ON DISK	
6686	036200	001424			BEQ	3\$:BRANCH IF GOOD	
6687	036202	010037	001124		MOV	R0,@#\$GDDAT	:GOOD DATA	
6688	036206	014137	001126		MOV	-(R1),@#\$BDDAT	:BAD DATA	
6689	036212	160237	047624		SUB	R2,@#ERWORD	:ERROR WORD NO	
6690	036216	005737	002006		TST	@#ERFLG\$:ANY ERRORS ALREADY THERE?	
6691	036222	001002			BNE	2\$:BRANCH IF YES	
6692	036224	104004			ERROR	4	:ERROR ON WRITE DATA COMMAND	
6693	036226	000401			BR	64\$:BRANCH TO AVOID PRINTING NEXT ERROR	
6694	036230	104005		2\$:	ERROR	5	:WORD NO GIVES WORD IN ERROR	
6695	036232	005721		64\$:	TST	(R1)+	:UNDO -(R1) FOR BAD DATA	
6696	036234	017746	142700		MOV	@SWR,-(SP)	:GET SWITCH SETTING	
6697	036240	042716	177177		BIC	#177177,(SP)	:KEEP ONLY SWITCH 7 AND 8	
6698	036244	022726	000200		CMP	#SW07,(SP)+	:IS 7 SET AND 8 RESET	
6699	036250	001402			BEQ	4\$:BRANCH OUT IF YES	
6700	036252	005302		3\$:	DEC	R2	:IF NOT COUNT 256 WORDS	
6701	036254	001345			BNE	1\$:BRANCH IF 256. NOT DONE	
6702								
6703	036256	013746	001736	4\$:	MOV	@#DS1,-(SP)	:GET RHDS1	
6704	036262	042716	001000		BIC	#PROG,(SP)	:CLEAR PROG	
6705	036266	022726	002700		CMP	#LST!DPR!DRY!VV,(SP)+	:IS 'LST' HIGH ?	
6706	036272	001412			BEQ	5\$:BRANCH IF GOOD	
6707	036274	013737	001662	042270	MOV	@#RHDS1,@#REGADR	:FAILING REG. ADDRESS	
6708	036302	012737	002700	001124	MOV	#LST!DPR!DRY!VV,@#\$GDDAT	:GOOD DATA	
6709	036310	013737	001736	001126	MOV	@#DS1,@#\$BDDAT	:BAD DATA	
6710	036316	104001			ERROR	1	: 'LST' DID NOT SET AFTER :LAST SECTOR ON LAST TRACK :ON LAST CYLINDER WAS :WRITTEN :VV BIT #6 MAY OR MAY NOT BE HIGH	
6711								
6712								
6713								
6714								
6715	036320	013737	001640	036330	5\$:	MOV	@#RHCS1,@#6\$:SET UP 'WAT' SUBROUTINE
6716	036326	104415			WAT			
6717	036330	000000			0		:RHCS1 ADDRESS	
6718	036332	000200			RDY		:WAIT FOR READY	
6719								
6720	036334	000137	037004		JMP	@#CAT	:DON'T DO THE RP04 'LST' TEST FOLLOWING	
6721								

```
6722
6723
6724
6725 036340          DOG:
6726 036340 000004    TST114: SCOPE
6727 036342 012706 001000    MOV #STACK,SP ;RESET STACK
6728 036346 012737 000114 002032    MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
6729 036354 004737 042620    JSR PC,@#CLDISK ;INIT DRIVE
6730 036360 012777 000001 143272    MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE
6731 036366 004037 045172    JSR RO,@#MAKECYL ;SUBROUTINE TO GIVE A SEEK
6732 036372 000632          410. ;CHANGE RHCC TO 410.
6733
6734
6735
6736 036374 000004    TST115: SCOPE
6737
6738 036376 012706 001000    MOV #STACK,SP ;RESET STACK
6739 036402 012737 000115 002032    MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
6740 036410 004037 042536    JSR RO,@#CLAREA ;CLEAR SIMULATED DISK
6741 036414 051422          .WORD DISK ;FROM
6742 036416 052446          .WORD TOLGAP+16 ;TO
6743 036420 000000          .WORD 0 ;DATA
6744 036422 012737 010632 047504    MOV #410,!FMT22,@#CYL;CYLINDER 410.
6745 036430 112737 000022 047507    MOV #18,@#SECOTR+1;TRACK 18.
6746 036436 112737 000025 047506    MOV #21,@#SECOTR ;SECTOR 21.
6747 036444 005037 047510    CLR @#KEY1 ;KEY1 0
6748 036450 005037 047512    CLR @#KEY2 ;KEY2 0
6749 036454 012737 000400 047552    MOV #256,@#NOWORD ;NO OF DATA WORDS
6750 036462 012737 000001 047514    MOV #1,@#X ;WRITE DATA
6751 036470 004537 044026    JSR R5,@#CRC ;GO TO CALCULATE CRC
6752 036474 047504          CYL
6753 036476 051404          WCRC
6754 036500 004037 042536    JSR RO,@#CLAREA ;FILL WRITE BUFFER WITH 377
6755 036504 002110          WRFROM ;FROM LOCATION
6756 036506 003110          WRFROM+<256.*2> ;TO LOCATION
6757 036510 000377          377 ;DATA
6758 036512 004737 042620    JSR PC,@#CLDISK ;SETUP GENERAL REGISTERS
6759 036516 012777 177400 143106    MOV #-256,@RHWC ;256. DATA WORDS
6760 036524 012777 002110 143102    MOV #WRFROM,@RHBA ;STARTING ADDRESS OF WRITE BUFFER
6761 036532 012746 000025          MOV #21,-(SP) ;SECTOR 21.
6762 036536 112766 000022 000001    MOV #18,1(SP) ;TRACK 18.
6763 036544 012677 143074          MOV (SP)+,@RHDST ;SECTOR 21. TRACK 18.
6764 036550 012777 010000 143072    MOV #FMT22,@RHOF ;16 BITS PER WORD FORMAT
6765 036556 012777 000632 143066    MOV #410,@RHCA ;CYLINDER 410.
6766 036564 004737 042654          JSR PC,@#CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
6767 036570 104401 062450          TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-
6768 036574 000000          HALT ;STOP THE TEST
6769 036576 013711 002062          MOV @#WRIDAT,@R1 ;WRITE DATA=60
6770 036602 005037 002006          CLR @#ERFLG$ ;CLEAR ERROR FLAG
6771 036606 004737 047344          JSR PC,@#COMHD ;WRITE DATA
6772 036612 004737 042224          JSR PC,@#PUTREG ;SAVE REGISTERS
6773 036616 005737 002006          TST @#ERFLG$ ;HAVE ANY ERRORS OCCURED?
6774 036622 001062          BNE 5$ ;BRANCH IF YES
6775 036624 012700 000377          MOV #377,R0 ;GOOD DATA
6776 036630 012701 051422          MOV #DISK,R1 ;DATA WRITTEN INTO 'DISK'
6777 036634 012702 000400          MOV #256.,R2 ;COUNTER
```



```
6778 036640 012737 000401 047624 1$: MOV #256.+1,@#ERWORD;FOR ERROR WORD
6779 036646 020021 CMP R0,(R1);+ ;COMPARE GOOD DATA WITH DATA ON DISK
6780 036650 001424 BEQ 3$ ;BRANCH IF GOOD
6781 036652 010037 001124 MOV R0,@#$GDDAT ;GOOD DATA
6782 036656 014137 001126 MOV -(R1),@#$BDDAT ;BAD DATA
6783 036662 160237 047624 SUB R2,@#ERWORD ;ERROR WORD NO
6784 036666 005737 002006 TST @#ERFLG$ ;ANY ERRORS ALREADY THERE?
6785 036672 001002 BNE 2$ ;BRANCH IF YES
6786 036674 104004 ERROR 4 ;ERROR ON WRITE DATA COMMAND
6787 036676 000401 BR 64$ ;BRANCH TO AVOID PRINTING NEXT ERROR
6788 036700 104005 2$: ERROR 5 ;WORD NO GIVES WORD IN ERROR
6789 036702 005721 64$: TST (R1)+ ;UNDO -(R1) FOR BAD DATA
6790 036704 017746 142230 MOV @SWR,-(SP) ;GET SWITCH SETTING
6791 036710 042716 177177 BIC #177177,(SP) ;KEEP ONLY SWITCH 7 AND 8
6792 036714 022726 000200 CMP #SW07,(SP)+ ;IS 7 SET AND 8 RESET
6793 036720 001402 BEQ 4$ ;BRANCH OUT IF YES
6794 036722 005302 3$: DEC R2 ;IF NOT COUNT 256 WORDS
6795 036724 001345 BNE 1$ ;BRANCH IF 256. NOT DONE
6796
6797 036726 013746 001736 4$: MOV @#DS1,-(SP) ;GET RHDS1
6798 036732 042716 001000 BIC #PROG,(SP) ;CLEAR PROG BIT
6799 036736 022726 002700 CMP #LST!DPR!DRY!VV,(SP)+ ;IS 'LST' HIGH ?
6800 036742 001412 BEQ 5$ ;WAIT FOR 'RDY' IF GOOD
6801 036744 013737 001662 042270 MOV @#RHDS1,@#REGADR ;FAILING REG. ADDRESS
6802 036752 012737 002700 001124 MOV #LST!DPR!DRY!VV,@#$GDDAT ;GOOD DATA
6803 036760 013737 001736 001126 MOV @#DS1,@#$BDDAT ;BAD DATA
6804 036766 104001 ERROR 1 ;'LST' DID NOT SET AFTER
6805 ;LAST SECTOR ON LAST TRACK ON LAST
6806 ;CYLINDER WAS WRITTEN - 'VV' BIT #6
6807 ;MAY OR MAY NOT BE HIGH
6808
6809 036770 013737 001640 037000 5$: MOV @#RHCS1,@#6$ ;SET UP 'WAT' SUBROUTINE
6810 036776 104415 WAT ;RHCS1 ADDRESS
6811 037000 000000 6$: 0 ;WAIT FOR 'RDY' BIT
6812 037002 000200 RDY
```

```
6813
6814
6815 037004          CAT:
6816 037004 000004   TST116: SCOPE
6817 037006 012706 001000   MOV #STACK,SP ;RESET STACK
6818 037012 012737 000116 002032   MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
6819 037020 004737 042620   JSR PC,@#CLDISK ;INIT AND SET UP GENERAL REG. CORRES.
6820 037024 004037 042536   JSR R0,@#CLAREA ;CLEAR SIMULATED DISK
6821 037030 051422   .WORD DISK ;FROM
6822 037032 052446   .WORD TOLGAP+16 ;TO
6823 037034 000000   .WORD 0 ;DATA
6824
6825 ;*THESE ARE TO SETUP FOR DISKLESS USE ONLY
6826 ;*AND WILL HANDLE RP04 OR RP06 DRIVES
6827
6828 037036 005737 002036   TST @#RP06 ;MOVE RP06 FLAG TO ITSELF TO TEST
6829 037042 001404   BEQ 10$ ;TREAT DRIVE AS RP04 IF = 0
6830
6831 037044 012737 011456 047504   MOV #814.!FMT22,@#CYL;CYLINDER 814., 16 BITS PER WORD
6832 037052 000403   BR 11$ ;TREAT DRIVE AS RP06
6833
6834 037054 012737 010632 047504 10$:   MOV #410.!FMT22,@#CYL;CYLINDER 410., 16 BITS PER WORD
6835 ;TREAT DRIVE AS RP04
6836
6837 037062 112737 000022 047507 11$:   MOVB #18.,@#SECOTR+1 ;TRACK 18.
6838 037070 112737 000025 047506   MOVB #21.,@#SECOTR ;SECTOR 21.
6839 037076 005037 047510   CLR @#KEY1 ;KEY1 0
6840 037102 005037 047512   CLR @#KEY2 ;KEY2 0
6841 037106 012737 000400 047552   MOV #256.,@#NOWORD ;NO OF DATA WORDS
6842 037114 012737 000001 047514   MOV #1,@#X ;WRITE DATA
6843 037122 004537 044026   JSR R5,@#CRC ;GO TO CALCULATE CRC
6844 037126 047504   CYL
6845 037130 051404   WCRC
6846
6847 ;*THESE ARE REGULAR SETUPS
6848
6849 037132 004037 042536   JSR R0,@#CLAREA ;FILL WRITE BUFFER WITH 377
6850 037136 002110   WRFROM ;FROM
6851 037140 003110   WRFROM+<256.*2> ;TO
6852 037142 000377   377 ;DATA
6853 037144 004737 042620   JSR PC,@#CLDISK ;SETUP GENERAL REGISTERS
6854 037150 012777 177272 142454   MOV #-326.,@#RHWC ;326. DATA WORDS
6855 037156 012777 002110 142450   MOV #WRFROM,@#RHBA ;STARTING ADDRESS OF WRITE BUFFER
6856 037164 012746 000025   MOV #21.,-(SP) ;SECTOR 21.
6857 037170 112766 000022 000001   MOVB #18.,1(SP) ;TRACK 18.
6858 037176 012677 142442   MOV (SP)+,@#RHDST ;SECTOR 21. TRACK 18.
6859 037202 012777 010000 142440   MOV #FMT22,@#RHOF ;16 BITS PER WORD FORMAT
6860
6861 ;*CHECK TO SEE WHAT TYPE OF DRIVE IS BEING TESTED
6862 ;*AND LOAD CYLINDER ADDRESS REGISTER WITH THE PROPER NUMBER
6863
6864 037210 005737 002036   TST @#RP06 ;MOVE FLAG TO ITSELF TO TEST
6865 037214 001404   BEQ 12$ ;TREAT AS RP04 IF = 0
6866 037216 012777 001456 142426   MOV #814.,@#RHCA ;CYLINDER 814.
6867 037224 000403   BR 13$ ;TREAT AS RP06
6868 037226 012777 000632 142416 12$:   MOV #410.,@#RHCA ;CYLINDER 410.
```

```
6869 037234 13$:
6870
6871 037234 004737 042654 JSR PC,@#CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
6872 037240 104401 062450 TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-
6873 037244 000000 HALT ;STOP THE TEST
6874 037246 013711 002062 MOV @#WRIDAT,@R1 ;WRITE DATA=60
6875 037252 005037 002006 CLR @#ERFLG$ ;CLEAR ERROR FLAG
6876
6877 ;*THE REGISTERS WILL BE SAVED IN REINTO BUFFER
6878
6879 037256 004037 043312 JSR R0,@#SAVER ;SAVE
6880 037262 001632 RHWC ;FROM
6881 037264 003154 REINTO ;TO
6882 037266 000023 19. ;NUMBER SAVED
6883
6884 ;*GIVE WRITE DATA COMMAND
6885
6886 037270 004737 047344 JSR PC,@#COMHD ;WRITE DATA COMMAND
6887
6888 ;*CHANGE SAVED REGISTERS TO EXPECTED VALUES
6889
6890 037274 005737 002040 TST @#RH70 ;CHECK FOR RH70 CONTROLLER
6891 037300 001407 BEQ 8$ ;SKIP RH70 CODE AND DO RH11 IF NOT
6892
6893 037302 012737 177702 003154 MOV #-76,@#REINTO ;SAVED RHWC SHOULD BE = 76 (OCTAL)
6894 037310 012737 003130 003156 MOV #WRFROM+<2*256.>+<2*8.>,@#REINTO+2
6895 ;SAVED RHBA SHOULD BE WRFROM+256+8
6896 037316 000406 BR 9$ ;SKIP NEXT RH11 CODE
6897
6898 037320 012737 177774 003154 8$: MOV #-4,@#REINTO ;SAVED RHWC SHOULD BE = 4
6899 037326 012737 003314 003156 MOV #WRFROM+<2*256.>+<2*66.>,@#REINTO+2
6900 ;SAVED RHBA SHOULD BE WRFROM+256+66
6901
6902 037334 052737 000200 003160 9$: BIS #OR,@#REINTO+4 ;SAVED RHCS2
6903 037342 042737 000100 003160 BIC #IR,@#REINTO+4 ;SAVED RHCS2
6904 037350 052737 140000 003162 BIS #SC!TRE,@#REINTO+6;SAVED RHCS1 SHOULD HAVE 'SC' & 'TRE'
6905 037356 012737 001000 003164 MOV #AOE,@#REINTO+10 ;SAVED RHER1 SHOULD HAVE 'AOE'
6906 037364 017737 142254 003166 MOV @RHDST,@#REINTO+12;SAVED RHDST SHOULD HAVE=
6907 ;RHDST IS UNDEFINED
6908
6909 ;*CHECK TO SEE WHAT TYPE OF DRIVE IS BEING TESTED
6910 ;*AND SET UP CYLINDER ADDRESS ACCORDINGLY
6911
6912 037372 005737 002036 TST @#RP06 ;MOVE RP06 FLAG TO ITSELF TO TEST
6913 037376 001404 BEQ 14$ ;TREAT AS RP04 IF = 0
6914 037400 012737 001457 003174 MOV #815.,@#REINTO+20;SAVED DESIRED CYLINDER ADDRESS
6915 037406 000403 BR 15$ ;TREAT AS RP06
6916 037410 012737 000633 003174 14$: MOV #411.,@#REINTO+20;SAVED DESIRED CYLINDER ADDRESS
6917
6918 037416 013737 002016 003200 15$: MOV @#ATTENT,@#REINTO+24 ;SAVED RHAS SHOULD HAVE APPRO. BIT
6919 037424 052737 000001 003202 BIS #DMD,@#REINTO+26;SAVED RHMR
6920 037432 052737 142000 003204 BIS #ATA!ERR!LST,@#REINTO+30 ;SAVED RHDS1
6921
6922 ;*AFTER A WRITE DATA COMMAND WITH 'AOE' ERROR
6923 ;*SAVE REGISTERS AGAIN SO THAT COMPARES CAN BE DONE
6924
```

```
6925 037440 004037 043312 JSR R0,@#SAVER ;SAVE
6926 037444 001632 RHW C ;FROM
6927 037446 002110 WRFROM ;TO
6928 037450 000021 17. ;NUMBER OF REGISTERS SAVED
6929
6930 ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
6931 ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
6932 ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
6933
6934 037452 113737 003201 002135 MOV B @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS
6935
6936 ;*COMPARE REGISTERS BEFORE WRITE DATA COMMAND
6937 ;*WITH AFTER COMMAND
6938
6939 037460 004037 043514 JSR R0,@#COMPAR ;COMPARE
6940 037464 003154 REINTO ;GOOD BUFFER
6941 037466 002110 WRFROM ;TEST BUFFER
6942 037470 000021 17. ;NUMBER OF REGISTERS
6943 037472 037500 1$ ;RETURN FOR ERROR
6944 037474 037500 1$ ;SAME
6945 037476 037520 2$ ;RETURN FOR GOOD COMPARISON
6946
6947 037500 013705 047624 1$: MOV @#ERWORD,R5 ;GETTING READY TO INDEX
6948 037504 060505 ADD R5,R5 ;DOUBLE ERROR WORD
6949 037506 016537 001630 042270 MOV RHW C-2(R5),@#REGADR ;FAILING REG. ADDRESS
6950 037514 104001 ERROR 1 ;FORCED AOE ERROR CAUSED IMPROPER
6951 ;REGISTER CHANGE
6952 037516 000207 RTS PC ;RETURN FOR FURTHER COMPARISONS
6953 ;NO ERRORS
6954 037520 005037 002006 2$: CLR @#ERFLG$ ;CLEAR ERROR FLAG
6955
6956
6957 ;*DATA IS TO BE CHECKED HERE
6958
6959 037524 004737 042224 JSR PC,@#PUTREG ;SAVE REGISTERS
6960 037530 012700 000377 MOV #377,R0 ;GOOD DATA
6961 037534 012701 051422 MOV #DISK,R1 ;DATA WRITTEN INTO 'DISK'
6962 037540 012702 000400 MOV #256.,R2 ;COUNTER
6963 037544 012737 000400 047624 3$: MOV #256.,@#ERWORD ;FOR ERROR WORD
6964 037552 020021 CMP R0,(R1)+ ;COMPARE GOOD DATA WITH DATA ON DISK
6965 037554 001424 BEQ 6$ ;BRANCH IF GOOD
6966 037556 010037 001124 MOV R0,@#$GDDAT ;GOOD DATA
6967 037562 014137 001126 MOV -(R1),@#$BDDAT ;BAD DATA
6968 037566 160237 047624 SUB R2,@#ERWORD ;ERROR WORD NO
6969 037572 005737 002006 TST @#ERFLG$ ;ANY ERRORS ALREADY THERE?
6970 037576 001002 BNE 4$ ;BRANCH IF YES
6971 037600 104004 ERROR 4 ;ERROR ON WRITE DATA COMMAND WITH FORCED 'AOE'
6972 037602 000401 BR 5$ ;BRANCH TO AVOID PRINTING NEXT ERROR
6973 037604 104005 4$: ERROR 5 ;WORD NO. GIVES WORD IN ERROR
6974 037606 005721 5$: TST (R1)+ ;UNDO -(R1) FOR BAD DATA
6975 037610 017746 141324 MOV @SWR,-(SP) ;GET SWITCH SETTING
6976 037614 042716 177177 BIC #177177,(SP) ;KEEP ONLY SWITCH 7 AND 8
6977 037620 022726 000200 CMP #SW07,(SP)+ ;IS 7 SET AND 8 RESET
6978 037624 001402 BEQ 7$ ;BRANCH OUT IF YES ----->
6979
6980 037626 005302 6$: DEC R2 ;IF NOT COUNT 256 WORDS
```



```

6984
6985 037632 000004
6986 037634 012706 001000
6987 037640 012737 000117 002032
6988 037646 004737 042620
6989 037652 012777 000001 142000
6990 037660 004037 045172
6991 037664 000000
6992
6993
6994 037666 000004
6995 037670 012706 001000
6996 037674 012737 000120 002032
6997 037702 004737 042620
6998
6999
7000 037706 012777 000001 141744
7001 037714 052777 000004 141736
7002 037722 042777 000004 141730
7003
7004
7005
7006
7007 037730 012777 177400 141674
7008 037736 012700 003154
7009 037742 010077 141666
7010
7011 037746 012720 010000
7012
7013 037752 012720 012000
7014 037756 005020
7015 037760 005020
7016 037762 012705 000400
7017 037766 012720 177777
7018 037772 005305
7019 037774 001374
7020 037776 012777 012000 141640
7021
7022 040004 004737 042654
7023 040010 104401 062450
7024 040014 000000
7025
7026 040016 013711 002070
7027
7028 040022 005037 002006
7029 040026 012777 010000 141614
7030 040034 005077 141612
7031
7032
7033 040040 004037 043312
7034 040044 001632
7035 040046 003154
7036 040050 000023
7037
7038
7039

```

TST117: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
JSR PC,@#CLDISK ;INIT DRIVE
MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE
JSR R0,@#MAKECYL ;SUBROUTINE TO GIVE A SEEK
0 ;CHANGE RHCC TO 0

TST120: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
JSR PC,@#CLDISK ;CLEAR REGISTERS AND SET UNIT NO.
;*GIVE INDEX PULSE
MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE
BIS #MINX,@RHMR ;SET INDEX
BIC #MINX,@RHMR ;CLEAR INDEX

;*THESE ARE REGULAR SETUPS
MOV #-256,@RHWC ;256 DATA WORDS 4 HEADER WORDS
MOV #REINTO,R0 ;THESE TWO INSTRUCTIONS GETS
MOV R0,@RHBA ;ADDR, OF WRFROM INTO R0 AND
;BUS ADDRESS REGISTER
MOV #FMT22,(R0)+ ;FORMAT=16 BIT WORDS
;CYLINDER=0
MOV #12000,(R0)+ ;TRACK=20 SECTOR=0 KEYS=0
CLR (R0)+ ;KEY1=0
CLR (R0)+ ;KEY2=0
MOV #256,R5 ;COUNTER
1\$: MOV #-1,(R0)+ ;MOVE ALL ONES FOR DATA
DEC R5
BNE 1\$;BRANCH IF DATA NOT COMPLETE
MOV #12000,@RH DST ;TRACK=20 SECTOR=0
JSR PC,@#CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-
HALT ;STOP THE TEST
MOV @#REFOR,@R1 ;GET READY FOR WRITE HEADER AND
;DATA WITH 62 IN RHCS1
CLR @#ERFLG\$;CLEAR ERROR FLAG
MOV #FMT22,@RHOF ;FORMAT BIT=1 (16 BIT WORDS)
CLR @RHCA ;CYLINDER =0

;*THE REGISTERS WILL BE SAVED IN REINTO BUFFER
JSR R0,@#SAVER ;SAVE
RHWC ;FROM
REINTO ;TO
19. ;NUMBER SAVED

;*GO TO WRITE HEADER AND DATA

```
7040 040052 013700 001660 MOV @#RHMR,R0 ;NOW R0 WAS MAINTENANCE REG. ADDR.
7041 040056 012710 000001 MOV #DMD,@R0 ;SET DIAGNOSTIC MODE
7042 040062 052777 000001 141550 BIS #GO,@RHCS1 ;GO
7043
7044 ;*CHANGE SAVED REGISTERS TO EXPECTED VALUE
7045 040070 052737 140000 003162 BIS #SC!TRE,@#REINTO+6 ;SAVED RHCS1
7046 040076 012737 002000 003164 MOV #IAE,@#REINTO+10 ;SAVED RHER1
7047 040104 012737 012001 003166 MOV #12001,@#REINTO+12;SAVED RHDST
7048 040112 013737 002016 003200 MOV @#ATTENT,@#REINTO+24 ;SAVED RHAS
7049 040120 052737 000001 003202 BIS #DMD,@#REINTO+26 ;SAVED RHMR
7050 040126 052737 140000 003204 BIS #ATA!ERR,@#REINTO+30 ;SAVED RHDS1
7051
7052 ;*SAVE REGISTERS AGAIN SO THAT COMPARES CAN BE DONE
7053 040134 004037 043312 JSR R0,@#SAVER ;SAVE
7054 040140 001632 RHCW ;FROM
7055 040142 002110 WRFROM ;TO
7056 040144 000023 19. ;NUMBER OF REGISTERS SAVED
7057
7058 ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
7059 ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
7060 ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
7061 040146 113737 003201 002135 MOVB @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS
7062
7063
7064 ;*COMPARE REGISTERS BEFORE READ IN PRESET COMMAND
7065 ;*WITH AFTER COMMAND
7066 040154 004037 043514 JSR R0,@#COMPAR ;COMPARE
7067 040160 003154 REINTO ;GOOD BUFFER
7068 040162 002110 WRFROM ;TEST BUFFER
7069 040164 000021 17. ;NUMBER OF REGISTERS
7070 040166 040174 2$ ;RETURN FOR ERROR
7071 040170 040174 2$ ;SAME
7072 040172 040214 3$ ;RETURN FOR GOOD COMPARISON
7073
7074 040174 013705 047624 2$: MOV @#ERWORD,R5 ;GETTING READY TO INDEX
7075 040200 060505 ADD R5,R5 ;DOUBLE ERROR WORD
7076 040202 016537 001630 042270 MOV RHCW-2(R5),@#REGADR ;FAILING REG. ADDRESS
7077 040210 104001 ERROR 1 ;FORCED IAE CAUSED IMPROPER
7078 ;REGISTER CHANGE
7079 040212 000207 RTS PC ;RETURN FOR FURTHER COMPARISONS
7080
7081 ;NO ERRORS
7082
7083 040214 004737 042620 3$: JSR PC,@#CLDISK ;CLEAR GO BIT
7084
```

```
7085
7086 040220 000004
7087 040222 012706 001000
7088 040226 012737 000121 002032
7089 040234 004737 042620
7090
7091
7092 040240 012777 000001 141412
7093 040246 052777 000004 141404
7094 040254 042777 000004 141376
7095
7096
7097
7098 040262 012777 177400 141342
7099 040270 012700 002110
7100 040274 010077 141334
7101
7102 040300 012720 010000
7103
7104 040304 012720 000026
7105 040310 005020
7106 040312 005020
7107 040314 012705 000400
7108 040320 012720 177777 1$:
7109 040324 005305
7110 040326 001374
7111 040330 012777 000026 141306
7112
7113 040336 004737 042654
7114 040342 104401 062450
7115 040346 000000
7116
7117 040350 013711 002064
7118
7119 040354 005037 002006
7120 040360 012777 010000 141262
7121 040366 005077 141260
7122
7123
7124
7125
7126
7127
7128
7129 040372 004037 043312
7130 040376 001642
7131 040400 003154
7132 040402 000015
7133
7134
7135
7136 040404 013700 001660
7137 040410 012710 000001
7138 040414 052777 000001 141216
7139
7140
```

TST121: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
JSR PC,@#CLDISK ;CLEAR REGISTERS AND SET UNIT NO.
;*GIVE INDEX PULSE
MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE
BIS #MINX,@RHMR ;SET INDEX
BIC #MINX,@RHMR ;CLEAR INDEX
;*THESE ARE REGULAR SETUPS
MOV #-256.,@RHWC ;256 DATA WORDS 4 HEADER WORDS
MOV #WRFROM,R0 ;THESE TWO INSTRUCTIONS GETS
MOV R0,@RHBA ;ADDR. OF WRFROM INTO R0 AND
;BUS ADDRESS REGISTER
MOV #FMT22,(R0)+ ;FORMAT=16 BIT WORDS
;CYLINDER=0
MOV #22.,(R0)+ ;TRACK=0, SECTOR=22, KEYS=0
CLR (R0)+ ;KEY1=0
CLR (R0)+ ;KEY2=0
MOV #256.,R5 ;COUNTER
MOV #-1,(R0)+ ;MOVE ALL ONES FOR DATA
DEC R5
BNE 1\$;BRANCH IF DATA NOT COMPLETE
MOV #22.,@RHDST ;TRACK=0 SECTOR=22
JSR PC,@#CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-
HALT ;STOP THE TEST
MOV @#WRIFOR,@R1 ;GET READY FOR WRITE HEADER AND
;DATA WITH 62 IN RHCS1
CLR @#ERFLG\$;CLEAR ERROR FLAG
MOV #FMT22,@RHOF ;FORMA BIT=1 (16 BIT WORDS)
CLR @RHCA ;CYLINDER =0
;*AS EXCEPTION IS ASSERTED BEFORE RUN IS
;*LATCHED RHWC,RHBA,RHCS1,RHCS2 CANNOT BE CHECKED
;*BECAUSE RHWC WILL VARY DEPENDING UPON GATE DELAYS
;*ON DIFFERENT UNITS
;*THE REGISTERS WILL BE SAVED IN REINTO BUFFER
JSR R0,@#SAVER ;SAVE
RHER1 ;FROM
REINTO ;TO
13. ;NUMBER SAVED
;*GO TO WRITE HEADER AND DATA
MOV @#RHMR,R0 ;NOW R0 HAS MAINTENANCE REG. ADDR.
MOV #DMD,@R0 ;SET DIAGNOSTIC MODE
BIS #GO,@RHCS1 ;GO
;*CHANGE SAVED REGISTERS TO EXPECTED VALUE


```
7141 040422 012737 002000 003154      MOV    #IAE,@#REINTO    ;SAVED RHER1
7142 040430 012737 000027 003156      MOV    #23,@#REINTO+2;SAVED RHDST
7143 040436 013737 002016 003170      MOV    @#ATTENT,@#REINTO+14 ;SAVED RHAS
7144 040444 052737 000001 003172      BIS    #DMD,@#REINTO+16 ;SAVED RHMR
7145 040452 052737 140000 003174      BIS    #ATA!ERR,@#REINTO+20 ;SAVED RHDS1
7146
7147                                     ;*SAVE REGISTERS AGAIN SO THAT COMPARES CAN BE DONE
7148 040460 004037 043312      JSR    R0,@#SAVER      ;SAVE
7149 040464 001642                                     RHER1      ;FROM
7150 040466 002110      WRFROM      ;TO
7151 040470 000015      13.        ;NUMBER OF REGISTERS SAVED
7152
7153                                     ;*AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
7154                                     ;*OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
7155                                     ;*SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
7156 040472 113737 003171 002125      MOVB   @#REINTO+15,@#WRFROM+15;SAVE UPPER RHAS
7157
7158
7159                                     ;*COMPARE REGISTERS BEFORE READ IN PRESET COMMAND
7160                                     ;*WITH AFTER COMMAND
7161 040500 004037 043514      JSR    R0,@#COMPAR     ;COMPARE
7162 040504 003154      REINTO      ;GOOD BUFFER
7163 040506 002110      WRFROM      ;TEST BUFFER
7164 040510 000015      13.        ;NUMBER OF REGISTERS
7165 040512 040520      2$         ;RETURN FOR ERROR
7166 040514 040520      2$         ;SAME
7167 040516 040540      3$         ;RETURN FOR GOOD COMPARISON
7168
7169 040520 013705 047624      2$:      MOV    @#ERWORD,R5    ;GETTING READY TO INDEX
7170 040524 060505      ADD    R5,R5          ;DOUBLE ERROR WORD
7171 040526 016537 001640 042270      MOV    RHER1-2(R5),@#REGADR ;FAILING REG. ADDRESS
7172 040534 104001      ERROR    1          ;FORCED IAE CAUSED IMPROPER
7173                                     ;REGISTER CHANGE
7174 040536 000207      RTS    PC            ;RETURN FOR FURTHER COMPARISONS
7175
7176                                     ;NO ERRORS
7177
7178 040540 004737 042620      3$:      JSR    PC,@#CLDISK    ;CLEAR GO BIT
7179
7180
```

```
7181
7182 040544 000004          TST122: SCOPE
7183 040546 012706 001000  MOV      #STACK,SP      ;RESET STACK
7184 040552 012737 000122 002032  MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
7185 040560 004737 042620  JSR      PC,@#CLDISK    ;CLEAR REGISTERS AND SET UNIT NO.
7186
7187          ;*GIVE INDEX PULSE
7188 040564 012777 000001 141066  MOV      #DMD,@RHMR     ;SET DIAGNOSTIC MODE
7189 040572 052777 000004 141060  BIS      #MINX,@RHMR    ;SET INDEX
7190 040600 042777 000004 141052  BIC      #MINX,@RHMR    ;CLEAR INDEX
7191
7192          ;*THESE ARE REGULAR SETUPS
7193
7194 040606 012777 177400 141016  MOV      #-256.,@RHWC   ;256 DATA WORDS 4 HEADER WORDS
7195 040614 012700 002110  MOV      #WRFROM,R0    ;THESE TWO INSTRUCTIONS GETS
7196 040620 010077 141010  MOV      R0,@RHBA      ;ADDR. OF WRFROM INTO R0 AND
7197          ;BUS ADDRESS REGISTER
7198 040624 012705 000400  MOV      #256.,R5      ;COUNTER
7199 040630 012720 177777 1$: MOV      #-1,(R0)+    ;MOVE ALL ONES FOR DATA
7200 040634 005305  DEC      R5
7201 040636 001374  BNE     1$            ;BRANCH IF DATA NOT COMPLETE
7202 040640 012777 000000 140776  MOV      #0.,@RH DST   ;TRACK=0 SECTOR=0
7203
7204 040646 004737 042654  JSR      PC,@#CHECKT   ;CHECK THAT DVA,RDY,DPR,DRY = 1
7205 040652 104401 062450  TYPE    ,CPHALT       ;AND THAT NO OTHERS = 1. CANNOT CON-
7206 040656 000000  HALT
7207
7208 040660 013711 002062  MOV      @#WRIDAT,@R1  ;GET READY FOR WRITE
7209          ;DATA WITH 60 IN RHCS1
7210 040664 005037 002006  CLR      @#ERFLG$     ;CLEAR ERROR FLAG
7211 040670 012777 010000 140752  MOV      #FMT22,@RHOF  ;FORMA BIT=1 (16 BIT WORDS)
7212
7213 040676 005737 002036  TST      @#RP06 ;MOVE FLAG TO ITSELF TO TEST
7214 040702 001404  BEQ     4$            ;TREAT DRIVE AS RP04 IF FLAG = 0
7215
7216 040704 012777 001457 140740  MOV      #815.,@RHCA   ;CYLINDER = 815 (ONE TOO MANY)
7217 040712 000403  BR      5$            ;TREAT DRIVE AS RP06
7218
7219 040714 012777 000633 140730 4$: MOV      #411.,@RHCA   ;CYLINDER = 411 (ONE TOO MANY)
7220          ;TREAT DRIVE AS RP04
7221
7222          ;*AS EXCEPTION IS ASSERTED BEFORE RUN IS
7223          ;*LATCHED RHWC,RHBA,RHCS1,RHCS2 CANNOT BE CHECKED
7224          ;*BECAUSE RHWC WILL VARY DEPENDING UPON GATE DELAYS
7225          ;*ON DIFFERENT UNITS
7226
7227          ;*THE REGISTERS WILL BE SAVED IN REINTO BUFFER
7228 040722 004037 043312 5$: JSR      R0,@#SAVER    ;SAVE
7229 040726 001642  RHER1    ;FROM
7230 040730 003154  REINTO   ;TO
7231 040732 000015  13.     ;NUMBER SAVED
7232
7233          ;*GO TO WRITE HEADER AND DATA
7234
7235 040734 013700 001660  MOV      @#RHMR,R0    ;NOW R0 HAS MAINTENANCE REG. ADDR.
7236 040740 012710 000001  MOV      #DMD,@R0    ;SET DIAGNOSTIC MODE
```

7
7
7
7
7
7
7
7
7
7
7


```
7278 041074 000004
7279 041076 012737 000001 001212
7280 041104 012737 000000 177776
7281 041112 104401 041120
7282 041116 000425
7283 041172 013746 001774
7284 041176 104405
7285 041200 104401 041206
7286 041204 000402
7287 041212 013746 001112
7288 041216 104405
7289 041220 005037 001112
7290 041224 005037 001102
7291 041230 005737 002002
7292
7293 041234 001413
7294
7295 041236 005237 001100
7296 041242 104401 041425
7297 041246 013746 001100
7298 041252 104405
7299 041254 104401 041422
7300 041260 000137 007544
7301
7302 041264 005337 001776
7303 041270 001413
7304 041272 013700 001774
7305 041276 012701 001754
7306 041302 022100
7307 041304 001401
7308 041306 000775
7309 041310 011137 001774
7310 041314 000137 007544
7311
7312
7313
7314
7315
7316
7317
7318 041320 000004
7319 041322 005037 001102
7320 041326 005037 001212
7321 041332 005237 001100
7322 041336 042737 100000 001100
7323 041344 005327
7324 041346 000001
7325 041350 003022
7326 041352 012737
7327 041354 000001
7328 041356 041346
7329 041360 104401 041425
7330 041364 013746 001100
7331 041370 104405
7332 041372 104401 041422
7333 041376 013700 000042

TST123: SCOPE
MOV #1,$TIMES ;;DO 1 ITERATION
MOV #0,PS ;;REINSTATE PS TO 0
TYPE ,65$ ;;TYPE ASCIZ STRING
BR 64$ ;;GET OVER THE ASCIZ
MOV @#UNIT,-(SP) ;;GET READY TO TYPE UNIT NUMBER
TYPDS
TYPE ,67$ ;;TYPE ASCIZ STRING
BR 66$ ;;GET OVER THE ASCIZ
MOV @#$ERTTL,-(SP) ;;GET READY TO TYPE NUMBER OF ERRORS
TYPDS
CLR @#$ERTTL ;;CLEAR TOTAL NUMBER OF ERRORS
CLR @#$TSTNM ;;CLEAR TEST NUMBER
TST @#$SELECT ;;STARTING FROM 210 ?

BEQ 3$ ;;TEST NEXT DRIVE IF NOT
;;CONTINUE ON THIS ONE IF SO
INC @#$PASS ;;INCREASE PASS COUNT
TYPE ,SENDMG ;;TYPE END PASS #
MOV @#$PASS,-(SP)
TYPDS
TYPE ,$ENULL
JMP @#TST5 ;;DO NEXT TESTS ----->

3$: DEC @#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 ;;GO FOR NEXT TESTS ----->

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

SCOPE
CLR $TSTNM ;;ZERO THE TEST NUMBER
CLR $TIMES ;;ZERO THE NUMBER OF ITERATIONS
INC $PASS ;;INCREMENT THE PASS NUMBER
BIC #100000,$PASS ;;DON'T ALLOW A NEG. NUMBER
DEC (PC)+ ;;LOOP?
$EOPCT: .WORD 1
BGT $DOAGN ;;YES
MOV (PC)+,@(PC)+ ;;RESTORE COUNTER
$ENDCT: .WORD 1
$EOPCT
TYPE ,SENDMG ;;TYPE 'END PASS #'
MOV $PASS,-(SP) ;;SAVE $PASS FOR TYPEOUT
TYPDS ;;GO TYPE--DECIMAL ASCII WITH SIGN
TYPE ,$ENULL ;;TYPE A NULL CHARACTER
$GET42: MOV @#42,R0 ;;GET MONITOR ADDRESS
```

7334 041402 001405
7335 041404 000005
7336 041406 004710
7337 041410 000240
7338 041412 000240
7339 041414 000240
7340 041416 000137
7341 041420 005500
7342 041422 377 377 000
7343 041425 015 042412 042116
7344 041432 050040 051501 020123
7345 041440 000043
7346
7347
7348
7349
7350
7351
7352
7353
7354
7355
7356
7357
7358
7359
7360
7361
7362
7363
7364
7365
7366
7367
7368 041442 000000
7369
7370 041444
7371 041444 005037 177776
7372 041450 104401 041456
7373 041454 000421
7374 041520 013746 002032
7375 041524 104402
7376 041526 104401 041534
7377 041532 000414
7378 041564 013746 001110
7379 041570 104402
7380 041572 104401 001223
7381 041576 104401 041604
7382 041602 000430
7383 041664 104401 041672
7384 041670 000430
7385 041752 104401 041760
7386 041756 000422
7387 042024 104412
7388 042026 062716 000002
7389 042032 012637 001106

BEQ \$DOAGN ;;BRANCH IF NO MONITOR
RESET ;;CLEAR THE WORLD
\$ENDAD: JSR PC,(R0) ;;GO TO MONITOR
NOP ;;SAVE ROOM
NOP ;;FOR
NOP ;;ACT11
JMP @ (PC)+ ;;RETURN
\$RTNAD: .WORD TST1
\$ENULL: .BYTE -1,-1,0 ;;NULL CHARACTER STRING
\$ENDMG: .ASCIZ <15><12>/END PASS #/

;**HERE IS A DETAILED EXPLANATION OF HOW THE LOOP ON ERROR WORKS.
;**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.

TESTAD: 0 ;FIRST ADDRESS OF TEST

OPERSEL:

CLR PS ;MAKE PROCESSOR STATUS ZERO
TYPE ,65\$;;TYPE ASCIZ STRING
BR 64\$;;GET OVER THE ASCIZ
MOV @#TSTNM,-(SP) ;GET READY TO TYPE TEST
TYPOC ;NUMBER
TYPE ,67\$;;TYPE ASCIZ STRING
BR 66\$;;GET OVER THE ASCIZ
MOV @#\$LPERR,-(SP) ;GET READY TO TYPE LOOP BACK PC
TYPOC
TYPE , \$CRLF
TYPE ,69\$;;TYPE ASCIZ STRING
BR 68\$;;GET OVER THE ASCIZ
TYPE ,71\$;;TYPE ASCIZ STRING
BR 70\$;;GET OVER THE ASCIZ
TYPE ,73\$;;TYPE ASCIZ STRING
BR 72\$;;GET OVER THE ASCIZ
RDOCT
ADD #2,(SP) ;GET LPADR
MOV (SP)+,@#\$LPADR

7390	042036	104401	042044	TYPE	75\$::TYPE ASCIZ STRING
7391	042042	000417		BR	74\$::GET OVER THE ASCIZ
7392	042102	104401	042110	TYPE	77\$::TYPE ASCIZ STRING
7393	042106	000440		BR	76\$::GET OVER THE ASCIZ
7394	042210	104412		RDOCT		
7395	042212	012637	001110	MOV	(SP)+, @#\$LPERR	;GET LPERR
7396	042216	013746	001106	MOV	@#\$LPADR, -(SP)	
7397	042222	000002		RTI		

7398
7399
7400

.SBTTL SAVE REGISTERS ROUTINE

7401
7402
7403
7404
7405
7406
7407
7408
7409
7410
7411

::*THIS SAVES THE CONTENTS OF ALL HARDWARE REGISTERS
::*IN MEMORY LOCATIONS TAGED FROM 'WC' TO 'EC2'
::*
::*
::*THIS IS DONE SO THAT COMPARES ARE DONE WITH SAVED LOCATIONS
::*AND NOT THE REGISTERS THEMSELVES. THIS WILL MAKE
::*ERROR PRINTOUTS FOR GOOD AND BAD DATA ALWAYS DIFFRENT

7412	042224		
7413	042224	010046	
7414	042226	010146	
7415	042230	010246	
7416	042232	012700	001632
7417	042236	012701	001706
7418	042242	012702	000023
7419	042246	013021	
7420	042250	005302	
7421	042252	001375	
7422	042254	012602	
7423	042256	012601	
7424	042260	012600	
7425	042262	000207	

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

7426
7427
7428
7429
7430

.SBTTL FLOAT 1 AND 0

7431
7432
7433
7434
7435
7436

::*FLOAT A ONE AND A ZERO THRU A DESIGNATED REGISTER
::*ABSOLUTE ADDRESS OF REG. UNDER TEST IS IN R4

7437	042264	000000	
7438	042266	000000	
7439	042270	000000	
7440			
7441	042272	012537	042264
7442	042276	012504	
7443	042300	010437	042270
7444	042304	010537	042266
7445	042310	062705	000004

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

```
7446 042314 012703 000001      MOV    #1,    R3      ;INITIALIZE DATA PATTERN
7447 042320 004737 042342      BLT1: JSR    PC,    BLT2 ;OUTPUT FLOATING ZERO
7448 042324 004737 042342      JSR    PC,    BLT2 ;OUTPUT FLOATING ONE
7449 042330 000241      CLC
7450 042332 006103      ROL    R3      ;SHIFT PATTERN
7451 042334 005703      TST    R3
7452 042336 001370      BNE    BLT1    ;BRANCH IF NOT COMPLETE
7453 042340 000205      RTS
7454 042342 005103      BLT2: COM    R3      ;COMPLEMENT PATTERN
7455 042344 012737 042352 042600      MOV    #BLT3, @#LAD ;SET SCOPE LOOP
7456 042352 032777 001000 136560      BLT3: BIT    #SW09,@SWR ;LOOP ON ERROR
7457 042360 001411      BEQ    4$      ;BRANCH IF NO
7458 042362 105737 001103      TSTB  @#$ERFLG ;ANY ERRORS
7459 042366 001406      BEQ    4$      ;BRANCH IF NO
7460 042370 000005      RESET ;START WITH AN INIT
7461 042372 013777 001774 137236      MOV    @#UNIT,@RHCS2 ;SET UNIT NUMBER UNDER TEST
7462 042400 004737 055176      JSR    PC,@#STKINT ;INITILIZE TK
7463                                     ;INIT FOR SCOPING LOOPS
7464 042404 010337 001124      4$:  MOV    R3,@#$GDDAT ;STORE GOOD DATA
7465 042410 005137 042264      COM    @#MASK    ;AND MASK WITH PATTERN
7466 042414 043737 042264 001124      BIC    @#MASK, @#$GDDAT ;CLEAR THE REST
7467 042422 005137 042264      COM    @#MASK    ;RESTORE MASK
7468 042426 013714 001124      MOV    @#$GDDAT,(R4) ;OUTPUT TO REGISTER
7469 042432 011437 001126      MOV    (R4),@#$BDDAT ;INPUT FROM REGISTER
7470 042436 005137 042264      COM    @#MASK
7471 042442 043737 042264 001126      BIC    @#MASK,@#$BDDAT ;AND MASK OUT RECEIVED DATA
7472 042450 005137 042264      COM    @#MASK    ;RESTORE MASK
7473 042454 023737 001124 001126      CMP    @#$GDDAT,@#$BDDAT ;IS DATA CORRECT
7474 042462 001424      BEQ    1$      ;BRANCH IF GOOD
7475 042464 011437 001126      MOV    (R4),@#$BDDAT
7476 042470 023704 001640      CMP    @#RHCS1,R4 ;REGISTER UNDER TEST RHCS1?
7477 042474 001004      BNE    2$      ;BRANCH IF NOT
7478 042476 052737 004200 001124      BIS    #RDY!DVA,@#$GDDAT ;SET RDY AND DVA
7479 042504 000410      BR     3$
7480 042506 023704 001636      2$:  CMP    @#RHCS2,R4 ;REGISTER UNDER TEST RHCS2?
7481 042512 001005      BNE    3$      ;BRANCH IF NOT
7482 042514 011446      MOV    @R4,-(SP) ;GET RHCS2
7483 042516 042716 177477      BIC    #^C<IR!OR>,(SP) ;KEEP IR AND OR BIT
7484 042522 052637 001124      BIS    (SP)+,@#$GDDAT ;SET IR OR BITS IF NEEDED
7485 042526 004777 177534      3$:  JSR    PC,    @LERR ;GO TO REPORT ERROR
7486 042532 000240      NOP      ;REPLACE BY 104420 FOR LOCAL SCOPE LOOP
7487 042534 000207      1$:  RTS    PC
```

```
7488
7489
7490      .SBTTL  CLEAR MEMORY ROUTINE
7491
7492      ;*      THIS CLEARS ANY BLOCK OF MEMORY
7493      ;*      FILLING IT WITH ANY DATA
7494      ;*
7495      ;*      CALL
7496      ;*      JSR    R0,CLAREA
7497      ;*      X      ;STARTING ADDRESS OF BLOCK
7498      ;*      Y
7499      ;*      Z      ;DATA TO BE FILLED
7500      ;*R1 WILL HAVE STARTING ADDRESS OF BLOCK TO BE FILLED
7501      ;*R2 AFTER SUBTRACTION WILL HAVE TWICE NUMBER OF LOCATIONS
```

```
7502      ;*R3 WILL HAVE DATA TO BE FILLED
7503      ;*TO AVOID DIVIDE ROUTINE TWO DECREMENT R2 WILL BE USED
7504
7505
7506 042536      CLAREA:
7507 042536 010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
7508 042540 010246      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
7509 042542 010346      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
7510 042544 012001      MOV      (R0)+,R1      ;FROM
7511 042546 012002      MOV      (R0)+,R2      ;TO
7512 042550 012003      MOV      (R0)+,R3      ;DATA
7513 042552 160102      SUB      R1,R2      ;NO. OF LOCATIONS MINUS TWO
7514 042554 062702 000002      ADD      #2,R2      ;GET TWICE NO OF LOCATIONS
7515 042560 010321      1$: MOV      R3,(R1)+      ;MOVE IN DATA
7516 042562 005302      DEC      R2
7517 042564 005302      DEC      R2
7518 042566 001374      BNE      1$          ;BRANCH IF NOT COMPLETE
7519 042570 012603      MOV      (SP)+,R3      ;;POP STACK INTO R3
7520 042572 012602      MOV      (SP)+,R2      ;;POP STACK INTO R2
7521 042574 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
7522 042576 000200      RTS      R0          ;RETURN
```



```
7523          .SBTTL LOCAL TRAPS
7524 042600 000000          LAD: 0
7525
7526 042602 032777 001000 136330 T.SCOPI: BIT    #SW09, @SWR
7527 042610 001402          BEQ    1$
7528 042612 013716 042600          MOV    @#LAD, (SP)
7529 042616 000002          1$: RTI
7530
7531          ;*EXAMPLE OF THE USE OF THE ABOVE
7532          ;*THIS WILL LOOP BETWEEN X: AND SCOP1 PROVIDED THERE IS NO 'NEWTST'
7533          ;*MOV    #X,    @#LAD
7534          ;*X:    ---    ---
7535          ;*      ---    ---
7536          ;*      ---    ---
7537          ;*      SCOP1
7538
7539          .SBTTL CLEAR DISK ROUTINE
7540
7541          CLDISK: MOV    @#RHCS1,    R1    ;R1 WILL BE CONTROL AND STATUS1
7542 042620 013701 001640          MOV    @#RHCS2,    R2    ;R2 WILL BE CONTROL AND STATUS2
7543 042624 013702 001636          MOV    @#RHDS1,    R3    ;R3 WILL BE DISK STATUS REGISTER1
7544 042630 013703 001662          MOV    @#RER1,    R4    ;R4 WILL BE ERROR REGISTER #1
7545 042634 013704 001642
7546
7547 042640 012712 000040          MOV    #CLR,@R2    ;CLEAR ALL REG.
7548 042644 013712 001774          MOV    @#UNIT,@R2 ;REINSTATE UNIT NO.
7549 042650 005011          CLR    @R1    ;CLEAR FUNCTION BITS
7550 042652 000207          RTS    PC
```

.SBTTL CHECK DISK STATUS ROUTINES

```

7551
7552
7553
7554
7555
7556      ;*THIS CHECKS THAT DEVICE AVAILABLE (DVA) AND READY (RDY) IN RHCS1 = 1
7557      ;*AND CHECKS THAT DEVICE PRESENT (DPR), DEVICE READY (DRY) IN RHDS1 = 1
7558      ;*IT ALSO CHECKS THAT NO OTHER BITS IN THESE REGISTERS = 1
7559
7560 042654 011637 002014 CHECKT: MOV (SP),@#PCJSR ;SAVE PC OF JSR+4
7561 042660 162737 000004 002014 SUB #4,@#PCJSR ;GET PC OF JSR
7562 042666 004737 042224 JSR PC,@#PUTREG ;SAVE REGISTERS
7563 042672 022737 004200 001714 CMP #DVA!RDY,@#CS1 ;RHCS1 SHOULD HAVE DEVICE AVAILABLE
7564 ;AND BE READY
7565 042700 001423 BEQ 3$ ;BRANCH IF GOOD
7566
7567 042702 032737 004000 001714 BIT #DVA,@#CS1 ;BAD SO TEST DEVICE AVAILABLE
7568 042710 001004 BNE 1$ ;BRANCH IF DVA THERE
7569 042712 010137 001122 MOV R1,@#$BDADR ;ADDRESS OF BAD REGISTER (RHCS1)
7570 042716 104026 ERROR 26 ;RHCS1 DID NOT HAVE DEVICE
7571 ;AVAILABLE AT START OF TEST
7572 042720 000413 BR 3$ ;BRANCH TO NEXT COMPARE
7573 042722 032737 000200 001714 1$: BIT #RDY,@#CS1 ;TEST READY
7574 042730 001003 BNE 2$ ;IF RDY THERE BRANCH
7575 042732 010137 001122 MOV R1,@#$BDADR ;ADDRESS OF BAD REGISTER (RHCS1)
7576 042736 104026 ERROR 26 ;RHCS1 DID NOT HAVE READY
7577 ;RIGHT AT START OF TEST
7578 042740 000403 BR 3$ ;BRANCH TO NEXT COMPARE
7579 042742 010137 001122 2$: MOV R1,@#$BDADR ;ADDRESS OF BAD REGISTER (RHCS1)
7580 042746 104026 ERROR 26 ;RHCS1 HAD SOME BITS OTHER
7581 ;THAN DVA AND RDY SET
7582 ;ALL OTHER BITS SHOULD BE 0
7583
7584 042750 013746 001736 3$: MOV @#DS1,-(SP) ;GET RHDS1
7585 042754 042716 001100 BIC #VV!PROG,(SP) ;CLEAR VV AND PROGRAMABLE BIT
7586 042760 022726 000600 CMP #DPR!DRY,(SP)+;RHDS1 SHOULD HAVE THESE SET
7587 042764 001424 BEQ 8$ ;BRANCH IF THEY ARE
7588
7589 042766 032737 000400 001736 4$: BIT #DPR,@#DS1 ;TEST DRIVE PRESENT
7590 042774 001004 BNE 5$ ;CONTINUE IF THERE
7591 042776 010337 001122 MOV R3,@#$BDADR ;ADDRESS OF BAD REGISTER (RHDS1)
7592 043002 104026 ERROR 26 ;RHDS1 DOES NOT HAVE DPR
7593 043004 000413 BR 7$ ;BRANCH OUT
7594 043006 032737 000200 001736 5$: BIT #DRY,@#DS1 ;TEST DRIVE READY
7595 043014 001004 BNE 6$ ;IF DPR WAS THERE, BRANCH IF GOOD
7596 043016 010337 001122 MOV R3,@#$BDADR ;ADDRESS OF BAD REGISTER (RHDS1)
7597 043022 104026 ERROR 26 ;RHDS1 DOES NOT HAVE DRY
7598 043024 000403 BR 7$ ;BRANCH OUT
7599 043026 010337 001122 6$: MOV R3,@#$BDADR ;ADDRESS OF BAD REGISTER (RHDS1)
7600 043032 104026 ERROR 26 ;RHDS1 HAS SOME BITS OTHER
7601 ;THAN MOL, DRY, DPR, SET
7602 ;ALL OTHER BITS SHOULD BE 0
7603 043034 000207 7$: RTS PC ;RETURN TO TEST AND HALT
7604
7605 043036 062716 000006 8$: ADD #6,(SP) ;ADJUST STACK TO GET OVER HALT IN TEST
7606 043042 000207 RTS PC ;RETURN TO TEST AND CONTINUE TESTING
  
```

CZRJGCO.RP04/5/6 DSKLS CTRLR1
CZRJGC.P11 26-JUL-78 10:10

MACY11 30A(1052) 27-JUL-78 12:41 I 14
CHECK DISK STATUS ROUTINES PAGE 179

SEQ 0178
SEQ 0177

7607

```
7608
7609
7610      ;*THIS CHECKS THAT DEVICE AVAILABLE (DVA) AND READY (RDY) IN RHCS1 = 1
7611      ;*AND CHECKS THAT DEVICE PRESENT (DPR), DEVICE READY (DRY) IN RHDS1 = 1
7612 043044 011637 002014      CHECKE: MOV      (SP),@#PCJSR      ;SAVE PC OF JSR+4
7613 043050 162737 000004 002014      SUB      #4,@#PCJSR      ;GET PC OF JSR
7614 043056 004737 042224      JSR      PC,@#PUTREG      ;READ & SAVE REGISTERS
7615 043062 032737 000200 001714      BIT      #RDY,@#CS1      ;RHCS1 SHOULD HAVE DEVICE AVAILABLE
7616      ;AND BE READY
7617 043070 001004      BNE      1$              ;BRANCH IF GOOD
7618 043072 010137 001122      MOV      R1,@#$BDADR      ;FAILING REGISTER
7619 043076 104026      ERROR   26              ;RHCS1 IS IN ERROR
7620      ;DOES NOT HAVE DVA, RDY
7621 043100 000427      BR       4$              ;BRANCH OUT
7622 043102 032737 004000 001714 1$:   BIT      #DVA,@#CS1      ;RHCS1 SHOULD HAVE DEVICE AVAILABLE
7623      ;AND BE READY
7624 043110 001004      BNE      2$              ;BRANCH IF GOOD
7625 043112 010137 001122      MOV      R1,@#$BDADR      ;FAILING REGISTER
7626 043116 104026      ERROR   26              ;RHCS1 IS IN ERROR
7627      ;DOES NOT HAVE DVA, RDY
7628 043120 000417      BR       4$              ;BRANCH OUT
7629 043122 032737 000200 001736 2$:   BIT      #DRY,@#DS1      ;RHDS1 SHOULD HAVE DPR,DRY
7630 043130 001004      BNE      3$              ;BRANCH IF THERE
7631 043132 010337 001122      MOV      R3,@#$BDADR      ;FAILING REGISTER RHDS1
7632 043136 104026      ERROR   26              ;RHDS1 DOES NOT HAVE DPR,DRY
7633 043140 000407      BR       4$              ;BRANCH OUT
7634 043142 032737 000400 001736 3$:   BIT      #DPR,@#DS1      ;RHDS1 SHOULD HAVE DPR,DRY
7635 043150 001004      BNE      5$              ;BRANCH OUT AND CONTINUE IF THERE
7636 043152 010337 001122      MOV      R3,@#$BDADR      ;FAILING REGISTER RHDS1
7637 043156 104026      ERROR   26              ;RHDS1 DOES NOT HAVE DPR,DRY
7638
7639      4$:   RTS      PC              ;RETURN TO TEST AND HALT
7640
7641      5$:   ADD      #6,(SP)      ;ADJUST STACK TO GET OVER HALT IN TEST
7642 043166 000207      RTS      PC              ;RETURN TO TEST AND CONTINUE TESTING
```

```
7643
7644
7645      :*      WAIT LOOP
7646      :*      ONE LOOP OR ONE COUNT = 5.15 MICROSEC WITH BIPOLAR MEMORY (MIN)
7647      :*      ONE LOOP OR ONE COUNT = 11.86 MICROSEC WITH CORE (MIN)
7648      :*      WITH CORE ERROR IS INDICATED AFTER ABOUT 650 MILLISEC (MIN)
7649
7650 043170 177777      TIMCNT: 177777      ;WAITING COUNT
7651 043172 010046      WAIT.T: MOV      R0,-(SP)      ;SAVE R0
7652 043174 016600 000002      MOV      2(SP),R0      ;GET ADDRESS OF REG. ADDRSS
7653 043200 010037 001204      MOV      R0,@#$TMP3      ;WAT PC+2 IN $TMP3
7654 043204 162737 000002 001204      SUB      #2,@#$TMP3      ;WAT PC FOR TYPEOUT
7655 043212 012037 001176      MOV      (R0)+,@#$TMP0      ;WAIT REGISTER ADDRESS
7656 043216 012037 001200      MOV      (R0)+,@#$TMP1      ;WAIT ON BIT
7657 043222 010066 000002      MOV      R0,2(SP)      ;RESTORE RETURN ON STACK
7658 043226 012600      MOV      (SP)+,R0      ;RESTORE R0
7659 043230 013737 043170 001202      MOV      @#TIMCNT,@#$TMP2      ;TEMPORARY COUNT
7660 043236 033777 001200 135732 1$:      BIT      @#$TMP1,@$TMP0      ;IS REQUIRED BIT THERE?
7661 043244 001021      BNE      2$      ;BRANCH IF YES
7662 043246 005337 001202      DEC      @#$TMP2      ;COUNT
7663 043252 001371      BNE      1$      ;BRANCH IF NOT TIME UP
7664 043254 013737 043170 001202      MOV      @#TIMCNT,@#$TMP2      ;TEMPORARY COUNT
7665 043262 033777 001200 135706 3$:      BIT      @#$TMP1,@$TMP0      ;IS REQUIRED BIT THERE?
7666 043270 001007      BNE      2$      ;BRANCH IF YES
7667 043272 005337 001202      DEC      @#$TMP2      ;COUNT
7668 043276 001371      BNE      3$      ;BRANCH IF NOT TIME UP
7669 043300 017737 135672 001126      MOV      @$TMP0,@#$BDDAT ;REGISTER CONTENTS
7670 043306 104016      ERROR   16      ;WAITED ON BIT FAILED TO SET
7671 043310 000002      2$:      RTI
7672
7673
7674
7675      :*      CALL FOR THE ABOVE WAITLOOP IS
7676      :*
7677      :*      MOV      @A,@#X$      ;A CONTAINS REGISTER ADDRESS
7678      :*      -      -      -      ;HENCE X$ WILL HAVE ABSOLUTE REG. ADR.
7679      :*      -      -      -
7680      :*      -      -      -
7681      :*      WAT
7682      :*X$: 0      ;ABSOLUTE REG. ADDRESS UNDER WAIT
7683      :*      .WORD 0      ;BIT WAITED FOR
7684      :*
7685
```

7686
 7687
 7688
 7689
 7690
 7691
 7692
 7693
 7694
 7695
 7696

7697 043312
 7698 043312 010146
 7699 043314 010246
 7700 043316 010346
 7701 043320 012001
 7702 043322 012002
 7703 043324 012003
 7704 043326 013122
 7705 043330 005303
 7706 043332 001375
 7707 043334 012603
 7708 043336 012602
 7709 043340 012601
 7710 043342 000200

7711
 7712
 7713
 7714
 7715
 7716
 7717
 7718
 7719

7720
 7721 043344 012737 010000 047504
 7722 043352 112737 000001 047507
 7723 043360 112737 000001 047506
 7724 043366 005037 047510
 7725 043372 005037 047512
 7726 043376 012737 000044 047564
 7727 043404 005037 047514
 7728 043410 004537 044026
 7729 043414 047504
 7730 043416 051404

7731
 7732

7733
 7734 043420 004737 042620
 7735
 7736 043424 012777 177730 136200
 7737 043432 012777 003154 136174
 7738 043440 112746 000001
 7739 043444 112766 000001 000001
 7740 043452 012677 136166
 7741 043456 012777 014000 136164

.SBTTL SAVE ROUTINE

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

SAVER:

```

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

.SBTTL WRITE CHECK ROUTINE

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

WRCHHD:

```

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

;*THESE ARE REGULAR SETUPS

```

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

7742							:ECC CORRECTION INHIBIT BECAUSE
7743							:ECC LOGIC IS NOT CHECKED YET
7744	043464	005077	136162	CLR	@RHCA		:CYLINDER=0
7745	043470	004737	042654	JSR	PC,@#CHECKT		:CHECK THAT DVA,RDY,DPR,DRY = 1
7746	043474	104401	062450	TYPE	,CPHALT		:AND THAT NO OTHERS = 1. CANNOT CON-
7747	043500	000000		HALT			:STOP THE TEST
7748	043502	013711	002060	MOV	@#WRCHDT,@R1		:WRITE CHECK HEADER AND DATA=52
7749							:INTO RHCS1
7750	043506	004737	047344	JSR	PC,@#COMHD		:WRITE CHECK HEADER AND DATA
7751							:SAME AS READ HEADER AND DATA
7752							
7753	043512	000207		RTS	PC		:RETURN TO WRITE CHECK TEST
7754							
7755							

.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:

7756						
7757						
7758						
7759						
7760						
7761						
7762						
7763						
7764						
7765						
7766						
7767						
7768						
7769	043514					
7770	043514	010146				
7771	043516	010246				
7772	043520	010346				
7773	043522	010446				
7774	043524	010546				
7775	043526	012001				
7776	043530	012002				
7777	043532	012003				
7778	043534	012037	001176			
7779	043540	012037	001200			
7780	043544	011000				
7781	043546	010304				
7782	043550	005204				
7783	043552	010437	047624			
7784	043556	022122				
7785	043560	001426				
7786						
7787	043562	014137	001124			
7788	043566	014237	001126			
7789	043572	160337	047624			
7790	043576	005737	002006			
7791	043602	001003				
7792	043604	004777	135366			
7793	043610	000402				
7794	043612	004777	135362			
7795	043616	022122				
7796	043620	017746	135314			
7797	043624	042716	177177			
7798	043630	022726	000200			
7799	043634	001402				
7800	043636	005303				
7801	043640	001344				
7802	043642					
7803	043642	012605				
7804	043644	012604				
7805	043646	012603				
7806	043650	012602				
7807	043652	012601				
7808	043654	000200				
7809						
7810						
7811						


```

7812
7813
7814
7815
7816
7817
7818 043656 012737 010000 047504 WRCHDA: MOV #FMT22,@#CYL ;CYLINDER 0 FORMAT 16 BIT WORDS
7819 043664 112737 000001 047507 MOVB #1,@#SECOTR+1 ;TRACK=1
7820 043672 112737 000001 047506 MOVB #1,@#SECOTR ;SECTOR=1
7821 043700 005037 047510 CLR @#KEY1 ;KEY1=0
7822 043704 005037 047512 CLR @#KEY2 ;KEY2=0
7823 043710 012737 000040 047564 MOV #32,@#DAWORD ;NO OF DATA WORDS
7824 043716 005037 047514 CLR @#X ;THIS IS A READ OPERATION
7825
7826 043722 004537 044026 JSR R5,@#CRC ;GO TO CALCULATE CRC
7827 043726 047504 CYL
7828 043730 051404 WCRC
7829
7830
7831
7832 043732 004737 042620 JSR PC,@#CLDISK ;SET UP GENERAL REGISTERS
7833 ;AND CLEAR DISK REGISTERS
7834
7835 043736 012777 177740 135666 MOV #-32,@#RHC ;36 DATA WORDS 4 HEADER WORDS
7836 043744 012777 003154 135662 MOV #REINTO,@#RHB ;STARTING ADDRESS OF READ BUFFER
7837 043752 112746 000001 MOVB #1,-(SP) ;SECTOR=1
7838 043756 112766 000001 000001 MOVB #1,1(SP) ;TRACK=1 IN UPPER BYTE
7839 043764 012677 135654 MOV (SP)+,@#RHDST ;TRACK=1, SECTOR=1 IN RHDST
7840 043770 012777 014000 135652 MOV #FMT22!ECI,@#RHOF ;16 BIT WORDS
7841 ;ECC CORRECTION INHIBIT BECAUSE
7842 ;ECC LOGIC IS NOT CHECKED YET
7843 043776 005077 135650 CLR @#RHCA ;CYLINDER=0
7844 044002 004737 042654 JSR PC,@#CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
7845 044006 104401 062450 TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-
7846 044012 000000 HALT ;STOP THE TEST
7847 044014 013711 002056 MOV @#WRCHK,@R1 ;WRITE CHECK DATA=50 INTO RHCS1
7848 044020 004737 047344 JSR PC,@#COMHD ;WRITE CHECK HEADER AND DATA
7849 ;SAME AS READ HEADER AND DATA
7850
7851 044024 000207 RTS PC ;RETURN TO WRITE CHECK TEST

```

7852
7853
7854
7855
7856
7857
7858
7859
7860
7861
7862
7863
7864
7865
7866
7867
7868
7869
7870
7871
7872
7873
7874
7875
7876
7877
7878
7879
7880
7881
7882
7883
7884
7885
7886
7887
7888
7889
7890
7891
7892
7893
7894
7895
7896
7897
7898
7899
7900
7901
7902
7903
7904
7905
7906
7907

.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
:*$TMP0 - NUMBER OF WORDS
:*$TMP2 - NUMBER OF BITS PER WORD = 16
:*$TMP3 - TEMPORARY REG.
:*$TMP4 - TEMPORARY REG TO TRANSFER CARRY
:*$TMP5 - 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
:*REPET 64 TIMES
:*CALL JSR R5,@#CRC
:*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 @#$TMP5
MOV #4,@#$TMP0 ;:WORD COUNT
MOV (R0)+,@#$TMP3 ;:TEMPORARY WORD STORAGE
MOV #16,$TMP2 ;:BIT COUNT
MOV @#$TMP3,@#$TMP4 ;:TEMPORARY WORD STORAGE
ROR @#$TMP3 ;:GET LSB INTO 'C'
ROR @#$TMP5 ;:GET ABOVE 'C' INTO $TMP5
BIT #BIT0,R1 ;:IS POSITION 15 HIGH
BEQ 1$ ;:BRANCH IF POSITION 16 LOW
MOV #BIT15,R3 ;:GET POSITION 16
BR 2$
1$: CLR R3 ;:GET POSITION 16
2$: ADD @#$TMP5,R3 ;:XOR POSITION 16 WITH D
;:TO GIVE B
BIT #BIT14,R1 ;:IS POSITION 2 HIGH
BEQ 3$ ;:BRANCH IF POSITION 2 LOW
MOV #BIT15,R2 ;:GET POSITION 2
BR 4$
3$: CLR R2 ;:GET POSITION 2
    
```

```

044026
044026 010046
044030 012500
044032 010146
044034 010246
044036 010346
044040 010446
044042 005001
044044 005037 001210
044050 012737 000004 001176
044056 012037 001204
044062 012737 000020 001202
044070 013737 001204 001206
044076 006037 001204
044102 006037 001210
044106 032701 000001
044112 001403
044114 012703 100000
044120 000401
044122 005003
044124 063703 001210
044130 032701 040000
044134 001403
044136 012702 100000
044142 000401
044144 005002
    
```

```

7908 044146 060302      4$:  ADD      R3,P2      ;XOR B WITH POSITION 2
7909                                ;TO GIVE C
7910 044150 032701 000002      BIT      #BIT1,R1      ;IS POSITION 15 HIGH
7911 044154 001403          BEQ      5$            ;BRANCH IF POSITION 15 LOW
7912 044156 012704 100000      MOV      #BIT15,R4     ;GET POSITION 15
7913 044162 000401          BR       6$            ;
7914 044164 005004      5$:  CLR      R4            ;GET POSITION 15
7915 044166 060304      6$:  ADD      R3,R4       ;XOR POSITION 15 WITH B
7916                                ;TO GIVE E
7917 044170 006037 001206      ROR      @#$TMP4      ;GET LSB INTO 'C'
7918 044174 006001          ROR      R1            ;GET ABOVE C INTO R1
7919 044176 005703          TST     R3            ;TEST B
7920 044200 100403          BMI     7$            ;BRANCH IF B=1
7921 044202 042701 100000      BIC     #BIT15,R1     ;SET B IN POSITION 1
7922 044206 000402          BR      10$           ;
7923 044210 052701 100000      7$:  BIS     #BIT15,R1   ;SET B IN POSITION 1
7924 044214 005702      10$: TST     R2            ;TEST C
7925 044216 100403          BMI     11$          ;BRANCH IF C=1
7926 044220 042701 020000      BIC     #BIT13,R1     ;GET C IN POSITION 3
7927 044224 000402          BR      12$           ;
7928 044226 052701 020000      11$: BIS     #BIT13,R1   ;GET C IN POSITION 3
7929 044232 005704      12$: TST     R4            ;TEST E
7930 044234 100403          BMI     13$          ;BRANCH IF E=1
7931 044236 042701 000001      BIC     #BIT0,R1      ;GET E IN POSITION 16
7932 044242 000402          BR      14$           ;
7933 044244 052701 000001      13$: BIS     #BIT0,R1   ;GET E IN POSITION 16
7934 044250 005337 001202      14$: DEC     @#$TMP2    ;BIT COUNTER
7935 044254 001310          BNE     15$          ;BRANCH IF 16 NOT DONE
7936 044256 005337 001176      DEC     @#$TMP0      ;WORD COUNTER
7937 044262 001275          BNE     16$          ;BRANCH IF 4 NOT DONE
7938 044264 010135          MOV     R1,@(R5)+     ;PUT CRC WHERE DESIRED
7939 044266 012604          MOV     (SP)+,R4      ;:POP STACK INTO R4
7940 044270 012603          MOV     (SP)+,R3      ;:POP STACK INTO R3
7941 044272 012602          MOV     (SP)+,R2      ;:POP STACK INTO R2
7942 044274 012601          MOV     (SP)+,R1      ;:POP STACK INTO R1
7943 044276 012600          MOV     (SP)+,R0      ;:POP STACK INTO R0
7944 044300 000205          RTS      R5
7945
7946
7947
7948
7949
7950
7951
7952
7953
7954
7955
7956
7957
7958
7959 044302          SETDSK:
7960 044302 010046          MOV     R0,-(SP)      ;:PUSH R0 ON STACK
7961 044304 010146          MOV     R1,-(SP)      ;:PUSH R1 ON STACK
7962 044306 010246          MOV     R2,-(SP)      ;:PUSH R2 ON STACK
7963 044310 012700 177400      MOV     #177400,R0    ;DATA IN THE DISK

```

```

;*THIS IS A SUBROUTINE TO SET UP THE SIMULATOR DISK FOR
;*CYLINDER 0 (16 BITS PER WORD)
;*TRACK 1, SECTOR 1
;*KEY1 1
;*KEY2 1
;*CRC THROUGH THE JSR R5,@#CRC
;*256 WORDS OF 177400

```

```

;*CALL JSR PC,@#SETDSK

```

SETDSK:

```

MOV     R0,-(SP)      ;:PUSH R0 ON STACK
MOV     R1,-(SP)      ;:PUSH R1 ON STACK
MOV     R2,-(SP)      ;:PUSH R2 ON STACK
MOV     #177400,R0    ;DATA IN THE DISK

```

```
7964 044314 012701 000400      MOV      #256.,R1      ;COUNTER
7965 044320 012702 051422      MOV      #DISK,R2     ;START OF SIMULATOR DISK
7966 044324 010022      1$: MOV      R0,(R2)+    ;MOVE IN DATA
7967 044326 005301      DEC      R1           ;COUNT FOR 256
7968 044330 001375      BNE     1$           ;BRANCH IF 256 NOT COMPLETE
7969 044332 012701 000021      MOV      #17.,R1     ;2 ECC WORDS, 1 DATA GAP
7970                                ;14 TOLERANCE GAP
7971 044336 005022      2$: CLR      (R2)+    ;CLEAR ECC,DATA GAP AND
7972                                ;TOLERANCE GAP
7973 044340 005301      DEC      R1           ;COUNT
7974 044342 001375      BNE     2$           ;BRANCH IF NOT COMPLETE
7975
7976                                ;*NOW SET UP FOR DISKLESS USE
7977
7978 044344 012737 010000 047504  MOV      #FMT22,@#CYL ;CYLINDER 0 (16 BIT WORDS)
7979 044352 112737 000001 047507  MOVB     #1,@#SECOTR+1 ;TRACK=1
7980 044360 112737 000001 047506  MOVB     #1,@#SECOTR  ;SECTOR=1
7981 044366 012737 000001 047510  MOV      #1,@#KEY1    ;KEY1=1
7982 044374 012737 000001 047512  MOV      #1,@#KEY2    ;KEY2=1
7983 044402 013737 000400 047564  MOV      256.,@#DAWORD ;NO. OF DATA WORDS
7984 044410 004537 044026  JSR      R5,@#CRC     ;GO TO CALCULATE CRC
7985 044414 047504      CYL                ;FIRST CRC WORD
7986 044416 051404      WCRC              ;PUT CALCULATED CRC
7987 044420 012602      MOV      (SP)+,R2   ;:POP STACK INTO R2
7988 044422 012601      MOV      (SP)+,R1   ;:POP STACK INTO R1
7989 044424 012600      MOV      (SP)+,R0   ;:POP STACK INTO R0
7990 044426 000207      RTS      PC
```

```
7991
7992      ;*THIS IS A SUBROUTINE TO CHECK HEADER COMPARE ERROR
7993      ;*(BIT #7) AND CRC ERROR (BIT #8)
7994
7995      ;*CALL JSR R0,@#HCCRCE
7996
7997      ;*      COM      ;COMMAND-READ HEADER AND DATA
7998      ;          -WRITE DATA
7999      ;*      C      ;CYLINDER
8000      ;*      S      ;SECTOR
8001      ;*      T      ;TRACK
8002      ;*      -N.    ;WORD COUNT
8003      ;*      B      ;RHBA BUFFER START
8004      ;*      X      ;1=WRITE DATA 0=READ
8005      ;*      H      ;H=1 HEADER CHECK, H=0 CRC CHECK
8006
8007 044430 010037 002014      HCCRCE: MOV R0,@#PCJSR      ;SAVE PC OF JSR+4
8008 044434 162737 000004 002014 SUB #4,@#PCJSR      ;GET PC OF JSR
8009 044442 004737 042620 JSR PC,@#CLDISK    ;INIT AND SETUP GENERAL REG.
8010 044446 004737 042654 JSR PC,@#CHECKT    ;CHECK THAT DVA,RDY,DPR,DRY = 1
8011 044452 104401 062450 TYPE ,CPHALT      ;AND THAT NO OTHERS = 1. CANNOT CON-
8012 044456 000000 HALT              ;STOP THE TEST
8013 044460 011037 001210 MOV (R0),@#STMP5   ;SAVE COMMAND
8014 044464 012011 MOV (R0)+,@R1      ;COMMAND
8015 044466 012077 135160 MOV (R0)+,@RHCA    ;CYLINDER
8016 044472 112046 MOV B (R0)+,-(SP)  ;SECTOR
8017 044474 105720 TSTB (R0)+         ;UP DATE R0
8018 044476 112066 000001 MOV B (R0)+,1(SP)  ;TRACK
8019 044502 105720 TSTB (R0)+         ;UPDATE R0
8020 044504 012677 135134 MOV (SP)+,@RHDST   ;TRACK SECTOR
8021 044510 012077 135116 MOV (R0)+,@RHWC    ;NO. OF DATA WORDS +4 HEADER
8022      ;IF A READ HEADER AND DATA
8023 044514 012077 135114 MOV (R0)+,@RHBA    ;STARTING ADDRESS OF BUFFER
8024 044520 012037 047514 MOV (R0)+,@#X      ;X=0 READ HEADER AND DATA
8025      ;X=1 WRITE DATA
8026 044524 012777 014000 135116 MOV #FMT22!ECI,@RHOF ;16 BITS PER WORD
8027      ;ECC CORRECTION INHIBIT
8028 044532 005037 002006 CLR @#ERFLG$      ;CLEAR ERROR FLAG
8029 044536 004737 047344 JSR PC,@#COMHD     ;COMMAND
8030
8031      ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
8032      ;*FROM THE 'COMHD' ROUTINE THAT MEANS SECTOR GAP,
8033      ;*FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
8034      ;*SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
8035      ;*DETECTED
8036      ;*HEADER AND DATA ARE TO BE CHECKED.
8037
8038 044542 004737 042224 JSR PC,@#PUTREG    ;SAVE REGISTERS
8039 044546 005737 002006 TST @#ERFLG$      ;ANY ERRORS ALREADY THERE
8040 044552 001034 BNE 10$           ;BRANCH IF YES
8041 044554 005737 047514 TST @#X           ;IS THIS A READ
8042 044560 001015 BNE 3$           ;IF A WRITE DATA BRANCH
```

```
8043
8044
8045
8046
8047
8048
8049
8050 044562 004037 043514 JSR RO,@#COMPAR ;CHECK
8051 044566 002110 WRFROM ;GOOD DATA
8052 044570 003154 REINTO ;TEST BUFFER
8053 044572 000400 256. ;4 HEADER 252 DATA
8054 044574 044602 1$ ;RETURN POINT FOR ERROR HEADER
8055 044576 044606 2$ ;RETURN POINT FOR ERROR DATA
8056 044600 044644 10$ ;RETURN FOR GOOD COMPARISON
8057 044602 104004 1$: ERROR 4 ;READ NEXT ERROR 5
8058 044604 000207 RTS PC ;RETURN TO COMPARISON SUBROUTINE
8059 044606 104005 2$: ERROR 5 ;WORD NO 1 THRU 4 ARE
8060 ;HEADER WORDS AND HENCE
8061 ;SHOULD BE READ AS WRITTEN ON
8062 ;DISK, WORD NOS. 5 ONWARDS
8063 ;SHOULD NOT BE READ AND HENCE
8064 ;READ INTO BUFFER
8065 ;SHOULD BE UNCHANGED
8066 044610 000207 RTS PC ;RETURN TO COMPARISON
8067
8068 044612 000414 BR 10$ ;JUMP OUT
8069
8070
8071
8072
8073
8074
8075
8076 044614 004037 043514 3$: JSR RO,@#COMPAR ;CHECK
8077 044620 003154 REINTO ;GOOD DATA BUFFER
8078 044622 051422 DISK ;TEST BUFFER
8079 044624 000400 256.
8080 044626 044634 4$ ;RETURN POINT FOR ERROR HEADER
8081 044630 044640 5$ ;RETURN POINT FOR ERROR DATA
8082 044632 044644 10$ ;RETURN POINT FOR GOOD COMPARISON
8083 044634 104004 4$: ERROR 4 ;READ NEXT ERROR 5
8084 044636 000207 RTS PC ;RETURN TO COMPARISON SUBROUTINE
8085 044640 104005 5$: ERROR 5 ;WORD NO ARE ALL DATA
8086 ;WORDS THE SHOULD NOT
8087 ;HAVE BEEN CHANGED BY THE
8088 ;WRITE COMMAND
8089 044642 000207 RTS PC ;RETURN TO COMPARISON SUBROUTINE
8090 044644 005720 10$: TST (R0)+ ;IS THIS A HCRC ON HCE CHECK?
8091 044646 001442 BEQ 6$ ;BRANCH IF HCRC
8092 044650 022737 000072 001210 CMP #72,@#$TMP5 ;IS THIS A READ COMMAND
8093 044656 001417 BEQ 11$ ;BRANCH IF YES
8094 044660 017737 134756 001126 MOV @RHER1,@#$BDDAT ;TEST DATA
8095 044666 022737 000200 001126 CMP #HCE,@#$BDDAT ;ONLY HEADER COMPARE BIT?
8096 ;SHOULD BE SET
8097 044674 001470 BEQ 7$ ;BRANCH IF GOOD
8098 044676 013737 001642 042270 MOV @RHER1,@#REGADR ;REGISTER ADDRESS RHER1
```

```

8099 044704 012737 000200 001124      MOV    #HCE,@#$GDDAT ;GOOD DATA
8100 044712 104027                ERROR  27             ;AFTER AN ERROR ON THE
8101                                ;HEADER ONLY HCE SHOULD
8102 044714 000460                BR     7$            ;BE SET
8103 044716                11$:
8104 044716 017737 134720 001126      MOV    @RHER1,@#$BDDAT ;TEST DATA
8105 044724 022737 100200 001126      CMP    #DCK!HCE,@#$BDDAT ;ONLY HEADER COMPARE BIT?
8106                                ;SHOULD BE SET
8107                                ;DCK IS SET BECAUSE ECC IS NOT READ
8108 044732 001451                BEQ    7$            ;BRANCH IF GOOD
8109 044734 013737 001642 042270      MOV    @RHER1,@#REGADR ;REGISTER ADDRESS RHER1
8110 044742 012737 100200 001124      MOV    #DCK!HCE,@#$GDDAT ;GOOD DATA
8111 044750 104027                ERROR  27             ;AFTER AN ERROR ON THE
8112                                ;HEADER ONLY HCE SHOULD
8113 044752 000441                BR     7$            ;BE SET
8114 044754 022737 000072 001210 6$:    CMP    #72,@#$TMP5    ;IS THIS A READ COMMAND?
8115 044762 001417                BEQ    12$           ;BRANCH IF A READ
8116 044764 017737 134652 001126      MOV    @RHER1,@#$BDDAT ;TEST DATA
8117 044772 022737 000400 001126      CMP    #HCRC,@#$BDDAT ;ONLY CRC ERROR SHOULD BE THERE
8118 045000 001426                BEQ    7$
8119 045002 013737 001642 042270      MOV    @RHER1,@#REGADR ;REG. ADDR = RHER1
8120 045010 012737 000400 001124      MOV    #HCRC,@#$GDDAT ;GOOD DATA
8121 045016 104027                ERROR  27             ;AFTER A CRC ERROR ONLY CRC
8122                                ;SHOULD BE SET
8123 045020 000416                BR     7$            ;BRANCH OUT
8124 045022 017737 134614 001126 12$:  MOV    @RHER1,@#$BDDAT ;TEST DATA
8125
8126 045030 022737 100400 001126      CMP    #DCK!HCRC,@#$BDDAT ;HCRC AND DCK SHOULD BE SET
8127                                ;DCK IS SET BECAUSE ECC IS NOT READ
8128 045036 001407                BEQ    7$            ;BRANCH IF GOOD
8129 045040 012737 100400 001124      MOV    #DCK!HCRC,@#$GDDAT ;GOOD DATA
8130 045046 013737 001642 042270      MOV    @RHER1,@#REGADR ;FAILING REGISTER RHER1
8131 045054 104027                ERROR  27             ;AFTER A CRC ERROR ON A READ
8132                                ;DCK AND HCRC SHOULD BE SET
8133                                ;DCK IS SET BECAUSE ECC IS NOT READ
8134 045056 000200                7$:    RTS     R0           ;RETURN TO MAIN TEST
8135
8136
8137
8138                                ;*THIS IS A SUBROUTINE TO LEAVE AT THE MIDDLE OF
8139                                ;*A WRITE HEADER AND DATA COMMAND
8140                                ;*IT TRYS TO GET SECTOR 10, TRACK 0, CYLINDER 0
8141                                ;*BUT COMES OUT AFTER ONE SECTOR
8142                                ;*THE COMMAND OS JSR PC,@#MIDDLE
8143                                ;*BAI IS SET
8144
8145                                MIDDLE:
8146 045060                MOV    R0,-(SP)      ;:PUSH R0 ON STACK
8147 045062 010046                MOV    R1,-(SP)      ;:PUSH R1 ON STACK
8148 045064 013777 002064 134546      MOV    @#WRIFOR,@RHCS1 ;WRITE HEADER AND DATA=62
8149                                ;:IN RHCS1
8150 045072 012777 177766 134532      MOV    #-10,@RHWC    ;:10 WORDS
8151 045100 012777 002110 134526      MOV    #WRFROM,@RHBA ;:BUS ADDRESS=WRFROM
8152 045106 012777 000010 134530      MOV    #10,@RHDST    ;:DESIRED TRACK=0 SECTOR=10
8153 045114 052777 000010 134514      BIS    #BAI,@RHCS2   ;:BUS ADDRESS INCREMENT INHIBIT
8154 045122 012777 010000 134520      MOV    #FMT22,@RHOF  ;:FORMAT 16 BIT WORDS
    
```

```

8155 045130 005077 134516          CLR      @RHCA      ;CYLINDER=0
8156 045134 012737 000001 045162  MOV      #1,@MID   ;SECTOR IS SET TO 1 SO THAT
8157                                     ;WE CAN GET OUT AT THE
8158                                     ;MIDDLE OF AN OPERATION
8159                                     ;LOOKING FOR SECTOR 10
8160 045142 012777 000001 134510  MOV      #DMD,@RHMR ;SET DIAGNOSTIC MODE
8161 045150 052777 000001 134462  BIS      #GO,@RHCS1 ;GO TO RHCS1 WITH 62
8162 045156 004137 053552          JSR      R1,@SEARCH
8163 045162 000000          MID:     .WORD    0      ;SECTOR
8164 045164 012601          MOV      (SP)+,R1   ;:POP STACK INTO R1
8165 045166 012600          MOV      (SP)+,R0   ;:POP STACK INTO R0
8166 045170 000207          RTS      PC
8167
8168
8169
8170                                     .SBTTL  JAM CURRENT CYLINDER ROUTINE
8171
8172                                     ;*THIS SUBROUTINE WILL CHANGE THE CURRENT CYLINDER REGISTER
8173                                     ;*THIS IS DONE BY GIVING A SEEK COMMAND THEN AN INIT
8174                                     ;*WHICH WILL LOAD THE CURRENT CYLINDER WITH THE DESIRED CYLINDER VALUE
8175                                     ;*
8176                                     ;*CALL IS
8177                                     ;*JSR  R0,@MAKECYL
8178                                     ;*XC      ;DESIRED VALUE OF CURRENT CYLINDER
8179
8180
8181
8182 045172          MAKECYL:
8183 045172 010546          MOV      R5,-(SP)   ;:PUSH R5 ON STACK
8184 045174 010037 002014  MOV      R0,@PCJSR ;:PC OF JSR+4
8185 045200 162737 000004 002014  SUB      #4,@PCJSR ;:SAVE PC OF JSR
8186 045206 012005          MOV      (R0)+,R5 ;:GETTING READY TO FILL DESIRED CYLINDER
8187 045210 010577 134436          MOV      R5,@RHCA ;:FILL DESIRED CYLINDER REGISTER
8188 045214 005077 134424          CLR      @RHDST   ;:MAKE SURE DESIRED SECTOR TRACK IS NOT ILLEGAL
8189 045220 013777 002072 134412  MOV      @#SEECOM,@RHCS1 ;:FILL SEEK COMMAND
8190 045226 012777 000001 134424  MOV      #DMD,@RHMR ;:SET DIAGNOSTIC MODE
8191 045234 052777 000001 134376  BIS      #GO,@RHCS1 ;:GO TO SEEK
8192 045242 000240          NOP          ;:ALLOW TIME FOR SEEK TO HANG UP
8193 045244 000240          NOP          ;:ALLOW TIME FOR SEEK TO HANG UP
8194 045246 000240          NOP          ;:ALLOW TIME FOR SEEK TO HANG UP
8195 045250 000240          NOP          ;:ALLOW TIME FOR SEEK TO HANG UP
8196 045252 004737 042620          JSR      PC,@CLDISK ;:GIVE INIT
8197 045256 017737 134414 001126  MOV      @RHCC,@#SBDDAT ;:TEST DATA
8198 045264 020537 001126          CMP      R5,@#SBDDAT ;:COMPARE CURRENT CYLINDER
8199 045270 001406          BEQ      1$      ;:BRANCH IF GOOD
8200 045272 010537 001124          MOV      R5,@#SGDDAT ;:GOOD VALUE OF RHCC
8201 045276 013737 001676 042270  MOV      @#RHCC,@#REGADR ;:FAILING REGISTER ADDRESS
8202 045304 104030          ERROR    30      ;:CURRENT CYLINDER DOES NOT MATCH DESIRED CYLINDER
8203                                     ;:REGISTER AFTER A SEEK AND AN INIT
8204 045306          1$:
8205 045306 012605          MOV      (SP)+,R5   ;:POP STACK INTO R5
8206 045310 000200          RTS      R0
8207
8208
8209                                     .SBTTL  ECC GENERATION AND COMPARISON ROUTINE
8210
    
```



```
8211
8212
8213
8214
8215
8216
8217
8218
8219
8220
8221      100000      PIE1      =100000
8222      040000      PIE2      =40000
8223      020000      PIE3      =20000
8224      010000      PIE4      =10000
8225      004000      PIE5      =4000
8226      002000      PIE6      =2000
8227      001000      PIE7      =1000
8228      000400      PIE8      =400
8229      000200      PIE9      =200
8230      000100      PIE10     =100
8231      000040      PIE11     =40
8232      000020      PIE12     =20
8233      000010      PIE13     =10
8234      000004      PIE14     =4
8235      000002      PIE15     =2
8236      000001      PIE16     =1
8237      100000      PIE17     =100000
8238      040000      PIE18     =40000
8239      020000      PIE19     =20000
8240      010000      PIE20     =10000
8241      004000      PIE21     =4000
8242      002000      PIE22     =2000
8243      001000      PIE23     =1000
8244      000400      PIE24     =400
8245      000200      PIE25     =200
8246      000100      PIE26     =100
8247      000040      PIE27     =40
8248      000020      PIE28     =20
8249      000010      PIE29     =10
8250      000004      PIE30     =4
8251      000002      PIE31     =2
8252      000001      PIE32     =1
8253
8254 045312 000000      ECDATA: 0      ;DATA BIT FOR ECC
8255                                     ;IF ALL ONES THEN CURRENT BIT IS A ONE
8256                                     ;IF ZERO THEN CURRENT BIT IS A ZERO
8257
8258 045314 000000      GECC1: 0      ;LOW ORDER ECC WORD TO BE GENERATED HERE
8259                                     ;=R1
8260
8261 045316 000000      GECC2: 0      ;HIGH ORDER ECC WORD TO BE GENERATED HERE
8262                                     ;=R2
8263
8264 045320 000000      TSECCG: 0     ;IF =177777 GENERATE AND TEST ECC FOR THIS BIT
8265                                     ;IF =0 DO NOT GENERATE AND TEST ECC FOR THIS BIT
8266
```

8267	045322	113713		NCODE: 38859.	:N-CODE WORD
8268	045324	000000		NCOUNT: 0	:TEMPORARY N CODE
8269	045326	000000		POSITI: 0	:POSITION REGISTER
8270	045330	010041		HARDER: 4129.	:HARD ERROR COUNT
8271					:TRUE COUNT IS 4128 BUT AS COMPARES ARE
8272					:DONE ONE STAGE LATER SO 4129
8273	045332	000000		DATENV: 0	:DATA ENVELOPE FOR TYPE OUT
8274					:MAX FOR WRITE IS 4096
8275					:MAX FOR READ IS 4128
8276	045334	000000		ZCODE: 0	:LEADING ZEROS ENVELOPE FOR TYPE OUT
8277					:THIS IS SHUT OFF WHEN POSITION COUNTER
8278					:IN ENABLED
8279					:MAX COUNT IS 38859
8280					
8281					
8282					
8283	045336	000000		HADTMP: 0	:TEMPORARY HARD ERROR COUNT
8284	045340	000000		P3: 0	
8285	045342	000000		P12: 0	
8286	045344	000000		P22: 0	
8287	045346	000000		P24: 0	
8288					
8289					
8290					
8291					
8292					
8293	045350			ECTEST:	
8294	045350	010046		MOV R0,-(SP)	::PUSH R0 ON STACK
8295	045352	010146		MOV R1,-(SP)	::PUSH R1 ON STACK
8296	045354	010246		MOV R2,-(SP)	::PUSH R2 ON STACK
8297	045356	010346		MOV R3,-(SP)	::PUSH R3 ON STACK
8298	045360	010446		MOV R4,-(SP)	::PUSH R4 ON STACK
8299	045362	010546		MOV R5,-(SP)	::PUSH R5 ON STACK
8300	045364	013701	045314	MOV @#GECC1,R1	:ECC1 WORD
8301	045370	013702	045316	MOV @#GECC2,R2	:ECC2 WORD
8302	045374	005737	045312	TST @#ECDATA	:IS CURRENT BIT A ONE
8303	045400	001406		BEQ 2\$:BRANCH IF CURRENT DATA D=0
8304					
8305					:*IF CARRY IS NOT ZERO THEN D=1
8306					:*INVERT X32 TO GIVE R0
8307					
8308	045402	010103		1\$: MOV R1,R3	
8309	045404	052703	177776	BIS #^CPIE32,R3	
8310	045410	005103		COM R3	
8311	045412	010300		MOV R3,R0	
8312	045414	000404		BR 3\$	
8313					
8314					:*IF CARRY IS ZERO THEN D=0
8315					:*X32 BECOMES R0
8316	045416	010103		2\$: MOV R1,R3	
8317	045420	042703	177776	BIC #^CPIE32,R3	
8318	045424	010300		MOV R3,R0	
8319					
8320	045426	000241		3\$: CLC	
8321	045430	006000		ROR R0	
8322	045432	006000		ROR R0	

```
8323 045434 005700          TST      R0
8324 045436 001462          BEQ      10$          ;BRANCH IF R0=0
8325                          ;*INVERT X2
8326
8327 045440 010203          MOV      R2,R3
8328 045442 052703 137777    BIS      #^CPIE2,R3
8329 045446 005103          COM      R3
8330 045450 010337 045340    MOV      R3,@#P3
8331 045454 006237 045340    ASR      @#P3
8332
8333                          ;*INVERT X11
8334
8335
8336 045460 010203          MOV      R2,R3
8337 045462 052703 177737    BIS      #^CPIE11,R3
8338 045466 005103          COM      R3
8339 045470 010337 045342    MOV      R3,@#P12
8340 045474 006237 045342    ASR      @#P12
8341
8342                          ;*INVERT X21
8343
8344 045500 010103          MOV      R1,R3
8345 045502 052703 173777    BIS      #^CPIE21,R3
8346 045506 005103          COM      R3
8347 045510 010337 045344    MOV      R3,@#P22
8348 045514 006237 045344    ASR      @#P22
8349
8350                          ;*INVERT X23
8351
8352 045520 010103          MOV      R1,R3
8353 045522 052703 176777    BIS      #^CPIE23,R3
8354 045526 005103          COM      R3
8355 045530 010337 045346    MOV      R3,@#P24
8356 045534 006237 045346    ASR      @#P24
8357
8358                          ;*NOW THAT R0 FOR POSITION 1
8359                          ;*      P3 FOR POSITION 3
8360                          ;*      P12 FOR POSITION 12
8361                          ;*      P22 FOR POSITION 22
8362                          ;*      P24 FOR POSITION 24
8363                          ;*ARE KNOWN THE ROTATE WILL BE DONE AND
8364                          ;*THESE BITS JAMED IN
8365
8366 045540 006002          ROR      R2
8367 045542 006001          ROR      R1
8368 045544 053700 045340    BIS      @#P3,R0
8369 045550 053700 045342    BIS      @#P12,R0
8370 045554 042702 120020    BIC      #PIE1!PIE3!PIE12,R2
8371 045560 050002          BIS      R0,R2
8372
8373 045562 005000          CLR      R0
8374 045564 053700 045344    BIS      @#P22,R0
8375 045570 053700 045346    BIS      @#P24,R0
8376 045574 042701 002400    BIC      #PIE22!PIE24,R1
8377 045600 050001          BIS      R0,R1
8378 045602 000404          BR      12$
```

```
8379
8380
8381
8382
8383 045604 006002      10$: ROR    R2
8384 045606 006001      ROR    R1
8385 045610 042702 100000  BIC    #PIE1,R2
8386 045614 010137 045314 12$: MOV    R1,@#GECC1      ;SAVE ECC1
8387 045620 010237 045316  MOV    R2,@#GECC2      ;SAVE ECC2
8388 045624 005737 045320  TST    @#TSECCG        ;IS HARDWARE TO BE CHECKED
8389
8390
8391 045630 001432      BEQ    14$              ;IF =1777777 TEST HARDWARE
8392
8393
8394
8395 045632 032777 000400 133300 ;*CHECK HARDWARE
8396 045640 001005      BIT    #SW8,@SWR        ;IS SWITCH 8 SET
8397 045642 032777 000100 133270 BNE    15$              ;BRANCH IF SW8 IS SET
8398 045650 001401      BIT    #SW6,@SWR        ;IS SWITCH 6 SET
8399 045652 000421      BEQ    15$              ;BRANCH IF SW6 IS NOT SET
8400
8401
8402 045654 010146      BR     14$              ;IF SWITCH 8 IS NOT SET AND
8403 045656 042716 174000 15$: MOV    R1,-(SP)        ;SWITCH 6 IS SET THEN
8404 045662 022677 134004      BIC    #174000,(SP)     ;DO NOT DO COMPARES
8405 045666 001404      CMP    (SP)+,@RHEC2     ;GOOD PATTERN REGISTER
8406
8407 045670 004737 042224      BEQ    13$              ;GET ONLY PATTERN BITS
8408 045674 104035      JSR    PC,@#PUTREG      ;COMPARE PATTERN REGISTER
8409 045676 000407      BR     14$              ;BRANCH IF GOOD
8410 045700 023777 045326 133762 13$: ;*TO SAVE TIME
8411 045706 001403      JSR    PC,@#PUTREG      ;SAVE REGISTERS
8412
8413 045710 004737 042224      ERROR  35              ;PATTERN REGISTER IN 11 BITS IN ERROR
8414 045714 104035      BR     14$              ;BRANCH OUT
8415
8416
8417
8418
8419
8420
8421
8422
8423
8424
8425
8426
8427
8428
8429
8430
8431
8432
8433 045716
8434 045716 012605      14$: MOV    (SP)+,R5      ;COMPARE POSITION REGISTER
                                ;BRANCH IF GOOD
                                ;*TO SAVE TIME
                                JSR    PC,@#PUTREG      ;SAVE REGISTERS
                                ERROR  35              ;POSITION REGISTER IN ERROR
                                ;'DATA ENVLOP' GIVES NUMBER OF CLOCK
                                ;PULSES FROM BEGINING OF COMMAND
                                ;THAT IS THE CLOCKS IN THE R/W DATA FIELD ENVELOPE
                                ;
                                ;IN A WRITE THERE ARE 10000 OCTAL CLOCKS
                                ;IN A READ THERE ARE 10040 OCTAL CLOCKS
                                ;
                                ;
                                ;'N-CODE ZEROS' GIVE THE NUMBER OF CLOCKS
                                ;GIVEN FOR THE LEADING ZEROS FIELD
                                ;MAX COUNT IS 113713 OCTAL
                                ;
                                ;'GOOD POSITION' GIVES NUMBER OF CLOCKS
                                ;GIVEN AFTER LEADING ZEROS WHICH IS FOR THE DATA
                                ;FIELD
                                ;MAX COUNT IS 10040 OR 10041 OCTAL
                                ;POP STACK INTO R5
```

8435	045720	012604	MOV	(SP)+,R4	::POP STACK INTO R4
8436	045722	012603	MOV	(SP)+,R3	::POP STACK INTO R3
8437	045724	012602	MOV	(SP)+,R2	::POP STACK INTO R2
8438	045726	012601	MOV	(SP)+,R1	::POP STACK INTO R1
8439	045730	012600	MOV	(SP)+,R0	::POP STACK INTO R0
8440	045732	000207	RTS	PC	

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

```
8441
8442
8443      ;*THIS SUBROUTINE WILL CONTROL THE ECC GENERATION ROUTINE
8444      ;*FOR ERROR CORRECTION PROCESS
8445      ;*CALL JSR, PC,@#ECORR
8446      ;* XP ;EXPECTED POSITION REGISTER WHEN CORRECTION IS COMPLETE
8447
8448
8449
8450 045734 000000 ERPOS: 0 ;POSITION REG. WHEN CORRECTION IS COMPLETE
8451
8452
8453
8454 045736 010037 002014 ECORR: MOV R0,@#PCJSR ;SAVE PC OF JSR + 4
8455 045742 162737 000004 002014 SUB #4,@#PCJSR ;SAVE PC OF JSR
8456 045750 012037 045734 MOV (R0)+,@#ERPOS ;GET POSITION REG. WHEN CORRECTION IS COMPLETE
8457 045754 010146 MOV R1,-($P) ;PUSH R1 ON STACK
8458 045756 013701 001660 MOV @#RHMR,R1 ;MAINTENANCE REGISTER
8459 045762 012711 000001 MOV #DMD,@R1 ;SET DIAGNOSTIC MODE BIT
8460 045766 005037 045312 CLR @#ECDATA ;ECC DATA IS ZERO
8461
8462
8463
8464 045772 005737 045326 1$: TST @#POSITI ;IS SOFTWARE POSITION NON ZERO
8465 045776 001007 BNE 2$ ;BRANCH IF N-CODE S COMPLETE
8466 046000 005337 045324 DEC @#NCGUNT ;DECREMENT N-CODE
8467 046004 001001 BNE 6$ ;BRANCH IF N-CODE IS NOT COMPLETE
8468 046006 000403 BR 2$ ;BRANCH AS N-CODE IS COMPLETE
8469 046010 005237 045334 6$: INC @#ZCODE ;INCREMENT CLOCKS GIVEN FOR LEADING ZEROS
8470 046014 000420 BR 3$ ;BRANCH AS N-CODE IS NOT COMPLETE
8471 ;GO TO GIVE CLOCK AND TEST ECC
8472 046016 005237 045326 2$: INC @#POSITI ;INCREMENT SOFTWARE POSITION
8473 046022 023737 045734 045326 CMP @#ERPOS,@#POSITI ;HAVE ENOUGH CLOCKS BEEN GIVEN TO DETECT ERROR
8474 046030 103012 BHIS 3$ ;BRANCH IF MORE CLOCKS TO BE GIVEN
8475 046032 023737 045336 045326 CMP @#HADTMP,@#POSITI ;HAVE ENOUGH CLOCKS BEEN GIVEN FOR HARD ERROR
8476 ;THAT IS HAVE 4128 MORE CLOCKS BEEN GIVEN
8477 046040 001415 BEQ 5$ ;BRANCH IF YES
8478 046042 032711 000400 BIT #ZER,@R1 ;CHECK ZERO DETECT BIT IN RHMR
8479 046046 001016 BNE 4$ ;BRANCH IS ZER SET
8480 ;*TO SAVE TIME
8481 046050 004737 042224 JSR PC,@#PUTREG ;SAVE REGISTERS
8482 046054 104034 ERROR 34 ;ZERO DETECT BIT NOT HIGH
8483 ;WHEN 21 BITS IN ECC 32 BIT REGISTER IS 0
8484
8485
8486 046056 052711 000002 3$: BIS #MCLK,@R1 ;SET CLOCK
8487 046062 042711 000002 BIC #MCLK,@R1 ;CLEAR CLOCK
8488 046066 004737 045350 JSR PC,@#ECTEST ;GO TO GENERATE AND TEST ECC
8489 046072 000737 BR 1$ ;CONTINUE
8490
8491 ;*THIS EXTRA CLOCK IS TO BRING ECH HIGH
8492 ;*AFTER THIS CLOCK POSITION REGISTER MAY BE 10040 OR 10041 OCTAL
8493
8494 046074 052711 000002 5$: BIS #MCLK,@R1 ;SET CLOCK
8495 046100 042711 000002 BIC #MCLK,@R1 ;CLEAR CLOCK
8496
```

8497 046104
 8498 046104 012601
 8499 046106 000200

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

8500
 8501
 8502
 8503
 8504
 8505
 8506 ;*THIS SUBROUTINE GENERATES THE ECC FOR WHAT IS ON DISK AND INSERTS THEM
 8507 ;*ON LOCATIONS 'DISK+1000' AND 'DISK+1002'
 8508
 8509

8510
 8511 046110

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 @#POSITI ;CLEAR POSITION
 CLR @#GECC1 ;CLEAR GECC1
 CLR @#GECC2 ;CLEAR
 MOV #DISK,R1 ;:POINTER TO DATA FOR ECC GENERATION
 MOV #256,R2 ;:COUNTER FOR NUMBER OF DATA WORDS
 9\$: MOV #16,R3 ;:COUNTER FOR NUMBER OF BITS PER WORD
 MOV (R1)+,R4 ;:DATA IN R4
 10\$: ROR R4 ;:GET ONE DATA BIT IN CARRY
 BCC 11\$;:BRANCH IF DATA BIT IS ZERO
 8512 046110 010046
 8513 046112 010146
 8514 046114 010246
 8515 046116 010346
 8516 046120 010446
 8517 046122 010546
 8518 046124 005037 045326
 8519 046130 005037 045314
 8520 046134 005037 045316
 8521 046140 012701 051422
 8522 046144 012702 000400
 8523 046150 012703 000020
 8524 046154 012104
 8525 046156 006004
 8526 046160 103004
 8527 046162 012737 177777 045312
 8528 046170 000402
 8529 046172 005037 045312
 8530 046176 004737 045350
 8531 046202 005303
 8532 046204 001364
 8533 046206 005302
 8534 046210 001357
 8535 046212 013737 045314 052422
 8536 046220 013737 045316 052424
 8537 046226 012605
 8538 046230 012604
 8539 046232 012603
 8540 046234 012602
 8541 046236 012601
 8542 046240 012600
 8543 046242 000207
 8544
 8545
 8546
 8547
 8548
 8549
 8550
 8551
 8552

.SBTTL RH BASE ADDRESS CHANGE ROUTINE

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

8553							
8554	046244			BASECH:	TYPE	,65\$::TYPE ASCIZ STRING
8555	046244	104401	046252		BR	64\$::GET OVER THE ASCIZ
8556	046250	000425			MOV	@#RHCS1,-(SP)	::GET READY TO TYPE OLD BASE
8557	046324	013746	001640		TYPOC		
8558	046330	104402			TYPE	,67\$::TYPE ASCIZ STRING
8559	046332	104401	046340		BR	66\$::GET OVER THE ASCIZ
8560	046336	000425			JSR	PC,@#STKINT	::INITIALIZE THE TTY KEYBOARD
8561	046412	004737	055176		RDOCT		
8562	046416	104412			MOV	#RHDB,R0	::GET STARTING ADDRESS OF REGISTERS
8563	046420	012700	001630		MOV	#22,R1	::NUMBER OF REGISTERS
8564	046424	012701	000026		MOV	#ADTIMO,@#4	::SET UP TO CHECK NEW ADDRESS
8565	046430	012737	047230	000004	CMP	@SP,@#RHCS1	::NEW CSR?
8566	046436	021637	001640		BEQ	1\$::NO-SKIP NEXT
8567	046442	001407			TST	@0(SP)	::ACCESS THE NEW ADDRESS
8568	046444	005776	000000		SUB	@#RHCS1,@SP	::GET THE ADDRESS OFFSET
8569	046450	163716	001640		ADD	@SP,(R0)+	::AND PLUG IT IN
8570	046454	061620		2\$:	DEC	R1	::DONE ALL OF THEM YET?
8571	046456	005301			BNE	2\$::NOT YET, SO DO MORE
8572	046460	001375					
8573	046462			1\$:			
8574	046462	104401	046470		TYPE	,69\$::TYPE ASCIZ STRING
8575	046466	000417			BR	68\$::GET OVER THE ASCIZ
8576	046526	013746	001626		MOV	@#RPVEC,-(SP)	::GET READY TO TYPE OLD VECTOR ADDRESS
8577	046532	104402			TYPOC		
8578	046534	104401	046542		TYPE	,71\$::TYPE ASCIZ STRING
8579	046540	000437			BR	70\$::GET OVER THE ASCIZ
8580	046640	104412			RDOCT		
8581	046642	012637	001626		MOV	(SP)+,@#RPVEC	::SETUP VECTOR ADDRESS
8582	046646	104401	046654		TYPE	,73\$::TYPE ASCIZ STRING
8583	046652	000416			BR	72\$::GET OVER THE ASCIZ
8584	046710	013746	001640		MOV	@#RHCS1,-(SP)	
8585	046714	104402			TYPOC		
8586	046716	104401	046724		TYPE	,75\$::TYPE ASCIZ STRING
8587	046722	000416			BR	74\$::GET OVER THE ASCIZ
8588	046760	013746	001626		MOV	@#RPVEC,-(SP)	
8589	046764	104402			TYPOC		
8590	046766	104401	046774		TYPE	,77\$::TYPE ASCIZ STRING
8591	046772	000417			BR	76\$::GET OVER THE ASCIZ
8592	047032	104401	047040		TYPE	,79\$::TYPE ASCIZ STRING
8593	047036	000402			BR	78\$::GET OVER THE ASCIZ
8594	047044	104401	047052		TYPE	,81\$::TYPE ASCIZ STRING
8595	047050	000424			BR	80\$::GET OVER THE ASCIZ
8596	047122	104401	047130		TYPE	,83\$::TYPE ASCIZ STRING
8597	047126	000426			BR	82\$::GET OVER THE ASCIZ
8598	047204	012746	000200		MOV	#RA,-(SP)	
8599	047210	104402			TYPOC		
8600	047212	104401	047220		TYPE	,85\$::TYPE ASCIZ STRING
8601	047216	000402			BR	84\$::GET OVER THE ASCIZ
8602	047224	000137	004240		JMP	@#BEGIN	::OK, NOW START OVER WITH NEW ADDRESS
8603	047230			ADTIMO:			
8604	047230	104401	047236		TYPE	,65\$::TYPE ASCIZ STRING
8605	047234	000424			BR	64\$::GET OVER THE ASCIZ
8606	047306	022626			CMP	(SP)+,(SP)+	::RESTORE THE STACK
8607	047310	000137	046244		JMP	@#BASECH	::AND DO IT AGAIN.
8608							

8609
8610
8611
8612
8613
8614
8615
8616
8617 047314 000000
8618 047316 004737 042620
8619 047322 013712 047314
8620 047326 005714
8621 047330 032712 010000
8622 047334 001401
8623 047336 000773
8624 047340 000772

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

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

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

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
 8705
 8706
 8707
 8708
 8709
 8710
 8711
 8712
 8713
 8714
 8715
 8716
 8717
 8718
 8719
 8720
 8721
 8722
 8723
 8724
 8725
 8726
 8727
 8728
 8729
 8730
 8731
 8732
 8733
 8734

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

;**THIS SUBROUTINE IS THE FIRST IN A SERIES OF NESTED SUBROUTINES

;**IT ISSURE DIAGNOSTIC MODE, AN EXTRA DIAGNOSTIC INDEX, AND THE
 ;**'GO' BIT

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

;** SEARCH
 ;** RDHEAD
 ;** WRDATA
 ;** REDATA

RUNCTR:	.WORD	0			
COMHD:	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 IS ISSUED BY THE TEST)
RUNWAT:	MOV	#75,@#RUNCTR			;LOAD STALL COUNT = APPROX. 450US
					;FOR 11/50 CPU WITH CORE MEMORY
1\$:	DEC	@#RUNCTR			;COUNT DOWN ONE
	BNE	1\$;CONTINUE UNTIL = 0
	MOV	SECTR,-(SP)			;GET DESIRED SECTOR/TRACK
	BIC	#177740,(SP)			;MAKE ONLY SECTOR
	MOV	(SP)+,@#TRK			;SAVE SECTOR
	JSR	R1,@#SEARCH			;ISSUE SECTOR CLOCKS <----->

8735	047456	000000	TRK:	.WORD	0	
8736	047460	012701		MOV	#+NOP,R1	:GOING TO MOVE NOPS
8737	047464	010137		MOV	R1,@#SSYN	:NOP INTO SSYN
8738	047470	010137		MOV	R1,@#HEDGAP	:NOP INTO HEDGAP
8739	047474	010137		MOV	R1,@#HEDSYN	:NOP INTO HEDSYN
8740	047500	004137		JSR	R1,@#RDHEAD	
8741						
8742						
8743						:*DUMMY ERROR CALL LOCATIONS FOR THE READ HEADER OPERATION
8744						
8745	047504	000000	CYL:	.WORD	0	:CYLINDER ADDRESS
8746	047506	000000	SECOTR:	.WORD	0	:SECTOR/TRACK ADDRESS
8747	047510	000000	KEY1:	.WORD	0	:KEY1 WORD
8748	047512	000000	KEY2:	.WORD	0	:KEY2 WORD
8749	047514	000000	X:	.WORD	0	:X=1 WRITE COMMAND :X=0 READ COMMAND
8750						
8751						
8752	047516	000240	SSYN:	NOP		:IF 'ERROR 2' INSERTED BY RDHEAD :SUBROUTINE THEN THE FIRST SYNC. :IS NOT DETECTED. NO BAD DATA :IS GIVEN BECAUSE SYNC=144000 :CANNOT BE READ. WORD NO :IS '1' BECAUSE THIS IS THE FIRST :WORD TESTED
8753						
8754						
8755						
8756						
8757						
8758						
8759						
8760	047520	000240	HEDGAP:	NOP		:IF 'ERROR 3' INSERTED BY :RDHEAD SUBROUTINE THEN THE :HEADER GAP 0'S WERE NOT :WRITTEN RIGHT. :IF 'WORD NO' CONTAINS, SAY :3(8), THEN IT IS THE THIRD :WORD OF A 5 WORD HEADER :GAP THAT IS WRONG :'BAD DATA' CONTAINS WHAT IS :GOING ON THE DISK
8761						
8762						
8763						
8764						
8765						
8766						
8767						
8768						
8769						
8770						
8771	047522	000240	HEDSYN:	NOP		:IF 'ERROR 3' INSERTED BY RDHEAD :SUBROUTINE THEN THE HEADER SYNC. :GENERATED BY DCL IS WRONG :OR THE LAST BYTE :OF THE HEADER GAP 0'S IS WRONG :IN EITHER CASE WORD NO=6 :RIGHT BYTE IS HEADER 0 :LEFT BYTE IS SYNC :'BAD DATA' HAS WHAT IS GOING :ON DISK
8772						
8773						
8774						
8775						
8776						
8777						
8778						
8779						
8780						
8781						
8782	047524	005737		TST	@#ERFLG\$:ARE ANY ERRORS DETECTED
8783	047530	001017		BNE	OUT	:IF YES, EXIT ----->
8784	047532	005737		TST	@#X	:IS IT A DATA WRITE ?
8785	047536	001410		BEQ	DAREAD	:NO...THEN DO A DATA READ
8786	047540	005737		TST	@#NOSYNC	:IS THIS FORCED HEADER ERROR COMMAND
8787						:IF YES NOSYNC=-1 THEN WRITE OR READ
8788						:IS SHUT OFF SO BRANCH OUT
8789						:IF NOSYNC=0 THEN CONTINUE
8790	047544	001011		BNE	OUT	:EXIT IF SET ----->

```
8791
8792 047546 004137 051072          JSR    R1,@#WRDATA      ;WRITE DATA <----->
8793
8794 047552 000000          NOWORD: .WORD 0        ;NO OF WORDS TO BE WRITTEN
8795 047554 000000          Y:      .WORD 0        ;
8796 047556 000404          BR      OUT            ;EXIT ----->
8797
8798 047560 004137 054026          DAREAD: JSR    R1,@#REDATA ;READ DATA <----->
8799 047564 000000          DAWORD: .WORD 0        ;NO OF WORDS TO BE READ
8800 047566 000000          .WORD 0
8801 047570          OUT:
8802 047570 012605          MOV     (SP)+,R5       ;;POP STACK INTO R5
8803 047572 012604          MOV     (SP)+,R4       ;;POP STACK INTO R4
8804 047574 012603          MOV     (SP)+,R3       ;;POP STACK INTO R3
8805 047576 012602          MOV     (SP)+,R2       ;;POP STACK INTO R2
8806 047600 012601          MOV     (SP)+,R1       ;;POP STACK INTO R1
8807 047602 012600          MOV     (SP)+,R0       ;;POP STACK INTO R0
8808 047604 000207          RTS     PC             ;EXIT ROUTINE
```

8809
8810
8811
8812
8813
8814
8815 047606 014400
8816 047610 000000
8817 047612 000000
8818 047614 0C0000
8819 047616 000000
8820
8821
8822
8823
8824
8825
8826
8827
8828
8829
8830
8831
8832

:*THE DISK SECTOR IS DEVIDED AS FOLLOWS
:*19 WORDS OF 0, ONE WORD 144000
:*THESE MAKE 39 BYTES FOR SECTOR GAP AND ONE SYNC. BYTE

RSYNC: 14400
RCYL: 0
RSETR: 0
RKEY1: 0
RKEY2: 0

:*5 WORDS OF 0 ONE WORD 144000
:*THESE MAKE 11 BYTES FOR HEADER GAP AND ONE SYNC. BYTE
:*THESE ARE DCL GENERATED

:*THERE ARE 256 WORDS OF DATA
:*THERE ARE 2 WORDS FOR ECC GENERATED BY DCL
:*15 WORDS OF 0 FOR DATA GAP AND TOLERANCE GAP

```
8833
8834
8835
8836
8837          ;*READ DISK HEADER
8838
8839
8840
8841
8842
8843 047620 000000      NOSYNC: 0          ;FORCED HEADER ERROR = -1
8844                                     ;NORMAL = 0
8845 047622 000000      TY: 0          ;ERROR TYPE NO.
8846 047624 000000      ERWORD: 0       ;ERROR WORD NO.
8847
8848
8849
8850
8851 047626 012137 047610      RDHEAD: MOV (R1)+, @#RCYL ;STORE CYLINDER ADDRESS
8852 047632 012137 047612      MOV (R1)+, @#RSETR ;STORE SECTOR AND TRACK ADDRESS
8853 047636 012137 047614      MOV (R1)+, @#RKEY1 ;STORE KEY1
8854 047642 012137 047616      MOV (R1)+, @#RKEY2 ;STORE KEY2
8855 047646 012137 050416      MOV (R1)+, @#COMPA ;STORE COMPARE OR NOT
8856 047652 010146      MOV R1, -(SP) ;:PUSH R1 ON STACK
8857 047654 013700 001660      MOV @#RHMR, R0 ;R0 CONTAINS MAINTANENCE REG.
8858 047660 012705 000002      MOV #2, R5 ;R5 IS A COUNTER FOR WORDS
8859 047664 012710 000001      MOV #DMD, @R0 ;DIAG. MODE
8860 047670 052710 000010      BIS #MSTCK, @R0 ;SET SECTOR FOR FIRST WORD
8861 047674 052710 000002      BIS #MCLK, @R0 ;SET CLOCK FOR FIRST WORD
8862 047700 042710 000012      BIC #MSTCK!MCLK, @R0 ;RESET SECTOR AND CLOCK
8863 047704 000404      BR 2$ ;BRANCH OVER GIVING SECTOR FOR FIRST TIME
8864 047706 012710 000013      1$: MOV #MSTCK!MCLK!DMD, @R0 ;SET SECTOR, CLOCK, DIAG. MODE, RESET INDEX
8865 047712 042710 000012      BIC #MSTCK!MCLK, @R0 ;RESET SECTOR, CLOCK
8866 047716 012702 000007      2$: MOV #7, R2 ;R2 IS A COUNTER FOR BYTES
8867 047722 052710 000002      3$: BIS #MCLK, @R0 ;SET CLOCK
8868 047726 042710 000002      BIC #MCLK, @R0 ;RESET CLOCK
8869 047732 005302      DEC R2 ;BYTE COUNTER
8870 047734 001372      BNE 3$ ;BRANCH IF BYTE NOT COMPLETE
8871 047736 005305      DEC R5 ;WORD COUNTER
8872 047740 001362      BNE 1$ ;BRANCH IF WORD NOT COMPLETE
8873 047742 012702 000022      MOV #18, R2 ;NO OF WORDS OF ZEROS
8874 047746 005037 050414      4$: CLR @#WORD ;READ 0
8875 047752 004737 050420      JSR PC, @#READ ;GO TO READ
8876 047756 005302      DEC R2 ;COUNT
8877 047760 001372      BNE 4$
8878 047762 013737 047606 050414      MOV @#RSYNC, @#WORD ;SYNC. WORD
8879 047770 004737 050420      JSR PC, @#READ
8880 047774 032710 001000      BIT #DTSY, @R0 ;SYNC. BYTE DETECTED?
8881 050000 001012      BNE 5$ ;BRANCH IF SYNC DETECTED
8882 050002 012737 000001 047624      MOV #1, @#ERWORD ;ERROR WORD NO
8883 050010 013737 047606 001124      MOV @#RSYNC, @#$GDDAT ;SYNC WORD
8884 050016 012737 104002 047516      MOV #104002, @#SSYN ;INSERT "ERROR 2" IN SSYN
8885 050024 000571      BR 13$ ;BRANCH OUT
8886 050026 013737 047610 050414      5$: MOV @#RCYL, @#WORD ;SETUP CYLINDER
8887 050034 004737 050420      JSR PC, @#READ ;READ
8888 050040 013737 047612 050414      MOV @#RSETR, @#WORD ;SETUP SECTOR/TRACK
```

```

8889 050046 004737 050420 JSR PC,@#READ ;READ
8890 050052 013737 047614 050414 MOV @#RKEY1,@#WORD ;SETUP KEY1
8891 050060 004737 050420 JSR PC,@#READ ;READ
8892 050064 013737 047616 050414 MOV @#RKEY2,@#WORD ;SETUP KEY2
8893 050072 004737 050420 JSR PC,@#READ ;READ
8894 050076 013737 051404 050414 MOV @#WCRC,@#WORD ;SETUP CRC
8895 050104 004737 050420 JSR PC,@#READ ;READ
8896 050110 005737 002026 TST @#TESDTE ;IS THIS A DRIVE TIMING ERROR
8897 050114 001135 BNE 13$ ;BRANCH OUT IF YES
8898 050116 005737 050416 TST @#COMPA ;IS THIS A READ OR WRITE COMMAND
8899 050122 001472 BEQ 11$
8900 050124 012705 051406 MOV #HEGAP,R5 ;POINTER FOR HEADER GAP
8901 050130 012702 000005 MOV #5,R2 ;NO OF WORDS OF ZEROS
8902 050134 012737 000006 047624 6$: MOV @#ERWORD ;ERROR WORD NO SET
8903 050142 004737 050652 JSR PC,@#WRITE ;FOR HEADER GAP
8904 050146 005737 050650 TST @#WORD ;TEST WRITTEN WORD
8905 050152 001413 BEQ 7$ ;BRANCH IF GOOD THAT IS 0
8906 050154 160237 047624 SUB R2,@#ERWORD ;WORD NO IN ERROR
8907 050160 005037 001124 CLR @#$GDDAT ;GOOD WORD SHOULD BE 0
8908 050164 013737 050650 001126 MOV @#WORD,$BDDAT ;BAD DATA
8909 050172 012737 104003 047520 MOV #104003,@#HEDGAP ;'ERROR 2' GOES IN HEDGAP
8910 050200 000503 BR 13$ ;BRANCH OUT
8911 050202 013725 050650 7$: MOV @#WORD,(R5)+ ;SAVE HEADER GAP
8912 050206 005302 DEC R2
8913 050210 001351 BNE 6$
8914 050212 004737 050652 JSR PC,@#WRITE ;WRITE HEADER (DATA) GAP SYNC
8915 050216 023737 047606 050650 CMP @#RSYNC,@#WORD
8916 050224 001426 BEQ 10$
8917 050226 005737 047620 TST @#NOSYNC ;IS THIS FORCED HEADER ERROR COMMAND
8918 ;IF YES NOSYNC=-1 THEN WRITE OR READ
8919 ;IS SHUT OFF SO BRANCH OUT
8920 ;IF NO NOSYNC=0 THEN CONTINUE
8921 050232 001406 BEQ 14$ ;BRANCH IF TRUE ERROR
8922 050234 005737 050650 TST @#WORD
8923 050240 001420 BEQ 10$ ;BRANCH IF GOOD
8924 050242 005037 001124 CLR @#$GDDAT ;IT SHOULD BE ZERO
8925 050246 000403 BR 15$ ;BRANCH TO TYPE ERROR
8926 050250 013737 047606 001124 14$: MOV @#RSYNC,@#$GDDAT ;GOOD DATA
8927 050256 013737 050650 001126 15$: MOV @#WORD,@#$BDDAT ;BAD DATA
8928 050264 012737 000006 047624 MOV #6,@#ERWORD
8929 050272 012737 104003 047522 MOV #104003,@#HEDSYN
8930 050300 000443 BR 13$ ;BRANCH OUT
8931 050302 013725 050650 10$: MOV @#WORD,(R5)+ ;SAVE DATA SYNC.
8932 050306 000440 BR 13$
8933 ;*READ COMMAND START FROM HERE
8934 050310 012702 000005 11$: MOV #5,R2
8935 050314 005037 050414 12$: CLR WORD
8936 050320 004737 050420 JSR PC,READ ;READ HEADER GAP
8937 050324 005302 DEC R2 ;IS 5 HEADER GAP ZEROS COMPLETE
8938 050326 001372 BNE 12$ ;IF NOT BRANCH
8939 050330 013737 047606 050414 MOV @#RSYNC,@#WORD ;SYNC WORD
8940 050336 004737 050420 JSR PC,READ ;READ HEADER (DATA) SYNC)
8941 050342 005737 047620 TST @#NOSYNC
8942 050346 001404 BEQ 16$ ;IF NOT ERROR COMMAND BRANCH
8943 050350 032710 001000 BIT #DTSY,@R0 ;SYNC. DETECTED
8944 050354 001415 BEQ 13$ ;IF ZERO BRANCH OUT

```



```
8945 050356 000403          BR      17$          ;IF NOT ZERO BRANCH TO ERROR
8946 050360 032710 001000    16$:   BIT      #DTSY, @R0      ;SYNC. DETECTED?
8947 050364 001011          BNE     13$          ;BRANCH IF YES
8948 050366 012737 000006 047624 17$:   MOV      #6,@#ERWORD      ;ERROR WORD NO.
8949 050374 013737 047606 001124    MOV      @#RSYNC,@#$GDDAT;SYNC WORD
8950 050402 012737 104002 047522    MOV      #104002,@#HEDSYN
8951 050410          13$:
8952 050410 012601          MOV      (SP)+,R1      ;;POP STACK INTO R1
8953 050412 000201          RTS      R1
8954
8955
```

8956
8957
8958
8959
8960
8961
8962
8963
8964
8965
8966
8967
8968
8969
8970
8971
8972
8973
8974
8975
8976
8977
8978
8979
8980
8981
8982
8983
8984
8985
8986
8987
8988
8989
8990
8991
8992
8993
8994
8995
8996
8997
8998
8999
9000
9001
9002
9003
9004
9005
9006
9007
9008
9009
9010
9011

050414 000000
050416 000000

050420
050420 010246
050422 012705 000002
050426 012710 000001
050432 006037 050414
050436 103002
050440 052710 000020
050444 012702 000007
050450 052710 000012
050454 005737 045320
050460 001411
050462 032710 000020
050466 001404
050470 012737 177777 045312
050476 000402
050500 005037 045312
050504 012746 000001
050510 006037 050414
050514 103002
050516 012716 000021
050522 012610
050524 005737 045320
050530 001404
050532 005237 045332
050536 004737 045350
050542 052710 000002
050546 005737 045320
050552 001411
050554 032710 000020
050560 001404
050562 012737 177777 045312
050570 000402
050572 005037 045312
050576 012746 000001
050602 006037 050414
050606 103002
050610 012716 000021
050614 012610
050616 005737 045320

;*READ ONE WORD IN 'WORD'

WORD: 0
COMPA: 0

READ:

```

MOV R2,-(SP)      ;;PUSH R2 ON STACK
MOV #2,R5         ;;WORD COUNTER
MOV #DMD,@R0     ;;SET DIAG. MODE
ROR @#WORD       ;;CHECKING IF THERE IS A ONE
BCC 1$           ;;IF NO ONE BRANCH
BIS #MRD,@R0     ;;SET BIT 4 IF DATA HAS ONE
1$: MOV #7,R2     ;;BYTE COUNTER
    BIS #MSTCK!MCLK,@R0 ;;SET CLOCK,DATA IF ANY, SECTOR
    TST @#TSECCG     ;;IS THIS BIT TO GENERATE AND TEST ECC
    BEQ 6$           ;;BRANCH IF NO
    BIT #MRD,@R0    ;;IS DATA BIT A ONE
    BEQ 5$           ;;BRANCH IF DATA BIT IS 0
    MOV #-1,@#ECDATA ;;ECC DATA BIT IS A ONE
    BR 6$           ;;BRANCH
5$: CLR @#ECDATA   ;;ECC DATA BIT IS A 0
6$: MOV #DMD,-(SP) ;;KEEP ONLY DIAG. MODE
    ROR @#WORD     ;;CHECKING IF THERE IS A ONE
    BCC 2$        ;;IF NO ONE BRANCH
    MOV #MRD!DMD,(SP) ;;KEEP DATA AND DIAG. MODE
    2$: MOV (SP)+,@R0 ;;PUT IN DATA,RESET CLOCK, SECTOR
    TST @#TSECCG   ;;IS ECC TO BE GENERATED FOR THIS BIT
    BEQ 3$        ;;BRANCH IF NO
    INC @#DATENV   ;;NUMBER OF CLOCKS GIVEN FOR DATA ENVELOPE
    JSR PC,@#ECTEST ;;GO TO GENERATE AND TEST ECC
    3$: BIS #MCLK,@R0 ;;SET CLOCK
    TST @#TSECCG   ;;IS THIS BIT TO GENERATE ECC
    BEQ 8$        ;;BRANCH IF NO
    BIT #MRD,@R0  ;;IS DATA BIT A ONE
    BEQ 7$        ;;BRANCH IF DATA BIT IS = 0
    MOV #-1,@#ECDATA ;;ECC DATA BIT IS A ONE
    BR 8$        ;;BRANCH
7$: CLR @#ECDATA  ;;ECC DATA BIT IS = 0
8$: MOV #DMD,-(SP) ;;KEEP DIAG. MODE
    ROR @#WORD     ;;CHECKING IF THERE IS A ONE
    BCC 4$        ;;BRANCH IF NO ONE
    MOV #MRD!DMD,(SP) ;;KEEP DIAG. MODE AND DATA
    4$: MOV (SP)+,@R0 ;;SET DATA, DIAG. MODE, CLEAR CLOCK
    TST @#TSECCG   ;;IS THIS BIT TO GENERATE ECC
    
```

9012 050622 001404
9013 050624 005237 045332
9014 050630 004737 045350
9015 050634 005302
9016 050636 001341
9017 050640 005305
9018 050642 001300
9019 050644 012602
9020 050646 000207
9021
9022

9\$:

BEQ 9\$
INC @#DATE/IV
JSR PC,@#ECTEST
DEC R2
BNE 3\$
DEC R5
BNE 1\$
MOV (SP)+,R2
RTS PC

:BRANCH IF NO
:NUMBER OF CLOCKS GIVEN FOR DATA ENVELOPE
:GO TO GENERATE AND TEST ECC
:BYTE COUNTER
:BRANCH IF ONE BYTE NOT COMPLETE
:WORD COUNTER
:BRANCH IF ONE WORD NOT COMPLETE
:;POP STACK INTO R2

```

9023
9024
9025
9026
9027          ;*WRITE ONE WORD WHICH COMES BACK IN 'WORD'
9028
9029
9030
9031
9032
9033
9034 050650 000000          WORD: 0
9035
9036
9037
9038
9039 050652          WRITE:
9040 050652 010046          MOV     R0,-(SP)          ;;PUSH R0 ON STACK
9041 050654 010246          MOV     R2,-(SP)          ;;PUSH R2 ON STACK
9042 050656 010346          MOV     R3,-(SP)          ;;PUSH R3 ON STACK
9043 050660 010546          MOV     R5,-(SP)          ;;PUSH R5 ON STACK
9044 050662 012705 000002          MOV     #2,R5          :WORD COUNTER
9045 050666 012710 000001          MOV     #1,@R0          :SET DIAG. MODE
9046                                     :RO HAS RHMR ADDRESS IN IT
9047 050672 012702 000007          1$:  MOV     #7,R2          :BYTE COUNTER
9048 050676 012710 000013          MOV     #MSTCK!MCLK!DMD,@R0;SET SECTOR AND CLOCK
9049 050702 032710 000040          BIT     #MWR,@R0          :CHECK WRITE BIT IN MAINT. REG.
9050 050706 001406          BEQ     2$          :BRANCH IF ZERO
9051 050710 012737 177777 045312          MOV     #-1,@#ECDATA          :ECC DATA BIT IS A ONE
9052 050716 000261          SEC          :SET CARRY
9053 050720 006003          ROR     R3          :MOVE 1 FORWARD
9054 050722 000404          BR     3$
9055 050724 005037 045312          2$:  CLR     @#ECDATA          :ECC DATA BIT IS = 0
9056 050730 000241          CLC          :CLEAR CARRY
9057 050732 006003          ROR     R3          :MOVE 0 FOR WORD
9058 050734 012710 000001          3$:  MOV     #DMD,@R0          :CLEAR SECTOR AND CLOCK
9059 050740 005737 045320          TST     @#TSECCG          :IS THIS BIT TO GENERATE ECC
9060 050744 001404          BEQ     4$          :BRANCH IF NO
9061 050746 005237 045332          INC     @#DATENV          :NUMBER OF CLOCKS GIVEN FOR DATA ENVELOPE
9062 050752 004737 045350          JSR     PC,@#ECTEST          :GO TO GENERATE AND TEST ECC
9063 050756 052710 000002          4$:  BIS     #MCLK,@R0          :SET CLOCK IN RHMR
9064 050762 032710 000040          BIT     #MWR,@R0          :CHECK WRITE BIT IN RHMR
9065 050766 001406          BEQ     5$          :BRANCH IF ZERO
9066 050770 012737 177777 045312          MOV     #-1,@#ECDATA          :ECC DATA BIT IS A ONE
9067 050776 000261          SEC          :SET CARRY
9068 051000 006003          ROR     R3          :MOVE 1 FOR WORD
9069 051002 000404          BR     6$
9070 051004 005037 045312          5$:  CLR     @#ECDATA          :ECC DATA BIT IS ZERO
9071 051010 000241          CLC          :CLEAR CARRY
9072 051012 006003          ROR     R3          :MOVE 0 FOR WORD
9073 051014 012710 000001          6$:  MOV     #DMD,@R0          :CLEAR CLOCK
9074 051020 005737 045320          TST     @#TSECCG          :IS THIS BIT TO GENERATE ECC
9075 051024 001404          BEQ     7$          :BRANCH IF NO
9076 051026 005237 045332          INC     @#DATENV          :NUMBER OF CLOCKS GIVEN FOR DATA ENVELOPE
9077 051032 004737 045350          JSR     PC,@#ECTEST          :GO TO GENERATE AND TEST ECC
9078 051036 005302          7$:  DEC     R2          :COUNT FOR BYTE END

```

9079	051040	001346	BNE	4\$:IF NOT BYTE END BRANCH
9080	051042	005305	DEC	R5	:COUNT FOR WORD END
9081	051044	001312	BNE	1\$:IF NOT WORD END BRANCH
9082					
9083	051046	010337 050650	MOV	R3,@#WORD	:STORE THE WORD
9084					
9085	051052	012605	MOV	(SP)+,R5	::POP STACK INTO R5
9086	051054	012603	MOV	(SP)+,R3	::POP STACK INTO R3
9087	051056	012602	MOV	(SP)+,R2	::POP STACK INTO R2
9088	051060	012600	MOV	(SP)+,R0	::POP STACK INTO R0
9089	051062	000207	RTS	PC	
9090					
9091					

```

9092
9093
9094
9095
9096          :*WRITE DATA - PUT DATA INTO 'DISK' AREA FROM 'WORD'
9097          :*ONE WORD AT A TIME
9098
9099
9100
9101
9102
9103
9104
9105 051064 000000          COUNTD: 0
9106 051066 000400          FORMAT: 256.
9107 051070 000000          ZWORDS: 0
9108 051072          WRDATA:
9109 051072 011137 051064          MOV      (R1),@#COUNTD      ;STORE NO. OF WORDS TO BE WRITTEN
9110 051076 012102          MOV      (R1)+,R2            ;SAME IN R2
9111 051100 012137 050416          MOV      (R1)+,@#COMPA      ;COMPARE OR NOT
9112
9113 051104 010046          MOV      R0,-(SP)           ;;PUSH R0 ON STACK
9114 051106 010146          MOV      R1,-(SP)           ;;PUSH R1 ON STACK
9115 051110 010246          MOV      R2,-(SP)           ;;PUSH R2 ON STACK
9116 051112 010346          MOV      R3,-(SP)           ;;PUSH R3 ON STACK
9117 051114 010446          MOV      R4,-(SP)           ;;PUSH R4 ON STACK
9118
9119 051116 012701 000016          MOV      #14,R1            ;NO. OF TOLERANCE GAP WORDS
9120 051122 012703 052430          MOV      #TOLGAP,R3        ;START OF TOLERANCE GAP TABLE
9121 051126 012723 177777          1$: MOV      #-1,(R3)+        ;MAKE IT 177777
9122 051132 005301          DEC      R1                ;IS 14 COMPLETED
9123 051134 001374          BNE     1$                 ;IF NO BRANCH
9124 051136 013700 001660          MOV      @#RHMR,R0         ;R0 CONTAINS MAINTANENCE REG.
9125 051142 013746 051066          MOV      @#FORMAT,-(SP)
9126 051146 163716 051064          SUB      @#COUNTD,(SP)
9127 051152 011637 051070          MOV      (SP),@#ZWORDS     ;NO. OF ZERO WORDS TO BE WRITTEN
9128 051156 012604          MOV      (SP)+,R4
9129 051160 005737 002024          TST     @#TSECC            ;IS THIS AN ECC TEST ?
9130 051164 001403          BEQ     7$                 ;BRANCH IF NO
9131 051166 012737 177777 045320          MOV      #-1,@#TSECCG      ;THESE BITS ARE TO GENERATE ECC
9132 051174 012703 051422          7$: MOV      #DISK,R3        ;ADDRESS THE 'DISK' AREA
9133 051200 004737 050652          2$: JSR     PC,@#WRITE      ;WRITE INTO 'WORD'
9134 051204 013723 050650          MOV      @#WORD,(R3)+     ;STORE ON SIMULATED DISK
9135 051210 005302          DEC      R2                ;COUNT DOWN
9136 051212 001372          BNE     2$                 ;CONTINUE IF ALL WORDS NOT WRITTEN
9137 051214 005704          TST     R4                ;ANY ZEROS TO BE WRITTEN ?
9138 051216 001406          BEQ     4$                 ;BRANCH IF NONE TO BE WRITTEN
9139 051220 004737 050652          3$: JSR     PC,@#WRITE      ;WRITE ZEROS INTO 'WORD'
9140 051224 013723 050650          MOV      @#WORD,(R3)+     ;STORE INTO 'DISK'
9141 051230 005304          DEC      R4
9142 051232 001372          BNE     3$
9143 051234 005037 045320          4$: CLR     @#TSECCG        ;NO MORE ECC TO BE GENERATED
9144 051240 012701 000002          MOV      #2,R1
9145 051244 004737 050652          5$: JSR     PC,WRITE        ;WRITE ECC1 AND ECC2 ON SIMULATED DISK
9146 051250 013723 050650          MOV      @#WORD,(R3)+     ;STORE ON WEEC1 AND WEEC2
9147 051254 005301          DEC      R1
  
```

```
9148 051256 001372          BNE      5$
9149 051260 004737 050652    JSR      PC,WRITE          ;WRITE DATA GAP INTO 'WORD'
9150 051264 013723 050650    MOV      @#WORD,(R3)+     ;STORE INTO 'DISK'
9151 051270 012701 000016    MOV      #14.,R1
9152 051274 004737 050652    6$:     JSR      PC,@#WRITE   ;WRITE TOLERANCE GAP ZEROS
9153 051300 013723 050650    MOV      @#WORD,(R3)+     ;STORE INTO 'DISK'
9154 051304 005301          DEC      R1
9155 051306 001372          BNE      6$
9156
9157 051310 012604          MOV      (SP)+,R4         ;;POP STACK INTO R4
9158 051312 012603          MOV      (SP)+,R3         ;;POP STACK INTO R3
9159 051314 012602          MOV      (SP)+,R2         ;;POP STACK INTO R2
9160 051316 012601          MOV      (SP)+,R1         ;;POP STACK INTO R1
9161 051320 012600          MOV      (SP)+,R0         ;;POP STACK INTO R0
9162
9163 051322 000201          RTS      R1
9164
9165
```

9166
9167
9168
9169
9170
9171
9172
9173
9174
9175
9176
9177
9178
9179
9180
9181
9182
9183
9184
9185
9186
9187
9188
9189
9190
9191
9192
9193
9194
9195
9196
9197

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

051324 000023
051372 000001
051374 000004
051404 000001
051406 000005
051420 00000i
051422
051422 000400
052422 000001
052424 000001
052426 000001
052430 000016

SECGAP: .BLKW 19.
WSSYNC: .BLKW 1
HEADER: .BLKW 4
WCRC: .BLKW 1
HEGAP: .BLKW 5
HDWSYN: .BLKW 1
SILOTB:
DISK: .BLKW 256.
WECC1: .BLKW 1
WECC2: .BLKW 1
DTAGAP: .BLKW 1
TOLGAP: .BLKW 14.

:SECTOR GAP 38 BYTES OF 0
:SECTOR GAP 1 BYTE OF 0 ONE SYNC BYTE
:HEADER = CYL, SECTOR/TRACK, KEY1, KEY2
:CRC
:HEADER GAP 10 BYTES OF 0
:HEADER GAP 1 BYTE OF 0 ONE SYNC. BYTE
:USED IN SILO TEST AS SILO TABLE
:DATA SPACE
:ECC1
:ECC2
:DATA GAP 2 BYTES OF 0
:TOLERANCE GAP 28 BYTES OF 0


```

9198
9199
9200
9201
9202           ;*WRITE HEADER AND DATA
9203
9204
9205
9206
9207           ;**THIS SUBROUTINE IS THE FIRST IN A SERIES OF NESTED SUBROUTINES
9208
9209           ;**IT ISSUES DIAGNOSTIC MODE, AN EXTRA DIAGNOSTIC INDEX, AND THE
9210           ;**'GO' BIT
9211
9212           ;**IT THEN CALL THREE OTHER SUBROUTINES, WHICH IN TURN CALL OTHER
9213           ;**SUBROUTINES.  THESE ARE:
9214
9215           ;**      SEARCH
9216           ;**      WRHEAD
9217           ;**      WRDATA
9218
9219
9220
9221
9222
9223
9224
9225 052464 000000          RNCTR1: .WORD 0           ;'RUN' LINE STALL COUNTER
9226 052466 011637 002014 COMWHD: MOV (SP),@#PCJSR      ;SAVE PC OF JSR + 4
9227 052472 162737 000004 002014     SUB #4,@#PCJSR      ;SAVE PC OF JSR
9228 052500 010046          MOV R0,-(SP)        ;:PUSH R0 ON STACK
9229 052502 010146          MOV R1,-(SP)        ;:PUSH R1 ON STACK
9230 052504 010246          MOV R2,-(SP)        ;:PUSH R2 ON STACK
9231 052506 010346          MOV R3,-(SP)        ;:PUSH R3 ON STACK
9232 052510 010446          MOV R4,-(SP)        ;:PUSH R4 ON STACK
9233 052512 010546          MOV R5,-(SP)        ;:PUSH R5 ON STACK
9234
9235 052514 012777 000001 127136     MOV #DMD,@RHMR      ;SET DIAGNOSTIC MODE
9236 052522 052777 000004 127130     BIS #MINX,@RHMR     ;SET DIAGNOSTIC INDEX
9237 052530 042777 000004 127122     BIC #MINX,@RHMR     ;CLEAR IT
9238 052536 052777 000001 127074     BIS #GO,@RHCS1     ;SET 'GO' BIT & STALL 'TILL 'RUN'
9239
9240 052544 012737 000113 052464     RNWAT1: MOV #75.,@#RNCTR1 ;LOAD STALL COUNTER = APPROX. 450US
9241                                     ;FOR 11/50 CPU WITH CORE MEMORY
9242 052552 005337 052464           1$: DEC @#RNCTR1      ;COUNT DOWN 1 TIME
9243 052556 001375                                     BNE 1$             ;CONTINUE UNTIL 0
9244
9245 052560 013746 052644           MOV @#WSECTR,-(SP) ;GET DESIRED SECTOR/TRACK
9246 052564 042716 177740           BIC #177740,(SP)  ;MAKE ONLY SECTOR
9247 052570 012637 052600           MOV (SP)+,@#WTRK  ;SAVE SECTOR
9248
9249 052574 004137 053552           JSR R1,@#SEARCH   ;ISSUE SECTOR CLOCKS <----->
9250 052600 000000           WTRK: .WORD 0     ;SECTOR NO.
9251
9252 052602 012701 000240           MOV #+NOP,R1     ;GOING TO MOVE NOPS
9253 052606 010137 052654           MOV R1,@#SEGP   ;NOP INTO SEGAP

```



```

9310                                     ;GOOD DATA IS WHAT SHOULD BE ON DISK
9311                                     ;BAD DATA IS WHAT IS GOING ON DISK
9312                                     ;WORD NO IS 5.
9313 052664 000240                     ERHDGP: NOP
9314                                     ;IF 'ERROR 6' INSERTED BY
9315                                     ;WRHEAD SUBROUTINE THEN HEADER
9316                                     ;GAP GOING ON DISK IS WRONG.
9317
9318                                     ;WORD NO. GIVES WHICH OF
9319                                     ;THE HEADER GAP WORDS
9320                                     ;ARE WRONG. FOR EXAMPLE:
9321
9322                                     ;WORD NO 1 = FIRST HEADER
9323                                     ;GAP WORD
9324                                     ;BAD WORD IS WHAT IS GOING ON DISK
9325
9326 052666 000240                     HDESYN: NOP
9327
9328                                     ;IF 'ERROR 6' INSERTED BY
9329                                     ;WRHEAD SUBROUTINE THEN LAST
9330                                     ;HEADER GAP BYTE OR HEADER
9331                                     ;SYNC BYTE GOING ON DISK IS WRONG.
9332
9333                                     ;WORD NO = 5
9334                                     ;BAD DATA IS WHAT IS GOING
9335                                     ;ON DISK, RIGHT BYTE IS HEADER
9336                                     ;GAP 0 BYTE, LEFT BYTE IS HEADER
9337                                     ;GAP SYNC.
9338
9339 052670 005737 002006                 TST    @#ERFLG$
9340 052674 001004                       BNE    FOUT
9341                                     ;ARE ANY ERRORS DETECTED
9342                                     ;IF YES EXIT ----->
9343 052676 004137 051072                 JSR    R1,@#WRDATA
9344 052702 000000                       .WORD 0
9345 052704 000000                       .WORD 0
9346                                     ;WRITE THE DATA <----->
9347                                     ;FORMAT COMMAND NO. OF DATA
9348
9349                                     FOUT:
9350                                     MOV    (SP)+,R5      ;;POP STACK INTO R5
9351                                     MOV    (SP)+,R4      ;;POP STACK INTO R4
9352                                     MOV    (SP)+,R3      ;;POP STACK INTO R3
9353                                     MOV    (SP)+,R2      ;;POP STACK INTO R2
9354                                     MOV    (SP)+,R1      ;;POP STACK INTO R1
9355                                     MOV    (SP)+,R0      ;;POP STACK INTO R0
9356                                     RTS    PC

```

```
9355
9356          :*WRITE HEADER
9357
9358
9359
9360
9361
9362          :*R0 = MAINT.REG.
9363          :*R1 = SIMULATED DISK
9364          :*R2 = BYTE COUNT
9365          :*R3 = WRITE WORD
9366          :*R5 = WORD COUNT
9367
9368
9369
9370 052724 000000          SCYL: 0
9371 052726 000000          SSECTR: 0
9372 052730 000000          SKEY1: 0
9373 052732 000000          SKEY2: 0
9374 052734 000000          SCRC: 0
9375
9376
9377 052736 012137 052724  WRHEAD: MOV (R1)+,@#SCYL
9378 052742 012137 052726          MOV (R1)+,@#SSECTR
9379 052746 012137 052730          MOV (R1)+,@#SKEY1
9380 052752 012137 052732          MOV (R1)+,@#SKEY2
9381 052756 012137 052734          MOV (R1)+,@#SCRC
9382 052762 010146          MOV R1,-(SP)          ;;PUSH R1 ON STACK
9383 052764 012701 051324          MOV #SECGAP,R1      ;;SIMULATED DISK INDICATOR
9384 052770 013700 001660          MOV @#RHMR,R0      ;;R0 NOW HAS MAINT. REG. ADDR.
9385 052774 012710 000001          MOV #DMD,@R0       ;;SET DIAG. MODE IN RHMR
9386 053000 012705 000002          MOV #2,R5          ;;WORD COUNTER
9387 053004 052710 000010          BIS #MSTCK,@R0     ;;SET SECTOR FOR FIRST BYTE
9388 053010 012710 000013 1$: MOV #MSTCK!MCLK!DMD,@R0 ;;SET SECTOR, CLOCK, DIAG. MODE, RESET INDEX
9389 053014 032710 000040          BIT #MWR,@R0      ;;CHECK WRITE BIT IN MAINT. REG.
9390 053020 001403          BEQ 2$
9391 053022 000261          SEC              ;;SET CARRY
9392 053024 006003          ROR R3          ;;MOVE ONE FORWARD
9393 053026 000402          BR 3$
9394 053030 000241 2$: CLC              ;;CLEAR CARRY
9395 053032 006003          ROR R3          ;;MOVE ZERO FORWARD
9396 053034 012710 000001 3$: MOV #DMD,@R0      ;;CLEAR CLOCK, SECTOR
9397 053040 012702 000007          MOV #7,R2        ;;BYTE COUNTER
9398 053044 052710 000002 4$: BIS #MCLK,@R0    ;;SET CLOCK
9399 053050 032710 000040          BIT #MWR,@R0    ;;CHECK WRITE BIT IN MAINT.REG.
9400 053054 001403          BEQ 5$          ;;BRANCH IF ZERO
9401 053056 000261          SEC              ;;SET CARRY
9402 053060 006003          ROR R3          ;;MOVE ONE FORWARD
9403 053062 000402          BR 6$
9404 053064 000241 5$: CLC              ;;CLEAR CARRY
9405 053066 006003          ROR R3          ;;MOVE ZERO FORWARD
9406 053070 012710 000001 6$: MOV #DMD,@R0      ;;SET DIAG. MODE AGAIN IN RHMR
9407 053074 005302          DEC R2
9408 053076 001362          BNE 4$
9409 053100 005305          DEC R5
9410 053102 001342          BNE 1$          ;;CONTINUE
```

```

9411
9412 053104 010321      MOV      R3,(R1)+
9413 053106 005703      TST      R3
9414 053110 001414      BEQ      7$
9415 053112 012737 000001 047624    MOV      #1,@#ERWORD
9416 053120 005037 001124      CLR      @#$GDDAT
9417 053124 010337 001126      MOV      R3,@#$BDDAT
9418 053130 012737 104006 052654    MOV      #104006,@#SEGP
9419 053136 000137 053544      JMP      @#17$ ;BRANCH OUT ----->
9420
9421 053142 012702 000022      7$: MOV      #18.,R2 ;COUNT NO. OF SECTOR GAP
9422 053146 012737 000024 047624 10$: MOV      #20.,@#ERWORD ;COUNT TO GIVE ERROR WORD
9423 053154 004737 050652      JSR      PC,@#WRITE ;WRITE SECTOR GAP
9424 053160 013721 050650      MOV      @#WORD,(R1)+ ;STORE SECTOR GAP WORD
9425 053164 001413      BEQ      11$
9426 053166 160237 047624      SUB      R2,@#ERWORD ;IF NOT GET ERROR WORD NO.
9427 053172 005037 001124      CLR      @#$GDDAT ;GOOD WORD
9428 053176 013737 050650 001126    MOV      @#WORD,@#$BDDAT ;BAD WORD
9429 053204 012737 104006 052654    MOV      #104006,@#SEGP ;STORE 'ERROR 6' IN SEGP
9430 053212 000554      BR       17$ ;BRANCH OUT ----->
9431
9432 053214 005302      11$: DEC      R2 ;HAVE 18 WORDS OF ZEROS BEEN WRITTEN ?
9433 053216 001353      BNE      10$ ;IF NOT DO SO
9434
9435 ;*AT THIS POINT THE SECTOR FOUND FLOP SHOULD
9436 ;*BE HIGH. SO THAT THE HEADER SYNC BYTE CAN BE GIVEN
9437
9438 ;*HOWEVER IN THE DRIVE TIMING ERROR TEST THE REST OF THE ROUTINE
9439 ;*IS ABORTED - HEADER SYNC BYTE IS NOT GIVEN
9440
9441 053220 005737 002026      TST      @#TESDTE ;IS THIS A DRIVE TIMING ERROR
9442 053224 001147      BNE      17$ ;BRANCH OUT IF YES
9443 053226 004737 050652      JSR      PC,@#WRITE ;WRITE ONE SECTOR GAP 0 BYTE
9444 ;AND ONE SYNC. BYTE = 230
9445 053232 013711 050650      MOV      @#WORD,(R1) ;SAVE 0 BYTE AND SYNC BYTE
9446 053236 023721 047606      CMP      @#RSYNC,(R1)+ ;IF SYNC. BYTE RIGHT
9447 053242 001414      BEQ      12$ ;IF YES BRANCH
9448 053244 012737 000024 047624    MOV      #20.,@#ERWORD ;IF NOT GET READY FOR ERROR
9449 053252 013737 047606 001124    MOV      @#RSYNC,@#$GDDAT ;GOOD WORD
9450 053260 014137 001126      MOV      -(R1),@#$BDDAT ;BAD WORD
9451 053264 012737 104006 052656    MOV      #104006,@#FSYNER ;INSERT 'ERROR 6' IN FSYNER
9452 053272 000524      BR       17$ ;BRANCH OUT ----->
9453
9454 053274 012702 000004      12$: MOV      #4,R2 ;FOUR HEADER WORDS
9455 053300 012703 052724      MOV      #SCYL,R3 ;POINTER FOR HEADER TABLE
9456 053304 012737 000005 047624 13$: MOV      #5,@#ERWORD ;ERROR WORD NO SET
9457 053312 004737 050652      JSR      PC,@#WRITE ;WRITE 4 HEADER WORDS
9458 053316 013711 050650      MOV      @#WORD,(R1) ;STORE WRITTEN WORD
9459 053322 022321      CMP      (R3)+,(R1)+ ;IS IT RIGHT?
9460 053324 001412      BEQ      14$ ;IF GOOD CONTINUE
9461 ;IF NOT GET READY FOR PRINT
9462 053326 160237 047624      SUB      R2,@#ERWORD ;WORD NO
9463 053332 014337 001124      MOV      -(R3),@#$GDDAT ;GOOD DATA
9464 053336 014137 001126      MOV      -(R1),@#$BDDAT ;BAD DATA
9465 053342 012737 104006 052660    MOV      #104006,@#ERHEAD ;INSERT 'ERROR 6'
9466 053350 000475      BR       17$ ;BRANCH OUT ----->

```

```

9467
9468 053352 005302          14$:  DEC      R2          ;ARE 4 HEADER WORDS DONE?
9469 053354 001353          BNE      13$          ;IF NOT DO THEM
9470 053356 004737 050652    JSR      PC,@#WRITE   ;WRITE CRC
9471 053362 013711 050650    MOV      @#WORD,(R1)  ;STORE CRC
9472 053366 022137 052652    CMP      (R1)+,@#GCRC ;COMPARE GOOD CRC
9473 053372 001414          BEQ      20$          ;BRANCH IF GOOD
9474 053374 014137 001126    MOV      -(R1),@#SBDDATA ;BAD CRC WRITTEN
9475 053400 013737 052652 001124    MOV      @#GCRC,@#SGDDAT ;GOOD CRC
9476 053406 012737 000005 047624    MOV      #5,@#ERWORD   ;ERROR WORD NO
9477 053414 012737 104006 052662    MOV      #104006,@#ERCRC ;INSERT ERROR 6
9478 053422 000450          BR       17$          ;EXIT ----->
9479
9480 053424 012702 000005          20$:  MOV      #5,R2          ;NO OF HEADER GAP
9481 053430 012737 000006 047624 15$:  MOV      #6,@#ERWORD   ;ERROR WORD NO SET
9482 053436 004737 050652    JSR      PC,@#WRITE   ;WRITE HEADER GAP
9483 053442 013721 050650    MOV      @#WORD,(R1)+ ;STORE
9484 053446 001412          BEQ      16$          ;IF GOOD BRANCH
9485 053450 160237 047624    SUB      R2,@#ERWORD   ;ERROR WORD NO
9486 053454 005037 001124    CLR      @#SGDDAT     ;GOOD DATA
9487 053460 014137 001126    MOV      -(R1),@#SBDDAT ;BAD DATA
9488 053464 012737 104006 052664    MOV      #104006,@#ERHDGP ;STORE 'ERROR 6'
9489 053472 000424          BR       17$          ;BRANCH OUT ----->
9490
9491 053474 005302          16$:  DEC      R2          ;ARE 5 HEADER GAP ZEROS DONE
9492 053476 001354          BNE      15$          ;IF NOT BRANCH
9493 053500 004737 050652    JSR      PC,@#WRITE   ;WRITE CRC
9494 053504 013711 050650    MOV      @#WORD,(R1)  ;STORE CRC
9495 053510 023721 047606    CMP      @#RSYNC,(R1)+ ;COMPARE GOOD CRC
9496 053514 001413          BEQ      17$          ;BRANCH IF GOOD
9497 053516 012737 000005 047624    MOV      #5,@#ERWORD   ;ERROR WORD NO
9498 053524 014137 001126    MOV      -(R1),@#SBDDAT ;BAD DATA
9499 053530 013737 047606 001124    MOV      @#RSYNC,@#SGDDAT ;GOOD DATA
9500 053536 012737 104006 052666    MOV      #104006,@#HDESYN ;STORE 'ERROR 6'
9501
9502 053544          17$:  MOV      (SP)+,R1     ;;POP STACK INTO R1
9503 053544 012601          MOV      (SP)+,R1     ;;POP STACK INTO R1
9504
9505 053546 000201          RTS      R1
9506
9507

```

9508
9509
9510
9511
9512
9513
9514
9515
9516
9517
9518
9519
9520
9521
9522
9523
9524
9525
9526
9527
9528
9529
9530
9531
9532
9533
9534
9535
9536
9537
9538
9539
9540
9541
9542
9543
9544
9545
9546
9547
9548
9549
9550
9551
9552
9553
9554
9555
9556
9557
9558
9559
9560
9561
9562
9563

;*SEARCH SECTOR

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

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

```

9564 053636 005703          TST    R3          ;IF SECTOR REQUIRED JUMP OUT
9565 053640 001461          BEQ    7$          ;BRANCH OF SECTOR ZERO REQUIRED
9566                               ;*NOW THE 304 WORDS WILL START
9567
9568
9569
9570                               ;*FOR FIRST BYTE SECTOR CLOCK WILL GO HIGH THEN CLOCK WILL GO HIGH
9571                               ;*BOTH WILL COME DOWN TOGETHER THEN SEVEN CLOCKS WILL BE GIVEN
9572                               ;*FOR SECOND BYTE AND ALL OTHER BYTES TILL NEXT SECTOR SECTOR CLOCK
9573                               ;*WILL BE IDENTICAL WITH ONE CLOCK
9574
9575
9576                               ;*ONE WORD ONLY
9577
9578 053642 012702 000010      1$:   MOV    #8.,   R2          ;BYTE COUNTER
9579 053646 012705 000002      MOV    #2.,   R5          ;BYTES PER WORD
9580 053652 052710 000010      BIS    #MSTCK,@R0        ;SET SECTOR CLOCK
9581 053656 052710 000002      BIS    #MCLK,@R0         ;SET CLOCK
9582 053662 000402              BR     3$                ;BRANCH TO CLEAR SECTOR AND CLOCK
9583 053664 052710 000012      2$:   BIS    #MSTCK!MCLK,@R0 ;SET SECTOR AND CLOCK
9584 053670 042710 000012      3$:   BIC    #MSTCK!MCLK,@R0 ;CLEAR SECTOR AND CLOCK
9585 053674 052710 000002      8$:   BIS    #MCLK, @R0      ;SET CLOCK
9586 053700 042710 000002      BIC    #MCLK, @R0        ;CLEAR CLOCK
9587 053704 005302              DEC    R2                ;BYTE COUNTER
9588 053706 001372              BNE    8$                ;BRANCH IF BYTE NOT COMPLETE
9589 053710 012702 000007      MOV    #7.,   R2          ;SETUP FOR SECOND BYTE
9590 053714 005305              DEC    R5                ;IS WORD COMPLETE?
9591 053716 001362              BNE    2$                ;BRANCH IF NOT COMPLETE
9592                               ;TO GIVE SECTOR CLOCK AND CLOCK
9593
9594
9595                               ;*NOW 303 WORDS ARE LEFT AND ALL ARE IDENTICAL
9596
9597 053720 012701 000457      MOV    #303., R1         ;WORDS PER SECTOR COUNTER
9598 053724 012705 000002      4$:   MOV    #2.,   R5          ;BYTES PER WORD COUNTER
9599 053730 012702 000007      5$:   MOV    #7.,   R2          ;BYTE COUNTER (CLOCK COUNTER)
9600 053734 052710 000012      BIS    #MSTCK!MCLK,@R0   ;SET SECTOR CLOCK AND CLOCK
9601 053740 042710 000012      BIC    #MSTCK!MCLK,@R0   ;CLEAR SECTOR CLOCK AND CLOCK
9602 053744 052710 000002      6$:   BIS    #MCLK, @R0      ;SET CLOCK
9603 053750 042710 000002      BIC    #MCLK, @R0        ;RESET CLOCK
9604 053754 005302              DEC    R2                ;IS BYTE COMPLETE?
9605 053756 001372              BNE    6$                ;BRANCH IF NOT COMPLETE
9606 053760 005305              DEC    R5                ;IS WORD COMPLETE?
9607 053762 001362              BNE    5$                ;BRANCH IF NOT
9608 053764 005301              DEC    R1                ;IS SECTOR COMPLETE
9609 053766 001356              BNE    4$                ;BRANCH IF NOT
9610 053770 052710 000010      BIS    #MSTCK,@R0        ;SET SECTOR
9611 053774 042710 000010      BIC    #MSTCK,@R0        ;CLEAR SECTOR
9612 054000 005303              DEC    R3                ;IS REQUIRED NO OF SECTORS COMPLETE
9613 054002 001317              BNE    1$                ;BRANCH IF NOT
9614
9615                               7$:
9616 054004 012605              MOV    (SP)+,R5          ;;POP STACK INTO R5
9617 054006 012604              MOV    (SP)+,R4          ;;POP STACK INTO R4
9618 054010 012603              MOV    (SP)+,R3          ;;POP STACK INTO R3
9619 054012 012602              MOV    (SP)+,R2          ;;POP STACK INTO R2

```



```

9620 054014 012601          MOV      (SP)+,R1          ;;POP STACK INTO R1
9621 054016 012600          MOV      (SP)+,R0          ;;POP STACK INTO R0
9622 054020 000201          RTS        R1
9623
9624
9625          ;*READ ONE SECTOR OF DATA
9626
9627 054022 000000          RNO:      0                ;NO. OF WORDS READ
9628 054024 000000          RCOM:     0                ;EXTRA STORAGE
9629
9630
9631
9632 054026 012137 054022          REDATA: MOV      (R1)+,@#RNO          ;SAVE NO. OF WORDS ONLY FOR INFORMATION
9633 054032 012137 054024          MOV      (R1)+,@#RCOM          ;EXTRA WORD ONLY FOR INFORMATION
9634 054036 010146          MOV      R1,-(SP)           ;;PUSH R1 ON STACK
9635 054040 005737 002024          TST      @#TSECC            ;IS THIS AN ECC TEST
9636 054044 001403          BEQ      1$                ;BRANCH IF NO
9637 054046 012737 177777 045320          MOV      #-1,@#TSECCG        ;THESE BITS ARE TO GENERATE ECC
9638 054054 012702 000402          1$:      MOV      #258.,R2          ;256 WORDS PER SECTOR
9639          ;PLUS 2 ECC WORDS
9640 054060 012703 051422          MOV      #DISK,R3           ;POINTE TO DISK SIMULATION
9641 054064 012337 050414          2$:      MOV      (R3)+,@#WORD          ;READY TO READ CONTENTS
9642 054070 004737 050420          JSR      PC,@#READ          ;READ
9643 054074 005302          DEC      R2                ;IS 256 WORDS DONE?
9644 054076 001372          BNE      2$                ;IF NOT BRANCH
9645 054100 005737 002024          TST      @#TSECC            ;IS THIS AN ECC TEST
9646 054104 001012          BNE      4$                ;BRANCH OUT IF YES
9647 054106 005037 045320          CLR      @#TSECCG          ;NO MORE ECC BITS ARE TO BE GENERATED
9648 054112 012702 000017          MOV      #15.,R2           ;ONE DATA GAP, 14 TOLERANCE GAP
9649 054116 012337 050414          3$:      MOV      (R3)+,@#WORD          ;READY TO READ CONTENTS OF WORD
9650 054122 004737 050420          JSR      PC,@#READ          ;READ
9651 054126 005302          DEC      R2                ;COUNT
9652 054130 001372          BNE      3$                ;BRANCH IF 14 NOT DONE
9653 054132          4$:
9654 054132 012601          MOV      (SP)+,R1          ;;POP STACK INTO R1
9655 054134 000201          RTS        R1                ;RETURN
9656
9657

```

```
9658  
9659  
9660 054136 RPVECT:  
9661 054136 104401 054144 TYPE ,65$ ;;TYPE ASCIZ STRING  
9662 054142 000421 BR 64$ ;;GET OVER THE ASCIZ  
9663 054206 104402 TYPOC ;TYPE FROM PC  
9664 054210 012777 054136 125410 MOV #RPVECT,@RPVEC ;RESTORE TRAP RP04 VECTOR  
9665 054216 000000 HALT ;CHANGE TO CONTINUE
```

```
.SBTTL SYSMAC LIBRARY ROUTINES
9666
9667
9668 054220 104407          CKSWR          ;;TEST FOR CHANGE IN SOFT-SWR
9669 054222 005037 047620  CLR          @#NOSYNC  ;;CLEAR FLAG FOR HEADER ERROR COMMANDS
9670 054226 005037 002024  CLR          @#TSECC  ;;CLEAR FLAG FOR ECC TEST
9671 054232 005037 045320  CLR          @#TSECCG  ;;EVEN IN AN ECC TEST EVERY CLOCK
9672 054236 005037 002026  CLR          @#TESDTE  ;;DRIVE TIMING ERROR TEST
9673 054242 032777 040000 124670 1$: BIT          #BIT14,@SWR  ;;LOOP ON PRESENT TEST?
9674 054250 001111          BNE          $OVER    ;;YES IF SW14=1
9675 054252 000416          $XTSTR: BR         6$    ;;IF RUNNING ON THE 'XOR' TESTER CHANGE
9676 054254 013746 000004          MOV          @#ERRVEC,-(SP) ;;SAVE THE CONTENTS OF THE ERROR VECTOR
9677 054260 012737 054300 000004          MOV          #5$,@#ERRVEC  ;;SET FOR TIMEOUT
9678 054266 005737 177060          TST          @#177060    ;;TIME OUT ON XOR?
9679 054272 012637 000004          MOV          (SP)+,@#ERRVEC ;;RESTORE THE ERROR VECTOR
9680 054276 000463          BR          $$VLAD     ;;GO TO THE NEXT TEST
9681 054300 022626          5$: CMP          (SP)+,(SP)+ ;;CLEAR THE STACK AFTER A TIME OUT
9682 054302 012637 000004          MOV          (SP)+,@#ERRVEC ;;RESTORE THE ERROR VECTOR
9683 054306 000423          BR          7$        ;;LOOP ON THE PRESENT TEST
9684 054310 032777 000400 124622          BIT          #BIT08,@SWR  ;;LOOP ON SPEC. TEST?
9685 054316 001404          BEQ          2$        ;;BR IF NO
9686 054320 127737 124614 001102          CMPB         @SWR,$STNM   ;;ON THE RIGHT TEST? SWR<7:0>
9687 054326 001462          BEQ          $OVER    ;;BR IF YES
9688 054330 105737 001103          2$: TSTB         $ERFLG   ;;HAS AN ERROR OCCURRED?
9689 054334 001421          BEQ          3$        ;;BR IF NO
9690 054336 123737 001115 001103          CMPB         $ERMAX,$ERFLG ;;MAX. ERRORS FOR THIS TEST OCCURRED?
9691 054344 101015          BHI          3$        ;;BR IF NO
9692 054346 032777 001000 124564          BIT          #BIT09,@SWR  ;;LOOP ON ERROR?
9693 054354 001404          BEQ          4$        ;;BR IF NO
9694 054356 013737 001110 001106 7$: MOV          $LPERR,$LPADR ;;SET LOOP ADDRESS TO LAST SCOPE
9695 054364 000443          BR          $OVER    ;;
9696 054366 105037 001103          4$: CLRB         $ERFLG   ;;ZERO THE ERROR FLAG
9697 054372 005037 001212          CLR          $TIMES    ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
9698 054376 000415          BR          1$        ;;ESCAPE TO THE NEXT TEST
9699 054400 032777 004000 124532 3$: BIT          #BIT11,@SWR  ;;INHIBIT ITERATIONS?
9700 054406 001011          BNE          1$        ;;BR IF YES
9701 054410 005737 001100          TST          $PASS    ;;IF FIRST PASS OF PROGRAM
9702 054414 001406          BEQ          1$        ;;INHIBIT ITERATIONS
9703 054416 005237 001104          INC          $ICNT    ;;INCREMENT ITERATION COUNT
9704 054422 023737 001212 001104          CMP          $TIMES,$ICNT ;;CHECK THE NUMBER OF ITERATIONS MADE
9705 054430 002021          BGE          $OVER    ;;BR IF MORE ITERATION REQUIRED
9706 054432 012737 000001 001104 1$: MOV          #1,$ICNT  ;;REINITIALIZE THE ITERATION COUNTER
9707 054440 013737 054510 001212          MOV          $MXCNT,$TIMES ;;SET NUMBER OF ITERATIONS TO DO
9708 054446 105237 001102          $$VLAD: INCB         $STNM  ;;COUNT TEST NUMBERS
9709 054452 011637 001106          MOV          (SP),$LPADR ;;SAVE SCOPE LOOP ADDRESS
9710 054456 011637 001110          MOV          (SP),$LPERR ;;SAVE ERROR LOOP ADDRESS
9711 054462 005037 001214          CLR          $ESCAPE  ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
9712 054466 112737 000001 001115          MOVB         #1,$ERMAX  ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
9713 054474 013777 001102 124440 $OVER: MOV          $STNM,@DISPLAY ;;DISPLAY TEST NUMBER
9714 054502 013716 001106          MOV          $LPADR,(SP) ;;FUDGE RETURN ADDRESS
9715 054506 000002          RTI           ;;FIXES PS
9716 054510 000004          $MXCNT: 4        ;;MAX. NUMBER OF ITERATIONS
```

9717	054512	010046				MOV	R0,-(SP)	::PUSH R0 ON STACK
9718	054514	010146				MOV	R1,-(SP)	::PUSH R1 ON STACK
9719	054516	010246				MOV	R2,-(SP)	::PUSH R2 ON STACK
9720	054520	010346				MOV	R3,-(SP)	::PUSH R3 ON STACK
9721	054522	010546				MOV	R5,-(SP)	::PUSH R5 ON STACK
9722	054524	012746	020200			MOV	#20200,-(SP)	::SET BLANK SWITCH AND SIGN
9723	054530	016605	000020			MOV	20(SP),R5	::GET THE INPUT NUMBER
9724	054534	100004				BPL	1\$::BR IF INPUT IS POS.
9725	054536	005405				NEG	R5	::MAKE THE BINARY NUMBER POS.
9726	054540	112766	000055	000001		MOVB	#'-,1(SP)	::MAKE THE ASCII NUMBER NEG.
9727	054546	005000			1\$:	CLR	R0	::ZERO THE CONSTANTS INDEX
9728	054550	012703	054726			MOV	#\$DBLK,R3	::SETUP THE OUTPUT POINTER
9729	054554	112723	000040			MOVB	#' ,(R3)+	::SET THE FIRST CHARACTER TO A BLANK
9730	054560	005002			2\$:	CLR	R2	::CLEAR THE BCD NUMBER
9731	054562	016001	054716			MOV	\$DTBL(R0),R1	::GET THE CONSTANT
9732	054566	160105			3\$:	SUB	R1,R5	::FORM THIS BCD DIGIT
9733	054570	002402				BLT	4\$::BR IF DONE
9734	054572	005202				INC	R2	::INCREASE THE BCD DIGIT BY 1
9735	054574	000774				BR	3\$	
9736	054576	060105			4\$:	ADD	R1,R5	::ADD BACK THE CONSTANT
9737	054600	005702				TST	R2	::CHECK IF BCD DIGIT=0
9738	054602	001002				BNE	5\$::FALL THROUGH IF 0
9739	054604	105716				TSTB	(SP)	::STILL DOING LEADING 0'S?
9740	054606	100407				BMI	7\$::BR IF YES
9741	054610	106316			5\$:	ASLB	(SP)	::MSD?
9742	054612	103003				BCC	6\$::BR IF NO
9743	054614	116663	000001	177777		MOVB	1(SP),-1(R3)	::YES--SET THE SIGN
9744	054622	052702	000060		6\$:	BIS	#'0,R2	::MAKE THE BCD DIGIT ASCII
9745	054626	052702	000040		7\$:	BIS	#' ,R2	::MAKE IT A SPACE IF NOT ALREADY A DIGIT
9746	054632	110223				MOVB	R2,(R3)+	::PUT THIS CHARACTER IN THE OUTPUT BUFFER
9747	054634	005720				TST	(R0)+	::JUST INCREMENTING
9748	054636	020027	000010			CMP	R0,#10	::CHECK THE TABLE INDEX
9749	054642	002746				BLT	2\$::GO DO THE NEXT DIGIT
9750	054644	003002				BGT	8\$::GO TO EXIT
9751	054646	010502				MOV	R5,R2	::GET THE LSD
9752	054650	000764				BR	6\$::GO CHANGE TO ASCII
9753	054652	105726			8\$:	TSTB	(SP)+	::WAS THE LSD THE FIRST NON-ZERO?
9754	054654	100003				BPL	9\$::BR IF NO
9755	054656	116663	177777	177776		MOVB	-1(SP),-2(R3)	::YES--SET THE SIGN FOR TYPING
9756	054664	105013			9\$:	CLRB	(R3)	::SET THE TERMINATOR
9757	054666	012605				MOV	(SP)+,R5	::POP STACK INTO R5
9758	054670	012603				MOV	(SP)+,R3	::POP STACK INTO R3
9759	054672	012602				MOV	(SP)+,R2	::POP STACK INTO R2
9760	054674	012601				MOV	(SP)+,R1	::POP STACK INTO R1
9761	054676	012600				MOV	(SP)+,R0	::POP STACK INTO R0
9762	054700	104401	054726			TYPE	\$DBLK	::NOW TYPE THE NUMBER
9763	054704	016666	000002	000004		MOV	2(SP),4(SP)	::ADJUST THE STACK
9764	054712	012616				MOV	(SP)+,(SP)	
9765	054714	000002				RTI		::RETURN TO USER
9766	054716	023420			\$DTBL:	10000.		
9767	054720	001750				1000.		
9768	054722	000144				100.		
9769	054724	000012				10.		

9770	054736	105737	001157		\$TYPE:	TSTB	\$TPFLG	:: IS THERE A TERMINAL?
9771	054742	100002				BPL	1\$:: BR IF YES
9772	054744	000000				HALT		:: HALT HERE IF NO TERMINAL
9773	054746	000407				BR	3\$:: LEAVE
9774	054750	010046			1\$:	MOV	R0,-(SP)	:: SAVE R0
9775	054752	017600	000002			MOV	@2(SP),R0	:: GET ADDRESS OF ASCIZ STRING
9776	054756	112046			2\$:	MOVB	(R0)+,-(SP)	:: PUSH CHARACTER TO BE TYPED ONTO STACK
9777	054760	001005				BNE	4\$:: BR IF IT ISN'T THE TERMINATOR
9778	054762	005726				TST	(SP)+	:: IF TERMINATOR POP IT OFF THE STACK
9779	054764	012600			60\$:	MOV	(SP)+,R0	:: RESTORE R0
9780	054766	062716	000002		3\$:	ADD	#2,(SP)	:: ADJUST RETURN PC
9781	054772	000002				RTI		:: RETURN
9782	054774	122716	000011		4\$:	CMPB	#HT,(SP)	:: BRANCH IF <HT>
9783	055000	001430				BEQ	8\$	
9784	055002	122716	000200			CMPB	#CRLF,(SP)	:: BRANCH IF NOT <CRLF>
9785	055006	001006				BNE	5\$	
9786	055010	005726				TST	(SP)+	:: POP <CR><LF> EQUIV
9787	055012	104401				TYPE		:: TYPE A CR AND LF
9788	055014	001223				\$CRLF		
9789	055016	105037	055152			CLRB	\$CHARCNT	:: CLEAR CHARACTER COUNT
9790	055022	000755				BR	2\$:: GET NEXT CHARACTER
9791	055024	004737	055106		5\$:	JSR	PC,\$TYPEC	:: GO TYPE THIS CHARACTER
9792	055030	123726	001156		6\$:	CMPB	\$FILLC,(SP)+	:: IS IT TIME FOR FILLER CHARS.?
9793	055034	001350				BNE	2\$:: IF NO GO GET NEXT CHAR.
9794	055036	013746	001154			MOV	\$NULL,-(SP)	:: GET # OF FILLER CHARS. NEEDED
9795	055042	105366	000001		7\$:	DECB	1(SP)	:: DOES A NULL NEED TO BE TYPED?
9796	055046	002770				BLT	6\$:: BR IF NO--GO POP THE NULL OFF OF STACK
9797	055050	004737	055106			JSR	PC,\$TYPEC	:: GO TYPE A NULL
9798	055054	105337	055152			DECB	\$CHARCNT	:: DO NOT COUNT AS A COUNT
9799	055060	000770				BR	7\$:: LOOP
9800	055062	112716	000040		8\$:	MOVB	#' ,(SP)	:: REPLACE TAB WITH SPACE
9801	055066	004737	055106		9\$:	JSR	PC,\$TYPEC	:: TYPE A SPACE
9802	055072	132737	000007	055152		BITB	#7,\$CHARCNT	:: BRANCH IF NOT AT
9803	055100	001372				BNE	9\$:: TAB STOP
9804	055102	005726				TST	(SP)+	:: POP SPACE OFF STACK
9805	055104	000724				BR	2\$:: GET NEXT CHARACTER
9806	055106	105777	124036		\$TYPEC:	TSTB	@\$TPS	:: WAIT UNTIL PRINTER IS READY
9807	055112	100375				BPL	\$TYPEC	
9808	055114	116677	000002	124030		MOVB	2(SP),@\$TPB	:: LOAD CHAR TO BE TYPED INTO DATA REG.
9809	055122	122766	000015	000002		CMPB	#CR,2(SP)	:: IS CHARACTER A CARRIAGE RETURN?
9810	055130	001003				BNE	1\$:: BRANCH IF NO
9811	055132	105037	055152			CLRB	\$CHARCNT	:: YES--CLEAR CHARACTER COUNT
9812	055136	000406				BR	\$TYPEX	:: EXIT
9813	055140	122766	000012	000002	1\$:	CMPB	#LF,2(SP)	:: IS CHARACTER A LINE FEED?
9814	055146	001402				BEQ	\$TYPEX	:: BRANCH IF YES
9815	055150	105227				INCB	(PC)+	:: COUNT THE CHARACTER
9816	055152	000000			\$CHARCNT:	.WORD	0	:: CHARACTER COUNT STORAGE
9817	055154	000207			\$TYPEX:	RTS	PC	

```
9818 055156 000000 $TKCNT: .WORD 0 ;;NUMBER OF ITEMS IN QUEUE
9819 055160 000000 $TKQIN: .WORD 0 ;;INPUT POINTER
9820 055162 000000 $TKQOUT: .WORD 0 ;;OUTPUT POINTER
9821 055176 005037 055156 $TKINT: CLR $TKCNT ;;CLEAR COUNT OF ITEMS IN QUEUE
9822 055202 012737 055164 055160 MOV # $TKQSR, $TKQIN ;;MOVE THE STARTING ADDRESS OF THE
9823 055210 013737 055160 055162 MOV $TKQIN, $TKQOUT ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
9824 055216 012737 055246 000060 MOV # $TKSRV, @ $TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
9825 055224 012737 000200 000062 MOV #200, @ $TKVEC+2 ;;'BR' LEVEL 4
9826 055232 005777 123710 TST @ $TKB ;;CLEAR DONE FLAG
9827 055236 012777 000100 123700 MOV #100, @ $TKS ;;ENABLE TTY KEYBOARD INTERRUPT
9828 055244 000207 RTS PC ;;RETURN TO CALLER
9829 055246 117746 123674 $TKSRV: MOVB @ $TKB, -(SP) ;;PICKUP THE CHARACTER
9830 055252 042716 177600 BIC #^C177, (SP) ;;STRIP THE JUNK
9831 055256 021627 000003 CMP (SP), #3 ;;IS IT A CONTROL C?
9832 055262 001007 BNE 1$ ;;BRANCH IF NO
9833 055264 104401 056235 TYPE , $CNTLC ;;TYPE A CONTROL-C (^C)
9834 055270 004737 055176 JSR PC, $TKINT ;;INIT THE KEYBOARD
9835 055274 005726 TST (SP)+ ;;CLEAN UP STACK
9836 055276 000137 041444 JMP OPERSEL ;;CONTROL C RESTART
9837 055302 021627 000007 1$: CMP (SP), #7 ;;IS IT A CONTROL G?
9838 055306 001004 BNE 2$ ;;BRANCH IF NO
9839 055310 022737 000176 001140 CMP #SWREG, SWR ;;IS SOFT-SWR SELECTED?
9840 055316 001500 BEG 6$ ;;GO TO SWR CHANGE
9841 055320 022737 000011 055156 CMP #9, $TKCNT ;;IS THE QUEUE FULL?
9842 055326 001004 BNE 3$ ;;BRANCH IF NO
9843 055330 104401 001216 TYPE , $BELL ;;RING THE TTY BELL
9844 055334 005726 TST (SP)+ ;;CLEAN CHARACTER OFF OF STACK
9845 055336 000451 BR 5$ ;;EXIT
9846 055340 021627 000023 3$: CMP (SP), #23 ;;IS IT A CONTROL-S?
9847 055344 001021 BNE 32$ ;;BRANCH IF NO
9848 055346 005077 123572 CLR @ $TKS ;;DISABLE TTY KEYBOARD INTERRUPTS
9849 055352 005726 TST (SP)+ ;;CLEAN CHAR OFF STACK
9850 055354 105777 123564 31$: TSTB @ $TKS ;;WAIT FOR A CHAR
9851 055360 100375 BPL 31$ ;;LOOP UNTIL ITS THERE
9852 055362 117746 123560 MOVB @ $TKB, -(SP) ;;GET THE CHARACTER
9853 055366 042716 177600 BIC #^C177, (SP) ;;MAKE IT 7-BIT ASCII
9854 055372 022627 000021 CMP (SP)+, #21 ;;IS IT A CONTROL-Q?
9855 055376 001366 BNE 31$ ;;BRANCH IF NO
9856 055400 012777 000100 123536 MOV #100, @ $TKS ;;REENABLE TTY KEYBOARD INTERRUPTS
9857 055406 000002 RTI ;;RETURN
9858 055410 005237 055156 32$: INC $TKCNT ;;COUNT THIS CHARACTER
9859 055414 021627 000140 CMP (SP), #140 ;;IS IT UPPER CASE?
9860 055420 002405 BLT 4$ ;;BRANCH IF YES
9861 055422 021627 000175 CMP (SP), #175 ;;IS IT A SPECIAL CHAR?
9862 055426 003002 BGT 4$ ;;BRANCH IF YES
9863 055430 042716 000040 BIC #40, (SP) ;;MAKE IT UPPER CASE
9864 055434 112677 177520 4$: MOVB (SP)+, @ $TKQIN ;;AND PUT IT IN QUEUE
9865 055440 005237 055160 INC $TKQIN ;;UPDATE THE POINTER
9866 055444 023727 055160 055175 CMP $TKQIN, # $TKQEND ;;GO OFF THE END?
9867 055452 001003 BNE 5$ ;;BRANCH IF NO
9868 055454 012737 055164 055160 MOV # $TKQSR, $TKQIN ;;RESET THE POINTER
9869 055462 000002 5$: RTI ;;RETURN
9870 055464 022737 000176 001140 $CKSWR: CMP #SWREG, SWR ;;IS THE SOFT-SWR SELECTED
9871 055472 001124 BNE 15$ ;;EXIT IF NOT
9872 055474 105777 123444 TSTB @ $TKS ;;IS A CHAR WAITING?
9873 055500 100121 BPL 15$ ;;IF NOT, EXIT
```

9874	055502	117746	123440			MOVB	@\$TKB,-(SP)	::YES
9875	055506	042716	177600			BIC	#^C177,(SP)	::MAKE IT 7-BIT ASCII
9876	055512	021627	000007			CMP	(SP),#7	::IS IT A CONTROL-G?
9877	055516	001300				BNE	2\$::IF NOT, PUT IT IN THE TTY QUEUE
9878	055520	123727	001134	000001	6\$:	CMPB	\$AUTOB,#1	::ARE WE RUNNING IN AUTO-MODE?
9879	055526	001674				BEQ	2\$::BRANCH IF YES
9880	055530	005726				TST	(SP)+	::CLEAR CONTROL-G OFF STACK
9881	055532	004737	055176			JSR	PC,\$TKINT	::FLUSH THE TTY INPUT QUEUE
9882	055536	005077	123402			CLR	@\$TKS	::DISABLE TTY KEYBOARD INTERRUPTS
9883	055542	112737	000001	001135		MOVB	#1,\$INTAG	::SET INTERRUPT MODE INDICATOR
9884	055550	104401	056247			TYPE	,\$CNTLG	::ECHO THE CONTROL-G (^G)
9885	055554	104401	056254		\$GTSWR:	TYPE	,\$MSWR	::TYPE CURRENT CONTENTS
9886	055560	013746	000176			MOV	SWREG,-(SP)	::SAVE SWREG FOR TYPEOUT
9887	055564	104402				TYPOC		::GO TYPE--OCTAL ASCII(ALL DIGITS)
9888	055566	104401	056265			TYPE	,\$MNEW	::PROMPT FOR NEW SWR
9889	055572	005046			19\$:	CLR	-(SP)	::CLEAR COUNTER
9890	055574	005046				CLR	-(SP)	::THE NEW SWR
9891	055576	105777	123342		7\$:	TSTB	@\$TKS	::CHAR THERE?
9892	055602	100375				BPL	7\$::IF NOT TRY AGAIN
9893	055604	117746	123336			MOVB	@\$TKB,-(SP)	::PICK UP CHAR
9894	055610	042716	177600			BIC	#^C177,(SP)	::MAKE IT 7-BIT ASCII
9895	055614	021627	000003			CMP	(SP),#3	::IS IT A CONTROL-C?
9896	055620	001015				BNE	9\$::BRANCH IF NOT
9897	055622	104401	056235			TYPE	,\$CNTLC	::YES, ECHO CONTROL-C (^C)
9898	055626	062706	000006			ADD	#6,SP	::CLEAN UP STACK
9899	055632	123727	001135	000001		CMPB	\$INTAG,#1	::REENABLE TTY KEYBOARD INTERRUPTS?
9900	055640	001003				BNE	8\$::BRANCH IF NO
9901	055642	012777	000100	123274		MOV	#100,@\$TKS	::ALLOW TTY KEYBOARD INTERRUPTS
9902	055650	000137	041444		8\$:	JMP	OPERSEL	::CONTROL-C RESTART
9903	055654	021627	000025		9\$:	CMP	(SP),#25	::IS IT A CONTROL-U?
9904	055660	001005				BNE	10\$::BRANCH IF NOT
9905	055662	104401	056242			TYPE	,\$CNTLU	::YES, ECHO CONTROL-U (^U)
9906	055666	062706	000006		20\$:	ADD	#6,SP	::IGNORE PREVIOUS INPUT
9907	055672	000737				BR	19\$::LET'S TRY IT AGAIN
9908	055674	021627	000015		10\$:	CMP	(SP),#15	::IS IT A <CR>?
9909	055700	001022				BNE	16\$::BRANCH IF NO
9910	055702	005766	000004			TST	4(SP)	::YES, IS IT THE FIRST CHAR?
9911	055706	001403				BEQ	11\$::BRANCH IF YES
9912	055710	016677	000002	123222		MOV	2(SP),@SWR	::SAVE NEW SWR
9913	055716	062706	000006		11\$:	ADD	#6,SP	::CLEAR UP STACK
9914	055722	104401	001223		14\$:	TYPE	,\$CRLF	::ECHO <CR> AND <LF>
9915	055726	123727	001135	000001		CMPB	\$INTAG,#1	::RE-ENABLE TTY KBD INTERRUPTS?
9916	055734	001003				BNE	15\$::BRANCH IF NOT
9917	055736	012777	000100	123200		MOV	#100,@\$TKS	::RE-ENABLE TTY KBD INTERRUPTS
9918	055744	000002			15\$:	RTI		::RETURN
9919	055746	004737	055106		16\$:	JSR	PC,\$TYPEC	::ECHO CHAR
9920	055752	021627	000060			CMP	(SP),#60	::CHAR < 0?
9921	055756	002420				BLT	18\$::BRANCH IF YES
9922	055760	021627	000067			CMP	(SP),#67	::CHAR > 7?
9923	055764	003015				BGT	18\$::BRANCH IF YES
9924	055766	042726	000060			BIC	#60,(SP)+	::STRIP-OFF ASCII
9925	055772	005766	000002			TST	2(SP)	::IS THIS THE FIRST CHAR
9926	055776	001403				BEQ	17\$::BRANCH IF YES
9927	056000	006316				ASL	(SP)	::NO, SHIFT PRESENT
9928	056002	006316				ASL	(SP)	:: CHAR OVER TO MAKE
9929	056004	006316				ASL	(SP)	:: ROOM FOR NEW ONE.

```

9930 056006 005266 000002      17$: INC 2(SP)           ;;KEEP COUNT OF CHAR
9931 056012 056616 177776      BIS -2(SP),(SP)        ;;SET IN NEW CHAR
9932 056016 000667           BR 7$                 ;;GET THE NEXT ONE
9933 056020 104401 001222      18$: TYPE , $QUES      ;;TYPE ?<CR><LF>
9934 056024 000720           BR 20$                ;;SIMULATE CONTROL-U
9935 056026 011646           $RDCHR: MOV (SP),-(SP)  ;;PUSH DOWN THE PC AND
9936 056030 016666 000004 000002 MOV 4(SP),2(SP)        ;;THE PS
9937 056036 005066 000004      CLR 4(SP)             ;;GET READY FOR A CHARACTER
9938 056042 005046           CLR -(SP)             ;;PUT NEW PS ON STACK
9939 056044 012746 056052      MOV #64$,-(SP)       ;;PUT NEW PC ON STACK
9940 056050 000002           RTI                  ;;POP NEW PC AND PS
9941 056052 005737 055156      1$: TST $TKCNT        ;;WAIT ON A CHARACTER
9942 056056 001775           BEQ 1$                ;;
9943 056060 005337 055156      DEC $TKCNT            ;;DECREMENT THE COUNTER
9944 056064 117766 177072 000004 MOVB @ $TKQOUT,4(SP)   ;;GET ONE CHARACTER
9945 056072 005237 055162      INC $TKQOUT           ;;UPDATE THE POINTER
9946 056076 023727 055162 055175 CMP $TKQOUT,$TKQEND   ;;DID IT GO OFF OF THE END?
9947 056104 001003           BNE 2$                ;;BRANCH IF NO
9948 056106 012737 055164 055162 MOV # $TKQSRRT,$TKQOUT ;;RESET THE POINTER
9949 056114 000002           RTI                  ;;RETURN
9950 056116 010346           $RDLIN: MOV R3,-(SP)   ;;SAVE R3
9951 056120 012703 056224      1$: MOV # $TTYIN,R3    ;;GET ADDRESS
9952 056124 022703 056235      2$: CMP # $TTYIN+9.,R3 ;;BUFFER FULL?
9953 056130 101405           BLOS 4$               ;;BR IF YES
9954 056132 104410           RDCHR                ;;GO READ ONE CHARACTER FROM THE TTY
9955 056134 112613           MOVB (SP)+,(R3)      ;;GET CHARACTER
9956 056136 122713 000177      10$: CMPB #177,(R3)    ;;IS IT A RUBOUT
9957 056142 001003           BNE 3$                ;;SKIP IF NOT
9958 056144 104401 001222      4$: TYPE , $QUES      ;;TYPE A '?'
9959 056150 000763           BR 1$                 ;;CLEAR THE BUFFER AND LOOP
9960 056152 111337 056222      3$: MOVB (R3),9$       ;;ECHO THE CHARACTER
9961 056156 104401 056222      TYPE ,9$              ;;
9962 056162 122723 000015      CMPB #15,(R3)+       ;;CHECK FOR RETURN
9963 056166 001356           BNE 2$                ;;LOOP IF NOT RETURN
9964 056170 105063 177777      CLRB -1(R3)           ;;CLEAR RETURN (THE 15)
9965 056174 104401 001224      TYPE , $LF            ;;TYPE A LINE FEED
9966 056200 012603           MOV (SP)+,R3         ;;RESTORE R3
9967 056202 011646           MOV (SP),-(SP)       ;;ADJUST THE STACK AND PUT ADDRESS OF THE
9968 056204 016666 000004 000002 MOV 4(SP),2(SP)        ;; FIRST ASCII CHARACTER ON IT
9969 056212 012766 056224 000004 MOV # $TTYIN,4(SP)    ;;
9970 056220 000002           RTI                  ;;RETURN
9971 056222 000           9$: .BYTE 0          ;;STORAGE FOR ASCII CHAR. TO TYPE
9972 056223 000           .BYTE 0            ;;TERMINATOR
9973 056235 136 006503 000012 $CNTLC: .ASCIZ / ^C / <15><12> ;;CONTROL 'C'
9974 056242 052536 005015 000 $CNTLU: .ASCIZ / ^U / <15><12> ;;CONTROL 'U'
9975 056247 136 006507 000012 $CNTLG: .ASCIZ / ^G / <15><12> ;;CONTROL 'G'
9976 056254 005015 053523 020122 $MSWR: .ASCIZ <15><12> / SWR = /
9977 056262 020075 000
9978 056265 040 047040 053505 $MNEW: .ASCIZ / NEW = /
9979 056272 036440 000040
9980

```

;FROM THE TTY


```

9981 056276 011646 $RDOCT: MOV (SP),-(SP) ;;PROVIDE SPACE FOR THE
9982 056300 016666 000004 000002 MOV 4(SP),2(SP) ;;INPUT NUMBER
9983 056306 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK
9984 056310 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
9985 056312 010246 MOV R2,-(SP) ;;PUSH R2 ON STACK
9986 056314 104411 1$: RDLIN ;;READ AN ASCIZ LINE
9987 056316 012600 MOV (SP)+,R0 ;;GET ADDRESS OF 1ST CHARACTER
9988 056320 010037 056424 MOV R0,5$ ;;AND SAVE IT
9989 056324 005001 CLR R1 ;;CLEAR DATA WORD
9990 056326 005002 CLR R2
9991 056330 112046 2$: MOV (R0)+,-(SP) ;;PICKUP THIS CHARACTER
9992 056332 001420 BEQ 3$ ;;IF ZERO GET OUT
9993 056334 122716 000060 CMPB #'0,(SP) ;;MAKE SURE THIS CHARACTER
9994 056340 003026 BGT 4$ ;;IS AN OCTAL DIGIT
9995 056342 122716 000067 CMPB #'7,(SP)
9996 056346 002423 BLT 4$
9997 056350 006301 ASL R1 ;;*2
9998 056352 006102 ROL R2
9999 056354 006301 ASL R1 ;;*4
10000 056356 006102 ROL R2
10001 056360 006301 ASL R1 ;;*8
10002 056362 006102 ROL R2
10003 056364 042716 177770 BIC #'C7,(SP) ;;STRIP THE ASCII JUNK
10004 056370 062601 ADD (SP)+,R1 ;;ADD IN THIS DIGIT
10005 056372 000756 BR 2$ ;;LOOP
10006 056374 005726 3$: TST (SP)+ ;;CLEAN TERMINATOR FROM STACK
10007 056376 010166 000012 MOV R1,12(SP) ;;SAVE THE RESULT
10008 056402 010237 056434 MOV R2,$HIOCT
10009 056406 012602 MOV (SP)+,R2 ;;POP STACK INTO R2
10010 056410 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
10011 056412 012600 MOV (SP)+,R0 ;;POP STACK INTO R0
10012 056414 000002 RTI ;;RETURN
10013 056416 005726 4$: TST (SP)+ ;;CLEAN PARTIAL FROM STACK
10014 056420 105010 CLRB (R0) ;;SET A TERMINATOR
10015 056422 104401 TYPE ;;TYPE UP THRU THE BAD CHAR.
10016 056424 000000 5$: .WORD 0
10017 056426 104401 001222 TYPE $QUES ;; '?' 'CR' & 'LF'
10018 056432 000730 BR 1$ ;;TRY AGAIN
10019 056434 000000 $HIOCT: .WORD 0 ;;HIGH ORDER BITS GO HERE
  
```

```
10020 056436 104407          CKSWR          ::TEST FOR CHANGE IN SOFT-SWR
10021 056440 012737 177777 002006  MOV    #-1,@#CRFLG$  ::SET ERROR FLAG
10022 056446 105237 001103 7$:    INCB   $ERFLG        ::SET THE ERROR FLAG
10023 056452 001775          BEQ     7$          ::DON'T LET THE FLAG GO TO ZERO
10024 056454 013777 001102 122460  MOV    $TSTNM,@DISPLAY ::DISPLAY TEST NUMBER AND ERROR FLAG
10025 056462 032777 002000 122450  BIT    #BIT10,@SWR    ::BELL ON ERROR?
10026 056470 001402          BEQ     1$          ::NO - SKIP
10027 056472 104401 001216          TYPE   , $BELL      ::RING BELL
10028 056476 005237 001112 1$:    INC    $ERTTL       ::COUNT THE NUMBER OF ERRORS
10029 056502 011637 001116          MOV    (SP), $ERRPC  ::GET ADDRESS OF ERROR INSTRUCTION
10030 056506 162737 000002 001116  SUB    #2,$ERRPC
10031 056514 117737 122376 001114  MOVB  @$ERRPC,$ITEMB ::STRIP AND SAVE THE ERROR ITEM CODE
10032 056522 032777 020000 122410  BIT    #BIT13,@SWR   ::SKIP TYPEOUT IF SET
10033 056530 001004          BNE    20$         ::SKIP TYPEOUTS
10034 056532 004737 056604          JSR    PC,$ERRTYP   ::GO TO USER ERROR ROUTINE
10035 056536 104401 001223          TYPE   , $CRLF
10036 056542 005777 122372 2$:    TST    @SWR        ::HALT ON ERROR
10037 056546 100002          BPL    3$          ::SKIP IF CONTINUE
10038 056550 000000          HALT
10039 056552 104407          CKSWR          ::TEST FOR CHANGE IN SOFT-SWR
10040 056554 032777 001000 122356 3$:    BIT    #BIT09,@SWR  ::LOOP ON ERROR SWITCH SET?
10041 056562 001402          BEQ    4$          ::BR IF NO
10042 056564 013716 001110          MOV    $LPERR,(SP)  ::FUDGE RETURN FOR LOOPING
10043 056570 005737 001214 4$:    TST    $ESCAPE     ::CHECK FOR AN ESCAPE ADDRESS
10044 056574 001402          BEQ    5$          ::BR IF NONE
10045 056576 013716 001214          MOV    $ESCAPE,(SP) ::FUDGE RETURN ADDRESS FOR ESCAPE
10046 056602 000002          RTI              ::RETURN
```

10047	056604	104401	001223		TYPE	,\$CRLF	:::'CARRIAGE RETURN' & 'LINE FEED'
10048	056610	010046			MOV	R0,-(SP)	:::SAVE R0
10049	056612	005000			CLR	R0	:::PICKUP THE ITEM INDEX
10050	056614	153700	001114		BISB	@#\$ITEMB,R0	
10051	056620	001004			BNE	1\$:::IF ITEM NUMBER IS ZERO, JUST
10052	056622	013746	001116		MOV	\$ERRPC,-(SP)	:::SAVE \$ERRPC FOR TYPEOUT
10053	056626	104402			TYPOC		:::GO TYPE--OCTAL ASCII(ALL DIGITS)
10054	056630	000445			BR	10\$:::GET OUT
10055	056632	005300		1\$:	DEC	R0	:::ADJUST THE INDEX SO THAT IT WILL
10056	056634	006300			ASL	R0	:::WORK FOR THE ERROR TABLE
10057	056636	006300			ASL	R0	
10058	056640	006300			ASL	R0	
10059	056642	062700	001226		ADD	#\$ERRTB,R0	:::FORM TABLE POINTER
10060	056646	012037	056656		MOV	(R0)+,2\$:::PICKUP 'ERROR MESSAGE' POINTER
10061	056652	001404			BEQ	3\$:::SKIP TYPEOUT IF NO POINTER
10062	056654	104401			TYPE		:::TYPE THE 'ERROR MESSAGE'
10063	056656	000000		2\$:	.WORD	0	:::'ERROR MESSAGE' POINTER GOES HERE
10064	056660	104401	001223		TYPE	,\$CRLF	:::'CARRIAGE RETURN' & 'LINE FEED'
10065	056664	012037	056674		MOV	(R0)+,4\$:::PICKUP 'DATA HEADER' POINTER
10066	056670	001404			BEQ	5\$:::SKIP TYPEOUT IF 0
10067	056672	104401			TYPE		:::TYPE THE 'DATA HEADER'
10068	056674	000000		4\$:	.WORD	0	:::'DATA HEADER' POINTER GOES HERE
10069	056676	104401	001223		TYPE	,\$CRLF	:::'CARRIAGE RETURN' & 'LINE FEED'
10070	056702	010146			MOV	R1,-(SP)	:::SAVE R1
10071	056704	012001			MOV	(R0)+,R1	:::PICKUP 'DATA TABLE' POINTER
10072	056706	001415			BEQ	9\$:::BR IF NO DATA TO BE TYPED
10073	056710	012000			MOV	(R0)+,R0	:::PICKUP 'DATA FORMAT' POINTER
10074	056712	105720			TSTB	(R0)+	:::'OCTAL' OR 'DECIMAL'
10075	056714	001003		6\$:	BNE	7\$:::BR IF DECIMAL
10076	056716	013146			MOV	@(R1)+,-(SP)	:::SAVE @(R1)+ FOR TYPEOUT
10077	056720	104402			TYPOC		:::GO TYPE--OCTAL ASCII(ALL DIGITS)
10078	056722	000402			BR	8\$	
10079	056724	013146			MOV	@(R1)+,-(SP)	:::SAVE @(R1)+ FOR TYPEOUT
10080	056726	104405			TYPDS		:::GO TYPE--DECIMAL ASCII WITH SIGN
10081	056730	005711		8\$:	TST	(R1)	:::IS THERE ANOTHER NUMBER?
10082	056732	001403			BEQ	9\$:::BR IF NO
10083	056734	104401	056754		TYPE	,11\$:::TYPE TWO(2) SPACES
10084	056740	000764			BR	6\$:::LOOP
10085	056742	012601		9\$:	MOV	(SP)+,R1	:::RESTORE R1
10086	056744	012600		10\$:	MOV	(SP)+,R0	:::RESTORE R0
10087	056746	104401	001223		TYPE	,\$CRLF	:::'CARRIAGE RETURN' & 'LINE FEED'
10088	056752	000207			RTS	PC	:::RETURN
10089	056754	020040	000	11\$:	.ASCIZ	/ /	:::TWO(2) SPACES

```

10090 056760 017646 000000          $TYPOS: MOV @ (SP),-(SP)      ;;PICKUP THE MODE
10091 056764 116637 000001 057203  MOVB 1(SP), $OFILL      ;;LOAD ZERO FILL SWITCH
10092 056772 112637 057205          MOVB (SP)+, $OMODE+1    ;;NUMBER OF DIGITS TO TYPE
10093 056776 062716 000002          ADD #2, (SP)           ;;ADJUST RETURN ADDRESS
10094 057002 000406                    BR $TYPON
10095 057004 112737 000001 057203  $TYPOC: MOVB #1, $OFILL    ;;SET THE ZERO FILL SWITCH
10096 057012 112737 000006 057205  MOVB #6, $OMODE+1      ;;SET FOR SIX(6) DIGITS
10097 057020 112737 000005 057202  $TYPON: MOVB #5, $OCNT   ;;SET THE ITERATION COUNT
10098 057026 010346                    MOV R3, -(SP)          ;;SAVE R3
10099 057030 010446                    MOV R4, -(SP)          ;;SAVE R4
10100 057032 010546                    MOV R5, -(SP)          ;;SAVE R5
10101 057034 113704 057205          MOVB $OMODE+1, R4      ;;GET THE NUMBER OF DIGITS TO TYPE
10102 057040 005404                    NEG R4
10103 057042 062704 000006          ADD #6, R4             ;;SUBTRACT IT FOR MAX. ALLOWED
10104 057046 110437 057204          MOVB R4, $OMODE        ;;SAVE IT FOR USE
10105 057052 113704 057203          MOVB $OFILL, R4        ;;GET THE ZERO FILL SWITCH
10106 057056 016605 000012          MOV 12(SP), R5         ;;PICKUP THE INPUT NUMBER
10107 057062 005003                    CLR R3                 ;;CLEAR THE OUTPUT WORD
10108 057064 006105          1$: ROL R5             ;;ROTATE MSB INTO 'C'
10109 057066 000404                    BR 3$
10110 057070 006105          2$: ROL R5             ;;FORM THIS DIGIT
10111 057072 006105          ROL R5
10112 057074 006105          ROL R5
10113 057076 010503          MOV R5, R3
10114 057100 006103          3$: ROL R3             ;;GET LSB OF THIS DIGIT
10115 057102 105337 057204          DECB $OMODE           ;;TYPE THIS DIGIT?
10116 057106 100016          BPL 7$                ;;BR IF NO
10117 057110 042703 177770          BIC #177770, R3       ;;GET RID OF JUNK
10118 057114 001002          BNE 4$                ;;TEST FOR 0
10119 057116 005704          TST R4                ;;SUPPRESS THIS 0?
10120 057120 001403          BEQ 5$                ;;BR IF YES
10121 057122 005204          4$: INC R4            ;;DON'T SUPPRESS ANYMORE 0'S
10122 057124 052703 000060          BIS #'0, R3           ;;MAKE THIS DIGIT ASCII
10123 057130 052703 000040          5$: BIS #' , R3       ;;MAKE ASCII IF NOT ALREADY
10124 057134 110337 057200          MOVB R3, 8$           ;;SAVE FOR TYPING
10125 057140 104401 057200          TYPE ,8$             ;;GO TYPE THIS DIGIT
10126 057144 105337 057202          7$: DECB $OCNT        ;;COUNT BY 1
10127 057150 003347          BGT 2$                ;;BR IF MORE TO DO
10128 057152 002402          BLT 6$                ;;BR IF DONE
10129 057154 005204          INC R4                ;;INSURE LAST DIGIT ISN'T A BLANK
10130 057156 000744          BR 2$                 ;;GO DO THE LAST DIGIT
10131 057160 012605          6$: MOV (SP)+, R5      ;;RESTORE R5
10132 057162 012604          MOV (SP)+, R4         ;;RESTORE R4
10133 057164 012603          MOV (SP)+, R3         ;;RESTORE R3
10134 057166 016666 000002 000004  MOV 2(SP), 4(SP)      ;;SET THE STACK FOR RETURNING
10135 057174 012616          MOV (SP)+, (SP)
10136 057176 000002          RTI                   ;;RETURN
10137 057200 000          8$: .BYTE 0          ;;STORAGE FOR ASCII DIGIT
10138 057201 000          .BYTE 0              ;;TERMINATOR FOR TYPE ROUTINE
10139 057202 000          $OCNT: .BYTE 0       ;;OCTAL DIGIT COUNTER
10140 057203 000          $OFILL: .BYTE 0      ;;ZERO FILL SWITCH
10141 057204 000000          $OMODE: .WORD 0      ;;NUMBER OF DIGITS TO TYPE

```

```
10142 057206 010046          $TRAP: MOV      RO,-(SP)          ::SAVE RO
10143 057210 016600 000002    MOV      2(SP),RO          ::GET TRAP ADDRESS
10144 057214 005740          TST      -(RO)             ::BACKUP BY 2
10145 057216 111000          MOV      (RO),RO           ::GET RIGHT BYTE OF TRAP
10146 057220 006300          ASL      RO                 ::POSITION FOR INDEXING
10147 057222 016000 057242    MOV      $TRPAD(RO),RO     ::INDEX TO TABLE
10148 057226 000200          RTS      RO                 ::GO TO ROUTINE
10149 057230 011646          $TRAP2: MOV     (SP),-(SP)    ::MOVE THE PC DOWN
10150 057232 016666 000004 000002  MOV     4(SP),2(SP)        ::MOVE THE PSW DOWN
10151 057240 000002          RTI                          ::RESTORE THE PSW
10152 057242 057230          $TRPAD: .WORD   $TRAP2
10153 057244 054736          $TYPE    ::CALL=TYPE        TRAP+1(104401) TTY TYPEOUT ROUTINE
10154 057246 057004          $TYPOC   ::CALL=TYPOC       TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
10155 057250 056760          $TYPOS   ::CALL=TYPOS       TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
10156 057252 057020          $TYPON   ::CALL=TYPON       TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
10157 057254 054512          $TYPDS   ::CALL=TYPDS        TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
10158 057256 055554          $GTSWR   ::CALL=GTSWR        TRAP+6(104406) GET SOFT-SWR SETTING
10159 057260 055464          $CKSWR   ::CALL=CKSWR        TRAP+7(104407) TEST FOR CHANGE IN SOFT-SWR
10160 057262 056026          $RDCHR   ::CALL=RDCHR        TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
10161 057264 056116          $RDLIN   ::CALL=RDLIN        TRAP+11(104411) TTY TYPEIN STRING ROUTINE
10162 057266 056276          $RDOCT   ::CALL=RDOCT         TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
10163 057270 042602          T.SCOPI  ::CALL=SCOPI           TRAP+13(104413) MY LOCAL SCOPES
10164 057272 042654          CHECKT   ::CALL=CHECKD         TRAP+14(104414) CHECK DVA,RDY,DPR,DRY
10165 057274 043172          WAIT.T   ::CALL=WAT             TRAP+15(104415) WAIT LOOP
```

```
10166 057276 012737 057442 000024 $PWRDN: MOV    #$ILLUP,@#PWRVEC  ;;SET FOR FAST UP
10167 057304 012737 000340 000026      MOV    #340,@#PWRVEC+2 ;;PRIO:7
10168 057312 010046          MOV    R0,-(SP)        ;;PUSH R0 ON STACK
10169 057314 010146          MOV    R1,-(SP)        ;;PUSH R1 ON STACK
10170 057316 010246          MOV    R2,-(SP)        ;;PUSH R2 ON STACK
10171 057320 010346          MOV    R3,-(SP)        ;;PUSH R3 ON STACK
10172 057322 010446          MOV    R4,-(SP)        ;;PUSH R4 ON STACK
10173 057324 010546          MOV    R5,-(SP)        ;;PUSH R5 ON STACK
10174 057326 017746 121606      MOV    @SWR,-(SP)      ;;PUSH @SWR ON STACK
10175 057332 010637 057446      MOV    SP,$SAVR6       ;;SAVE SP
10176 057336 012737 057350 000024      MOV    #PWRUP,@#PWRVEC ;;SET UP VECTOR
10177 057344 000000          HALT
10178 057346 000776          BR     .-2             ;;HANG UP
10179 057350 012737 057442 000024 $PWRUP: MOV    #$ILLUP,@#PWRVEC  ;;SET FOR FAST DOWN
10180 057356 013706 057446      MOV    $SAVR6,SP       ;;GET SP
10181 057362 005037 057446      CLR    $SAVR6          ;;WAIT LOOP FOR THE TTY
10182 057366 005237 057446      1$:   INC    $SAVR6     ;;WAIT FOR THE INC
10183 057372 001375          BNE    1$              ;;OF WORD
10184 057374 012677 121540      MOV    (SP)+,@SWR      ;;POP STACK INTO @SWR
10185 057400 012605          MOV    (SP)+,R5       ;;POP STACK INTO R5
10186 057402 012604          MOV    (SP)+,R4       ;;POP STACK INTO R4
10187 057404 012603          MOV    (SP)+,R3       ;;POP STACK INTO R3
10188 057406 012602          MOV    (SP)+,R2       ;;POP STACK INTO R2
10189 057410 012601          MOV    (SP)+,R1       ;;POP STACK INTO R1
10190 057412 012600          MOV    (SP)+,R0       ;;POP STACK INTO R0
10191 057414 012737 057276 000024      MOV    #PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
10192 057422 012737 000340 000026      MOV    #340,@#PWRVEC+2 ;;PRIO:7
10193 057430 104401          TYPE    ;REPORT THE POWER FAILURE
10194 057432 057450          $PWRMG: .WORD $POWER  ;;POWER FAIL MESSAGE POINTER
10195 057434 012716          MOV    (PC)+,(SP)     ;;RESTART AT BEGIN
10196 057436 004240          $PWRAD: .WORD BEGIN   ;;RESTART ADDRESS
10197 057440 000002          RTI
10198 057442 000000          $ILLUP: HALT          ;;THE POWER UP SEQUENCE WAS STARTED
10199 057444 000776          BR     .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
10200 057446 000000          $SAVR6: 0              ;;PUT THE SP HERE
10201 057450 005015 047520 042527 $POWER: .ASCIZ <15><12>'POWER''
10202 057456 000122
```

10203
10204
10205
10206
10207
10208
10209
10210
10211
10212
10213
10214
10215
10216
10217
10218
10219
10220
10221
10222
10223
10224
10225
10226
10227
10228
10229
10230
10231
10232
10233
10234
10235
10236
10237
10238
10239
10240
10241
10242
10243
10244
10245
10246
10247
10248
10249
10250
10251
10252
10253
10254
10255
10256
10257
10258

057460 051127 047117 020107
057466 040504 040524 044440
057474 020116 042522 042101
057502 047111 020107 051117
057510 053440 044522 044524
057516 043516 044040 051101
057524 053504 051101 020105
057532 042522 044507 052123
057540 051105 000
057543 105 051122 051117
057550 047440 020116 042040
057556 052101 020101 047503
057564 046515 047101 000104
057572 051105 047522 020122
057600 047117 053440 044522
057606 042524 044040 040505
057614 042504 020122 047101
057622 020104 040504 040524
057630 000
057631 103 047117 051124
057636 046117 042514 020122
057644 051117 042040 044522
057652 042526 051440 040524
057660 052524 000123
057664 042522 044507 052123
057672 051105 043040 044501
057700 042514 000104
057704 047516 020116 054105
057712 051511 042524 052116
057720 051040 043505 051511
057726 042524 026122 020040
057734 051120 043517 040522
057742 020115 041101 051117
057750 042524 027104 000
057755 127 044501 020124
057762 047514 050117 043040
057770 044501 042514 000104
057776 051127 052111 020105
060004 044103 041505 020113
060012 040506 046111 047111
060020 000107
060022 042522 044507 052123
060030 051105 043040 044501
060036 044514 043516 000
060043 111 052116 051105
060050 052522 052120 043040
060056 044501 044514 043516

*
*ERROR AND MESSAGE TABLE CONDIMENTS
*

EM1: .ASCIZ /WRONG DATA IN READING OR WRITING HARDWARE REGISTER/
EM2: .ASCIZ /ERROR ON DATA COMMAND/
EM6: .ASCIZ /ERROR ON WRITE HEADER AND DATA/
EM11: .ASCIZ /CONTROLLER OR DRIVE STATUS/
EM14: .ASCIZ /REGISTER FAILED/
EM15: .ASCIZ /NON EXISTENT REGISTER, PROGRAM ABORTED./
EM16: .ASCIZ /WAIT LOOP FAILED/
EM17: .ASCIZ /WRITE CHECK FAILING/
EM20: .ASCIZ /REGISTER FAILING/
EM21: .ASCIZ /INTERRUPT FAILING/

10259	060064	000					
10260	060065	105	051122	051117	EM22:	.ASCII	/ERROR ON DRIVES PRESENT -/<15><12>
10261	060072	047440	020116	051104			
10262	060100	053111	051505	050040			
10263	060106	042522	042523	052116			
10264	060114	026440	005015				
10265	060120	044124	020105	047125		.ASCII	/THE UNIT NO'S FOUND BY SETTING RHAS USING RHER1/<15><12>
10266	060126	052111	047040	023517			
10267	060134	020123	047506	047125			
10268	060142	020104	054502	051440			
10269	060150	052105	044524	043516			
10270	060156	051040	040510	020123			
10271	060164	051525	047111	020107			
10272	060172	044122	051105	006461			
10273	060200	012					
10274	060201	050	032124	020051		.ASCII	/(T4) DO NOT AGREE WITH THE UNIT NO'S FOUND/<15><12>
10275	060206	047504	047040	052117			
10276	060214	040440	051107	042505			
10277	060222	053440	052111	020110			
10278	060230	044124	020105	047125			
10279	060236	052111	047040	023517			
10280	060244	020123	047506	047125			
10281	060252	006504	012				
10282	060255	102	020131	047514		.ASCII	/BY LOOKING FOR 'NED' = 0 IN RHCS2 (BIT #12)/<15><12><15><12>
10283	060262	045517	047111	020107			
10284	060270	047506	020122	047047			
10285	060276	042105	020047	020075			
10286	060304	020060	047111	051040			
10287	060312	041510	031123	024040			
10288	060320	044502	020124	030443			
10289	060326	024462	005015	005015			
10290	060334	047516	042524	020072		.ASCII	/NOTE: ON DUAL PORT SYSTEM, A DRIVE ON OTHER PORT WILL /<15><12>
10291	060342	047117	042040	040525			
10292	060350	020114	047520	052122			
10293	060356	051440	051531	042524			
10294	060364	026115	040440	042040			
10295	060372	044522	042526	047440			
10296	060400	020116	052117	042510			
10297	060406	020122	047520	052122			
10298	060414	053440	046111	020114			
10299	060422	005015					
10300	060424	047516	020124	044507		.ASCII	/NOT GIVE 'NED', BUT WILL GIVE RHAS RESPONSES/<15><12>
10301	060432	042526	023440	042516			
10302	060440	023504	020054	052502			
10303	060446	020124	044527	046114			
10304	060454	043440	053111	020105			
10305	060462	044122	051501	051040			
10306	060470	051505	047520	051516			
10307	060476	051505	005015				
10308	060502	042510	041516	020105		.ASCIZ	/HENCE THERE WILL BE AN EXTRA DRIVE/
10309	060510	044124	051105	020105			
10310	060516	044527	046114	041040			
10311	060524	020105	047101	042440			
10312	060532	052130	040522	042040			
10313	060540	044522	042526	000			
10314	060545	114	047517	020113	EM24:	.ASCIZ	/LOOK AHEAD REGISTER AT THE BEGINNING OF SECTOR IS IN ERROR/

10315	060552	044101	040505	020104
10316	060560	042522	044507	052123
10317	060566	051105	040440	020124
10318	060574	044124	020105	042502
10319	060602	044507	047116	047111
10320	060610	020107	043117	051440
10321	060616	041505	047524	020122
10322	060624	051511	044440	020116
10323	060632	051105	047522	000122
10324	060640	047514	045517	040440
10325	060646	042510	042101	051040
10326	060654	043505	051511	042524
10327	060662	020122	051511	044440
10328	060670	020116	051105	047522
10329	060676	000122		
10330	060700	052503	051122	047105
10331	060706	020124	054503	044514
10332	060714	042116	051105	042040
10333	060722	042517	020123	047516
10334	060730	020124	040515	041524
10335	060736	020110	042504	044523
10336	060744	042522	020104	054503
10337	060752	044514	042116	051105
10338	060760	051040	043505	051511
10339	060766	042524	006522	012
10340	060773	101	052106	051105
10341	061000	040440	051440	042505
10342	061006	020113	047101	020104
10343	061014	047111	052111	000
10344	061021	105	041503	043440
10345	061026	047105	051105	052101
10346	061034	042105	044440	020123
10347	061042	047111	047503	051122
10348	061050	041505	006524	012
10349	061055	105	042526	054522
10350	061062	053440	051117	020104
10351	061070	047117	052040	044510
10352	061076	020123	042523	052103
10353	061104	051117	044440	020123
10354	061112	044124	052101	043440
10355	061120	053111	047105	044440
10356	061126	020116	042042	052101
10357	061134	020101	051525	042105
10358	061142	000042		
10359	061144	047117	051040	040505
10360	061152	020104	047503	046515
10361	061160	047101	026104	040440
10362	061166	052106	051105	042040
10363	061174	052101	020101	047101
10364	061202	020104	041505	020103
10365	061210	040510	042526	041040
10366	061216	042505	020116	042522
10367	061224	042101	006454	012
10368	061231	105	041503	051040
10369	061236	043505	051511	042524
10370	061244	051522	047440	020122

EM25: .ASCIZ /LOOK AHEAD REGISTER IS IN ERROR/

EM30: .ASCII /CURRENT CYLINDER DOES NOT MATCH DESIRED CYLINDER REGISTER/<15><12>

.ASCIZ /AFTER A SEEK AND INIT/

EM31: .ASCII /ECC GENERATED IS INCORRECT/<15><12>

.ASCIZ /EVERY WORD ON THIS SECTOR IS THAT GIVEN IN 'DATA USED'/'

EM32: .ASCII /ON READ COMMAND, AFTER DATA AND ECC HAVE BEEN READ,/<15><12>

.ASCII /ECC REGISTERS OR RHER1 ARE IN ERROR/<15><12>

10371	061252	044122	051105	020061
10372	061260	051101	020105	047111
10373	061266	042440	051122	051117
10374	061274	005015		
10375	061276	047117	054514	046040
10376	061304	053517	051105	030440
10377	061312	020061	044502	051524
10378	061320	047440	020106	040520
10379	061326	052124	051105	020116
10380	061334	042522	027107	041440
10381	061342	047101	041040	020105
10382	061350	042522	042101	005015
10383	061356	044124	051511	051440
10384	061364	047510	046125	020104
10385	061372	040515	041524	020110
10386	061400	047514	042527	020122
10387	061406	030461	041040	052111
10388	061414	020123	043117	043440
10389	061422	047517	020104	041505
10390	061430	030503	000	
10391	061433	110	043511	020110
10392	061440	047503	047125	020124
10393	061446	044502	020124	047516
10394	061454	020124	042523	020124
10395	061462	043101	042524	020122
10396	061470	034063	032470	020071
10397	061476	046103	041517	051513
10398	061504	000		
10399	061505	132	051105	020117
10400	061512	042504	042524	052103
10401	061520	041040	052111	047040
10402	061526	052117	044040	043511
10403	061534	020110	044127	047105
10404	061542	031440	020062	044502
10405	061550	020124	041505	020103
10406	061556	042522	027107	044040
10407	061564	051501	031040	020061
10408	061572	042532	047522	000123
10409	061600	047520	044523	044524
10410	061606	047117	051040	043505
10411	061614	051511	042524	020122
10412	061622	051117	030440	020061
10413	061630	044502	051524	047440
10414	061636	020106	040520	052124
10415	061644	051105	020116	042522
10416	061652	044507	052123	051105
10417	061660	044440	041516	051117
10418	061666	042522	052103	005015
10419	061674	047514	042527	020122
10420	061702	030461	041040	052111
10421	061710	020123	043117	050040
10422	061716	052101	042524	047122
10423	061724	051040	043505	051511
10424	061732	042524	020122	044123
10425	061740	052517	042114	046440
10426	061746	052101	044103	046040

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

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

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

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

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

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

10427	061754	053517	051105	005015	
10428	061762	030461	041040	052111	.ASCII /11 BITS OF GOOD ECC1/<15><12>
10429	061770	020123	043117	043440	
10430	061776	047517	020104	041505	
10431	062004	030503	005015		
10432	062010	040504	020124	047105	.ASCIZ /DAT ENVLOP GOOD POSITION AND N-CODE ZEROS ARE IN OCTAL/
10433	062016	046126	050117	043440	
10434	062024	047517	020104	047520	
10435	062032	044523	044524	047117	
10436	062040	040440	042116	047040	
10437	062046	041455	042117	020105	
10438	062054	042532	047522	020123	
10439	062062	051101	020105	047111	
10440	062070	047440	052103	046101	
10441	062076	000			
10442	062077	117	020116	042522	EM36: .ASCIZ /ON READ COMMAND WITH NON-CORRECTABLE ERROR 'DCK' AND 'ECH' SHOULD BE SE
10443	062104	042101	041440	046517	
10444	062112	040515	042116	053440	
10445	062120	052111	020110	047516	
10446	062126	026516	047503	051122	
10447	062134	041505	040524	046102	
10448	062142	020105	051105	047522	
10449	062150	020122	042047	045503	
10450	062156	020047	047101	020104	
10451	062164	042447	044103	020047	
10452	062172	044123	052517	042114	
10453	062200	041040	020105	042523	
10454	062206	000124			
10455	062210	051120	043517	040522	EM37: .ASCII /PROGRAM ERROR BIT #10 IN RHCS2 DID NOT SET/<15><12>
10456	062216	020115	051105	047522	
10457	062224	020122	044502	020124	
10458	062232	030443	020060	047111	
10459	062240	051040	041510	031123	
10460	062246	042040	042111	047040	
10461	062254	052117	051440	052105	
10462	062262	005015			
10463	062264	043111	050040	051517	.ASCIZ /IF POSITION REGISTER =10040 OR 10041, IT IS GOOD/
10464	062272	052111	047511	020116	
10465	062300	042522	044507	052123	
10466	062306	051105	036440	030061	
10467	062314	032060	020060	051117	
10468	062322	030440	030060	030464	
10469	062330	020054	052111	044440	
10470	062336	020123	047507	042117	
10471	062344	000			
10472					
10473	062345	122	053510	020103	EM40: .ASCII /RHWC DID NOT = 0 UPON COMPLETION OF READ/<15><12>
10474	062352	044504	020104	047516	
10475	062360	020124	020075	020060	
10476	062366	050125	047117	041440	
10477	062374	046517	046120	052105	
10478	062402	047511	020116	043117	
10479	062410	051040	040505	006504	
10480	062416	012			
10481	062417	117	020122	051127	.ASCIZ /OR WRITE HEADER AND DATA/
10482	062424	052111	020105	042510	

10486										
10487	062450	040506	040524	020114	CPHALT: .ASCII	/FATAL ERROR - SEE DOCUMENT LISTING/	<15>	<12>		
10488	062456	051105	047522	020122						
10489	062464	020055	042523	020105						
10490	062472	047504	052503	042515						
10491	062500	052116	046040	051511						
10492	062506	044524	043516	005015						
10493	062514	006440	103412	177777	.ASCII	/ /	<15>	<12>	<207>	<377>
10494	062522	177607	103777	177777						
10495	062530	044124	020105	047503	.ASCII	/THE CONTROLLER OR DEVICE HAS GONE OFFLINE, LOST/	<15>	<12>		
10496	062536	052116	047522	046114						
10497	062544	051105	047440	020122						
10498	062552	042504	044526	042503						
10499	062560	044040	051501	043440						
10500	062566	047117	020105	043117						
10501	062574	046106	047111	026105						
10502	062602	046040	051517	006524						
10503	062610	012								
10504	062611	047	042522	042101	.ASCII	/'READY', BECOME UNAVAILABLE, OR HAS STATUS BITS/	<15>	<12>		
10505	062616	023531	020054	042502						
10506	062624	047503	042515	052440						
10507	062632	040516	040526	046111						
10508	062640	041101	042514	020054						
10509	062646	051117	044040	051501						
10510	062654	051440	040524	052524						
10511	062662	020123	044502	051524						
10512	062670	005015								
10513	062672	044127	041511	020110	.ASCIZ	/WHICH CANNOT BE CLEARED/				
10514	062700	040503	047116	052117						
10515	062706	041040	020105	046103						
10516	062714	040505	042522	000104						
10517										
10518	062722	020040	020040	020040	SPACE8: .ASCII	/ / /				
10519	062730	020040	000		SPACE2: .ASCIZ	/ / /				
10520										
10521										
10522	062733	120	020103	020040	DH1: .ASCII	/PC TEST REG. GOOD RECEIVED/	<15>	<12>		
10523	062740	020040	052040	051505						
10524	062746	020124	020040	051040						
10525	062754	043505	020056	020040						
10526	062762	043440	047517	020104						
10527	062770	020040	051040	041505						
10528	062776	044505	042526	006504						
10529	063004	012								
10530	063005	040	020040	020040	.ASCIZ	/ NO ADDR. DATA DATA /				
10531	063012	020040	047040	020117						
10532	063020	020040	020040	040440						
10533	063026	042104	027122	020040						
10534	063034	042040	052101	020101						
10535	063042	020040	042040	052101						
10536	063050	020101	020040	000040						
10537	063056	041520	020040	020040	DH2: .ASCII	/PC TEST WORD GOOD BAD /	<15>	<12>		
10538	063064	020040	042524	052123						
10539	063072	020040	020040	047527						
10540	063100	042122	020040	020040						
10541	063106	047507	042117	020040						

10934									
10935	067354	001116	002032	001122	DT11:	.WORD	\$ERRPC,TSTNM,\$BDADR,CS1,CS2,DS1,ER1,0		
10936	067362	001714	001712	001736					
10937	067370	001716	000000						
10938	067374	001116	002032	001122	DT14:	.WORD	\$ERRPC,TSTNM,\$BDADR,\$BDDAT,CS1,CS2,DS1,ER1,0		
10939	067402	001126	001714	001712					
10940	067410	001736	001716	000000					
10941	067416	001116	002032	001200	DT15:	.WORD	\$ERRPC,TSTNM,\$TMP1,0		
10942	067424	000000							
10943	067426	001116	002032	001204	DT16:	.WORD	\$ERRPC,TSTNM,\$TMP3,\$TMP1,\$TMP0,\$BDDAT,0		
10944	067434	001200	001176	001126					
10945	067442	000000							
10946	067444	001116	002032	001710	DT17:	.WORD	\$ERRPC,TSTNM,BA,DB,WC,CS1,CS2,0		
10947	067452	001704	001706	001714					
10948	067460	001712	000000						
10949									
10950	067464	001116	002032	001716	DT20:	.WORD	\$ERRPC,TSTNM,ER1,ER2,ER3,AS,DS1,0		
10951	067472	001722	001730	001732					
10952	067500	001736	000000						
10953	067504	001116	002032	001714	DT21:	.WORD	\$ERRPC,TSTNM,CS1,AS,DS1,0		
10954	067512	001732	001736	000000					
10955	067520	001116	002032	000000	DT22:	.WORD	\$ERRPC,TSTNM,0		
10956	067526	001116	002032	001720	DT24:	.WORD	\$ERRPC,TSTNM,DST,\$BDDAT,\$TMP1,\$TMP2,\$TMP3,0		
10957	067534	001126	001200	001202					
10958	067542	001204	000000						
10959	067546	001116	002032	002014	DT26:	.WORD	\$ERRPC,TSTNM,PCJSR,\$BDADR,CS1,CS2,DS1,ER1,0		
10960	067554	001122	001714	001712					
10961	067562	001736	001716	000000					
10962	067570	001116	002032	002014	DT27:	.WORD	\$ERRPC,TSTNM,PCJSR,REGADR,\$GDDAT,\$BDDAT,0		
10963	067576	042270	001124	001126					
10964	067604	000000							
10965									
10966	067606	001116	002032	002014	DT30:	.WORD	\$ERRPC,TSTNM,PCJSR,REGADR,\$GDDAT,\$BDDAT,0		
10967	067614	042270	001124	001126					
10968	067622	000000							
10969	067624	001116	002032	047624	DT31:	.WORD	\$ERRPC,TSTNM,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1,0		
10970	067632	001124	001126	001714					
10971	067640	001736	001716	000000					
10972	067646	001116	002032	002014	DT32:	.WORD	\$ERRPC,TSTNM,PCJSR,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1,0		
10973	067654	047624	001124	001126					
10974	067662	001714	001736	001716					
10975	067670	000000							
10976	067672	001116	002032	002014	DT33:	.WORD	\$ERRPC,TSTNM,PCJSR,ERWORD,\$GDDAT,CS1,DS1,ER1,0		
10977	067700	047624	001124	001714					
10978	067706	001736	001716	000000					
10979	067714	001116	002032	045314	DT34:	.WORD	\$ERRPC,TSTNM,GECC1,GECC2,WECC1,WECC2,DISK,0		
10980	067722	045316	052422	052424					
10981	067730	051422	000000						
10982	067734	001116	002032	045314	DT35:	.WORD	\$ERRPC,TSTNM,GECC1,GECC2,EC2,EC1,POSITI,ER1,0		
10983	067742	045316	001746	001744					
10984	067750	045326	001716	000000					
10985	067756	001116	002032	002014	DT36:	.WORD	\$ERRPC,TSTNM,PCJSR,MR,EC1,EC2,0		
10986	067764	001734	001744	001746					
10987	067772	000000							
10988	067774	001116	002032	001744	DT37:	.WORD	\$ERRPC,TSTNM,EC1,POSITI,GECC1,GECC2,EC2,DATENV,ZCODE,0		
10989	070002	045326	045314	045316					

10990	070010	001746	045332	045334			
10991	070016	000000					
10992							
10993	070020	001116	002032	001126	DT40:	.WORD	\$ERRPC,TSTNM,\$BDDAT,0
10994	070026	000000					
10995							
10996	070030	000	000	000	DF1:	.BYTE	0,0,0,0,0
10997	070033	000	000				
10998	070035	000	000	001	DF2:	.BYTE	0,0,1,0
10999	070040	000					
11000	070041	000	000	001	DF3:	.BYTE	0,0,1,0,0
11001	070044	000	000				
11002							
11003	070046	000	000	000	DF11:	.BYTE	0,0,0,0,0,0,0
11004	070051	000	000	000			
11005	070054	000					
11006	070055	000	000	000	DF14:	.BYTE	0,0,0,0,0,0,0,0
11007	070060	000	000	000			
11008	070063	000	000				
11009	070065	000	000	000	DF15:	.BYTE	0,0,0
11010	070070	000	000	000	DF16:	.BYTE	0,0,0,0,0
11011	070073	000	000				
11012	070075	000	000	000	DF17:	.BYTE	0,0,0,0,0,0,0
11013	070100	000	000	000			
11014	070103	000					
11015							
11016	070104	000	000	000	DF20:	.BYTE	0,0,0,0,0,0,0
11017	070107	000	000	000			
11018	070112	000					
11019	070113	000	000	000	DF21:	.BYTE	0,0,0,0,0
11020	070116	000	000				
11021	070120	000	000	000	DF22:	.BYTE	0,0,0,0
11022	070123	000					
11023	070124	000	000	000	DF24:	.BYTE	0,0,0,0,0,0,0
11024	070127	000	000	000			
11025	070132	000					
11026	070133	000	000	000	DF26:	.BYTE	0,0,0,0,0,0,0,0
11027	070136	000	000	000			
11028	070141	000	000				
11029	070143	000	000	000	DF27:	.BYTE	0,0,0,0,0,0
11030	070146	000	000	000			
11031							
11032	070151	000	000	000	DF30:	.BYTE	0,0,0,0,0,0
11033	070154	000	000	000			
11034	070157	000	000	001	DF31:	.BYTE	0,0,1,0,0,0,0,0
11035	070162	000	000	000			
11036	070165	000	000				
11037	070167	000	000	000	DF32:	.BYTE	0,0,0,1,0,0,0,0,0
11038	070172	001	000	000			
11039	070175	000	000	000			
11040	070200	000	000	000	DF33:	.BYTE	0,0,0,1,0,0,0,0
11041	070203	001	000	000			
11042	070206	000	000				
11043	070210	000	000	000	DF34:	.BYTE	0,0,0,0,0,0,0
11044	070213	000	000	000			
11045	070216	000					

11046	070217	000	000	000	DF35:	.BYTE	0,0,0,0,0,0,0
11047	070222	000	000	000			
11048	070225	000	000				
11049	070227	000	000	000	DF36:	.BYTE	0,0,0,0,0,0
11050	070232	000	000	000			
11051	070235	000	000	000	DF37:	.BYTE	0,0,0,0,0,0,0,0,0
11052	070240	000	000	000			
11053	070243	000	000	000			
11054							
11055	070246	000	000	000	DF40:	.BYTE	0,0,0
11056		070252				.EVEN	
11057							
11058		000001			.END		

DF11	070046	693	992	11003#										
DF14	070055	724	11006#											
DF15	070065	733	11009#											
DF16	070070	744	11010#											
DF17	070075	757	11012#											
DF2	070035	10998#												
DF20	070104	770	11016#											
DF21	070113	781	11019#											
DF22	070120	796	11021#											
DF24	070124	817	830	11023#										
DF26	070133	843	11026#											
DF27	070143	858	11029#											
DF3	070041	673	680	11000#										
DF30	070151	871	11032#											
DF31	070157	636	644	11034#										
DF32	070167	620	661	11037#										
DF33	070200	603	11040#											
DF34	070210	889	11043#											
DF35	070217	908	980	11046#										
DF36	070227	923	940	11049#										
DF37	070235	963	11051#											
DF40	070246	1001	11055#											
DF5	= 000001	1060#												
DH1	062733	580	699	10522#										
DH11	063201	685	984	10551#										
DH14	063361	716	10571#											
DH15	063561	730	10594#											
DH16	063612	738	10599#											
DH17	063755	749	10617#											
DH2	063056	667	10537#											
DH20	064140	762	10637#											
DH21	064323	775	10657#											
DH22	064443	791	10672#											
DH24	064546	810	823	10684#										
DH26	064731	834	10704#											
DH27	065134	850	10727#											
DH30	065277	864	10745#											
DH31	065442	626	10763#											
DH32	065645	609	650	10785#										
DH33	066070	594	10811#											
DH34	066273	879	10833#											
DH35	066456	899	970	10853#										
DH36	066661	914	931	10875#										
DH37	067020	951	10892#											
DH40	067236	997	10917#											
DIGB	= 000004	1062#												
DISK	051422	2133	2141	2194	2208	4722	4826	4939	5049	5124	5198	5273	5308	5337
		5338	5473	5474	5478	5485	5486	5592	5598	5605	5606	5807	5808	5913
		5914	6023	6029	6036	6037	6042*	6153	6246	6288	6647	6682	6741	6776
		6821	6961	7965	8078	8521	8535*	8536*	9132	9191#	9640	10979		
DISPLA	001142	547#	1459*	1466*	9713*	10024*								
DISPRE	000174	503#	1466											
DLT	= 100000	1034#	2635	2803	2878	2881								
DL64	= 000020	1064#												
DMD	= 000001	1097#	1988	1991	2012	2289	2484	2556	3082	3099	3182	3240	3297	3380
		3401	3602	3690	3743	3803	3855	3924	3980	3997	3998	4004	4063	4114

EM17	057776	748	10249#																
EM2	057543	592	607	624	10221#														
EM20	060022	761	10253#																
EM21	060043	774	10256#																
EM22	060065	785	10260#																
EM24	060545	807	10314#																
EM25	060640	820	10324#																
EM30	060700	863	10330#																
EM31	061021	876	10344#																
EM32	061144	893	10359#																
EM33	061433	913	10391#																
EM34	061505	926	10399#																
EM35	061600	945	10409#																
EM36	062077	968	10442#																
EM37	062210	983	10455#																
EM40	062345	995	10473#																
EM6	057572	648	666	10225#															
ERCLFC	020570	3413	3422	3431	3443	3457	3466	3474	3482	3490	3498	3506	3515	3523					
		3533	3541	3550	3558	3566	3582#												
ERCRC	052662	9256*	9305#	9477*															
ERCS2C	013434	2317	2325	2335	2343	2351	2359	2367	2375	2383	2391	2399	2408	2418					
		2427	2436	2444	2453	2470#													
ERFLG\$	002006	1347#	2128*	2274*	2750*	2764	4681*	4702	4716*	4783*	4801	4820*	4895*	4915					
		4933*	5004*	5080*	5085	5155*	5160	5229*	5234	5302*	5305	5316	5375*	5386					
		5409	5539*	5635*	5846*	5857*	5877	5949*	6043*	6101*	6184*	6189	6277*	6285					
		6296	6676*	6679	6690	6770*	6773	6784	6875*	6954*	6969	7028*	7119*	7210*					
		7790	8028*	8039	8782	9339	10021*												
ERHDGP	052664	9257*	9313#	9488*															
ERHEAD	052660	9255*	9293#	9465*															
ERPOS	045734	8450#	8456*	8473															
ERR =	040000	1074#	3124	5726	5746	5782	6368	6920	7050	7145	7244								
ERRVEC =	000004	502#	1456	1457*	1467*	1532*	1539*	1568*	9676	9677*	9679*	9682*							
ERSTAR	047316	8618#																	
ERUNIT	047314	8617#	8619																
ERWORD	047624	2525	2612	3224	3284	3352	3676	3730	3791	3841	3912	3958	4051	4097					
		4172	4218	4295	4340	4402	4616	5310*	5315*	5403*	5408*	5871*	5876*	6290*					
		6295*	6395	6684*	6689*	6778*	6783*	6947	6963*	6968*	7074	7169	7268	7783*					
		7789*	8846#	8882*	8902*	8906*	8928*	8948*	9415*	9422*	9426*	9448*	9456*	9462*					
		9476*	9481*	9485*	9497*	10930	10932	10969	10972	10976	10976	10959	10969	10972	10976				
ER1	001716	1320#	3103	3114	5790	6222	6315	10935	10938	10950	10959	10969	10972	10976					
		10982																	
ER2	001722	1322#	10950																
ER3	001730	1325#	10950																
EXT1 =	000001	1138#																	
EXT10 =	000010	1141#																	
EXT2 =	000002	1139#																	
EXT20 =	000020	1142#																	
EXT4 =	000004	1140#																	
EXT40 =	000040	1143#																	
FEN =	000200	1164#																	
FER =	000020	1082#	6222	6315															
FILLEC	046110	8511#																	
FIRST	002034	1376#	1474	1499*															
FMT22 =	010000	1201#	3195	4432	4651	4667	4682	4750	4766	4784	4861	4878	4896	4975					
		4991	5005	5058	5074	5091	5133	5149	5166	5207	5223	5240	5276	5296					
		5348	5368	5449	5516	5818	5838	5920	5941	6178	6271	6346	6429	6469					

PROG = 001000	1069#	2415	2585	3123	3530	5725	5745	5781	6704	6798	7585			
PRWC = 011210	1940*	1941#												
PRO = 000000	502#													
PR1 = 000040	502#													
PR2 = 000100	502#													
PR3 = 000140	502#													
PR4 = 000200	502#													
PR5 = 000240	502#													
PR6 = 000300	502#													
PR7 = 000340	502#													
PS = 177776	502#	1471*	7280*	7371*										
PSEL = 002000	1052#													
PSU = 000001	1231#													
PSW = 177776	502#													
PUTREG 042224	2636	2656	2682	2697	2745	2801	2827	2847	3102	3113	3150	5012	5304	
	5497	5618	5715	5735	5773	6054	6220	6284	6678	6772	6959	7412#	7562	
	7614	8038	8407	8413	8481									
PWRVEC= 000024	502#	1449*	1450*	10166*	10167*	10176*	10179*	10191*	10192*					
P12 045342	8285#	8339*	8340*	8369										
P22 045344	8286#	8347*	8348*	8374										
P24 045346	8287#	8355*	8356*	8375										
P3 045340	8284#	8330*	8331*	8368										
RA 000200	512#	8598												
RAW = 000020	1179#													
RCOM 054024	9628#	9633*												
RCYL 047610	8816#	8851*	8886											
RDCHR = 104410	9954	10160#												
RDHEAD 047626	8740	8851#												
RDLIN = 104411	9986	10161#												
RDOCT = 104412	1517	7387	7394	8562	8580	10162#								
RDY = 000200	1049#	1905	1908	2575	2645	2657	2905	2907	2915	2917	2939	2941	2949	
	2951	3257	3325	5956	5958	6370	6718	6812	7478	7563	7573	7615		
READ 050420	8875	8879	8887	8889	8891	8893	8895	8936	8940	8973#	9642	9650		
READAT 002066	1399#	5362	5832	5934										
READIN 002102	1405#	3180												
RECALI 002046	1391#	4246												
REDATA 054026	8798	9632#												
REFOR 002070	1400#	5079	5154	5228	6183	7026								
REGADR 042270	1605*	1909*	1920*	1929*	1996*	2018*	2299*	2470*	2527*	2614*	2711*	2718*	2763*	
	2811*	2863*	2871*	2884*	2908*	2918*	2942*	2952*	2979*	2994*	3015*	3023*	3042*	
	3050*	3226*	3286*	3354*	3582*	3678*	3732*	3793*	3843*	3914*	3960*	4053*	4099*	
	4174*	4220*	4297*	4342*	4404*	4618*	5738*	5959*	5969*	5977*	6397*	6707*	6801*	
	6949*	7076*	7171*	7270*	7439#	7443*	8098*	8109*	8119*	8130*	8201*	10928	10962	
	10966													
REGSA1 056446	1530	10022#												
REINTO 003154	1413#	2490	2497*	2511	2518	2562	2571*	2574*	2575*	2576*	2577*	2578*	2579*	
	2580*	2581*	2584	2588*	2589*	2590*	2601	2605	3187	3194*	3195*	3197*	3198*	
	3211	3217	3637	3645*	3646*	3647*	3663	3669	3696*	3697*	3698*	3699*	3700*	
	3716	3722	3753	3761*	3762*	3763*	3778	3785	3809*	3810*	3811*	3812*	3813*	
	3827	3833	3865	3872*	3873*	3874*	3899	3906	3929*	3930*	3931*	3932*	3946	
	3950	4014	4021*	4022*	4023*	4037	4044	4068*	4069*	4070*	4071*	4072*	4085	
	4089	4128	4144*	4145*	4146*	4159	4165	4189*	4190*	4191*	4192*	4204	4209	
	4251	4267*	4268*	4269*	4282	4288	4312*	4313*	4314*	4315*	4327	4332	4366	
	4375*	4376*	4389	4395	4439	4584*	4585*	4587*	4588*	4589*	4603	4609	4713	
	4714	4721	4812	4813	4816	4817	4825	4926	4927	4930	4931	4938	5070	
	5104	5145	5179	5219	5253	5342	5343	5364	5401	5448	5454	5455	5459	

		5466	5467	5543	5573	5578	5585	5586	5658	5694	5708	5812	5813	5834
		5869	6004	6009	6016	6017	6078	6092	6124	6174	6208	6353	6365*	6356*
		6367*	6368*	6369*	6370*	6383	6389	6425	6426	6443	6465	6466	6483	6509
		6510	6545	6546	6568	6569	6586	6605	6606	6881	6893*	6894*	6898*	6899*
		6902*	6903*	6904*	6905*	6906*	6914*	6916*	6918*	6919*	6920*	6934	6940	7008
		7035	7045*	7046*	7047*	7048*	7049*	7050*	7061	7067	7131	7141*	7142*	7143*
		7144*	7145*	7156	7162	7230	7240*	7241*	7242*	7243*	7244*	7255	7261	7737
		7836	8052	8077										
RELEAS	002052	1393#	4361											
RESVEC=	000010	502#												
RETCL	002076	1403#	4123											
RHAS	001656	1292#	1599	1628	5703	5772*								
RHBA	001634	1275#	1955	2280*	2547*	3033	3175*	3371*	4665*	4764*	4876*	4989*	5070*	5145*
		5219*	5292*	5364*	5834*	5936*	5972	5977	6174*	6269*	6343*	6666*	6760*	6855*
		7009*	7100*	7196*	7737*	7836*	8023*	8151*						
RHBAE	001700	1304#	1558											
RHCA	001652	1290#	2086	2287*	2554*	3178*	3378*	3622*	3631*	3748*	3991*	4431*	4683*	4785*
		4897*	5006*	5075*	5150*	5224*	5297*	5371*	5841*	5944*	6179*	6272*	6333	6347*
		6671*	6765*	6866*	6868*	7030*	7121*	7216*	7219*	7744*	7843*	8015*	8155*	8187*
RHCC	001676	1300#	3607	3991	7418	8197	8201							
RHCS1	001640	1285#	1879	2151	2282*	2485*	2494*	2549*	3181*	3191*	3373*	3606*	3641*	3747*
		3757*	3859*	3869*	4008*	4018*	4123*	4132*	4246*	4255*	4361*	4370*	4457*	5008*
		5010*	5734*	6340*	6352	6362*	6376	6397	6715	6809	7042*	7138*	7237*	7476
		7542	8148*	8161*	8189*	8191*	8557	8566	8569	8584	8723*	9238*		
RHCS2	001636	1276#	1582	1629	1680*	1771*	1863	1864*	1880*	1897*	1941*	1956*	1971*	2038*
		2055*	2071*	2087*	2099*	2281*	2548*	2568*	2691	2840	3305*	3372*	5015	5712*
		5713*	5845*	5948*	6335*	6336*	6345*	7461*	7480	7543	8153*			
RHCS3	001702	1305#												
RHDB	001630	1273#	1534	2278*	2294	2633	2689*	2690*	2704	2705	2711	2718	2739*	2753
		2763	2797*	2807	2811	2833	2853	3369*	3404	8563				
RHDST	001644	1287#	2037	2284*	2551*	3176*	3375*	3605*	4429*	4453	4473	4544	4672*	4775*
		4887*	4996*	5073*	5148*	5222*	5295*	5367*	5837*	5940*	6177*	6270*	6344*	6669*
		6763*	6858*	6906	7020*	7111*	7202*	7740*	7839*	8020*	8152*	8188*		
RHDS1	001662	1294#	3131	3248	3316	6707	6801	7544						
RHDT	001664	1295#	1681	1683	1688	1690	1693	1695	1710	1773	1775	1794	1797	1799
		1802	1804	1807	1809	1830	1832	2163	2165	2168	2170	2173	2175	
RHEC1	001670	1297#	8410											
RHEC2	001672	1298#	3244	3303	3312	8404								
RHER1	001642	1286#	1632	1899*	1970	2283*	2550*	3374*	5696	7130	7149	7171	7229	7248
		7270	7545	8094	8098	8104	8109	8116	8119	8124	8130			
RHER2	001646	1288#	2054	2285*	2552*	3376*	5698							
RHER3	001654	1291#	2098	2288*	2555*	3379*	5700							
RHLA	001674	1299#	3699	3812	3932	4072	4192	4315	4376	4472	4542			
RHMR	001660	1293#	1985	2289*	2290*	2484*	2556*	2557*	3082*	3099*	3182*	3240*	3297*	3380*
		3381*	3401*	3602*	3690*	3743*	3803*	3855*	3924*	3980*	3985*	3986*	3997*	3998*
		4004*	4063*	4114*	4119*	4120*	4137*	4138*	4184*	4236*	4243*	4244*	4260*	4261*
		4307*	4357*	4371*	4418*	4452	4628*	5007*	6361*	6413*	6453*	6493*	6529*	6634*
		6730*	6989*	7000*	7001*	7002*	7040	7092*	7093*	7094*	7136	7188*	7189*	7190*
		7235	8160*	8190*	8458	8720*	8721*	8722*	8857	9124	9235*	9236*	9237*	9384
		9552												
RHOF	001650	1289#	2070	2286*	2553*	3177*	3377*	4009*	4432*	4682*	4784*	4896*	5005*	5074*
		5149*	5223*	5296*	5368*	5838*	5941*	6178*	6271*	6346*	6670*	6764*	6859*	7029*
		7120*	7211*	7741*	7840*	8026*	8154*							
RHSN	001666	1296#	1790	1829	1831									
RHWC	001632	1274#	1940	2279*	2489	2504	2527	2546*	2561	2594	2614	3006	3174*	3186
		3204	3226	3242	3244	3263	3286	3302	3303	3310	3312	3331	3354	3370*

Symbol	Value	527#	1438	1445	1451	1452	562#	563#	562#	561#	562#	563#	564#	565#	566#	567#	568#	569#	9833	9897	9973#	9884	9975#	9905	9974#	573#	1717	2238	7380	9788	9818	9914	9973	10020	10035	10047	10064	10069			
\$CMTAG	001100																																								
\$CM1 =	000006	557#	558#	559#	560#	561#																																			
\$CM2 =	000014	557#	558#	559#	560#	561#																																			
\$CM3 =	000006	556#	557																																						
\$CM4 =	000006	563#	564#	565#	566#	567#																																			
\$CNTLC	056235	9833	9897	9973#																																					
\$CNTLG	056247	9884	9975#																																						
\$CNTLU	056242	9905	9974#																																						
\$CRLF	001223	573#	1717	2238	7380	9788	9818	9914	9973	10020	10035	10047	10064	10069																											
\$DBLK	054726	10087																																							
\$DOAGN	041416	9728	9762	9770#																																					
\$DTBL	054716	7325	7334	7340#																																					
\$ENDAD	041406	9731	9766#																																						
\$ENDCT	041354	506	7336#																																						
\$ENDMG	041425	7327#																																							
\$ENULL	041422	7296	7329	7343#																																					
\$EOP	041320	7299	7332	7342#																																					
\$EOPCT	041346	1554	1660	7303	7318#																																				
\$ERFLG	001103	7324#	7328																																						
\$ERMAX	001115	529#	7458	9668	9688	9690	9696*	9717	10022*	10047																															
\$ERROR	056436	535#	1453*	9690	9712*	9717																																			
\$ERRPC	001116	1445	1566	10020#																																					
		536#	10029*	10030*	10031	10047	10052	10928	10930	10932	10935	10938	10941	10943																											
		10946	10950	10953	10955	10956	10959	10962	10966	10969	10972	10976	10979	10982																											
		10985	10988	10993																																					
		575#	10059																																						
\$ERRTB	001226	10034	10047#																																						
\$ERRTY	056604	533#	7287	7289*	10028*	10047																																			
\$ERTTL	001112	570#	1452*	9711*	10043	10045	10047																																		
\$ESCAP	001214	554#	9792	9818																																					
\$FILLC	001156	553#	9818																																						
\$FILLS	001155	537#																																							
\$GDADR	001120	539#	1604*	1908*	1916*	1918	1928*	1992*	2016*	2209*	2216	2220	2298*	2301																											
\$GDDAT	001124	2313*	2321*	2331*	2333	2339*	2347*	2355*	2363*	2371*	2379*	2387*	2396*	2397																											
		2404*	2412*	2423*	2432*	2440*	2449*	2708*	2716*	2762*	2808*	2809	2864*	2872*																											
		2882*	2907*	2917*	2941*	2951*	2977*	2978*	2989*	3014*	3022*	3041*	3049*	3409*																											
		3418*	3427*	3439*	3441	3450*	3452*	3455	3462*	3470*	3478*	3486*	3494*	3502*																											
		3510*	3511*	3513	3519*	3527*	3537*	3546*	3554*	3562*	5313*	5406*	5739*	5874*																											
		5958*	5965*	5967	5976*	6293*	6687*	6708*	6781*	6802*	6966*	7464*	7466*	7468																											
		7473	7478*	7484*	7787*	8099*	8110*	8120*	8129*	8200*	8883*	8907*	8924*	8926*																											
		8949*	9416*	9427*	9449*	9463*	9475*	9486*	9499*	10928	10930	10932	10962	10966																											
		10969	10972	10976																																					
		7333#																																							
\$GET42	041376	9885#	10158																																						
\$GTSWR	055554	486																																							
\$HD =	000000	10008*	10019#																																						
\$HIOCT	056434	530#	9703*	9704	9706*	9716																																			
\$ICNT	001104	10166	10179	10198#																																					

\$XTSTR 054252	9675#																			
\$\$GET4= 000000	7335#																			
\$OFILL 057203	10091*	10095*	10105	10140#																
\$4OCAT= ***** U	9673	10034																		
= 070252	503#	506#	507#	508#	511#	525#	527#	575	1338#	1411#	1412#	1413#	1441							
	1454	1455	1482#	1485#	1492#	1494#	1562#	1566#	1621#	1653#	1655#	1659#	1705#							
	1710#	1714#	1786#	1820#	1824#	1828#	1843#	1845#	1851#	3883#	7287#	7342	7346							
	7383#	7392#	8576#	8592#	8598#	9184#	9185#	9186#	9187#	9188#	9189#	9191#	9192#							
	9193#	9194#	9195#	9716	9717	9770#	9818	9821#	9973#	9980	10020	10047	10090#							
	10178	10199	11056#																	

. ABS. 070252 000

ERRORS DETECTED: 0

CZRJGC,DSKW:CZRJGC.SEQ/SOL/NL:TOC:MC:ME:CND=SYSMAC.SML[400,1066],CZRJGC.P11[400,1424]

RUN-TIME: 37 42 2 SECONDS
RUN-TIME RATIO: 469/82=5.6
CORE USED: 38K (75 PAGES)

DOCUMENT PAGES: 274