

RP04/5/6

DISKLESS 2
CZRJHC0

AH-9215C-MC
COPYRIGHT © 74-78
FICHE 1 OF 2

DEC 1978
digital
MADE IN USA

This image shows a microfiche card with a grid of 14 columns and 18 rows of frames. Each frame contains a small, illegible image of a document page, likely a technical manual or report. The text within the frames is too small to be read. The card is labeled with 'RP04/5/6' in the top left, 'DISKLESS 2 CZRJHC0' in the top center, 'AH-9215C-MC COPYRIGHT © 74-78 FICHE 1 OF 2' in the top right, and 'DEC 1978 digital MADE IN USA' in the top right corner.

RP04/5/6

DISKLESS 2
CZRJHC0

AH-9215C-MC
COPYRIGHT © 74-78
FICHE 2 OF 2

DEC 1978
digital
MADE IN USA

This microfiche card contains a grid of frames. The frames are arranged in approximately 12 rows and 4 columns. Each frame contains a small amount of data, likely representing a page of text or a table. The data is too small to be legible in this image, but the overall structure is a standard microfiche layout. The card is dark blue with a lighter blue grid pattern.

.REM @

IDENTIFICATION

PRODUCT CODE: AC-9213C-MC
PRODUCT NAME: CZRJHCO RP04/5/6 DISKLESS CONTROLLER TEST-PART II
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 MASSBUS
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. PROGRAM DESCRIPTION

1.0 ABSTRACT

THIS DIAGNOSTIC TESTS THE RH11 AND DCL OF AN RP04/5/6 SUBSYSTEM. IT DOES NOT USE THE DISK SURFACE OR ANY SIGNALS FROM THE MDLI. IT REQUIRES THAT THE DCL CABLE BE PLUGGED INTO THE MDLI OR BE APPROPRIATELY TERMINATED. IF THE DISK IS POWERED UP, IT IS REQUIRED TO GET THE DISK TO THE 'HEADS UNLOADED' POSITION. AFTER A SUCCESSFUL RUN (WITH NO ERRORS) OF THIS DIAGNOSTIC IT CAN BE ASSERTED THAT, 'THAT PART OF THE DCL THAT HANDLES DATA OR DATA ASSOCIATED LOGIC IS WORKING PROPERLY'. THIS IMPLIES THAT, THAT PART OF THE LOGIC WHICH HANDLES MECHANICAL COMMANDS OR ITS ASSOCIATED LOGIC IS NOT TESTED IN THIS DIAGNOSTIC. ALL DATA COMMANDS USE THE MAINTENANCE REGISTER IN THE WRAPAROUND MODE.

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

2.0 REQUIREMENTS

2.1 EQUIPMENT

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

2.2 STORAGE

THIS PROGRAM REQUIRES 16K WORDS OF MEMORY.

2.3 PRELIMINARY PROGRAMS

THIS PROGRAM ASSUMES THAT MAINDEC-11-DZRJG- (LATEST REV) HAS BEEN RUN WITHOUT ERRORS

3.0 LOADING PROCEDURE

USE STANDARD PROCEDURE FOR LOADING .ABS TAPES

4.0 STARTING PROCEDURE

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

4.1 CONTROL SWITCH SETTINGS

SEE SECTION 5.1

4.2 STARTING ADDRESS

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

200 START

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

204 RESTART

SAME AS 200 START WITH FOLLOWING EXCEPTIONS: PROGRAM WILL QUERY OPERATOR FOR THE CORRECT CSR AND VECTOR ADDRESS OF THE RHXX CONTROLLER. WHEN THIS ACTION HAS BEEN COMPLETED, THE PROGRAM WILL AUTOMATICALLY RESTART FROM ADDRESS 200, WITH THE SAME CONVENTIONS AS DESCRIBED FOR A 200 START.

210 START

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

4.3 PROGRAM AND/OR OPERATOR ACTION

1. LOAD THE PROGRAM INTO MEMORY.
2. SET STARTING ADDRESS ON THE SWITCH REGISTER
3. PRESS 'LOAD ADDRESS'.
4. SET 'OPERATIONAL SWITCH SETTINGS' (SEE SECTION 5.1) WORST CASE IS ALL SWITCHES DOWN.
5. PRESS 'START'.
6. FOR THE FIRST PASS EACH TEST WILL BE EXECUTED ONCE ON THE DRIVES PRESENT OR DRIVE SELECTED BEFORE 'END PASS' IS PRINTED. THE FIRST PASS WILL REQUIRE OPERATOR INTERVENTION IF THE PROGRAM IS NOT RUN UNDER AN 'ACT-11' MONITOR. THE SECOND AND SUBSEQUENT PASSES WILL EXECUTE EACH TEST FOUR TIMES ON EACH DRIVES PRESENT OR DRIVE SELECTED BEFORE 'END PASS' IS PRINTED. THE SECOND AND SUBSEQUENT PASSED DO NOT NEED ANY OPERATOR INTERVENTION.

5.0 OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS

IF THE PROGRAM IS BEING RUN ON A SWITCHLESS PROCESSOR (I.E. AN 11/34) THE PROGRAM WILL DETERMINE THAT THE HARDWARE SWITCH

REGISTER IS NOT PRESENT AND WILL USE AN 'SOFTWARE' SWITCH REGISTER. THE SETTINGS OF THE 'SOFTWARE' SWITCHES ARE CONTROLLED THROUGH A KEYBOARD ROUTINE WHICH IS CALLED BY TYPING A 'CONTROL G'. THE PROGRAM WILL RECOGNIZE THE 'CONTROL G' AT ANY TIME EXCEPT WHEN THE PROGRAM IS AT A HIGHER PRIORITY PROCESSING AN RP04/5/6 INTERRUPT. THE 'SOFTWARE' SWITCH VALUES ARE ENTERED AS AN OCTAL NUMBER IN RESPONSE TO THE PROMPT FROM THE SWITCH ENTRY ROUTINE:

'SWR = NNNNNN NEW ='

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

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

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

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

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

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

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

ARE CONNECTED TO AN RHT1 CONTROLLER.

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

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

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

NOTE: ALSO SEE SECTION 8.3

SWITCH 8 - LOOP ON TEST IN SWR <7:0>
THIS IS A SPECIAL SWITCH. WHEN SET SWITCHES 0 THRU 7 HAVE ONE MEANING AND WHEN RESET SWITCHES 0 THRU 7 HAVE ANOTHER MEANING. THIS MEANS THAT ANY SETTING OF SWITCH 0 THRU 7 MUST BE DONE WITH SWITCH 8 IN THE APPROPRIATE POSITION. WHEN THIS SWITCH IS SET THEN SWITCHES 0 THRU 7 GIVE THE TEST NUMBER TO BE LOOPED ON. FOR EXAMPLE WITH SWITCH 8 SET AND SWITCH 3 SET THE PROGRAM WILL LOOP ON TEST 10. HOWEVER THIS SETTING MUST BE DONE AT THE BEGINNING OF THE PROGRAM THEN ALL THE TESTS FROM 1 TO 10 WILL BE EXECUTED AND THEN TEST 10 WILL BE REPEATED OVER AND OVER AGAIN. WHEN THIS SWITCH IS NOT SET THEN SWITCHES 0 THRU 7 HAVE THE MEANING ITS NAME INDICATES. FOR EXAMPLE SWITCH 7 IS 'STOP FURTHER COMPARES: THAT IS IF SWITCH 8 IS NOT SET AND SWITCH 7 IS SET THEN WHEN A DATA ERROR IS DETECTED NO FURTHER COMPARES WILL BE DONE. FOR EXAMPLE IN A 256 WORD BUFFER IF ALL THE WORDS ARE IN ERROR THEN AFTER SEEING THE PRINTOUT FOR THE FIRST FEW WORDS SETTING SWITCH 7 ONLY WILL STOP FURTHER PRINTOUTS OF THIS ERROR AND GO ON WITH THE TEST RATHER THAN PRINT ALL THE 256 WORDS. HOWEVER IF THIS WAS DONE WITH SWITCH 11

THEN THE NEXT ERROR THAT THE PROGRAM DETECTS IN A 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 REGISTERS OR GOOD/RECEIVED DATA. IF THE ERROR OCCURRED IN A SUBROUTINE, THE ADDRESS OF THE SUBROUTINE CALL IS ALSO GIVEN. REFER TO THE PROGRAM LISTING AT THE STATED ADDRESS TO DETERMINE THE CAUSE OF THE ERROR.

6.1 'FATAL' ERRORS

IN THE EVENT THAT THE DISK DRIVE BECOMES UNAVAILABLE TO THE

CONTROLLER, POWERS DOWN, OR CERTAIN CRITICAL STATUS BITS CANNOT BE CLEARED PRIOR TO THE START OF A TEST SEQUENCE - THIS INFORMATION WILL BE COMMUNICATED TO THE OPERATOR. IN ADDITION, THE TTY BELL WILL RING AND THE PROGRAM WILL HALT. IT IS SUGGESTED THAT IF THIS HAPPENS THE OPERATOR LOAD ADDRESS 200 (210) AND RESTART THE PROGRAM AS A FIRST ATTEMPT TO SOLVE THE PROBLEM. IF THE FAILURE CONTINUES TO OCCUR, THERE ARE TWO OPTIONS FOR THE OPERATOR:

1. LOOK IN THE TEST LISTING FOR THE 'HALT' INSTRUCTION AND REPLACE IT PLUS THE TWO WORDS ('TYPE ,CPHALT') ABOVE WITH 'NOP'S. WITH TTY ERROR PRINTOUTS INHIBITED, A SCOPE LOOP CAN BE INITIATED FOR THE TEST IN QUESTION.
2. GO BACK AND RERUN DZRPS AS IT IS QUITE POSSIBLE THAT A HARD FAILURE HAS OCCURRED IN ONE OF THE HARDWARE REGISTERS.

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

7.0 RESTRICTIONS

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

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

8.0 MISCELLANEOUS

8.1 EXECUTION TIME

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

8.2 STACK POINTER

THE STACK IS INITIALLY SET TO 1000

8.3 OPERATOR SELECTABLE SCOPE LOOPS

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

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

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

8.4 PROGRAM REVISION HISTORY

9.0 PROGRAM DESCRIPTION

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

@

522	001100	000000	\$PASS:	.WORD	0	::CONTAINS	PASS COUNT
523	001102	000	\$TSTNM:	.BYTE	0	::CONTAINS	THE TEST NUMBER
524	001103	000	\$ERFLG:	.BYTE	0	::CONTAINS	ERROR FLAG
525	001104	000000	\$ICNT:	.WORD	0	::CONTAINS	SUBTEST ITERATION COUNT
526	001106	000000	\$LPADR:	.WORD	0	::CONTAINS	SCOPE LOOP ADDRESS
527	001110	000000	\$LPERR:	.WORD	0	::CONTAINS	SCOPE RETURN FOR ERRORS
528	001112	000000	\$ERTTL:	.WORD	0	::CONTAINS	TOTAL ERRORS DETECTED
529	001114	000	\$ITEMB:	.BYTE	0	::CONTAINS	ITEM CONTROL BYTE
530	001115	001	\$ERMAX:	.BYTE	1	::CONTAINS	MAX. ERRORS PER TEST
531	001116	000000	\$ERRPC:	.WORD	0	::CONTAINS	PC OF LAST ERROR INSTRUCTION
532	001120	000000	\$GDADR:	.WORD	0	::CONTAINS	ADDRESS OF 'GOOD' DATA
533	001122	000000	\$BDADR:	.WORD	0	::CONTAINS	ADDRESS OF 'BAD' DATA
534	001124	000000	\$GDDAT:	.WORD	0	::CONTAINS	'GOOD' DATA
535	001126	000000	\$BDDAT:	.WORD	0	::CONTAINS	'BAD' DATA
536	001130	000000		.WORD	0	::RESERVED--NOT TO BE USED	
537	001132	000000		.WORD	0		
538	001134	000	\$AUTOB:	.BYTE	0	::AUTOMATIC MODE INDICATOR	
539	001135	000	\$INTAG:	.BYTE	0	::INTERRUPT MODE INDICATOR	
540	001136	000000		.WORD	0		
541	001140	177570	\$SWR:	.WORD	DSWR	::ADDRESS OF SWITCH REGISTER	
542	001142	177570	\$DISPLAY:	.WORD	DDISP	::ADDRESS OF DISPLAY REGISTER	
543	001144	177560	\$TKS:	177560		::TTY KBD STATUS	
544	001146	177562	\$TKB:	177562		::TTY KBD BUFFER	
545	001150	177564	\$TPS:	177564		::TTY PRINTER STATUS REG. ADDRESS	
546	001152	177566	\$TPB:	177566		::TTY PRINTER BUFFER REG. ADDRESS	
547	001154	000	\$NULL:	.BYTE	0	::CONTAINS NULL CHARACTER FOR FILLS	
548	001155	002	\$FILLS:	.BYTE	2	::CONTAINS # OF FILLER CHARACTERS REQUIRED	
549	001156	012	\$FILLC:	.BYTE	12	::INSERT FILL CHARS. AFTER A 'LINE FEED'	
550	001157	000	\$TPFLG:	.BYTE	0	::'TERMINAL AVAILABLE' FLAG (BIT<07>=0=YES)	
551	001160	000000	\$REGAD:	.WORD	0	::CONTAINS THE ADDRESS FROM	
552	001162	000000	\$REG0:	.WORD	0	::CONTAINS ((\$REGAD)+0)	
553	001164	000000	\$REG1:	.WORD	0	::CONTAINS ((\$REGAD)+2)	
554	001166	000000	\$REG2:	.WORD	0	::CONTAINS ((\$REGAD)+4)	
555	001170	000000	\$REG3:	.WORD	0	::CONTAINS ((\$REGAD)+6)	
556	001172	000000	\$REG4:	.WORD	0	::CONTAINS ((\$REGAD)+10)	
557	001174	000000	\$REG5:	.WORD	0	::CONTAINS ((\$REGAD)+12)	
558	001176	000000	\$TMP0:	.WORD	0	::USER DEFINED	
559	001200	000000	\$TMP1:	.WORD	0	::USER DEFINED	
560	001202	000000	\$TMP2:	.WORD	0	::USER DEFINED	
561	001204	000000	\$TMP3:	.WORD	0	::USER DEFINED	
562	001206	000000	\$TMP4:	.WORD	0	::USER DEFINED	
563	001210	000000	\$TMP5:	.WORD	0	::USER DEFINED	
564	001212	000000	\$TIMES:	0		::MAX. NUMBER OF ITERATIONS	
565	001214	000000	\$ESCAPE:	0		::ESCAPE ON ERROR ADDRESS	
566	001216	177607	\$BELL:	.ASCIZ	<207><377><377>	::CODE FOR BELL	
567	001222	077	\$QUES:	.ASCII	/?/	::QUESTION MARK	
568	001223	015	\$CRLF:	.ASCII	<15>	::CARRIAGE RETURN	
569	001224	000012	\$LF:	.ASCIZ	<12>	::LINE FEED	

000377

570					
571					
572					
573					
574					
575			:ITEM1		
576	001226	002136	EM1		:WRONG DATA IN READING OR WRITING HARDWARE REGISTER
577	001230	005451	DH1		:PC
578					:REG. ADDR.
579					:GOOD DATA
580					:RECEIVED DATA
581	001232	011764	DT1		:\$ERRPC,\$STSTM,REGADR,\$GDDAT,\$BDDAT
582	001234	012516	DF1		:0,0,0,0,0
583					
584					
585					
586					
587					
588			:ITEM2		
589	001236	002221	EM2		:ERROR ON DATA COMMAND
590					
591	001240	010463	DH33		:PC
592					:PC OF JSR
593					:TEST NO
594					:WORD NO.
595					:GOOD DATA
596					:CONTENTS OF RHCS1
597					:CONTENTS OF RHDS1
598					:CONTENTS OF RHER1
599	001242	012350	DT33		:\$ERRPC,PCJSR,\$STSTM,ERWORD,\$GDDAT,CS1,DS1,ER1
600	001244	012666	DF33		:0,0,0,1,0,0,0,0
601					
602					
603			:ITEM3		
604	001246	002221	EM2		:ERROR ON DATA COMMAND
605					
606	001250	010246	DH32		:PC
607					:PC OF JSR
608					:TEST NO
609					:WORD NO.
610					:GOOD DATA
611					:BAD DATA
612					:CONTENTS OF RHCS1
613					:CONTENTS OF RHDS1
614					:CONTENTS OF RHER1
615					
616	001252	012324	DT32		:\$ERRPC,PCJSR,\$STSTM,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1
617	001254	012655	DF32		:0,0,0,1,0,0,0,0,0,
618					
619					
620			:ITEM4		
621	001256	002221	EM2		:ERROR ON DATA COMMAND
622					
623	001260	010050	DH31		:PC
624					:TEST NO
625					:WORD NO.

626				:GOOD DATA
627				:BAD DATA
628				:CONTENTS OF RHCS1
629				:CONTENTS OF RHDS1
630				:CONTENTS OF RHER1
631				
632	001262	012302	DT31	:\$ERRPC,\$STSTNM,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1
633	001264	012645	DF31	:0,0,1,0,0,0,0,0,
634				
635				
636				
637			:ITEM5	
638	001266	000000	0	:
639	001270	000000	0	:
640	001272	012302	DT31	:\$ERRPC,\$STSTNM,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1
641	001274	012645	DF31	:0,0,1,0,0,0,0,0,
642				
643				
644			:ITEM6	
645	001276	002250	EM6	:ERROR ON WRITE HEADER AND DATA
646				
647	001300	010246	DH32	:PC
648				:PC OF JSR
649				:TEST iW
650				:WORD NO.
651				:GOOD DATA
652				:BAD DATA
653				:CONTENTS OF RHCS1
654				:CONTENTS OF RHDS1
655				:CONTENTS OF RHER1
656				
657	001302	012324	DT32	:\$ERRPC,PCJSR,\$STSTNM,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1
658	001304	012655	DF32	:0,0,0,1,0,0,0,0,0,
659				
660				
661				
662			:ITEM7	
663	001306	002250	EM6	:ERROR ON WRITE HEADER AND DATA
664	001310	005567	DH2	:PC
665				:TEST NO
666				:WORD NO.
667				:GOOD DATA
668				:BAD DATA
669	001312	012012	DT3	:\$ERRPC,\$STSTNM,ERWORD,\$GDDAT,\$BDDAT
670	001314	012527	DF3	:0,0,1,0,0,
671				
672				
673			:ITEM10	
674	001316	000000	0	:
675	001320	000000	0	:
676	001322	012012	DT3	:\$ERRPC,\$STSTNM,ERWORD,\$GDDAT,\$BDDAT
677	001324	012527	DF3	:0,0,1,0,0,
678				
679				
680			:ITEM11	
681	001326	002307	EM11	:CONTROLLER OR DRIVE STATUS

682	001330	005701	DH11	:PC
683				:TEST NO
684				:FAILING REG. ADDR
685				:CONTENTS OF RHCS1
686				:CONTENTS OF RHCS2
687				:CONTENTS OF RHDS1
688				:CONTENTS OF RHER1
689	001332	012026	DT11	:\$ERRPC,\$TSTNM,\$BDADR,CS1,CS2,DS1,ER1
690	001334	012534	DF11	:0,0,0,0,0,0
691				
692				
693			:ITEM12	
694	001336	002307	EM11	:WRONG DATA FROM SILO
695				
696	001340	005451	DH1	:PC
697				:REG.ADDR
698				:GOOD DATA
699				:RECEIVED DATA
700	001342	011764	DT1	:\$ERRPC,REGADR,\$GDDAT,\$BDDAT
701	001344	012516	DF1	:0,0,0,0
702				
703				
704			:ITEM13	
705	001346	000000	0	
706	001350	000000	0	
707	001352	011764	DT1	:\$ERRPC,TSTNM,REGADR,\$GDDAT,\$BDDAT
708	001354	012516	DF1	:0,0,0,0,0
709				
710				
711			:ITEM14	
712	001356	002342	EM14	:REGISTER FAILED
713	001360	006056	DH14	:PC
714				:FAILING REG. ADDR
715				:CONTENTS OF FAILING REG.
716				:CONTENTS OF RHCS1
717				:CONTENTS OF RHCS2
718				:CONTENTS OF RHDS1
719				:CONTENTS OF RHER1
720	001362	012046	DT14	:\$ERRPC,\$BDADR,\$BDDAT,CS1,CS2,DS1,ER1
721	001364	012543	DF14	:0,0,0,0,0,0,0
722				
723				
724			:ITEM15	
725	001366	002362	EM15	:SPECIFIED REG. NON EXISTANT SO ABORT
726				:PROGRAM
727	001370	006255	DH15	:PC
728				:ADDR. OF REG
729	001372	012070	DT15	:\$ERRPC,TEMP1
730	001374	012553	DF15	:0,0
731				
732				
733			:ITEM16	
734	001376	002432	EM16	:WAIT LOOP FAILED
735	001400	006305	DH16	:PC
736				:WAT PC
737				:BIT WANTED

738				:REG. ADR.
739				:REG. CONT.
740	001402	012100	DT16	:\$ERRPC,\$TMP3,\$TMP1,\$TMP0,\$BDDAT
741	001404	012556	DF16	:0,0,0,0
742				
743				
744			:ITEM17	
745	001406	002453	EM17	:WRITE CHECK FAILING
746	001410	006443	DH17	:PC
747				:TEST NO
748				:CONTENTS OF RHBA
749				:CONTENTS OF RHDB
750				:CONTENTS OF RHWC
751				:CONTENTS OF RHCS1
752				:CONTENTS OF RHCS2
753	001412	012116	DT17	:\$ERRPC,\$TSTNM,\$BA,DB,WC,CS1,CS2
754	001414	012563	DF17	:0,0,0,0,0,0,0
755				
756				
757			:ITEM20	
758	001416	002477	EM20	:REGISTER FAILING
759	001420	006620	DH20	:PC
760				:TEST NO
761				:CONTENTS OF RHER1
762				:CONTENTS OF RHER2
763				:CONTENTS OF RHER3
764				:CONTENTS OF RHAS
765				:CONTENTS OF RHDS1
766	001422	012136	DT20	:\$ERRPC,\$TSTNM,ER1,ER2,ER3,AS,DS1
767	001424	012572	DF20	:0,0,0,0,0,0,0
768				
769			:ITEM21	
770				
771	001426	002520	EM21	:INTERRUPT FAILING
772	001430	006774	DH21	:PC
773				:TEST NO
774				:CONTENTS OF RHCS1
775				:CONTENTS OF RHAS
776				:CONTENTS OF RHDS1
777	001432	012156	DT21	:\$ERRPC,\$TSTNM,CS1,AS,DS1
778	001434	012601	DF21	:0,0,0,0,0
779				
780				
781			:ITEM22	
782	001436	002542	EM22	:MISMATCH IN DRIVE PRESENT
783				:LOOKING AT RHAS AND RHCS2-NED(BIT#12)
784				:DRIVE PRESENT DO NOT AGREE
785				:NOTE: ON DUAL PORT SYSTEM
786				:DRIVE ON OTHER PORT WILL NOT GIVE NED
787				:HENCE THERE WILL BE A MISMATCH
788				:177777-MEANS NOT PRESENT
789	001440	007110	DH22	:PC
790				:TEST NO
791				:RHAS UNIT
792				:RHCS2 UNIT
793				:

794	001442	012172	DT22		;\$ERRPC,TSTNMS,\$GDDAT,\$BDDAT
795	001444	012606	DF22		:0,0,0,0
796					
797					
798				:ITEM23	
799	001446	000000	0		:MISSMATCH IN DRIVE PRESENT
800					:LOOKING AT RHAS AND RHCS2-NED(BIT#12)
801					:DRIVE PRESENT DO NOT AGREE
802					:177777-MEANS NOT PRESENT
803	001450	000000	0		:PC
804					:TEST NO
805					:RHAS UNIT
806					:RHCS2 UNIT
807					:
808	001452	012172	DT22		;\$ERRPC,TSTNMS,\$GDDAT,\$BDDAT
809	001454	012606	DF22		:0,0,0,0
810					
811					
812					
813				:ITEM 24	
814	001456	003143	EM24		:LOOK AHEAD REGISTER AT THE
815					:BEGINNING OF A SECTOR IS IN
816					:ERROR
817	001460	007204	DH24		:PC
818					:RHDST
819					:BAD RHLA
820					:GOOD RHLA
821					:SECTOR NO
822					:SECTOR CLOCK
823	001462	012204	DT24		;\$ERRPC,DST,\$BDDAT,\$TMP1,\$TMP2,\$TMP3
824	001464	012612	DF24		:0,0,0,0,0
825					
826				:ITEM 25	
827	001466	003236	EM25		:LOOK AHEAD REGISTER IS
828					:IN ERROR
829					
830	001470	007204	DH24		:PC
831					:RHDST
832					:BAD RHLA
833					:GOOD RHLA
834					:SECTOR NO
835					:SECTOR CLOCK
836	001472	012204	DT24		;\$ERRPC,DST,\$BDDAT,\$TMP1,\$TMP2,\$TMP3
837	001474	012612	DF24		:0,0,0,0,0
838				:ITEM26	
839	001476	002307	EM11		:CONTROLLER OR DRIVE STATUS
840					
841	001500	007362	DH26		:PC
842					:PC OF JSR
843					:FAILING REGISTER ADDRESS
844					:CONTENTS OF RHCS1
845					:CONTENTS OF RHCS2
846					:CONTENTS OF RHDS1
847					:CONTENTS OF RHER1
848					
849	001502	012224	DT26		;\$ERRPC,PCJSR,\$BDADR,CS1,CS2,DS1,ER1

850	001504	012621	DF26	:0,0,0,0,0,0,
851				
852				
853				
854			:ITEM27	
855	001506	002136	EM1	:ERROR IN READING OR WRITING HARDWARE REGISTER
856				
857	001510	007560	DH27	:PC
858				:PC OF JSR
859				:TEST NUMBER
860				:FAILING REGISTER
861				:GOOD DATA
862				:RECEIVED DATA
863				
864	001512	012246	DT27	:\$ERRPC,PCJSR,TSTNM,REGADR,\$GDDAT,\$BDDAT
865	001514	012631	DF27	:0,0,0,0,0,0
866				
867				
868				
869			:ITEM30	
870	001516	003276	EM30	:CURRENT CYLINDER DOES NOT REFLECT DESIRED CYLINDER REG.
871	001520	007716	DH30	:PC
872				:PC OF JSR
873				:REGISTER ADDRESS
874				:GOOD DATA
875				:BAD DATA
876				
877	001522	012264	DT30	:\$ERRPC,PCJSR,REGADR,\$GDDAT,\$BDDAT
878	001524	012637	DF30	:0,0,0,0,0
879				
880				
881				
882			:ITEM31	
883	001526	003420	EM31	:ECC GENERATED IS INCORRECT
884				:EVERY WORD IN THIS SECTOR IS GIVEN IN 'DATA USED'
885				
886	001530	010662	DH34	:PC
887				:TEST NUMBER
888				:GOOD ECC1
889				:GOOD EC2C
890				:WRITTEN ECC1
891				:WRITTEN ECC2
892				:DATA USED
893				
894	001532	012372	DT34	:\$ERRPC,TSTNM,GECC1,GECC2,WECC1,WECC2,DISK
895				
896	001534	012676	DF34	:0,0,0,0,0,0,0
897				
898				
899			:ITEM32	
900	001536	003543	EM32	:ON READ COMMAND AFTER DATA AND ECC HAVE BEEN READ
901				:ECC REGISTER OR RHER1 IS IN ERROR
902				:ONLY LOWER 11 BITS OF PATTERN REGISTER
903				:CAN BE READ
904				:THIS SHUOLD MATCH LOWER 11 BITS OF ECC1
905				

1200						
1201						*****
1202						*****
1203						ERROR AND MESSAGE TABLE CONDIMENTS
1204						*****
1205						*****
1206						
1207						
1208						
1209						
1210	002136	051127	047117	020107	EM1:	.ASCIZ /WRONG DATA IN READING OR WRITING HARDWARE REGISTER/
1211	002144	040504	040524	044440		
1212	002152	020116	042522	042101		
1213	002160	047111	020107	051117		
1214	002166	053440	044522	044524		
1215	002174	043516	044040	051101		
1216	002202	053504	051101	020105		
1217	002210	042522	044507	052123		
1218	002216	051105	000			
1219	002221	105	051122	051117	EM2:	.ASCIZ /ERROR ON DATA COMMAND/
1220	002226	047440	020116	042040		
1221	002234	052101	020101	047503		
1222	002242	046515	047101	000104		
1223	002250	051105	047522	020122	EM6:	.ASCIZ /ERROR ON WRITE HEADER AND DATA/
1224	002256	047117	053440	044522		
1225	002264	042524	044040	040505		
1226	002272	042504	020122	047101		
1227	002300	020104	040504	040524		
1228	002306	000				
1229	002307	103	047117	051124	EM11:	.ASCIZ /CONTROLLER OR DRIVE STATUS/
1230	002314	046117	042514	020122		
1231	002322	051117	042040	044522		
1232	002330	042526	051440	040524		
1233	002336	052524	000123			
1234	002342	042522	044507	052123	EM14:	.ASCIZ /REGISTER FAILED/
1235	002350	051105	043040	044501		
1236	002356	042514	000104			
1237	002362	047516	020116	054105	EM15:	.ASCIZ /NON EXISTENT REGISTER, PROGRAM ABORTED./
1238	002370	051511	042524	052116		
1239	002376	051040	043505	051511		
1240	002404	042524	026122	050040		
1241	002412	047522	051107	046501		
1242	002420	040440	047502	052122		
1243	002426	042105	000056			
1244	002432	040527	052111	046040	EM16:	.ASCIZ /WAIT LOOP FAILED/
1245	002440	047517	020120	040506		
1246	002446	046111	042105	000		
1247	002453	127	044522	042524	EM17:	.ASCIZ /WRITE CHECK FAILING/
1248	002460	041440	042510	045503		
1249	002466	043040	044501	044514		
1250	002474	043516	000			
1251	002477	122	043505	051511	EM20:	.ASCIZ /REGISTER FAILING/
1252	002504	042524	020122	040506		
1253	002512	046111	047111	000107		
1254	002520	047111	042524	051122	EM21:	.ASCIZ /INTERRUPT FAILING/
1255	002526	050125	020124	040506		

1256	002534	046111	047111	000107	
1257	002542	051105	047522	020122	EM22: .ASCII /ERROR ON DRIVE PRESENT/<15><12>
1258	002550	047117	042040	044522	
1259	002556	042526	050040	042522	
1260	002564	042523	052116	005015	
1261	002572	044124	020105	047125	.ASCII /THE UNIT NO'S FOUND BY SETTING RHAS/<15><12>
1262	002600	052111	047040	023517	
1263	002606	020123	047506	047125	
1264	002614	020104	054502	051440	
1265	002622	052105	044524	043516	
1266	002630	051040	040510	006523	
1267	002636	012			
1268	002637	104	020117	047516	.ASCII /DO NOT AGREE WITH THE UNIT NO. FOUND FROM/<15><12>
1269	002644	020124	043501	042522	
1270	002652	020105	044527	044124	
1271	002660	052040	042510	052440	
1272	002666	044516	020124	047516	
1273	002674	020056	047506	047125	
1274	002702	020104	051106	046517	
1275	002710	005015			
1276	002712	044122	051503	026462	.ASCII /RHCS2-'NED' BIT #12/<15><12>
1277	002720	047047	042105	020047	
1278	002726	044502	020124	030443	
1279	002734	006462	012		
1280	002737	061	033467	033467	.ASCII /177777-MEANS NO UNIT FOUND/<15><12>
1281	002744	026467	042515	047101	
1282	002752	020123	047516	052440	
1283	002760	044516	020124	047506	
1284	002766	047125	006504	012	
1285	002773	116	052117	035105	.ASCII /NOTE: ON DUAL PORT SYSTEM, DRIVE ON OTHER PORT WILL NOT GIVE/<15><12>
1286	003000	047440	020116	052504	
1287	003006	046101	050040	051117	
1288	003014	020124	054523	052123	
1289	003022	046505	020054	051104	
1290	003030	053111	020105	047117	
1291	003036	047440	044124	051105	
1292	003044	050040	051117	020124	
1293	003052	044527	046114	047040	
1294	003060	052117	043440	053111	
1295	003066	006505	012		
1296	003071	047	042516	023504	.ASCIIZ /'NED', HENCE THERE WILL BE AN EXTRA DRIVE/
1297	003076	020054	042510	041516	
1298	003104	020105	044124	051105	
1299	003112	020105	044527	046114	
1300	003120	041040	020105	047101	
1301	003126	042440	052130	040522	
1302	003134	042040	044522	042526	
1303	003142	000			
1304	003143	114	047517	020113	EM24: .ASCIIZ /LOOK AHEAD REGISTER AT THE BEGINNING OF SECTOR IS IN ERROR/
1305	003150	044101	040505	020104	
1306	003156	042522	044507	052123	
1307	003164	051105	040440	020124	
1308	003172	044124	020105	042502	
1309	003200	044507	047116	047111	
1310	003206	020107	043117	051440	
1311	003214	041505	047524	020122	

1312	003222	051511	044440	020116	
1313	003230	051105	047522	000122	
1314	003236	047514	045517	040440	EM25: .ASCIZ /LOOK AHEAD REGISTER IS IN ERROR/
1315	003244	042510	042101	051040	
1316	003252	043505	051511	042524	
1317	003260	020122	051511	044440	
1318	003266	020116	051105	047522	
1319	003274	000122			
1320	003276	052503	051122	047105	EM30: .ASCII /CURRENT CYLINDER DOES NOT MATCH DESIRED CYLINDER REGISTER/<15><12>
1321	003304	020124	054503	044514	
1322	003312	042116	051105	042040	
1323	003320	042517	020123	047516	
1324	003326	020124	040515	041524	
1325	003334	020110	042504	044523	
1326	003342	042522	020104	054503	
1327	003350	044514	042116	051105	
1328	003356	051040	043505	044111	
1329	003364	052123	051105	005015	
1330	003372	043101	042524	020122	.ASCIZ /AFTER A SEEK AND INIT/
1331	003400	020101	042523	045505	
1332	003406	040440	042116	044440	
1333	003414	044516	000124		
1334	003420	041505	020103	042507	EM31: .ASCII /ECC GENERATED IS INCORRECT/<15><12>
1335	003426	042516	040522	042524	
1336	003434	020104	051511	044440	
1337	003442	041516	051117	042522	
1338	003450	052103	005015		
1339	003454	053105	051105	020131	.ASCIZ /EVERY WORD ON THIS SECTOR IS THAT GIVEN IN 'DATA USED'/
1340	003462	047527	042122	047440	
1341	003470	020116	044124	051511	
1342	003476	051440	041505	047524	
1343	003504	020122	051511	052040	
1344	003512	040510	020124	044507	
1345	003520	042526	020116	047111	
1346	003526	021040	040504	040524	
1347	003534	052440	042523	021104	
1348	003542	000			
1349	003543	117	020116	042522	EM32: .ASCII /ON READ COMMAND, AFTER DATA AND ECC HAVE BEEN READ,/<15><12>
1350	003550	042101	041440	046517	
1351	003556	040515	042116	020054	
1352	003564	043101	042524	020122	
1353	003572	040504	040524	040440	
1354	003600	042116	042440	041503	
1355	003606	044040	053101	020105	
1356	003614	042502	047105	051040	
1357	003622	040505	026104	005015	
1358	003630	041505	020103	042522	.ASCII /ECC REGISTERS OR RHER1 ARE IN ERROR/<15><12>
1359	003636	044507	052123	051105	
1360	003644	020123	051117	051040	
1361	003652	042510	030522	040440	
1362	003660	042522	044440	020116	
1363	003666	051105	047522	006522	
1364	003674	012			
1365	003675	117	046116	020131	.ASCII /ONLY LOWER 11 BITS OF PATTERN REG. CAN BE READ/<15><12>
1366	003702	047514	042527	020122	
1367	003710	030461	041040	052111	

1368	003716	020123	043117	050040	
1369	003724	052101	042524	047122	
1370	003732	051040	043505	020056	
1371	003740	040503	020116	042502	
1372	003746	051040	040505	006504	
1373	003754	012			
1374	003755	124	044510	020123	.ASCIZ /THIS SHOULD MATCH LOWER 11 BITS OF GOOD ECC1/
1375	003762	044123	052517	042114	
1376	003770	046440	052101	044103	
1377	003776	046040	053517	051105	
1378	004004	030440	020061	044502	
1379	004012	051524	047440	020106	
1380	004020	047507	042117	042440	
1381	004026	041503	000061		
1382	004032	044510	044107	041440	EM33: .ASCIZ /HIGH COUNT BIT NOT SET AFTER 38859 CLOCKS/
1383	004040	052517	052116	041040	
1384	004046	052111	047040	052117	
1385	004054	051440	052105	040440	
1386	004062	052106	051105	031440	
1387	004070	034070	034465	041440	
1388	004076	047514	045503	000123	
1389	004104	042532	047522	042040	EM34: .ASCIZ /ZERO DETECT BIT NOT HIGH WHEN 32 BIT ECC REG. HAS 21 ZEROS/
1390	004112	052105	041505	020124	
1391	004120	044502	020124	047516	
1392	004126	020124	044510	044107	
1393	004134	053440	042510	020116	
1394	004142	031063	041040	052111	
1395	004150	042440	041503	051040	
1396	004156	043505	020056	040510	
1397	004164	020123	030462	055040	
1398	004172	051105	051517	000	
1399	004177	120	051517	052111	EM35: .ASCII /POSITION REGISTER OR 11 BITS OF PATTERN REGISTER INCORRECT/<15><12>
1400	004204	047511	020116	042522	
1401	004212	044507	052123	051105	
1402	004220	047440	020122	030461	
1403	004226	041040	052111	020123	
1404	004234	043117	050040	052101	
1405	004242	042524	047122	051040	
1406	004250	043505	051511	042524	
1407	004256	020122	047111	047503	
1408	004264	051122	041505	006524	
1409	004272	012			
1410	004273	114	053517	051105	.ASCII /LOWER 11 BITS OF PATTERN REGISTER SHOULD MATCH LOWER/<15><12>
1411	004300	030440	020061	044502	
1412	004306	051524	047440	020106	
1413	004314	040520	052124	051105	
1414	004322	020116	042522	044507	
1415	004330	052123	051105	051440	
1416	004336	047510	046125	020104	
1417	004344	040515	041524	020110	
1418	004352	047514	042527	006522	
1419	004360	012			
1420	004361	061	020061	044502	.ASCII /11 BITS OF GOOD ECC1/<15><12>
1421	004366	051524	047440	020106	
1422	004374	047507	042117	042440	
1423	004402	041503	006461	012	

1424	004407	104	052101	042440		.ASCIZ /DAT ENVELOP GOOD POSITION AND N-CODE ZEROS ARE IN OCTAL/
1425	004414	053116	047514	020120		
1426	004422	047507	042117	050040		
1427	004430	051517	052111	047511		
1428	004436	020116	047101	020104		
1429	004444	026516	047503	042504		
1430	004452	055040	051105	051517		
1431	004460	040440	042522	044440		
1432	004466	020116	041517	040524		
1433	004474	000114				
1434	004476	047117	051040	040505	EM36:	.ASCII /ON READ COMMAND WITH NON-CORRECTABLE ERROR DCK AND ECH SHOULD BE SET/<1
1435	004504	020104	047503	046515		
1436	004512	047101	020104	044527		
1437	004520	044124	047040	047117		
1438	004526	041455	051117	042522		
1439	004534	052103	041101	042514		
1440	004542	042440	051122	051117		
1441	004550	042040	045503	040440		
1442	004556	042116	042440	044103		
1443	004564	051440	047510	046125		
1444	004572	020104	042502	051440		
1445	004600	052105	005015			
1446	004604	043111	050040	051517		.ASCIZ /IF POSITION REGISTER -10040 OR 10041 IT IS GOOD/
1447	004612	052111	047511	020116		
1448	004620	042522	044507	052123		
1449	004626	051105	036440	030061		
1450	004634	032060	020060	051117		
1451	004642	030440	030060	030464		
1452	004650	044440	020124	051511		
1453	004656	043440	047517	000104		
1454	004664	051127	052111	047111	EM37:	.ASCIZ /WRITING WITH BUS ADDRESS HIGHER THAN 28K CAUSED ERROR/
1455	004672	020107	044527	044124		
1456	004700	041040	051525	040440		
1457	004706	042104	042522	051523		
1458	004714	044040	043511	042510		
1459	004722	020122	044124	047101		
1460	004730	031040	045470	041440		
1461	004736	052501	042523	020104		
1462	004744	051105	047522	000122		
1463	004752	044124	051105	020105	EM40:	.ASCII /THERE WAS A READ-WRITE HEADER & DATA ERROR DURING 'DTE'<15><12>
1464	004760	040527	020123	020101		
1465	004766	042522	042101	053455		
1466	004774	044522	042524	044040		
1467	005002	040505	042504	020122		
1468	005010	020046	040504	040524		
1469	005016	042440	051122	051117		
1470	005024	042040	051125	047111		
1471	005032	020107	042047	042524		
1472	005040	006447	012			
1473	005043	124	051505	020124		.ASCIZ /TEST SETUP - THE TEST WAS ABORTED AT THAT POINT/
1474	005050	042523	052524	020120		
1475	005056	020055	044124	020105		
1476	005064	042524	052123	053440		
1477	005072	051501	040440	047502		
1478	005100	052122	042105	040440		
1479	005106	020124	044124	052101		

1480 005114 050040 044517 052116
1481 005122 000

1594	006270	020124	020040	051040																	
1595	006276	043505	040440	051104																	
1596	006304	000																			
1597	006305	120	020103	020040	DH16:	.ASCII	/PC	TEST	WAT	BIT	REG	REG/<15><12>									
1598	006312	020040	052040	051505																	
1599	006320	020124	020040	053440																	
1600	006326	052101	020040	020040																	
1601	006334	041040	052111	020040																	
1602	006342	020040	051040	043505																	
1603	006350	020040	020040	051040																	
1604	006356	043505	005015																		
1605	006362	020040	020040	020040		.ASCIZ	/	NO	PC	WANTED	ADDRESS	CONTENTS/									
1606	006370	020040	047516	020040																	
1607	006376	020040	020040	041520																	
1608	006404	020040	020040	020040																	
1609	006412	040527	052116	042105																	
1610	006420	020040	042101	051104																	
1611	006426	051505	020123	047503																	
1612	006434	052116	047105	051524																	
1613	006442	000																			
1614	006443	120	020103	020040	DH17:	.ASCII	/PC	TEST	CONT.	CONT.	CONT.	CONT.	CONT./<15><12>								
1615	006450	020040	052040	051505																	
1616	006456	020124	020040	041440																	
1617	006464	047117	027124	020040																	
1618	006472	041440	047117	027124																	
1619	006500	020040	041440	047117																	
1620	006506	027124	020040	041440																	
1621	006514	047117	027124	020040																	
1622	006522	041440	047117	027124																	
1623	006530	005015																			
1624	006532	020040	020040	020040		.ASCIZ	/	NO	RHBA	RHDB	RHWC	RHCS1	RHCS2/								
1625	006540	020040	047516	020040																	
1626	006546	020040	020040	044122																	
1627	006554	040502	020040	020040																	
1628	006562	044122	041104	020040																	
1629	006570	020040	044122	041527																	
1630	006576	020040	020040	044122																	
1631	006604	051503	020061	020040																	
1632	006612	044122	051503	000062																	
1633	006620	041520	020040	020040	DH20:	.ASCII	/PC	TEST	CONT	CONT	CONT	CONT	CONT/<15><12>								
1634	006626	020040	042524	052123																	
1635	006634	020040	020040	047503																	
1636	006642	052116	020040	020040																	
1637	006650	047503	052116	020040																	
1638	006656	020040	047503	052116																	
1639	006664	020040	020040	047503																	
1640	006672	052116	020040	020040																	
1641	006700	047503	052116	005015																	
1642	006706	020040	020040	020040		.ASCIZ	/	NO	RHER1	RHER2	RHER3	RHAS	RHDS1/								
1643	006714	020040	047516	020040																	
1644	006722	020040	020040	044122																	
1645	006730	051105	020061	020040																	
1646	006736	044122	051105	020062																	
1647	006744	020040	044122	051105																	
1648	006752	020063	020040	044122																	
1649	006760	051501	020040	020040																	

CZR
CZR
2
2
2
2

1874	011364	043505	000056																
1875	011370	041520	020040	020040	DH37:	.ASCII	/PC	TEST	POSITON	POSITON	GOOD	GOOD	PATTERN	DATA	N-CODE/				
1876	011376	020040	042524	052123															
1877	011404	020040	020040	047520															
1878	011412	044523	047524	020116															
1879	011420	047520	044523	047524															
1880	011426	020116	047507	042117															
1881	011434	020040	020040	047507															
1882	011442	042117	020040	020040															
1883	011450	040520	052124	051105															
1884	011456	020116	040504	040524															
1885	011464	020040	020040	026516															
1886	011472	047503	042504	005015															
1887	011500	020040	020040	020040		.ASCIIZ	/	NO	ECC	GOOD	ECC1	ECC2	ECC	ENVELOPE	ZEROS				
1888	011506	020040	047516	020040															
1889	011514	020040	020040	041505															
1890	011522	020103	020040	020040															
1891	011530	047507	042117	020040															
1892	011536	020040	041505	030503															
1893	011544	020040	020040	041505															
1894	011552	031103	020040	020040															
1895	011560	041505	020103	020040															
1896	011566	020040	047105	046126															
1897	011574	050117	020105	042532															
1898	011602	047522	004523	000															
1899																			
1900	011607	120	020103	020040	DH40:	.ASCII	/PC	TEST	FAILING	CONT.	CONT.	CONT.	CONT./	<15><12>					
1901	011614	020040	052040	051505															
1902	011622	020124	020040	043040															
1903	011630	044501	044514	043516															
1904	011636	041440	047117	027124															
1905	011644	020040	041440	047117															
1906	011652	027124	020040	041440															
1907	011660	047117	027124	020040															
1908	011666	041440	047117	027124															
1909	011674	005015																	
1910	011676	020040	020040	020040		.ASCIIZ	/	NO	REG	ADR	RHCS1	RHCS2	RHDS1	RHER1/					
1911	011704	020040	047516	020040															
1912	011712	020040	020040	042522															
1913	011720	020107	042101	020122															
1914	011726	044122	051503	020061															
1915	011734	020040	044122	051503															
1916	011742	020062	020040	044122															
1917	011750	051504	020061	020040															
1918	011756	044122	051105	000061															
1919																			
1920						.EVEN													
1921																			
1922	011764	001116	017330	045534	DT1:	.WORD		\$ERRPC	TSTNM	REGADR	\$GDDAT	\$BDDAT	0						
1923	011772	001124	001126	000000															
1924	012000	001116	017330	052774	DT2:	.WORD		\$ERRPC	TSTNM	ERWORD	\$GDDAT	0							
1925	012006	001124	000000																
1926	012012	001116	017330	052774	DT3:	.WORD		\$ERRPC	TSTNM	ERWORD	\$GDDAT	\$BDDAT	0						
1927	012020	001124	001126	000000															
1928																			
1929	012026	001116	017330	001122	DT11:	.WORD		\$ERRPC	TSTNM	\$BDADR	CS1	CS2	DS1	ER1	0				

1986	012474	000000							
1987	012476	001116	017330	001122	DT40:	.WORD	\$ERRPC,TSTNM,\$BDADR,CS1,CS2,DS1,ER1,0		
1988	012504	015032	015030	015054					
1989	012512	015034	000000						
1990									
1991									
1992									
1993									
1994	012516	000	000	000	DF1:	.BYTE	0,0,0,0,0		
1995	012521	000	000						
1996	012523	000	000	001	DF2:	.BYTE	0,0,1,0		
1997	012526	000							
1998	012527	000	000	001	DF3:	.BYTE	0,0,1,0,0		
1999	012532	000	000						
2000									
2001	012534	000	000	000	DF11:	.BYTE	0,0,0,0,0,0,0		
2002	012537	000	000	000					
2003	012542	000							
2004	012543	000	000	000	DF14:	.BYTE	0,0,0,0,0,0,0,0		
2005	012546	000	000	000					
2006	012551	000	000						
2007	012553	000	000	000	DF15:	.BYTE	0,0,0		
2008	012556	000	000	000	DF16:	.BYTE	0,0,0,0,0		
2009	012561	000	000						
2010	012563	000	000	000	DF17:	.BYTE	0,0,0,0,0,0,0		
2011	012566	000	000	000					
2012	012571	000							
2013	012572	000	000	000	DF20:	.BYTE	0,0,0,0,0,0,0		
2014	012575	000	000	000					
2015	012600	000							
2016									
2017	012601	000	000	000	DF21:	.BYTE	0,0,0,0,0		
2018	012604	000	000						
2019	012606	000	000	000	DF22:	.BYTE	0,0,0,0		
2020	012611	000							
2021	012612	000	000	000	DF24:	.BYTE	0,0,0,0,0,0,0		
2022	012615	000	000	000					
2023	012620	000							
2024	012621	000	000	000	DF26:	.BYTE	0,0,0,0,0,0,0,0		
2025	012624	000	000	000					
2026	012627	000	000						
2027	012631	000	000	000	DF27:	.BYTE	0,0,0,0,0,0		
2028	012634	000	000	000					
2029	012637	000	000	000	DF30:	.BYTE	0,0,0,0,0,0		
2030	012642	000	000	000					
2031									
2032	012645	000	000	001	DF31:	.BYTE	0,0,1,0,0,0,0,0		
2033	012650	000	000	000					
2034	012653	000	000						
2035	012655	000	000	000	DF32:	.BYTE	0,0,0,1,0,0,0,0,0		
2036	012660	001	000	000					
2037	012663	000	000	000					
2038	012666	000	000	000	DF33:	.BYTE	0,0,0,1,0,0,0,0		
2039	012671	001	000	000					
2040	012674	000	000						
2041	012676	000	000	000	DF34:	.BYTE	0,0,0,0,0,0,0,0		

2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505

000001
000002
000010
000020
000040
000100
040000
100000

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

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

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

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

2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549

014744 000254

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

015016 176752
015020 176750

.SBTTL REGISTER ADDRESSES

:RP04/5/6 VECTOR ADDRESS

RPVEC: 254 ;RP04/5/6 VECTOR ADDRESS

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

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

:ADDITIONAL REGISTERS LOCATED IN THE RH70 CONTROLLER

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


```
2550
2551 ;THE FOLLOWING LOCATIONS ARE RESERVED FOR REGISTER SAVES
2552 ;ANY TIME THERE IS AN ERROR ALL THESE WILL BE FILLED
2553 ;ONLY SOME MAY BE PRINTED BUT ALL WILL BE FILLED TRUE
2554 ;FOR THE TIME JUST AFTER THE 'ERROR' ERROR COMMAND
2555
2556 015022 000000 DB: 0 ;DATA BUFFER
2557 015024 000000 WC: 0 ;WORD COUNT
2558 015026 000000 BA: 0 ;BUS ADDRESS
2559 015030 000000 CS2: 0 ;CONTROL AND STATUS 2
2560
2561
2562 015032 000000 CS1: 0 ;CONTROL AND STATUS 1
2563 015034 000000 ER1: 0 ;ERROR #1
2564 015036 000000 DST: 0 ;DESIRED SECTOR/TRACK ADDRESS
2565 015040 000000 ER2: 0 ;ERROR #2
2566 015042 000000 OF: 0 ;OFFSET
2567 015044 000000 CA: 0 ;DESIRED CYLINDER ADDRESS
2568 015046 000000 ER3: 0 ;ERROR #3
2569 015050 000000 AS: 0 ;ATTENTION SUMMARY
2570 015052 000000 MR: 0 ;MAINTAINABILITY
2571 015054 000000 DS1: 0 ;DRIVE STATUS
2572 015056 000000 DT: 0 ;DRIVE TYPE
2573 015060 000000 SN: 0 ;SERIAL NUMBER
2574 015062 000000 EC1: 0 ;ECC POSITION
2575 015064 000000 EC2: 0 ;ECC PATTERN
2576 015066 000000 LA: 0 ;LOOK-AHEAD
2577 015070 000000 CC: 0 ;CURRENT CYLINDER ADDRESS
2578
2579
```

```
2580 ;FLAGS AND INTERNAL PROGRAM CONTROL WORDS
2581
2582
2583
2584 015072 000010 UNITS: .BLKW 8. ;THIS IS FILLED WITH -1
2585 015112 000000 UNIT: .WORD 0 ;UNIT UNDER TEST
2586 015114 000000 NOUNIT: .WORD 0 ;NUMBER OF UNITS PRESENT
2587 ;USED TO KEEP TRACK OF UNIT UNDER TEST
2588 015116 000000 NUNIT: .WORD 0 ;USED TO DETERMIN IF THERE ARE MORE
2589 ;THAN ONE UNIT
2590 015120 000000 SELECT: .WORD 0 ;ALL ONES INDICATE UNIT TO BE SELECTED
2591 015122 000000 UNITS_: .WORD 0 ;UNIT NO. SELECTED
2592
2593 015124 000000 ERFLG$: 0 ;ERROR FLAG
2594
2595 015126 000000 SAVDT: 0 ;SAVE DRIVE TYPE REGISTER
2596 ;FOR COMPARISON IN DRIVE CLEAR TEST
2597 ;AND RH INIT TEST
2598 015130 000000 SAVSN: 0 ;SAVE SERIAL NUMBER REGISTER
2599 ;FOR COMPARISON IN DRIVE CLEAR TEST
2600 ;AND RH INIT TEST
2601
2602 015132 000000 PCJSR: 0 ;SAVE PC OF JSR WHICH GAVE THE ERROR
2603
2604 015134 000000 ATTENT: 0 ;ATTENTION BIT FOR PRESENT UNIT
2605 015136 000000 TOTALAT: 0 ;TOTAL ATTENTION BITS
2606
2607 015140 000000 TMPILL: 0 ;TEMPORARY ILLEGAL FUNCTION
2608
2609 015142 000000 TSECC: 0 ;FLAG TO SAY IF ECC TEST OR NOT
2610 ;WHEN =177777 IT IS AN ECC TEST
2611 ;WHEN =0IT IS NOT AN ECC TEST
2612
2613 015144 000000 TESDTE: 0 ;FLAG TO SAY IF DRIVE TIMING ERROR OR NOT
2614 ;WHEN = 177777 IT IS A DTE TEST
2615 ;WHEN = 0 IT IS NOT A DTE TEST
2616
2617 015146 000000 TAGDTE: 0 ;TEMPORARY TAG USED IN DRIVE TIMING
2618 ;ERROR TEST
2619
```

```
2620
2621
2622                ;FUNCTION EQUATES
2623
2624                ;TABLE OF COMMAND FUNCTIONS FOR RHCS1
2625                ;THEN 'GO' BIT HAS TO BE SET
2626
2627 015150          FUTABL:
2628 015150          NOPERA: 0                ;NO OPERATION
2629 015152          UNLOAD: 2              ;UNLOAD (STAND BY)
2630 015154          000006                ;RECALIBRATE
2631 015156          000010                ;DRIVE CLEAR
2632 015160          000012                ;RELEASE (DUAL-PORT OPERATION)
2633 015162          000030                ;SEARCH COMMAND
2634 015164          000050                ;WRITE CHECK DATA
2635 015166          000052                ;WRITE CHECK HEADER AND DATA
2636 015170          000060                ;WRITE DATA
2637 015172          000062                ;WRITE HEADER AND DATA (FORMAT)
2638 015174          000070                ;READ DATA
2639 015176          000072                ;READ HEADER AND DATA
2640 015200          000004                ;SEEK COMMAND
2641 015202          000014                ;OFFSET COMMAND
2642 015204          000016                ;RETURN TO CENTERLINE
2643 015206          000022                ;PACK ACKNOWLEDGE
2644 015210          000020                ;READ IN
2645 015212          000000                ;COMPUTED ILLEGAL FUNCTION
2646
2647
2648
2649                ;DATA BUFFER FOR READ WRITE
2650
2651
2652 015220          000422                WRFROM: .BLKW 274.                ;WRITE FROM THIS BUFFER
2653 016264          000422                REINTO: .BLKW 274.                ;READ INTO THIS BUFFER
2654
2655
2656 017330          000000                TSTNM: 0                ;TEST NUMBER
2657 017332          000000                FIRST: 0                ;IF ZERO WILL TYPE HEADER
2658                                     ;IF ONES WILL NOT TYPE HEADER
2659
2660 017334          000000                RH70: 0                ;FLAG = 1 FOR RH70 CONTROLLER
2661                                     ;FLAG = 0 FOR RH11
2662
2663
2664
2665
2666                ;TABLE FOR ATTENTION BITS
2667                ;ATTENTION TABLE
2668
2669 017336          001 002 004          ATABLE: .BYTE 1,2,4,10,20,40,100,200
2670 017341          010 020 040
2671 017344          100 200
```



```
2912 022322 012700 000010      MOV      #8,R0          ;COUNTER
2913 022326 011137 001176      MOV      @R1,@STMP0    ;TEMPORARY STORAGE
2914 022332 006037 001176      4$: ROR      @STMP0      ;SET CARRY IF ONE IN 0 BIT
2915
2916 022336 103120                BCC      5$
2917 022340 010577 172410      MOV      R5,@RHCS2     ;INSERT UNIT NUMBER
2918 022344 022777 024020 172430  CMP      #24020,@RHDT  ;IS THIS A DUAL PORT RP04 ?
2919 022352 001503                BEQ      6$            ;TYPE DRIVE NO. IF SO
2920 022354 022777 020020 172420  CMP      #20020,@RHDT  ;IS THIS A SINGLE PORT RP04 ?
2921 022362 001477                BEQ      6$            ;TYPE DRIVE NO. IF YES
2922
2923
2924 022364 022777 024021 172410  CMP      #24021,@RHDT  ;IS THIS A DUAL PORT RP05 ?
2925 022372 001473                BEQ      6$            ;TYPE UNIT NO. OUT
2926 022374 022777 020021 172400  CMP      #20021,@RHDT  ;IS THIS A SINGLE PORT RP05 ?
2927 022402 001467                BEQ      6$            ;TYPE UNIT NO. IF SO
2928
2929 022404 022777 024022 172370  CMP      #24022,@RHDT  ;IS THIS A DUAL PORT RP06 ?
2930 022412 001463                BEQ      6$            ;TYPE THE NO IF SO
2931 022414 022777 020022 172360  CMP      #20022,@RHDT  ;IS THIS A SINGLE PORT RP06 ?
2932 022422 001457                BEQ      6$            ;TYPE THE NO IF SO
2933
2934
2935                                ;*NO...IT'S NOT AN RP04/RP05/RP06 DEVICE
2936                                ;*SO TYPE OUT THE DEVICE TYPE
2937
2938 022424 104401 022432      TYPE     ,65$          ;;TYPE ASCIZ STRING
2939 022430 000410                BR       64$          ;;GET OVER THE ASCIZ
2940 022452 010546                MOV      R5,-(SP)     ;GET READY TO TYPE UNIT NUMBER
2941 022454 104405                TYPDS
2942 022456 104401 022464      TYPE     ,67$          ;;TYPE ASCIZ STRING
2943 022462 000406                BR       66$          ;;GET OVER THE ASCIZ
2944 022500 017746 172276      MOV      @RHDT,-(SP)  ;GET READY TO TYPE RHDT
2945 022504 104402                TYPOC
2946 022506 104401 022514      TYPE     ,69$          ;;TYPE ASCIZ STRING
2947 022512 000422                BR       68$          ;;GET OVER THE ASCIZ
2948 022560 000407                BR       5$           ;NO RP04/5/6 FOUND SO BRANCH
2949 022562 010523                6$: MOV      R5,(R3)+
2950 022564 104401 001223      TYPE     ,$CRLF
2951 022570 010546                MOV      R5,-(SP)     ;PUT DRIVE NO. ON STACK
2952 022572 104405                TYPDS
2953 022574 005237 015114      INC      @#NUNIT       ;INCR TOTAL NO. OF UNITS
2954
2955 022600 005205                5$: INC      R5         ;'RHCS2' UNIT ADDRESS
2956 022602 005300                DEC      R0           ;DRIVE COUNTER DOWN ONE
2957 022604 001252                BNE     4$           ;TEST AND DO NEXT UNIT IF 8 NOT DONE
2958
2959 022606 013737 015072 015112  MOV      @#UNITS,@#UNIT ;SET UNIT NO. TO FIRST ONE FOUND/OR 0
2960 022614 013737 015114 015116  MOV      @#NUNIT,@#NUNIT ;SAVE NO. OF UNITS
2961 022622 005337 015116                DEC      @#NUNIT      ;IF NUNIT = 0 THEN ONLY ONE UNIT
2962                                ;IF NUNIT > 0 THEN MORE THAN ONE UNIT
2963
2964 022626 005737 015120      SELTST: TST     @#SELECT ;STARTING ADDRESS 200 ?
2965 022632 001403                BEQ     TST          ;BRANCH IF STARTING FROM 200
2966 022634 013737 015122 015112  MOV      @#UNITSL,@#UNIT ;CHANGE UNIT NUMBER TO SELECTED ONE
2967
```



```

3024 023106 104402          TYPOC          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3025
3026
3027
3028
3029
3030 023110 022777 024020 171664      CMP      #24020,@RHDT      ;DUAL PORT RP04 ?
3031 023116 001425          BEQ      4$              ;TYPE ASCII MSG OUT
3032 023120 022777 020020 171654      CMP      #20020,@RHDT      ;SINGLE PORT RP04 ?
3033 023126 001421          BEQ      4$              ;TYPE THE MESSAGE
3034
3035 023130 022777 024021 171644      CMP      #24021,@RHDT      ;DUAL PORT RP05 ?
3036 023136 001434          BEQ      5$              ;TYPE THE MESSAGE
3037 023140 022777 020021 171634      CMP      #20021,@RHDT      ;SINGLE PORT RP05 ?
3038 023146 001430          BEQ      5$              ;TYPE THE MESSAGE
3039
3040 023150 022777 024022 171624      CMP      #24022,@RHDT      ;DUAL PORT RP06 ?
3041 023156 001443          BEQ      6$              ;TYPE THE MESSAGE
3042 023160 022777 020022 171614      CMP      #20022,@RHDT      ;SINGLE PORT RP06 ?
3043 023166 001437          BEQ      6$              ;TYPE IT OUT
3044 023170 000454          BR       1$              ;DRIVE IS NOT RP04/5/6 - SO
3045                                ;DO NOT TYPE ANY MESSAGE OUT
3046
3047                                ;-SHOULD NEVER HAPPEN AT THIS POINT
3048                                ;UNLESS DRIVE GOT SICK WHILE TESTING
3049                                ;WAS IN PROGRESS
3050
3051                                4$:
3052 023172 104401 023200          TYPE     ,71$            ;;TYPE ASCIZ STRING
3053 023176 000413          BR      ,70$            ;;GET OVER THE ASCIZ
3054 023226 000435          BR      1$              ;SKIP NEXT ONES
3055                                5$:
3056 023230 104401 023236          TYPE     ,73$            ;;TYPE ASCIZ STRING
3057 023234 000413          BR      ,72$            ;;GET OVER THE ASCIZ
3058 023264 000416          BR      1$              ;SKIP NEXT
3059                                6$:
3060 023266 104401 023274          TYPE     ,75$            ;;TYPE ASCIZ STRING
3061 023272 000413          BR      ,74$            ;;GET OVER THE ASCIZ
3062
3063 023322 005777 171456          1$:  TST    @RHSN          ;READ SERIAL NO. AND DRIVE TYPE
3064 023326 005777 171450          TST    @RHDT            ;THESE TWO ARE TO HELP SCOPE LOOPS
3065 023332 017737 171446 015130      MOV    @RHSN,@#SAVSN     ;SAVE TO CHECK IF CLR RHCS2 BIT 5 CLEARS ANY BITS
3066 023340 017737 171436 015126      MOV    @RHDT,@#SAVDT     ;SAVE TO CHECK IF CLR RHCS2 BIT 5 CLEARS ANY BITS

```

33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99

```
3067
3068
3069 023346 000004
3070 023350 012737 000006 017330 TST6: SCOPE
3071 023356 004737 045770 MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
3072 023362 032713 010000 JSR PC,@#CLDISK ;GIVE INITILIZE
3073 023366 001550 BIT #MOL,@R3 ;CHECK MOL IN RHDS1
3074 BEQ TST7 ;BRANCH IF MOL LOW
3075 023370 104401 023376 TYPE ,65$ ;;TYPE ASCIZ STRING
3076 023374 000421 BR 64$ ;;GET OVER THE ASCIZ
3077 023440 104401 023446 TYPE ,67$ ;;TYPE ASCIZ STRING
3078 023444 000424 BR 66$ ;;GET OVER THE ASCIZ
3079 023516 104401 023524 TYPE ,69$ ;;TYPE ASCIZ STRING
3080 023522 000430 BR 68$ ;;GET OVER THE ASCIZ
3081
3082 023604 032713 010000 1$: BIT #MOL,@R3 ;CHECK MOL IN RHDS1
3083 023610 001375 BNE 1$ ;BRANCH IF MOL IS HIGH
3084 023612 104401 023620 TYPE ,71$ ;;TYPE ASCIZ STRING
3085 023616 000434 BR 70$ ;;GET OVER THE ASCIZ
3086
3087
3088
```

```

3089
3090 023710 000004          TST7: SCOPE
3091 023712 012706 001000  MOV    #STACK,SP      ;RESET STACK
3092 023716 012737 000007 017330  MOV    #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
3093
3094 023724 004737 045770          JSR    PC,@#CLDISK    ;INIT AND SET UP GENERAL REG.
3095                                ;AND UNIT NUMBER
3096 023730 012777 000001 171040  MOV    #DMD,@RHMR     ;SET DIAGNOSTIC MODE
3097
3098 023736 013777 015206 171012  MOV    @#PKACK,@RHCS1 ;LOAD PACK ACKNOWLEDGE COMMAND INTO RHCS1
3099
3100                                ;SAVE REGISTERS FOR COMPARISON AFTER GO
3101 023744 004037 046462          JSR    R0,@#SAVER     ;SAVE
3102 023750 014750          RHW    ;FROM
3103 023752 016264          REINTO ;TO
3104 023754 000023          19.      ;NUMBER OF REGISTERS SAVED
3105
3106                                ;GIVE GO TO PACK ACKNOWLEDGE COMMAND
3107 023756 052777 000001 170772  BIS    #GO,@RHCS1    ;GO TO PACK ACKNOWLEDGE COMMAND
3108
3109                                ;CHANGE SAVED REGISTERS TO EXPECTED VALUES
3110 023764 052737 000100 016314  BIS    #VV,@#REINTO+30 ;SAVED RHDS1
3111
3112                                ;AFTER GO HAS BEEN GIVEN TO PACK ACKNOWLEDGE COMMAND
3113                                ;SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN
3114                                ;BE DONE
3115 023772 004037 046462          JSR    R0,@#SAVER     ;SAVE
3116 023776 014750          RHW    ;FROM
3117 024000 015220          WRF    ;FROM
3118 024002 000023          19.      ;NUMBER OF REGISTERS SAVED
3119
3120                                ;AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
3121                                ;OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
3122                                ;SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
3123 024004 113737 016311 015245  MOV    @#REINTO+25,@#WRF+25;SAVE UPPER RHAS
3124
3125
3126                                ;COMPARE REGISTERS BEFORE PACK ACKNOWLEDGE COMMAND
3127                                ;WITH AFTER GO
3128 024012 004037 046664          JSR    R0,@#COMPAR    ;COMPARE
3129 024016 016264          REINTO ;GOOD BUFFER
3130 024020 015220          WRF    ;TEST BUFFER
3131 024022 000023          19.      ;NUMBER
3132 024024 024032          1$      ;RETURN FOR ERROR
3133 024026 024032          1$      ;SAME
3134 024030 024052          2$      ;RETURN FOR GOOD COMPARISON
3135 024032 013705 052774          1$: MOV    @#ERWORD,R5    ;GETTING READY TO INDEX
3136 024036 060505          ADD    R5,R5          ;DOUBLE ERROR WORD
3137 024040 016537 014746 045534  MOV    RHWC-2(R5),@#REGADR ;FAILING REGISTER ADDRESS
3138
3139 024046 104001          ERROR 1              ;IMPROPER REGISTER CHANGE
3140                                ;AFTER PACK ACKNOWLEDGE COMMAND
3141                                ;WITH GO IS GIVEN
3142 024050 000207          RTS    PC              ;RETURN TO COMPARISION
3143
3144 024052          2$:

```

```
3145
3146 024052 000004          TST10: SCOPE
3147 024054 012706 001000  MOV      #STACK,SP      ;RESET STACK
3148 024060 012737 000010 017330  MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
3149 024066 004737 045770          JSR      PC,@#CLDISK    ;INIT DRIVE
3150 024072 012777 000001 170676  MOV      #DMD,@RHMR     ;SET DIAGNOSTIC MODE
3151 024100 004037 050342          JSR      RO,@#MAKECYL   ;SUBROUTINE TO GIVE A SEEK
3152 024104 000000          0                      ;THIS SHUOLD CHANGE RHCC TO 0
3153
3154          .SBTTL
3155          .SBTTL ***DIAGNOSTIC CODE***
3156          .SBTTL
```

```
3157          000004          TIMOT=4
3158          .REM %
3159          THE FOLLOWING TESTS WILL ATTEMPT TO CATCH THOSE BUGS WHICH FAULT
3160          INSERTION HAS SHOWN US WE MISSED IN THE FIRST PART OF THIS TEST.
3161
3162          THIS TEST WILL ASCERTAIN THAT THE ALL RH11 REGISTERS WILL
3163          RESPOND TO I.E. NOT CAUSE A NON-EXISTANT MEMORY ERROR WHEN ACCESSED
3164          BY A 'TST' INSRUCTION.
3165          %
3166 024106 000004          TST11: SCOPE
3167 024110 012737 000001 001212          MOV      #1,$TIMES          ;;DO 1 ITERATION
3168 024116 012737 177777 017330          MOV      #TTNO,@#TSTNM      ;THIS SAVES TEST NUMBER
3169 024124 012777 000040 170622          MOV      #CLR,@RHCS2        ;CLEAR RH11 CONTROLLER
3170
3171 024132 012705 014756          MOV      #RHCS1,R5          ;R5=LIST POINTER.
3172
3173 024136 013737 000004 001176          MOV      @#TIMOT,$TMP0      ;SAVE TIMEOUT VECTOR.
3174 024144 012737 024202 000004          MOV      #E00,@#TIMOT      ;SET VECTOR TO E00
3175
3176 024152 012706 001000          L00:    MOV      #STACK,SP    ;SET STACK POINTER.
3177
3178 024156 011502          I00:    MOV      (R5),R2        ;R2=RH11 ADDRESS.
3179 024160 010237 045534          MOV      R2,REGADR
3180 024164 005712          TST      (R2)                ;DOES THE RH11 REGISTER RESPOND?
3181
3182 024166 062705 000002          ADD      #2,R5                ;UPDATE ADDRESS
3183 024172 020527 015010          CMP      R5,#RHEC2           ;AT THE END OF LEGAL RH11 REGISTERS?
3184 024176 101767          BLOS    100                  ;NOPE
3185 024200 000401          BR      000                  ;YES! GOTO 000.
3186
3187 024202 104042          E00:    ERROR 42
3188
3189 024204 013737 001176 000004 000:    MOV      $TMP0,@#TIMOT      ;RESTORE TIMEOUT VECTOR.
```

CZRJHCO.RP04/5/6 DSKLS CTRLR2
CZRJHC.P12 10-NOV-77 11:09

MACY11 30A(1052) 27-JUL-78 11:16 PAGE 72
T11 BCTA LEGAL REGISTER RESPONSE TEST

F 6

SEQ 0070

3190

CZF
CZF

3191
 3192
 3193
 3194
 3195
 3196
 3197
 3198
 3199
 3200
 3201
 3202
 3203
 3204
 3205
 3206
 3207
 3208
 3209
 3210
 3211
 3212
 3213
 3214
 3215
 3216
 3217
 3218
 3219
 3220
 3221
 3222
 3223
 3224

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

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

%

```

TST12: SCOPE
MOV #1,$TIMES ;:DO 1 ITERATION
MOV #TNO,@#TSTNM ;:THIS SAVES TEST NUMBER
MOV #CLR,@RHCS2 ;:CLEAR RH11 CONTROLLER

MOV #STACK,SP ;:SET STACK POINTER.

MOV RHWC,R2 ;:R2=RH11 WC ADDRESS.
MOV R2,REGADR ;:REGISTER ADDRESS TO REGADR FOR TYPING.
MOV #377,$GDDAT ;:$GDDAT=S/B.

L02: CLR (R2) ;:CLEAR RH11 WC.

I02: MOVB #377,(R2) ;:SET WC TO LO BYTE.
MOV (R2),$BDDAT ;:$BDDAT=ACTUAL WAS

CMP $BDDAT,$GDDAT ;:COMPARE RESULTS
BEQ TST13 ;:NEXT TEST
ERROR 44
  
```

UNUS

CZRJHCO, RPO4/5/6 DSKLS CTRLR2
CZRJHC.P12 10-NOV-77 11:09

MACY11 30A(1052) 27-JUL-78 11:16 PAGE 81
T21 BCTB (NED) NON-EXISTANT DRIVE TEST. (SET)

SEQ 0079

```
3455 025560 042737 167770 001126      BIC      #^C<NED'US4'US2!US1>,$BDDAT;$BDDAT=SAVED DATA.
3456
3457 025566 023737 001126 001124      CMP      $BDDAT,$GDDAT ;COMPARE RESULTS.
3458 025574 001005                        BNE      E11
3459
3460 025576 005201                        N11:    INC      R1
3461 025600 020127 000010                        CMP      R1,#8. ;TESTED ALL DRIVES YET.
3462 025604                        X11:
3463 025604 001402                        BEQ      TST22 ;NEXT TEST
3464 025606 000746                        BR       S11
3465
3466 025610 104050                        E11:    ERROR   50
3467
3468
```

CZ
CZ

```
3469
3470 025612 000004          TST22: SCOPE
3471 025614 012737 000001 001212  MOV    #1,$TIMES      ;;DO 1 ITERATION
3472 025622 012737 000022 017330  MOV    #TINO,@TSTNM  ;THIS SAVES TEST NUMBER
3473 025630 012777 000040 167116  MOV    #CLR,@RHCS2   ;CLEAR RH11 CONTROLLER
3474
3475 025636 012706 001000          MOV    #STACK,SP     ;SET STACK POINTER.
3476
3477 025642 013702 014754          MOV    RHCS2,R2      ;R2=RH11 RHCS2 ADDRESS.
3478 025646 010237 045534          MOV    R2,REGADR
3479 025652 013700 015136          MOV    @TOTALAT,R0
3480 025656 005001          CLR    R1
3481 025660 006000          S12:  ROR    R0
3482 025662 103020          BCC   N12
3483
3484 025664 010137 001124          MOV    R1,$GDDAT
3485
3486 025670 013705 014756          MOV    RHCS1,R5      ;ADDRESS OF A DEVICE REGISTER.
3487
3488 025674 010112          L12:  MOV    R1,(R2)  ;SELECT UNIT.
3489
3490 025676 011537 001176          MOV    (R5),$TMPO    ;ATTEMPT TO READ FROM DEVICE.
3491
3492 025702 011237 001126          MOV    (R2),$BDDAT   ;$BDDAT=WAS.
3493 025706 042737 167770 001126  BIC    #^C<NED.US4!US2!US1>,$BDDAT ;$BDDAT=SAVED DATA.
3494
3495 025714 023737 001126 001124  CMP    $BDDAT,$GDDAT ;COMPARE RESULTS.
3496 025722 001005          BNE   E12
3497
3498 025724 005201          N12:  INC    R1
3499 025726 020127 000010          CMP    R1,#8.       ;TESTED ALL DRIVES.
3500 025732 001402          BEQ   TST23        ;NEXT TEST
3501
3502 025734 000751          BR    S12
3503 025736 104050          E12:  ERROR 50
3504
3505
```

3506
3507
3508
3509
3510
3511
3512
3513
3514
3515
3516
3517
3518
3519
3520
3521
3522
3523
3524
3525
3526
3527
3528
3529
3530
3531
3532
3533
3534
3535
3536
3537
3538
3539
3540

.REM %
TEST TO SEE IF WE CAN READ FROM THE 'AS' REGISTER AND NOT CAUSE
A NED NON-EXISTANT DRIVE ERROR

```
TST23: SCOPE
MOV #1,$TIMES ;:DO 1 ITERATION
MOV #TNO,@#TSTNM ;:THIS SAVES TEST NUMBER
MOV #CLR,@RHCS2 ;:CLEAR RH11 CONTROLLER
MOV #STACK,SP ;:SET STACK POINTER.
MOV RHCS2,R2 ;:R2=RH11 CS2 ADDRESS.
MOV #US4!US2!US1,$GDDAT;$GDDAT=S/B.
MOV RHCS1,R5 ;:ADDRESS OF A DEVICE REGISTER.
L13: MOV #US4!US2!US1,(R2);LOAD A NON-EXISTANT DEVICE.
MOV RHAS,R2 ;:R2= 'AS'.
MOV (R2),$TMP0 ;:ATTEMPT TO READ FROM DEVICE AS.
CLR (R2) ;:CLEAR AS REGISTER.
MOV RHCS2,R2 ;:R2-RH11 CS2 ADDRESS.
MOV R2,REGADR
MOV (R2),$BDDAT ;:$BDDAT=WAS.
BIC #^C<NED.US4!US2!US1>,$BDDAT;SAVE NED AND DEVICE SELECTED.
CMP $BDDAT,$GDDAT ;:COMPARE RESULTS.
BEQ TST24 ;:NEXT TEST
ERROR 51
```

```
3541 .REM %
3542 THE NEXT THREE TESTS WILL TEST THE BUS ADDRESS REGISTER IN BOTH WORD
3543 AND BYTE MODE. THE PATTERNS ARE CHOSEN TO PICK UP ANY DROPPED OR STUCK
3544 BITS.
3545 %
3546
3547 026060 000004 TST24: SCOPE
3548 026062 012737 000001 001212 MOV #1,$TIMES ;;DO 1 ITERATION
3549 026070 012737 000024 017330 MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
3550 026076 012777 000040 166650 MOV #CLR,@RHCS2 ;CLEAR RH11 CONTROLLER
3551
3552 026104 012706 001000 MOV #STACK,SP ;SET STACK POINTER.
3553
3554 026110 013702 014752 MOV RHBA,R2 ;R2=RH11 BA ADDRESS.
3555 026114 010237 045534 MOV R2,REGADR
3556 026120 012705 026160 MOV #LST14A,R5 ;R5=TEST LIST ADDRESS.
3557
3558 026124 012537 001124 L14: MOV (R5)+,$GDDAT ;$GDDAT=S/B.
3559
3560 026130 013712 001124 I14: MOV $GDDAT,(R2) ;SET BUS ADDRESS REGISTER.
3561 026134 011237 001126 MOV (R2),$BDDAT ;READ BUS ADDRESS REGISTER.
3562
3563 026140 023737 001126 001124 CMP $BDDAT,$GDDAT ;COMPARE RESULTS.
3564 026146 001401 BEQ N14 ;GET NEXT TEST DATA.
3565 026150 104052 ERROR 52
3566
3567 026152 005715 N14: TST (R5) ;AT END OF LIST
3568 026154 001363 BNE L14 ;NO!
3569 026156 000440 BR TST25 ;NEXT TEST
3570 026160 000002 LST14A: 2
3571 026162 000004 4
3572 026164 000010 10
3573 026166 000020 20
3574 026170 000040 40
3575 026172 000100 100
3576 026174 000200 200
3577 026176 000400 400
3578 026200 001000 1000
3579 026202 002000 2000
3580 026204 004000 4000
3581 026206 010000 10000
3582 026210 020000 20000
3583 026212 040000 40000
3584 026214 100000 100000
3585 026216 177776 177776
3586 026220 177774 177774
3587 026222 177772 177772
3588 026224 177766 177766
3589 026226 177756 177756
3590 026230 177736 177736
3591 026232 177676 177676
3592 026234 177576 177576
3593 026236 177376 177376
3594 026240 176776 176776
3595 026242 175776 175776
3596 026244 173776 173776
```

CZRJHCO,RP04/5/6 DSKLS CTRLR2
CZRJHC.P12 10-NOV-77 11:09

MACY11 30A(1052) 27-JUL-78 11:16 PAGE 85
T24 BCTC BUS ADDRESS REGISTER

F 7

SEQ 0083

3597	026246	167776	167776
3598	026250	157776	157776
3599	026252	137776	137776
3600	026254	077776	077776
3601	026256	000000	0
3602			

CZR
CZR

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


```
3756 .REM %  
3757 SET THE MXF FLOP AND READ IT BACK.  
3758 %  
3759 027000 000004 TST31: SCOPE  
3760 027002 012737 000001 001212 MOV #1,$TIMES ;;DO 1 ITERATION  
3761 027010 012737 000031 017330 MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER  
3762 027016 005737 017334 TST @#RH70 ;TEST FOR RH70 CONTROLLER  
3763 027022 001402 BEQ 30$ ;IF FLAG = 1, SKIP THIS TEST  
3764 027024 000137 027110 JMP TST32 ;JUMP TO NEXT TEST -----)  
3765 027030 012777 000040 165716 MOV #CLR,@RHCS2 ;CLEAR RH11 CONTROLLER  
3766  
3767 027036 012706 001000 MOV #STACK,SP ;SET STACK POINTER.  
3768 027042 013702 014754 MOV RHCS2,R2 ;R2=RH11 CS2 ADDRESS.  
3769 027046 010237 045534 MOV R2,REGADR  
3770 027052 012737 001000 001124 MOV #MXF,$GDDAT ;$GDDAT=S/B.  
3771  
3772 027060 012712 001000 124: MOV #MXF,(R2) ;LOAD MXF  
3773  
3774 027064 011237 001126 MOV (R2),$BDDAT ;$BDDAT=WAS.  
3775  
3776 027070 042737 176777 001126 BIC #^C<MXF>,$BDDAT ;SAVE ONLY MXF  
3777  
3778 027076 023737 001126 001124 CMP $BDDAT,$GDDAT ;COMPARE RESULTS  
3779 027104 001401 BEQ TST32 ;NEXT TEST  
3780 027106 104056 ERROR 56  
3781  
3782
```

41
41
41
41
41


```
3874  
3875  
3876  
3877  
3878  
3879 027434 000004  
3880 027436 012737 000001 001212  
3881 027444 012737 000035 017330  
3882 027452 005737 017334  
3883 027456 001402  
3884 027460 000137 027550  
3885  
3886 027464 012777 000040 165262  
3887  
3888 027472 012706 001000  
3889  
3890 027476 013702 014754  
3891 027502 012737 040000 001124  
3892  
3893 027510 012712 020000 L35:  
3894  
3895 027514 013702 014756  
3896 027520 010237 045534  
3897  
3898 027524 011237 001126  
3899 027530 042737 137777 001126  
3900  
3901 027536 023737 001126 001124  
3902 027544 001401  
3903 027546 104061  
3904  
3905
```

```
.REM %  
SET THE UPE FLOP UNIBUS PARITY ERROR. VERIFY THAT THE SETTING OF (UPE)  
ALSO SETS THE TRANSFER ERROR (TRE) FLOP  
%  
TST35: SCOPE  
MOV #1,$TIMES ;;DO 1 ITERATION  
MOV #TNO,@TSTNM ;THIS SAVES TEST NUMBER  
TST @RH70 ;TEST FOR RH70 CONTROLLER  
BEQ 30$ ;IF FLAG = 1, SKIP THIS TEST  
JMP TST36 ;JUMP TO NEXT TEST -----)  
  
MOV #CLR,@RHCS2 ;CLEAR RH11 CONTROLLER  
  
MOV #STACK,SP ;SET STACK POINTER.  
  
MOV RHCS2,R2 ;R2=RHCS2 ADDRESS.  
MOV #TRE,$GDDAT ;$GDDAT=S/B.  
  
MOV #UPE,(R2) ;SET UNIBUS PARITY ERROR.  
  
MOV RHCS1,R2 ;R2=RHCS1 ADDRESS.  
MOV R2,REGADR  
  
MOV (R2),$BDDAT ;READ REGISTER  
BIC #^C<TRE>,$BDDAT ;SAVE TRANSFER ERROR.  
  
CMP $BDDAT,$GDDAT ;COMPARE RESULTS  
BEQ TST36 ;NEXT TEST  
ERROR 61
```

```
3906
3907
3908 .REM %
3909 SET THE NED BIT NON EXISTANT DRIVE VERIFY THAT THIS SETS THE (TRE) BIT.
3910 TST36: %
3911 027550 000004 SCOPE
3912 027552 012737 000001 001212 MOV #1,$TIMES ;;DO 1 ITERATION
3913 027560 012737 000036 017330 MOV #TINO,@TSTNM ;THIS SAVES TEST NUMBER
3914 027566 012777 000040 165160 MOV #CLR,@RHCS2 ;CLEAR RH11 CONTROLLER
3915 027574 012706 001000 MOV #STACK,SP ;SET STACK POINTER.
3916
3917 027600 013702 014754 MOV RHCS2,R2 ;R2=CS2 ADDRESS.
3918 027604 012737 040000 001124 MOV #TRE,$GDDAT ;$GDDAT=S/B.
3919
3920 027612 013700 015136 MOV @TOTALAT,R0 ;GET ALL AVAILABLE DRIVES DATA
3921 027616 005001 CLR R1
3922 027620 006000 S36: ROR R0
3923 027622 103420 BCS N36
3924 027624 010112 L36: MOV R1,(R2) ;SET NED.
3925
3926 027626 013702 014756 MOV RHCS1,R2 ;R2=RHCS1 ADDRESS.
3927 027632 010237 045534 MOV R2,REGADR
3928
3929 027636 011237 001126 MOV (R2),$BDDAT ;READ REGISTER.
3930 027642 042737 137777 001126 BIC #^C<TRE>,$BDDAT ;SAVE TRE.
3931
3932 027650 023737 001126 001124 CMP $BDDAT,$GDDAT ;COMPARE RESULTS.
3933 027656 001406 BEQ TST37 ;NEXT TEST
3934 027660 104062 ERROR 62
3935 027662 000404 BR TST37 ;NEXT TEST
3936
3937 027664 005201 N36: INC R1
3938 027666 020127 000010 CMP R1,#8.
3939 027672 001352 BNE S36
3940
3941
```

```
3942  
3943  
3944  
3945  
3946 027674 000004  
3947 027676 012737 000001 001212  
3948 027704 012737 000037 017330  
3949 027712 012777 000040 165034  
3950  
3951 027720 012706 001000  
3952  
3953 027724 013702 014756  
3954 027730 010237 045534  
3955 027734 005037 001124  
3956  
3957 027740 012712 002000  
3958  
3959 027744 012777 000040 165002  
3960  
3961 027752 011237 001126  
3962 027756 042737 175777 001126  
3963  
3964 027764 023737 001126 001124  
3965 027772 001401  
3966 027774 104064  
3967
```

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



```
3968 .REM %  
3969 VERIFY THAT CONTROLLER CLEAR WILL CLEAR THE COMMAND REGISTER.  
3970 %  
3971  
3972 027776 000004 TST40: SCOPE  
3973 030000 012737 000001 00121c MOV #1,$TIMES ;;DO 1 ITERATION  
3974 030006 012737 000040 017330 MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER  
3975 030014 012777 000040 164732 MOV #CLR,@RHCS2 ;CLEAR RH11 CONTROLLER  
3976  
3977 030022 012706 001000 MOV #STACK,SP ;SET STACK POINTER.  
3978  
3979 030026 013702 014756 MOV RHCS1,R2 ;R2=RHCS1 ADDRESS.  
3980 030032 010237 045534 MOV R2,REGADR  
3981 030036 005037 001124 CLR $GDDAT ;$GDDAT=S/B.  
3982  
3983 030042 012712 000011 L41: MOV #11,(R2) ;ISSUE A DRIVE CLR AND GO.  
3984  
3985 030046 012777 000040 164700 MOV #CLR,@RHCS2 ;CONTROLLER CLEAR.  
3986  
3987 030054 011237 001126 MOV (R2),$BDDAT ;READ COMMAND REGISTER.  
3988 030060 042737 177770 001126 BIC #^C<7>,$BDDAT ;SAVE ONLY THE COMMAND BITS.  
3989  
3990 030066 023737 001126 001124 CMP $B DAT,$GDDAT ;COMPARE RESULTS.  
3991 030074 001401 BEQ TST41 ;NEXT TEST  
3992 030076 104065 ERROR 65  
3993
```

```
3994  
3995  
3996  
3997  
3998  
3999  
4000  
4001  
4002 030100 000004  
4003 030102 012737 000001 001212  
4004 030110 012737 000041 017330  
4005 030116 012777 000040 164630  
4006  
4007 030124 012706 001000  
4008  
4009 030130 013702 014756  
4010 030134 010237 045534  
4011 030140 005037 001124  
4012  
4013 030144 012737 000340 177776  
4014  
4015 030152 012712 000011  
4016  
4017 030156 011237 001176  
4018  
4019 030162 011237 001126  
4020 030166 042737 177770 001126  
4021  
4022 030174 023737 001126 001124  
4023 030202 001401  
4024 030204 104066  
4025
```

.REM %
THE COMMAND REGISTER IS SUPPOSED TO BE SELF CLEARING THAT IS WHEN THE
RH11 IS SELECTED WHEATHER OR NOT WE ADDRESS IT DIRECTLY THE LOGIC
SHOULD CLEAR THE COMMAND REGISTER.
THIS IS TESTED BY LOADING A COMMAND INTO IT READING ANOTHER REGISTER
THAN RETURNING TO CHECK THE COMMAND REGISTER TO DETERMINE IF IT CLEARED.

TST41: %
SCOPE
MOV #1,\$TIMES ;:DO 1 ITERATION
MOV #TINO,@#TSTNM ;:THIS SAVES TEST NUMBER
MOV #CLR,@RHCS2 ;:CLEAR RH11 CONTROLLER
MOV #STACK,SP ;:SET STACK POINTER.
MOV RHCS1,R2 ;:R2=RHCS1 ADDRESS.
MOV R2,REGADR
CLR \$GDDAT ;:\$GDDAT-S/B.
MOV #340,PS ;:SET PRIORITY TO #7.
L42: MOV #11,(R2) ;:ISSUE A DRIVE CLR AND GO
MOV (R2),\$TMP0 ;:DO A RESELECT OF THE REGISTER.
MOV (R2),\$BDDAT ;:READ THE COMMAND REGISTER.
BIC #^C<7>,\$BDDAT ;:SAVE ONLY THE COMMAND BITS.
CMP \$BDDAT,\$GDDAT ;:COMPARE RESULTS.
BEQ TST42 ;:NEXT TEST
ERROR 66

```
4026  
4027  
4028  
4029  
4030 030206 000004  
4031 030210 012737 000001 001212  
4032 030216 012737 000042 017330  
4033 030224 012777 000040 164522  
4034  
4035 030232 013702 014756  
4036 030236 010237 045534  
4037 030242 005037 001126  
4038 030246 005037 001124  
4039  
4040 030252 017737 164466 001176  
4041 030260 012777 030314 164456  
4042  
4043 030266 012706 001000  
4044 030272 005037 177776  
4045  
4046 030276 052712 000300  
4047 030302 000240  
4048 030304 000240  
4049  
4050 030306 011237 001124  
4051 030312 104067  
4052  
4053 030314 013777 001176 164422 043:
```

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


```
4090  
4091  
4092 .REM %  
4093 HERE WE VERIFY THAT WRITING INTO THE 'AS' REGISTER OWILL NOT CAUSE  
4094 AN (NED) ERROR.  
4095 TST44: %  
4096 030452 000004 SCOPE  
4097 030454 012737 000001 001212 MOV #1,$TIMES ;;DO 1 ITERATION  
4098 030462 012737 000044 017330 MOV #TINO,@#TSTNM ;THIS SAVES TEST NUMBER  
4099 030470 012777 000040 164256 MOV #CLR,@RHCS2 ;CLEAR RH11 CONTROLLER  
4100 030476 012706 001000 MOV #STACK,SP ;SET STACK POINTER.  
4101  
4102 030502 013702 014754 MOV RHCS2,R2 ;R2=RH11 CS2 ADDRESS.  
4103 030506 012737 000007 001124 MOV #US4!US2!US1,$GDDAT;$GDDAT=S/B.  
4104  
4105 030514 013705 014756 MOV RHCS1,R5 ;ADDRESS OF A DEVICE REGISTER.  
4106  
4107 030520 012712 000007 145: MOV #US4!US2!US1,(R2);LOAD A NON EXISTANT DEVICE.  
4108  
4109 030524 013702 014774 MOV RHAS,R2 ;R2='AS' ADDRESS.  
4110  
4111 030530 012712 000377 MOV #377,(R2) ;ATTEMPT TO LOAD 'AS'.  
4112  
4113 030534 005012 CLR (R2) ;CLEAR 'AS'.  
4114  
4115 030536 013702 014754 MOV RHCS2,R2 ;R2=RH11 CS2 ADDRESS.  
4116 030542 010237 045534 MOV R2,REGADR  
4117  
4118 030546 011237 001126 MOV (R2),$BDDAT ;$BDDAT=WAS.  
4119 030552 042737 167770 001126 BIC #^C<NED!US4.US2!US1>,$BDDAT;SAVE NED AND DEVICE SELECTED.  
4120  
4121 030560 023737 001126 001124 CMP $BDDAT,$GDDAT ;COMPARE RESULTS.  
4122 030566 001401 BEQ TST45 ;NEXT TEST  
4123 030570 104071 ERROR 71  
4124
```

```
.SBTTL EXTENDED MEMORY, DRIVE TIMING & SECTOR SELECT TESTS

4125
4126
4127
4128 030572 000004          TST45: SCOPE
4129 030574 012706 001000  MOV      #STACK,SP          ;RESET STACK
4130 030600 012737 000045 017330  MOV      #TTNO,@#TSTNM      ;THIS SAVES TEST NUMBER
4131 030606 004737 045770          JSR      PC,@#CLDISK        ;INIT DRIVE
4132 030612 012777 000001 164156  MOV      #DMD,@RHMR        ;SET DIAGNOSTIC MODE
4133 030620 004037 050342          JSR      RO,@#MAKECYL      ;SUBROUTINE TO GIVE A SEEK
4134 030624 000000          0                          ;THIS SHUOLD CHANGE RHCC TO 0
4135
4136
4137
4138
4139
4140
4141 030626 000004          TST46: SCOPE
4142
4143 030630 012706 001000          MOV      #STACK,SP          ;RESET STACK
4144 030634 012737 000046 017330  MOV      #TTNO,@#TSTNM      ;THIS SAVES TEST NUMBER
4145 030642 005737 017334          TST      @#RH70             ;TEST FOR RH70 CONTROLLER
4146 030646 001402          BEQ      30$                ;IF FLAG = 1, SKIP THIS TEST
4147 030650 000137 031160          JMP      TST47              ;JUMP TO NEXT TEST -----)
4148
4149 030654 004037 045706          JSR      RO,@#CLAREA        ;CLEAR SIMULATED DISK
4150 030660 054572          .WORD   DISK                ;FROM
4151 030662 055616          .WORD   TOLGAP+16          ;TO
4152 030664 000000          .WORD   0                   ;DATA
4153
4154          ;THESE ARE SETUP FOR DISKLESS USE ONLY
4155
4156 030666 012737 010000 052654  MOV      #FMT22,@#CYL;CYLINDER 0
4157          ;16 BITS PER WORD
4158 030674 005037 052656          CLR      @#SECOTR          ;SECTOR 0 TRACK 0
4159 030700 005037 052660          CLR      @#KEY1            ;KEY1 0
4160 030704 005037 052662          CLR      @#KEY2            ;KEY2 0
4161 030710 012737 000400 052722  MOV      #256,@#NOWORD      ;NO OF DATA WORDS
4162 030716 012737 000001 052664  MOV      #1,@#X             ;WRITE DATA
4163 030724 004537 047176          JSR      R5,@#CRC          ;GO TO CALCULATE CRC
4164 030730 052654          CYL
4165 030732 054554          WCRC
4166
4167          ;THESE ARE REGULAR SETUPS
4168
4169
4170 030734 004737 045770          JSR      PC,@#CLDISK        ;SETUP GENERAL REGISTERS
4171 030740 012777 177400 164002  MOV      #-256,@RHWC        ;256 DATA WORDS
4172 030746 013777 001144 163776  MOV      @#STKS,@RHBA       ;STARTING ADDRESS OF WRITE BUFFER
4173 030754 017737 150164 015140  MOV      @#STKS,@#TMPILL     ;TEMPORARY STORAGE OF DATA
4174 030762 005077 163774          CLR      @RHDS             ;SECTOR 0 TRACK 0
4175 030766 012777 010000 163772  MOV      #FMT22,@RHOF       ;16 BITS PER WORD FORMAT
4176 030774 005077 163770          CLR      @RHCA             ;CYLINDER 0
4177 031000 004737 046024          JSR      PC,@#CHECKT        ;CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DON'T
4178 031004 104401 005123          TYPE      ,CPHALT          ;CANNOT CONTINUE TESTING IF ANY OF THE
4179 031010 000000          HALT                       ;STOP THE TEST
4180 031012 013746 015170          MOV      @#WRIDAT,-(SP)    ;WRITE DATA=60
```



```
4333 : *NOW READ & SAVE REGISTERS AGAIN SO THAT COMPARISONS
4334 : *CAN BE DONE .USE THE 'WRFROM' SAVE BUFFER THIS TIME)
4335
4336 031546 004037 046462 JSR RO,@#SAVER ;READ IN SEQUENCE
4337 031552 014750 RHWC ;FROM HARDWARE REGISTER
4338 031554 015220 WRFROM ;INTO CORE BUFFER
4339 031556 000023 19. ;NUMBER OF REGISTERS TO READ
4340
4341 : *FOR RHAS UPPER BYTE
4342 031560 113737 015051 015245 MOVB @#AS+1,@#WRFROM+25 ;UPPER RHAS
4343
4344 : *COMPARE THE HEADER READ
4345
4346 031566 004037 046664 JSR RO,@#COMPAR ;COMPARE
4347 031572 052654 CYL ;GOOD BUFFER
4348 031574 016264 REINTO ;TEST BUFFER
4349 031576 000004 4. ;NUMBER
4350 031600 031606 6$ ;RETURN FOR ERROR
4351 031602 031606 6$ ;SAME
4352 031604 031612 7$ ;RETURN FOR GOOD COMPARISON
4353 031606 104010 6$: ERROR 10 ;HEADER READ 'N DURING THIS TEST IS
4354 ;IN ERROR
4355
4356 031610 000207 RTS PC ;RETURN
4357
4358 031612 7$: ;GOOD COMPARISON, CONTINUE
4359
4360 : *COMPARE REGISTERS BEFORE COMMAND WITH REGISTERS AFTER COMMAND
4361
4362
4363 031612 004037 046664 JSR RO,@#COMPAR ;COMPARE
4364 031616 015024 WC ;INITIAL SNAPSHOT BUFFER (CHANGED)
4365 031620 015220 WRFROM ;TEST SNAPSHOT BUFFER
4366 031622 000022 18. ;NUMBER OF REGISTERS
4367 031624 031632 2$ ;RETURN FOR ERROR
4368 031626 031632 2$ ;SAME
4369 031630 031652 3$ ;RETURN FOR GOOD COMPARISON
4370
4371 031632 013705 052774 2$: MOV @#ERWORD,R5 ;GETTING READY TO INDEX
4372 031636 060505 ADD R5,R5 ;DOUBLE ERROR WORD
4373 031640 016537 014746 045534 MOV RHWC-2(R5),@#REGADR ;FAILING REGISTER
4374 031646 104001 ERROR 1 ;IMPROPER REGISTER
4375 ;CHANGE AFTER FORCING
4376 ;'DTE' ERROR
4377 031650 000207 RTS PC ;RETURN
4378
4379 031652 3$: ;GOOD - REGISTERS OK, GO ON TO NEXT TEST
4380
4381
```

```

4382
4383 031652 000004          TST50: SCOPE
4384 031654 012706 001000  MOV      #STACK,SP      ;RESET STACK
4385 031660 012737 000050 017330  MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
4386                                     ;*THESE ARE TO SETUP FOR DISKLESS USE
4387
4388 031666 012737 010000 052654  MOV      #FMT22,@#CYL   ;16 BITS PER WORD
4389                                     ;CYLINDER 0
4390 031674 005037 052656      CLR      @#SECOTR      ;SECTOR 0
4391                                     ;TRACK 0
4392 031700 005037 052660      CLR      @#KEY1        ;KEY1 = 0
4393 031704 005037 052662      CLR      @#KEY2        ;KEY2 = 0
4394 031710 012737 177777 052664  MOV      #-1,@#X       ;THIS IS FOR WRITE DATA COMMAND
4395 031716 004537 047176      JSR      R5,@#CRC      ;GO TO CALCULATE CRC
4396 031722 052654      CYL
4397 031724 054554      WCRC
4398
4399                                     ;* THESE          ARE REGULAR SETUPS & CHECKS
4400
4401 031726 004737 045770      JSR      PC,@#CLDISK   ;SETUP GENERAL REGISTERS
4402 031732 012777 177400 163010  MOV      #-256,@#RHWC  ;256 DATA WORDS
4403 031740 012777 015220 163004  MOV      #WRFROM,@#RHBA ;STARTING ADDRESS OF BUFFER
4404 031746 005077 163010      CLR      @#RH DST     ;TRACK = 0
4405                                     ;SECTOR = 0
4406 031752 012777 010000 163006  MOV      #FMT22,@#RHOF ;16 BITS PER WORD
4407                                     ;ECC CORRECTION INHIBITED
4408 031760 005077 163004      CLR      @#RHCA       ;CYLINDER = 0
4409 031764 004737 046024      JSR      PC,@#CHECKT   ;CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DON'T
4410 031770 104401 005123      TYPE     ,CPHALT      ;CANNOT CONTINUE TESTING IF ANY OF THE
4411 031774 000000      HALT                    ;STOP THE TEST
4412 031776 013711 015170      MOV      @#WRIDAT,@R1 ;WRITE DATA = 60
4413
4414                                     ;*READ & SAVE REGISTERS FOR COMPARISON AFTER SIMULATED 'DTE'
4415
4416 032002 004037 046462      JSR      R0,@#SAVER    ;READ REGISTERS IN SEQUENCE
4417 032006 014750      RHWC                    ;FROM HARDWARE REGISTER
4418 032010 015024      WC                      ;INTO CORE BUFFER LOCATION
4419 032012 000023      19.                    ;NUMBER OF REGISTERS TO READ
4420
4421                                     ;*NOW 'GO' WILL BE GIVEN, EVERYTHING WILL BE TREATED
4422                                     ;*NORMALLY FOR THE HEADER. WHEN IT IS TIME TO WRITE
4423                                     ;*DATA, ONLY SECTOR CLOCKS WILL BE GIVEN, NO DIAGNOSTIC DATA
4424                                     ;*CLOCKS WILL BE GIVEN. THIS SHOULD BRING SECTOR PULSF HIGH
4425                                     ;*WITHOUT PUTTING READ DOWN, HENCE 'DTE' WILL COME UP.
4426
4427 032014 012737 177777 015144  MOV      #-1,@#TESDTE  ;SET DTE TEST
4428 032022 012737 177777 015124  MOV      #-1,@#ERFLG$  ;THIS WILL BRING THE READ HEADER
4429                                     ;AND DATA PROCESS OUT AFTER THE
4430                                     ;HEADER HAS BEEN CORRECTLY READ
4431
4432 032030 004737 052514      JSR      PC,@#COMHD    ;ISSUE 'GO', SEARCH FOR SECTOR,
4433                                     ;READ HEADER AND DATA.
4434
4435 032034 017737 162716 001176  SETCK2: MOV      @#RHCS1,@#STMPO ;READ CS1 TO CHECK FOR READ ERRORS
4436 032042 032737 100000 001176  BIT      #SC,@#STMPO   ;TEST FOR 'SPECIAL CONDITION' - 'SC'
4437 032050 001405      BEQ     6$            ;CONTINUE TESTING IF NO ERROR
    
```

```

4438 032052 004737 045470 JSR PC,@PUTREG ;READ & SAVE REGISTERS AGAIN IF UNDE-
4439 ;FINED ERROR HAS OCCURRED
4440 032056 104040 ERROR 40 ;THERE WAS A READ/WRITE HEADER ERROR
4441 ;DURING 'DTE' TEST SETUP
4442 032060 000137 032376 JMP TST51 ; ABORT THE TEST
4443
4444 032064 6$: ;NOW THE HEADER HAS BEEN READ,
4445 ;*560 SECTOR CLOCKS WILL BE GIVEN -
4446 ;*GAP 11 BYTES, SYNC 1 BYTE, DATA 512, ECC 4 BYTES
4447 ;*GAP 2 BYTES, TOLERANCE 28 BYTES, EXTRA 2
4448
4449 ;*THESE 560 SECTOR CLOCKS ARE DIVIDED INTO TWO GROUPS
4450 ;*24 SECTOR CLOCKS WITH NORMAL DIAGNOSTIC DATA CLOCKS
4451 ;*AND 536 SECTOR CLOCKS WITHOUT ANY DIAGNOSTIC DATA CLOCKS
4452
4453
4454 ;*THIS GIVES 24 SECTOR CLOCKS WITH DIAGNOSTIC DATA CLOCKS
4455
4456 032064 012701 000030 MOV #24.,R1 ;LOAD SECTOR CLOCK COUNTER
4457 032070 013700 014776 MOV @RHMR,R0 ;GET RHMR ADDRESS
4458 032074 012710 000001 MCV #DMD,@R0 ;SET DIAGNOSTIC MODE
4459 032100 052710 000012 1$: BIS #MSTCK!MCLK,@R0 ;SET SECTOR CLOCK AND DATA CLOCK
4460 032104 042710 000012 BIC #MSTCK!MCLK,@R0 ;CLEAR SECTOR CLOCK AND DATA CLOCK
4461 032110 012702 000007 MOV #7,R2 ;LOAD COUNTER FOR DIAGNOSTIC DATA CLOCKS
4462 032114 052710 000002 4$: BIS #MCLK,@R0 ;SET CLOCK (DATA)
4463 032120 042710 000002 BIC #MCLK,@R0 ;CLEAR CLOCK (DATA)
4464 032124 005302 DEC R2 ;COUNT
4465 032126 001372 BNE 4$ ;BRANCH IF 7 NOT DONE
4466 032130 005301 DEC R1 ;COUNT
4467 032132 001362 BNE 1$ ;BRANCH IF 24 NOT DONE
4468
4469 ;*THIS GIVES 536 SECTOR CLOCKS WITHOUT DIAGNOSTIC DATA CLOCKS
4470
4471 032134 012701 001030 MOV #536.,R1 ;LOAD SECTOR CLOCK COUNTER
4472 032140 052710 000010 5$: BIS #MSTCK,@R0 ;SET SECTOR CLOCK
4473 032144 042710 000010 BIC #MSTCK,@R0 ;CLEAR SECTOR CLOCK
4474 032150 005301 DEC R1 ;COUNT
4475 032152 001372 BNE 5$ ;BRANCH IF 536 NOT DONE
4476
4477
4478
4479 ;*ECC PATTERN REGISTER IS NOT CHECKED
4480 032154 017737 162630 015064 MOV @RHEC2,@#EC2 ;RHEC2 IS NOT CHECKED
4481
4482
4483 ;*NOW 'DTE' SHOULD BE SET, CHANGE SAVED REGISTERS TO EXPECTED VALUES
4484
4485 032162 005737 017334 TST @RH70 ;CHECK FOR RH70 CONTROLLER
4486 032166 001412 BEQ 7$ ;SKIP RH70 CODE AND DO RH11 IF NOT
4487
4488 032170 012737 177416 015024 MOV #-24.,@#WC ;SAVED RHWC
4489 032176 012737 015254 015026 MOV #WRFROM+<14.*2>,@#BA ;SAVED RHBA
4490 032204 052737 000300 015030 BIS #IR!OR,@#CS2 ;SAVED RHCS2
4491 032212 000414 BR 8$ ;SKIP NEXT RH11 CODE
4492
4493 032214 012737 177511 015024 7$: MOV #-183.,@#WC ;SAVED RHWC
    
```

```

4494 032222 012737 015442 015026      MOV    #WRFROM+<73.*2>, @#BA ;SAVED RHBA
4495 032230 042737 000100 015030      BIC    #IR, @#CS2 ;SAVED RHCS2
4496 032236 052737 000200 015030      BIS    #OR, @#CS2 ;SAVED RHCS2
4497
4498 032244 052737 140000 015032 8$:    BIS    #SC!TRE, @#CS1 ;SAVED RHCS1
4499 032252 052737 010000 015034      BIS    #DTE, @#ER1 ;SAVED RHER1
4500 032260 012737 000201 015052      MOV    #DENVL!DMD, @#MR ;SAVED RHMR
4501 032266 052737 140000 015054      BIS    #ATA!ERR, @#DS1 ;SAVED RHDS1
4502 032274 012737 000100 015066      MOV    #100, @#LA ;SAVED RHLA
4503 032302 012737 000001 015036      MOV    #1, @#DST ;SAVED RHDST
4504 032310 0:3737 015134 015050      MOV    @#ATTENT, @#AS ;SAVED RHAS
4505
4506 ;*NOW READ & SAVE REGISTERS AGAIN SO THAT COMPARISONS
4507 ;CAN BE DONE (USING 'WRFROM' BUFFER THIS TIME)
4508
4509 032316 004037 046462      JSR    R0, @#SAVER ;READ IN SEQUENCE
4510 032322 014750      RHC   ;FROM HARDWARE REGISTER
4511 032324 016264      REINTO ;INTO CORE BUFFER LOCATION
4512 032326 000023      19. ;NUMBER OF REGISTERS TO READ
4513
4514 ;*FOR RHAS UPPER BYTE
4515 032330 113737 015051 016311      MOV    @#AS+1, @#REINTO+25 ;UPPER RHAS
4516
4517
4518 ;*COMPARE CHANGED REGISTER SNAPSHOT BEFORE COMMAND WITH
4519 ;*SNAPSHOT AFTER COMMAND
4520
4521 032336 004037 046664      JSR    R0, @#COMPAR ;COMPARE
4522 032342 015024      WC   ;CHANGED INITIAL SNAPSHOT BUFFER
4523 032344 016264      REINTO ;SNAPSHOT BUFFER AFTER COMMAND
4524 032346 000022      18. ;NUMBER OF REGISTERS TO COMPARE
4525 032350 032356      2$ ;RETURN FOR ERROR
4526 032352 032356      2$ ;SAME
4527 032354 032376      3$ ;RETURN FOR GOOD COMPARISON
4528
4529 032356 013705 052774 2$:    MOV    @#ERWORD, R5 ;GETTING READY TO INDEX
4530 032362 060505      ADD   R5, R5 ;DOUBLE ERROR WORD
4531 032364 016537 014746 045534      MOV    RHC-2(R5), @#REGADR ;FAILING REGISTER
4532 032372 104001      ERROR 1 ;IMPROPER REGISTER
4533 ;CHANGE AFTER FORCING
4534 ;'DTE' ERROR
4535 032374 000207      RTS   PC ;RETURN
4536
4537 032376 3$: ;GOOD, REGISTERS OK - GO ON TO NEXT TEST
4538
4539

```

```

4540
4541 032376 000004          TST51: SCOPE
4542 032400 012706 001000  MOV      #STACK,SP      ;RESET STACK
4543 032404 012737 000051 017330  MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
4544
4545                          ;*THESE ARE TO SET UP FOR DISKLESS USE ONLY
4546
4547 032412 012737 010000 056012  MOV      #FMT22,@#WCYL  ;FORMAT 22=16 BITWORDS AND
4548                          ;CYLINDER 0
4549 032420 005037 056014          CLR      @#WSECTR       ;TRACK=0, SECTOR=0
4550 032424 005037 056016          CLR      @#WKEY1       ;KEY1=0
4551 032430 005037 056020          CLR      @#WKEY2       ;KEY2=0
4552 032434 012737 000400 056052  MOV      #256,@#FNWORD  ;256 DATAWORDS
4553 032442 004537 047176          JSR      R5,@#CRC      ;GO TO CALCULATE CRC
4554 032446 056012          WCYL
4555 032450 056022          GCRC
4556
4557                          ;* THESE ARE REGULAR SETUPS & CHECKS
4558
4559 032452 004737 045770          JSR      PC,@#CLDISK   ;SETUP GENERAL REGISTERS
4560 032456 012777 177374 162264  MOV      #-260,@#RHWC  ;256 DATA WORDS & 4 HEADER WORDS
4561 032464 012777 015220 162260  MOV      #WRFROM,@#HBA ;STARTING ADDRESS OF BUFFER
4562 032472 005077 162264          CLR      @#RHST       ;TRACK = 0
4563                          ;SECTOR = 0
4564 032476 012777 014000 162262  MOV      #FMT22.ECI,@#RHOF ;16 BITS PER WORD
4565                          ;ECC CORRECTION INHIBITED
4566 032504 005077 162260          CLR      @#RHCA       ;CYLINDER = 0
4567 032510 004737 046024          JSR      PC,@#CHECKT   ;CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DON'T
4568 032514 104401 005123          TYPE    ,CPHALT      ;CANNOT CONTINUE TESTING IF ANY OF THE
4569 032520 000000          HALT                ;STOP THE TEST
4570 032522 013711 015172          MOV      @#WRIFOR,@#R1 ;WRITE HEADER AND DATA = 62
4571
4572                          ;*READ & SAVE REGISTERS FOR COMPARISON AFTER SIMULATED 'DTE'
4573
4574 032526 004037 046462          JSR      R0,@#SAVER   ;READ IN SEQUENCE
4575 032532 014750          RHWC                ;FROM HARDWARE REGISTER
4576 032534 015024          WC                  ;INTO CORE BUFFER LOCATION
4577 032536 000023          19.                 ;NUMBER OF REGISTERS TO READ
  
```

```
4578
4579
4580
4581
4582
4583
4584
4585 032540 012737 177777 015144      MOV    #-1,@#TESDTE    ;SET DTE TEST
4586 032546 012737 177777 015124      MOV    #-1,@#ERFLG$   ;THIS WILL BRING THE READ HEADER
4587                                     ;AND DATA PROCESS OUT AFTER THE
4588                                     ;HEADER HAS BEEN CORRECTLY READ
4589
4590 032554 004737 055636                JSR    PC,@#COMWHD     ;ISSUE 'GO', SEARCH FOR SECTOR,
4591                                     ;WRITE HEADER AND DATA.
4592
4593 032560 017737 162172 001176  SETCK3: MOV    @RHCS1,@#$TMP0 ;READ CS1 TO CHECK FOR ERRORS DURING WRITE
4594 032566 032737 100000 001176      BIT    #SC,@#$TMP0    ;TEST FOR 'SPECIAL CONDITION' - 'SC'
4595 032574 001405                BEQ    4$              ;CONTINUE TEST IF NO ERROR ('SC' - 0)
4596 032576 004737 045470                JSR    PC,@#PUTREG     ;READ & SAVE REGISTERS AGAIN IF ERROR
4597 032602 104040                ERROR  40              ;THERE WAS A READ/WRITE HEADER ERROR
4598                                     ;DURING 'DTE' TEST SETUP
4599 032604 000137 033036                JMP    TST52           ;      ABORT THE TEST AT THAT POINT
4600
4601 032610                4$:      ;NOW SECTOR HAS BEEN FOUND OK
4602
4603                                     ;*609 SECTOR CLOCKS WILL BE GIVEN,
4604                                     ;*39 BYTES FOR SECTOR GAP,
4605                                     ;*1 BYTE FOR HEADER SYNC,
4606                                     ;*8 BYTES FOR HEADER,
4607                                     ;*GAP 11 BYTES, SYNC 1 BYTE, DATA 512, ECC 4 BYTES
4608                                     ;*GAP 2 BYTES, TOLERANCE 28 BYTES, EXTRA 3
4609
4610
4611                                     ;*THIS GIVES 609 SECTOR CLOCKS WITHOUT DIAGNOSTIC DATA CLOCKS
4612
4613 032610 012701 001141                5$:      MOV    #609.,R1        ;LOAD SECTOR CLOCK COUNTER
4614 032614 052710 000010                BIS    #MSTCK,@R0     ;SET SECTOR CLOCK
4615 032620 042710 000010                BIC    #MSTCK,@R0     ;CLEAR SECTOR CLOCK
4616 032624 005301                DEC    R1              ;COUNT
4617 032626 001372                BNE    5$              ;BRANCH IF 536 NOT DONE
4618
4619                                     ;*NOW 'DTE' SHOULD BE SET, CHANGE SAVED REGISTERS
4620                                     ;TO EXPECTED VALUES
4621
4622 032630 005737 017334                TST    @#RH70         ;CHECK FOR RH70 CONTROLLER
4623 032634 001407                BEQ    6$              ;SKIP RH70 CODE AND DO RH11 IF NOT
4624
4625 032636 012737 177404 015024      MOV    #-252.,@#WC     ;SAVED RHWC
4626 032644 012737 015240 015026      MOV    #WRFROM+<8.*2>,@#BA ;SAVED RHBA
4627 032652 000406                BR     7$              ;SKIP NEXT RH11 CODE
4628
4629 032654 012737 177477 015024  6$:      MOV    #-193.,@#WC     ;SAVED RHWC
4630 032662 012737 015426 015026      MOV    #WRFROM+<67.*2>,@#BA ;SAVED RHBA
4631
4632 032670 052737 140000 015032  7$:      BIS    #SC!TRE,@#CS1   ;SAVED RHCS1
4633 032676 042737 000100 015030      BIC    #IR,@#CS2     ;SAVED RHCS2
```



```

4634 032704 052737 000200 015030 BIS #OR,@#CS2 ;SAVED RHCS2
4635 032712 052737 010000 015034 BIS #DTE,@#ER1 ;SAVED RHER1
4636 032720 012737 000401 015052 MOV #401,@#MR ;SAVED RHMR
4637 032726 052737 140000 015054 BIS #ATA'ERR,@#DS1 ;SAVED RHDS1
4638 032734 012737 000100 015066 MOV #100,@#LA ;SAVED RHLA
4639 032742 012737 000001 015036 MOV #1,@#DST ;SAVED RHDST
4640 032750 013737 015134 015050 MOV @#ATTENT,@#AS ;SAVED RHAS
4641
4642 ;*NOW READ & SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN BE DONE
4643
4644 032756 004037 046462 JSR RO,@#SAVER ;READ IN SEQUENCE
4645 032762 014750 RHWC ;FROM HARDWARE REGISTER
4646 032764 016264 REINTO ;INTO CORE BUFFER LOCATION
4647 032766 000023 19. ;NUMBER OF REGISTERS TO READ
4648
4649 ;*FOR RHAS UPPER BYTE
4650 032770 113737 015051 016311 MOVB @#AS+1,@#REINTO+25 ;UPPER RHAS
4651
4652 ;*COMPARE CHANGED REGISTER SNAPSHOT BEFORE COMMAND
4653 ;*WITH REGISTER SNAPSHOT AFTER COMMAND
4654
4655 032776 004037 046664 JSR RO,@#COMPAR ;COMPARE
4656 033002 015024 WC ;CHANGED REGISTER SNAPSHOT
4657 033004 016264 REINTO ;SNAPSHOT AFTER COMMAND
4658 033006 000022 18. ;NUMBER OF REGISTERS TO COMPARE
4659 033010 033016 2$ ;RETURN FOR ERROR
4660 033012 033016 2$ ;SAME
4661 033014 033036 3$ ;RETURN FOR GOOD COMPARISON
4662
4663 033016 013705 052774 2$: MOV @#ERWORD,R5 ;GETTING READY TO INDEX
4664 033022 060505 ADD R5,R5 ;DOUBLE ERROR WORD
4665 033024 016537 014746 045534 MOV RHWC-2(R5),@#REGADR ;FAILING REGISTER
4666 033032 104001 ERROR 1 ;IMPROPER REGISTER
4667 ;CHANGE AFTER FORCING
4668 ;'DTE' ERROR
4669 033034 000207 RTS PC ;RETURN
4670
4671 033036 3$: ;GOOD, REGISTERS COMPARE OK
4672 ;GO ON TO THE NEXT TEST
4673

```

```
4674 033036 000004          TST52: SCOPE
4675 033040 012706 001000    MOV      #STACK,SP      ;RESET STACK
4676 033044 012737 000052 017330  MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
4677 033052 004737 045770    JSR      PC,@#CLDISK    ;SETUP GENERAL REGISTERS & CLEAR
4678                                ;THE DRIVE
4679 033056 012737 000026 015146  MOV      #22.,@#TAGDTE  ;22 SECTORS
4680                                ;THIS TEST REPEATS
4681                                ;ITSELF 22 TIMES
4682
4683                                ;*THE FOLLOWING INITIALIZES FOR SECTOR 0
4684
4685 033064 005037 033170    CLR      @#SS3+2        ;HEADER (SECTOR)
4686 033070 012737 000025 033174  MOV      #21.,@#SS4+2   ;HEADER (KEY1)
4687 033076 012737 000025 033200  MOV      #21.,@#SS5+2   ;HEADER (KEY2)
4688 033104 005037 033226    CLR      @#SS7+2        ;DATA (SECTOR)
4689 033110 005037 033310    CLR      @#SS10+2       ;DATA
4690 033114 005037 033340    CLR      @#SS12+2       ;SECTOR (SIMULATED DISK)
4691 033120 012737 000025 033346  MOV      #21.,@#SS13+2  ;KEY1 (SIMULATED DISK)
4692 033126 012737 000025 033354  MOV      #21.,@#SS14+2  ;KEY2 (SIMULATED DISK)
4693 033134 005037 033414    CLR      @#SS15+2       ;SECTOR (RHDST)
4694
4695                                ;*CLEAR SIMULATED DISK AREA
4696 033140          SS1:
4697 033140 012700 054474    1$:  MOV      #SECGAP,R0    ;POINTER
4698 033144 012701 000460    MOV      #304.,R1      ;COUNTER
4699 033150 005020          2$:  CLR      (R0)+          ;CLEAR SIMULATED DISK AREA
4700 033152 005301          DEC      R1             ;COUNT
4701 033154 001375          BNE     2$
4702
4703                                ;*SETUP WRITE FROM BUFFER
4704                                MOV      #WRFROM,R0
4705 033156 012700 015220
4706
4707                                ;*HEADER
4708 033162 012720 010000    MOV      #FMT22,(R0)+  ;FORMAT 16 BITS PER WORD
4709                                ;CYLINDER 0
4710 033166 012720 000000    SS3:  MOV      #0,(R0)+      ;SECTOR TO VARY
4711 033172 012720 000025    SS4:  MOV      #21.,(R0)+   ;KEY1 TO VARY
4712 033176 012720 000025    SS5:  MOV      #21.,(R0)+   ;KEY2 TO VARY
4713
4714                                ;*DATA IN WRITE FROM BUFFER ALTHOUGH THIS IS DATA AND NOT
4715                                ;*HEADER, THE SECTOR WITH SYNC BYTES WILL BE GIVEN AS DATA.
4716
4717                                ;*DATA IS - 19 WORDS OF ZEROS - SYNC WORDS, 4 HEADER WORDS
4718                                ;*1 CRC WORD, 5 WORDS OF ZEROS, 1 SYNC WORD, 100 ZEROS
4719                                ;*(DATA), 1 SYNC WORD, 70 SECTOR NUMBER TO VARY
4720
4721 033202 012705 000023          6$:  MOV      #19.,R5        ;COUNTER
4722 033206 005020          CLR      (R0)+          ;19 ZEROS
4723 033210 005305          DEC      R5             ;COUNT
4724 033212 001375          BNE     6$             ;19 DONE?
4725 033214 013720 052756    MOV      @#RSYNC,(R0)+ ;SYNC = 14400
4726 033220 012720 010000    MOV      #FMT22,(R0)+ ;CYLINDER 0
4727 033224 012720 000000    SS7:  MOV      #0,(R0)+      ;SECTOR TO VARY
4728 033230 005020          CLR      (R0)+
4729 033232 005020          CLR      (R0)+
```

```
4730 033234 004537 047176 JSR R5,@#CRC ;CALCULATE CRC FOR ABOVE 4 WORDS
4731 033240 015270 WRFROM+50 ;4 WORDS START FROM HERE
4732 033242 015300 WRFROM+60 ;PUT CRC HERE
4733
4734 033244 005720 TST (R0)+ ;INCREMENT R0
4735
4736 033246 012705 000005 8$: MOV #5.,R5
4737 033252 005020 CLR (R0)+ ;5 WORDS OF ZEROS
4738 033254 005305 DEC R5 ;COUNT
4739 033256 001375 BNE 8$ ;BRANCH IF 5 NOT DONE
4740
4741 033260 013720 052756 MOV @#RSYNC,(R0)+ ;SYNC = 14400
4742
4743 033264 012705 000144 9$: MOV #100.,R5
4744 033270 005020 CLR (R0)+ ;100 WORDS OF ZEROS
4745 033272 005305 DEC R5
4746 033274 001375 BNE 9$
4747
4748 033276 013720 052756 MOV @#RSYNC,(R0)+ ;SYNC = 14400
4749 033302 012705 000106 MOV #70.,R5
4750 033306 012720 000000 SS10: MOV #0,(R0)+ ;SECTOR TO VARY
4751 033312 005305 DEC R5
4752 033314 001374 BNE SS10
4753
4754 ;*CLEAR REST OF 256 WORDS THAT IS 54 WORDS OF ZEROS
4755
4756 033316 012705 000066 11$: MOV #54.,R5
4757 033322 005020 CLR (R0)+
4758 033324 005305 DEC R5
4759 033326 001375 BNE 11$
4760
4761 ;*THESE ARE TO BE SET UP FOR DISKLESS USE ONLY
4762
4763 033330 012737 010000 056012 MOV #FMT22,@#WCYL ;FORMAT = 16 BIT WORDS
4764 ;CYLINDER = 0
4765 033336 012737 000000 056014 SS12: MOV #0,@#WSECTR ;SECTOR TO VARY
4766 033344 012737 000025 056016 SS13: MOV #21.,@#WKEY1 ;KEY1 TO VARY
4767 033352 012737 000025 056020 SS14: MOV #21.,@#WKEY2 ;KEY2 TO VARY
4768 033360 012737 000312 056052 MOV #202.,@#FNWORD ;202 DATA WORDS
4769 033366 004537 047176 JSR R5,@#CRC ;CALCULATE CRC
4770 033372 056012 WCYL ;FIRST WORD
4771 033374 056022 GCRC ;PUT HERE
4772
4773 ;*THESE ARE REGULAR SETUPS
4774
4775 033376 012777 177400 161344 MOV #-256.,@RHWC ;202 DATA, 4 HEADER
4776 033404 012777 015220 161340 MOV #WRFROM,@RHBA ;FILL BUS ADDRESS
4777 033412 012777 000000 161342 SS15: MOV #0,@RH DST ;SECTOR TO VARY
4778 033420 013777 015172 161330 MOV @#WRIFOR,@RHCS1 ;GET READY TO DO
4779 ;WRITE HEADER AND DATA
4780 ;WITH 62 FUNCTION CODE IN RHCS1
4781 033426 012777 010000 161332 MOV #FMT22,@RHOF ;16 BITS PER WORD FORMAT
4782 033434 005077 161330 CLR @RHCA ;CYLINDER = 0
4783
4784 033440 005037 015124 CLR @#ERFLG$ ;CLEAR ERROR FLAG
4785
```

```
4786 033444 004737 046024 JSR PC,@#CHECKT ;CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DON'T
4787 033450 104401 005123 TYPE ,CPHALT ;CANNOT CONTINUE TESTING IF ANY OF THE
4788 033454 000000 HALT ;STOP THE TEST
4789
4790 033456 004737 055636 JSR PC,@#COMWHD ;ISSUE 'GO', COUNT SECTOR CLOCKS,
4791 ;WRITE HEADER AND DATA
4792 033462 005737 015124 TST @#ERFLG$ ;HAVE ANY ERRORS OCCURRED ?
4793 033466 001051 BNE TST53 ; EXIT IF YES-----)
4794
4795 033470 004737 046214 JSR PC,@#CHECKE ;CHECK THAT DVA,RDY,DPR,DRY = 1
4796 033474 104401 005123 TYPE ,CPHALT ;CANNOT CONTINUE IF THEY DON'T
4797 033500 000000 HALT ;STOP THE TEST AND RESTART PROGRAM
4798
4799 ;*NOW COMPARE 'DISK' BUFFER WITH 'REINTO' BUFFER
4800 033502 004037 046664 JSR RO,@#COMPAR ;CHECK
4801 033506 015230 WRFROM+8. ;GOOD BUFFER
4802 033510 054572 DISK ;TEST BUFFER
4803 033512 000400 256. ;NUMBER OF WORDS
4804 033514 033522 16$ ;RETURN POINT FOR ERROR HEADER
4805 033516 033526 17$ ;RETURN POINT FOR ERROR DATA
4806 033520 033532 18$ ;RETURN FOR GOOD COMPARISON
4807 033522 104007 16$: ERROR 7
4808 033524 000207 RTS PC
4809 033526 104010 17$: ERROR 10
4810 033530 000207 RTS PC
4811
4812 ;*THE FOLLOWING INCREMENTS ARE TO CHANGE THE ABOVE SET UP
4813 ;*TO WRITE ON THE NEXT SECTOR
4814
4815 033532 005237 033170 18$: INC @#SS3+2 ;HEADER (SECTOR)
4816 033536 005337 033174 DEC @#SS4+2 ;HEADER (KEY1)
4817 033542 005337 033200 DEC @#SS5+2 ;HEADER (KEY2)
4818 033546 005237 033226 INC @#SS7+2 ;DATA (SECTOR)
4819 033552 005237 033310 INC @#SS10+2 ;DATA
4820 033556 005237 033340 INC @#SS12+2 ;SECTOR (SIMULATED DISK)
4821 033562 005337 033346 DEC @#SS13+2 ;KEY1 (SIMULATED DISK)
4822 033566 005337 033354 DEC @#SS14+2 ;KEY2 (SIMULATED DISK)
4823 033572 005237 033414 INC @#SS15+2 ;SECTOR (RHDST)
4824
4825 033576 005337 015146 SS2: DEC @#TAGDTE ;COUNT DOWN FOR 22 SECTORS
4826 033602 001001 BNE 1$ ;BRANCH IF 22 SECTORS NOT DONE
4827 033604 000402 BR TST53 ;ALL DONE - GO TO NEXT TEST
4828 033606 000137 033140 1$: JMP @#SS1 ;GO BACK TO NEXT SECTOR
4829
4830
4831
```

```

4832          .SBTTL  DATA TRANSFER TESTS USING ECC
4833
4834
4835
4836 033612 000004          TST53: SCOPE
4837 033614 012706 001000      MOV      #STACK,SP          ;RESET STACK
4838
4839 033620 012737 000053 017330      MOV      #TTNO,@#TSTNM      ;THIS SAVES TEST NUMBER
4840 033626 012700 054474          MOV      #SECGAP,RO         ;POINTER
4841 033632 012701 000402          MOV      #258.,R1          ;COUNTER
4842 033636 012720 177777          1$:     MOV      #-1,(R0)+      ;FILL SIMULATOR DISK WITH ONES
4843 033642 005301          DEC      R1
4844 033644 001374          BNE     1$
4845 033646 004737 045770          JSR     PC,@#CLDISK        ;THIS IS USED TO SET GENERAL REGISTERS
4846
4847          ;*THESE ARE FOR ECC TEST ONLY
4848
4849 033652 012737 177777 015142      MOV      #-1,@#TSECC        ;THIS IS AN ECC TEST
4850 033660 005037 050476          CLR     @#POSITI          ;CLEAR ERROR POSITION COUNTER
4851 033664 013737 050472 050474      MOV      @#NCODE,@#NCOUNT    ;TEMPORARY N-CODE COUNTER
4852 033672 013737 050500 050506      MOV      @#HARDER,@#HADTMP    ;TEMPORARY HARD ERROR COUNTER
4853 033700 005037 050464          CLR     @#GECC1           ;ECC LOW ORDER TO BE GENERATED
4854 033704 005037 050466          CLR     @#GECC2           ;ECC HIGH ORDER TO BE GENERATED
4855 033710 005037 050502          CLR     @#DATENV          ;CLEAR DATA ENVELOPE CLOCK COUNT
4856 033714 005037 050504          CLR     @#ZCODE           ;CLEAR LEADING ZEROS CLOCK COUNT
4857
4858
4859
4860
4861          ;*THESE ARE TO BE SETUP FOR DISKLESS USE ONLY
4862
4863 033720 012737 010000 056012      MOV      #FMT22,@#WCYL      ;FORMAT22=16BIT WORDS AND
4864          ;CYLINDER 0
4865 033726 012737 000001 056014      MOV      #1,@#WSECTR        ;TRACK=0, SECTOR 1
4866 033734 005037 056016          CLR     @#WKEY1           ;KEY1=0
4867 033740 005037 056020          CLR     @#WKEY2           ;KEY2=0
4868 033744 012737 000400 056052      MOV      #256.,@#FNWORD     ;256 DATA WORDS
4869 033752 004537 047176          JSR     R5,@#CRC           ;GO TO CALCULATE CRC
4870 033756 056012          WCYL
4871 033760 056022          GCRC
4872
4873          ;*THESE ARE REGULAR SETUPS
4874
4875 033762 012777 177374 160760      MOV      #-260.,@#RHWC      ;256 DATA WORDS 4 HEADER WORDS
4876 033770 012700 015220          MOV      #WRFROM,RO         ;THESE TWO INSTRUCTIONS GETS
4877 033774 010077 160752          MOV      RO,@#RHBA          ;ADDR. OF WRFROM INTO RO AND
4878          ;BUS ADDRESS REGISTER
4879 034000 012720 010000          MOV      #FMT22,(R0)+      ;FORMAT=16 BIT WORDS
4880          ;CYLINDER=0
4881 034004 012720 000001          2$:     MOV      #1,(R0)+        ;TRACK=0, SECTOR=1, KEYS=0
4882 034010 005020          CLR     (R0)+             ;KEY1=0
4883 034012 005020          CLR     (R0)+             ;KEY2=0
4884 034014 012705 000400          MOV      #256.,R5          ;COUNTER
4885 034020 012720 000000          3$:     MOV      #0,(R0)+        ;MOVE ALL ZEROS FOR DATA
4886 034024 005305          DEC     R5
4887 034026 001374          BNE     3$                ;BRANCH IF DATA NOT COMPLETE

```

```

4888 034030 012777 000001 160724      MOV    #1, @RH DST      ; TRACK=0 SECTOR=1
4889
4890 034036 004737 046024      JSR    PC, @#CHECKT    ; CHECK DVA, RDY, DPR, DRY - 1 AND OTHERS DON'T
4891 034042 104401 005123      TYPE  ,CPHALT          ; CANNOT CONTINUE TESTING IF ANY OF THE
4892 034046 000000      HALT                    ; STOP THE TEST
4893
4894 034050 013711 015172      MOV    @#WRIFOR, @R1    ; GET READY FOR WRITE HEADER AND
4895                                ; DATA WITH 62 IN RHCS1
4896 034054 005037 015124      CLR    @#ERFLGS         ; CLEAR ERROR FLAG
4897 034060 012777 010000 160700    MOV    #FMT22, @RHOF    ; FORMAT BIT-1 (16 BIT WORDS)
4898 034066 005077 160676      CLR    @RHCA            ; CYLINDER =0
4899 034072 004737 055636      JSR    PC, @#COMWHD     ; WRITE HEADER AND DATA
4900
4901                                ; *IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
4902                                ; *FROM THE 'COMWHD' ROUTINE THAT MEANS ALL HEADER ON DISK
4903                                ; *IS GOOD IE. ONLY DATA IS TO BE CHECKED TO SEE IF THEY ARE
4904                                ; *ALL ZEROS AND WRITE DATA GAP AND TOLERANCE GAP TO SEE IF
4905                                ; *THEY ARE ALL ZEROS
4906
4907 034076 005737 015124      TST    @#ERFLGS         ; HAS ANY ERRORS OCCURED?
4908                                ; IF WRITE ERROR OCCURS ECC IS NOT CHECKED
4909 034102 001056      BNE    TST54            ; BRANCH IF YES
4910
4911                                ; *COMPARE SOFTWARE GENERATED ECC WITH THAT GENERATED BY HARDWARE
4912
4913
4914 034104 023737 050464 055572    CMP    @#GECC1, @#WECC1 ; COMPARE SOFTWARE ECC WITH HARDWARE ECC
4915 034112 001402      BEQ    6$              ; BRANCH IF GOOD
4916 034114 104031      ERROR  31              ; LOW ORDER ECC IN ERROR
4917 034116 000405      BR     7$              ; BRANCH TO CONTINUE
4918 034120 023737 050466 055574 6$:    CMP    @#GECC2, @#WECC2 ; COMPARE SOFTWARE ECC WITH HARDWARE ECC
4919 034126 001401      BEQ    7$              ; BRANCH IF GOOD
4920 034130 104031      ERROR  31              ; HIGH ORDER ECC IN ERROR
4921
4922
4923 034132 004737 046214 7$:      JSR    PC, @#CHECKE     ; CHECK THAT DVA, RDY, DPR, DRY - 1
4924 034132 004737 046214      TYPE  ,CPHALT          ; CANNOT CONTINUE IF THEY DON'T
4925 034136 104401 005123      HALT                    ; STOP THE TEST AND RESTART PROGRAM
4926 034142 000000
4927
4928
4929                                ; *FILL 'REINTO' BUFFER WITH EXPECTED DATA
4930
4931
4932 034144 004037 045706      JSR    RO, @#CLAREA     ; FILL REINTO BUFFER
4933 034150 016264      REINTO ; FROM
4934 034152 017262      REINTO+<255.*2> ; TO
4935 034154 000000      .WORD  0               ; DATA
4936
4937 034156 013737 050464 017264    MOV    @#GECC1, @#REINTO+<256.*2>; FILL ECC1
4938 034164 013737 050466 017266    MOV    @#GECC2, @#REINTO+<257.*2>; FILL ECC2
4939 034172 004037 045706      JSR    RO, @#CLAREA     ; FILL REST
4940 034176 017270      REINTO+<258.*2> ; FROM
4941 034200 017324      REINTO+<272.*2> ; TO
4942 034202 000000      0                       ; DATA
4943

```



```
4973
4974 034240 000004          TST54: SCOPE
4975 034242 012706 001000  MOV      #STACK,SP      ;RESET STACK
4976
4977 034246 012737 000054 017330  MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
4978
4979
4980          :          SETUP FOR WHAT IS TO BE READ
4981          :          HEADER CRC IS RESTORED FROM A SUBROUTINE
4982
4983 034254 012746 000000      MOV      #0,-(SP)        ;DATA TO BE READ
4984 034260 012705 000400      MOV      #256.,R5       ;COUNTER
4985 034264 012700 054572      MOV      #DISK,R0       ;START OF SIMULATED DISK DATA
4986 034270 011620          1$:  MOV      (SP),(R0)+      ;MOVE IN DATA ON TO SIMULATED DISK
4987 034272 005305          DEC      R5              ;COUNT
4988 034274 001375          BNE     1$              ;BRANCH IF 256 NOT COMPLETE
4989 034276 005726          TST     (SP)+           ;UNDO -(SP)
4990 034300 022020          CMP     (R0)+,(R0)+    ;JUMP OVER THE TWO ECC WORDS
4991 034302 012705 000017      MOV      #15.,R5       ;1 DATA GAP
4992
4993 034306 005020          2$:  CLR     (R0)+           ;CLEAR DATA GAP, AND
4994 034310 005305          DEC     R5              ;TOLERANCE GAP
4995 034312 001375          BNE     2$              ;BRANCH IF NOT COMPLETE
4996
4997
4998 034314 004737 051260      JSR     PC,@#FILLEC    ;INSERT THE TWO ECC WORDS ON THE DISK
4999
5000
5001          ;*THESE ARE FOR ECC TEST ONLY
5002
5003 034320 012737 177777 015142  MOV      #-1,@#TSECC    ;THIS IS AN ECC TEST
5004 034326 005037 050476      CLR     @#POSITI       ;CLEAR ERROR POSITION COUNTER
5005 034332 013737 050472 050474  MOV      @#NCODE,@#NCOUNT ;TEMPORARY N-CODE COUNTER
5006 034340 013737 050500 050506  MOV      @#HARDER,@#HADTMP ;TEMPORARY HARD ERROR COUNTER
5007 034346 005037 050464      CLR     @#GECC1        ;ECC LOW ORDER TO BE GENERATED
5008 034352 005037 050466      CLR     @#GECC2        ;ECC HIGH ORDER TO BE GENERATED
5009 034356 005037 050502      CLR     @#DATENV       ;CLEAR DATA ENVELOPE CLOCK COUNT
5010 034362 005037 050504      CLR     @#ZCODE        ;CLEAR LEADING ZEROS CLOCK COUNT
5011
5012
5013          ;*THESE ARE TO SETUP FOR DISKLESS USE ONLY
5014
5015 034366 012737 010000 052654  MOV      #FMT22,@#CYL   ;16 BITS PER WORD
5016
5017 034374 112737 000000 052657  MOVB    #0,@#SECOTR+1   ;CYLINDER 0, FORMAT 16 BITS
5018 034402 112737 000000 052656  MOVB    #0,@#SECOTR     ;TRACK 0
5019 034410 012737 000000 052660  MOV     #0,@#KEY1       ;SECTOR 0
5020 034416 012737 000000 052662  MOV     #0,@#KEY2       ;KEY1=0
5021 034424 012737 000400 052734  MOV     #0,@#KEY2       ;KEY2=0
5022 034432 005037 052664      MOV     #256.,@#DAWORD  ;NO. OF DATA WORDS
5023 034436 004537 047176      CLR     @#X             ;THIS IS A READ COMMAND
5024 034442 052654          JSR     R5,@#CRC        ;GO TO CALCULATE CRC
5025 034444 054554          CYL
5026
5027
5028          WCRC
```



```
5029
5030 ;*THESE ARE REGULAR SETUPS
5031
5032 034446 004737 045770 JSR @#CLDISK ;SETUP GENERAL REGISTERS
5033 034452 012777 177374 160270 MOV #-256.-4.,@RHWC ;256. DATA 4 HEADER WORDS
5034 034460 012777 016264 160264 MOV #REINTO,@RHBA ;STARTING ADDRESS OF READ BUFFER
5035 034466 112746 000000 MOVB #0,-(SP) ;IN LOWER BYTE GET SECTOR
5036 034472 112766 000000 000001 MOVB #0,1(SP) ;GET TRACK IN HIGHER BYTE
5037 034500 012677 160256 MOV (SP)+,@RHDST ;TRACK/SECTOR IN RHDST
5038 034504 012777 010000 160254 MOV #FMT22,@RHOF ;16 BITS PER WORD
5039 ;ECC CORRECTION NOT INHIBIT
5040 ;BECAUSE ECC IS NOT GOING
5041 ;TO BE CHECKED
5042 034512 005077 160252 CLR @RHCA ;CYLINDER 0
5043
5044 034516 004737 046024 JSR PC,@#CHECKT ;CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DON'T
5045 034522 104401 005123 TYPE ,CPHALT ;CANNOT CONTINUE TESTING IF ANY OF THE
5046 034526 000000 HALT ;STOP THE TEST
5047
5048 034530 013711 015176 MOV @#REFOR,@R1 ;READ HEADER AND DATA=72
5049 034534 005037 015124 CLR @#ERFLG$ ;CLEAR ERROR FLAG
5050 034540 004737 052514 JSR PC,@#COMHD ;READ HEADER AND DATA
5051 ;IF THERE ARE READ ERRORS THEN
5052 ;ECC WILL NOT BE CHECKED
5053
5054
5055 ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
5056 ;*FROM THE 'COMHD' ROUTINE THAT MEANS SECTOR GAP,
5057 ;*FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
5058 ;*SYNC BYTE HAVE GONE BY AND SYNCs WERE CORRECTLY
5059 ;*DETECTED
5060 ;*HEADER AND DATA ARE TO BE CHECKED.
5061 ;*IN CHECKING READ DATA THE WRITE FROM BUFFER
5062 ;*'WRFROM' IS FILLED WITH EXPECTED DATA AND
5063 ;*COMPARISONS ARE MADE
5064
5065 034544 005737 015124 TST @#ERFLG$ ;ANY ERRORS ALREADY THERE
5066 034550 001102 BNE TST55 ;BRANCH IF YES
5067 034552 004737 045470 JSR PC,@#PUTREG ;SAVE REGISTERS
5068 034556 005737 015034 TST @#ER1 ;NO ERRORS SHOULD BE SET
5069 034562 001401 BEQ 6$ ;BRANCH IF NO ERRORS SET
5070 034564 104032 ERROR 32 ;32 BIT ECC REGISTER SHOULD BE ZERO
5071 ;ONLY 11 OF THE 32 BITS CAN BE SEEN
5072 ;IN THE PATERN REGISTER
5073 ;DCK SHOULD BE SET IN RHER1
5074 034566 013746 050464 6$: MOV @#GECC1,-(SP) ;GET PATTERN REGISTER
5075 034572 042716 174000 BIC #174000,(SP) ;KEEP ONLY 11 BITS
5076 034576 022637 015064 CMP (SP)+,@#EC2 ;COMPARE PATTERN REGISTER
5077 034602 001401 BEQ 7$ ;BRANCH IF GOOD
5078 034604 104032 ERROR 32 ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
5079
5080
5081
5082
5083 ;*ADD 16 MAINTENANCE CLOCKS TO
5084 ;*BRING EBL DOWN
```

```
5085
5086 034606 012700 000020 7$: MOV #16.,R0 ;COUNTER
5087 034612 052777 000002 160156 8$: BIS #MCLK,@RHMR ;SET CLOCK
5088 034620 042777 000002 160150 BIC #MCLK,@RHMR ;CLEAR CLOCK
5089 034626 005300 DEC R0 ;COUNT
5090 034630 001370 BNE 8$ ;BRANCH IF 16 CLOCKS NOT DONE
5091 034632 004737 046214 JSR PC,@#CHECKE ;CHECK THAT DVA,RDY,DPR,DRY = 1
5092 034636 10440; 005123 TYPE ,CPHALT ;CANNOT CONTINUE IF THEY DON'T
5093 034642 000000 HALT ;STOP THE TEST AND RESTART PROGRAM
5094 034644 012700 015220 MOV #WRFROM,R0 ;GETTING READY TO FILL EXPECTED DATA
5095 034650 012720 010000 MOV #0!FMT22,(R0)+ ;CYLINDER 0
5096 034654 112746 000000 MOV#B #0,-(SP) ;IN LOWER BYTE GET SECTOR
5097 034660 112766 000000 000001 MOV#B #0,1(SP) ;GET TRACK IN HIGHER BYTE
5098 034666 012620 MOV (SP)+,(R0)+ ;GET TRACK/SECTOR IN BUFFER
5099 034670 012720 000000 MOV #0,(R0)+ ;KEY1 IN BUFFER
5100 034674 012720 000000 MOV #0,(R0)+ ;KEY2 IN BUFFER
5101 034700 012701 000400 MOV #256.,R1 ;DATA WORD COUNTER
5102 034704 012702 000000 MOV #0,R2 ;DATA
5103 034710 010220 3$: MOV R2,(R0)+ ;DATA INTO BUFFER
5104 034712 005301 DEC R1 ;COUNT
5105 034714 001375 BNE 3$ ;BRANCH IF 256 NOT DONE
5106
5107
5108 034716 005037 015124 CLR @#ERFLG$ ;CLEAR ERROR FLAG
5109 034722 004737 045470 JSR PC,@#PUTREG ;SAVE REGISTERS
5110
5111
5112
5113 ;*NOW READ DATA BUFFER WILL BE CHECKED
5114
5115 034726 004037 046664 JSR R0,@#COMPAR ;CHECK
5116 034732 015220 WRFROM ;GOOD BUFFER
5117 034734 016264 REINTO ;TEST BUFFER
5118 034736 000404 4+256. ;NUMBER OF WORDS CHECKED
5119 034740 034746 4$ ;RETURN POINT FOR ERROR HEADER
5120 034742 034752 5$ ;RETURN POINT FOR ERROR DATA
5121 034744 034756 TST55 ;RETURN FOR GOOD COMPARISON
5122 034746 104004 4$: ERROR 4 ;READ NEXT ERROR
5123 034750 000207 RTS PC ;RETURN TO 'COMPAR'
5124 034752 104005 5$: ERROR 5 ;WORD NOS 1 TO 4 ARE
5125 ;HEADER WORDS
5126 ;5 TO 260 ARE DATA WORDS
5127 034754 000207 RTS PC ;RETURN TO 'COMPAR'
5128
5129
5130
5131
```

```

5132
5133 034756 000004          TST55: SCOPE
5134 034760 012706 001000  MOV      #STACK,SP      ;RESET STACK
5135
5136 034764 012737 000055 017330  MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
5137
5138
5139                ;*SETUP FOR WHAT IS TO BE READ
5140                ;*HEADER CRC IS RESTORED FROM A SUBROUTINE
5141
5142 034772 012746 000000          MOV      #0,-(SP)        ;DATA TO BE READ
5143 034776 012705 000400          MOV      #256.,R5       ;COUNTER
5144 035002 012700 054572          MOV      #DISK,R0       ;START OF SIMULATED DISK DATA
5145 035006 011620          1$:  MOV      (SP),(R0)+     ;MOVE IN DATA ON TO SIMULATED DISK
5146 035010 005305          DEC      R5             ;COUNT
5147 035012 001375          BNE     1$             ;BRANCH IF 256 NOT COMPLETE
5148 035014 005726          TST     (SP)+         ;UNDO -(SP)
5149 035016 022020          CMP     (R0)+,(R0)+   ;JUMP OVER THE TWO ECC WORDS
5150 035020 012705 000017          MOV      #15., R5     ;1 DATA GAP
5151
5152 035024 005020          2$:  CLR     (R0)+         ;CLEAR DATA GAP, AND
5153 035026 005305          DEC     R5            ;TOLERANCE GAP
5154 035030 001375          BNE     2$           ;BRANCH IF NOT COMPLETE
5155
5156
5157 035032 004737 051260          JSR     PC,@#FILLEC   ;INSERT ECC IN PROPER PLACE ON DISK
5158
5159
5160
5161                ;*THESE ARE FOR ECC TEST ONLY
5162
5163 035036 012737 177777 015142  MOV      #-1,@#TSECC   ;THIS IS AN ECC TEST
5164 035044 005037 050476          CLR     @#POSITI     ;CLEAR ERROR POSITION COUNTER
5165 035050 013737 050472 050474  MOV      @#NCODE,@#NCOUNT ;TEMPORARY N-CODE COUNTER
5166 035056 013737 050500 050506  MOV      @#HARDER,@#HADTMP ;TEMPORARY HARD ERROR COUNTER
5167 035064 005037 050464          CLR     @#GECC1      ;ECC LOW ORDER TO BE GENERATED
5168 035070 005037 050466          CLR     @#GECC2      ;ECC HIGH ORDER TO BE GENERATED
5169 035074 005037 050502          CLR     @#DATENV     ;CLEAR DATA ENVELOPE CLOCK COUNT
5170 035100 005037 050504          CLR     @#ZCODE      ;CLEAR LEADING ZEROS CLOCK COUNT
5171
5172
5173                ;*THESE ARE TO SETUP FOR DISKLESS USE ONLY
5174
5175 035104 012737 010000 052654  MOV      #FMT22,@#CYL ;16 BITS PER WORD
5176
5177 035112 112737 000000 052657  MOVB    #0,@#SECOTR+1 ;CYLINDER 0, FORMAT 16 BITS
5178 035120 112737 000000 052656  MOVB    #0,@#SECOTR   ;TRACK 0
5179 035126 012737 000000 052660  MOV     #0,@#KEY1     ;SECTOR 0
5180 035134 012737 000000 052662  MOV     #0,@#KEY2     ;KEY1=0
5181 035142 012737 000400 052734  MOV     #256., @#DAWORD ;KEY2=0
5182 035150 005037 052664          CLR     @#X           ;NO. OF DATA WORDS
5183 035154 004537 047176          JSR     R5,@#CRC     ;THIS IS A READ COMMAND
5184 035160 052654          CYL                    ;GO TO CALCULATE CRC
5185 035162 054554          WCRC
5186
5187

```

```

5188 ;*THIS IS TO INSERT ERROR
5189 ;*THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
5190 ;*THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
5191 ;*THIS MOVE
5192
5193 035164 012737 100000 054574 MOV #100000,@#DISK+2 ;FORCE ERROR ON BIT NUMBER 32
5194 ;SO ERROR POSITION REGISTER WILL SHOW
5195 ;22
5196 035172 012737 000026 035346 MOV #22.,@#8$ ;INSERT POSITION REG.
5197
5198 ;*THESE ARE REGULAR SETUPS
5199
5200
5201 035200 004737 045770 JSR PC,@#CLDISK ;SETUP GENERAL REGISTERS
5202 035204 012777 177374 157536 MOV #-256.-4.,@RHWC ;256. DATA 4 HEADER WORDS
5203 035212 012777 016264 157532 MOV #REINTO,@RHBA ;STARTING ADDRESS OF READ BUFFER
5204 035220 112746 000000 MOVB #0,-(SP) ;IN LOWER BYTE GET SECTOR
5205 035224 112766 000000 000001 MOVB #0,1(SP) ;GET TRACK IN HIGHER BYTE
5206 035232 012677 157524 MOV (SP)+,@RHDST ;TRACK/SECTOR IN RHDST
5207 035236 012777 010000 157522 MOV #FMT22,@RHOF ;16 BITS PER WORD
5208 ;ECC CORRECTION NOT INHIBIT
5209 ;BECAUSE ECC IS NOT GOING
5210 ;TO BE CHECKED
5211 035244 005077 157520 CLR @RHCA ;CYLINDER 0
5212
5213 035250 004737 046024 JSR PC,@#CHECKT ;CHECK DVA,RDY,DPR,DRY 1 AND OTHERS DON'T
5214 035254 104401 005123 TYPE ,CPHALT ;CANNOT CONTINUE TESTING IF ANY OF THE
5215 035260 000000 HALT ;STOP THE TEST
5216
5217 035262 013711 015176 MOV @#REFOR,@R1 ;READ HEADER AND DATA=72
5218 035266 005037 015124 CLR @#ERFLG$ ;CLEAR ERROR FLAG
5219 035272 004737 052514 JSR PC,@#COMHD ;READ HEADER AND DATA
5220 ;IF THERE ARE READ ERRORS THEN
5221 ;ECC WILL NOT BE CHECKED
5222
5223
5224 ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
5225 ;*FROM THE 'COMHD' ROUTINE THAT MEANS SECTOR GAP,
5226 ;*FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
5227 ;*SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
5228 ;*DETECTED
5229 ;*HEADER AND DATA ARE TO BE CHECKED.
5230 ;*IN CHECKING READ DATA THE WRITE FROM BUFFER
5231 ;*'WRFROM' IS FILLED WITH EXPECTED DATA AND
5232 ;*COMPARISONS ARE MADE
5233
5234 035276 005737 015124 TST @#ERFLG$ ;ANY ERRORS ALREADY THERE
5235 035302 001077 BNE TST56 ;BRANCH IF YES
5236 035304 004737 045470 JSR PC,@#PUTREG ;SAVE REGISTERS
5237 035310 022737 100000 015034 CMP #DCK,@#ER1 ;ONLY DATA CHECK ERROR SHOULD BE SET
5238 035316 001401 BEQ 6$ ;BRANCH IF YES
5239 035320 104032 ERROR 32 ;32 BIT ECC REGISTER SHOULD BE NON
5240 ;ZERO
5241 ;ONLY 11 OF THE 32 BITS CAN BE SEEN
5242 ;IN THE PATTERN REGISTER
5243 ;DCK SHOULD BE SET IN RHER1

```

```

5244 035322 013746 050464      6$:  MOV    @#GECC1,-(SP)    ;GET PATTERN REGISTER
5245 035326 042716 174000      BIC    #174000,(SP)    ;KEEP ONLY 11 BITS
5246 035332 022637 015064      CMP    (SP)+,@#EC2    ;COMPARE PATTERN REGISTER
5247 035336 001401      BEQ    7$             ;BRANCH IF GOOD
5248 035340 104032      ERROR  32            ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
5249
5250 035342 004037 051106      7$:  JSR    R0,@#ECORR    ;GO TO ECC CORRECTION PROCESS
5251 035346 000026      8$:  22.                ;EXPECTED POSITION REG. WHEN CORRECTION
5252                                ;IS COMPLETE
5253
5254
5255
5256 035350 004737 046214      JSR    PC,@#CHECKE    ;CHECK THAT DVA,RDY,DPR,DRY 1
5257 035354 104401 005123      TYPE  .CPHALT        ;CANNOT CONTINUE IF THEY DON'T
5258 035360 000000      HALT                    ;STOP THE TEST AND RESTART PROGRAM
5259 035362 012700 015220      MOV    #WRFROM,R0    ;GETTING READY TO FILL EXPECTED DATA
5260 035366 012720 010000      MOV    #0!FMT22,(R0)+ ;CYLINDER 0
5261 035372 112746 000000      MOV    #0,-(SP)      ;IN LOWER BYTE GET SECTOR
5262 035376 112766 000000 000001      MOV    #0,1(SP)      ;GET TRACK IN HIGHER BYTE
5263 035404 012620      MOV    (SP)+,(R0)+   ;GET TRACK/SECTOR IN BUFFER
5264 035406 012720 000000      MOV    #0,(R0)+      ;KEY1 IN BUFFER
5265 035412 012720 000000      MOV    #0,(R0)+      ;KEY2 IN BUFFER
5266 035416 012701 000400      MOV    #256.,R1      ;DATA WORD COUNTER
5267 035422 012702 000000      MOV    #0,R2         ;DATA
5268 035426 010220      3$:  MOV    R2,(R0)+      ;DATA INTO BUFFER
5269 035430 005301      DEC    R1             ;COUNT
5270 035432 001375      BNE    3$            ;BRANCH IF 256 NOT DONE
5271
5272                                ;*ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
5273                                ;*NOW THE INSERTED ERROR WILL BE PUT IN
5274 035434 012737 100000 015232      MOV    #100000,@#WRFROM+<5*2> ;INSERTED ERROR
5275
5276
5277
5278 035442 005037 015124      CLR    @#ERFLG$      ;CLEAR ERROR FLAG
5279 035446 004737 045470      JSR    PC,@#PUTREG   ;SAVE REGISTERS
5280
5281
5282                                ;*NOW READ DATA BUFFER WILL BE CHECKED
5283
5284 035452 004037 046664      JSR    R0,@#COMPAR   ;CHECK
5285 035456 015220      WRFROM                ;GOOD BUFFER
5286 035460 016264      REINTO                ;TEST BUFFER
5287 035462 000404      4+256.                ;NUMBER OF WORDS CHECKED
5288 035464 035472      4$                     ;RETURN POINT FOR ERROR HEADER
5289 035466 035476      5$                     ;RETURN POINT FOR ERROR DATA
5290 035470 035502      TST56                 ;RETURN FOR GOOD COMPARISON
5291 035472 104004      4$:  ERROR  4            ;READ NEXT ERROR
5292 035474 000207      RTS    PC              ;RETURN TO 'COMPAR'
5293 035476 104005      5$:  ERROR  5            ;WORD NOS 1 TO 4 ARE
5294                                ;HEADER WORDS
5295                                ;5 TO 260 ARE DATA WORDS
5296 035500 000207      RTS    PC              ;RETURN TO 'COMPAR'
5297

```

```

5298
5299 035502 000004          TST56: SCOPE
5300 035504 012706 001000    MOV      #STACK,SP      ;RESET STACK
5301
5302 035510 012737 000056 017330    MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
5303
5304
5305          :          SETUP FOR WHAT IS TO BE READ
5306          :          HEADER CRC IS RESTORED FROM A SUBROUTINE
5307
5308 035516 012746 000000    MOV      #0,-(SP)       ;DATA TO BE READ
5309 035522 012705 000400    MOV      #256.,R5      ;COUNTER
5310 035526 012700 054572    MOV      #DISK,R0      ;START OF SIMULATED DISK DATA
5311 035532 011620          1$:  MOV      (SP),(R0)+    ;MOVE IN DATA ON TO SIMULATED DISK
5312 035534 005305          DEC      R5            ;COUNT
5313 035536 001375          BNE     1$            ;BRANCH IF 256 NOT COMPLETE
5314 035540 005726          TST     (SP)+         ;UNDO -(SP)
5315 035542 022020          CMP     (R0)+,(R0)+  ;JUMP OVER THE TWO ECC WORDS
5316 035544 012705 000017    MOV      #15.,R5      ;1 DATA GAP
5317
5318 035550 005020          2$:  CLR     (R0)+         ;CLEAR DATA GAP, AND
5319 035552 005305          DEC     R5            ;TOLERANCE GAP
5320 035554 001375          BNE     2$            ;BRANCH IF NOT COMPLETE
5321
5322
5323 035556 004737 051260    JSR     PC,@#FILLEC   ;INSERT THE TWO ECC WORDS ON THE DISK
5324
5325
5326          ;*THESE ARE FOR ECC TEST ONLY
5327
5328 035562 012737 177777 015142    MOV     #-1,@#TSECC   ;THIS IS AN ECC TEST
5329 035570 005037 050476    CLR     @#POSITI     ;CLEAR ERROR POSITION COUNTER
5330 035574 013737 050472 050474    MOV     @#NCODE,@#NCOUNT ;TEMPORARY N-CODE COUNTER
5331 035602 013737 050500 050506    MOV     @#HARDER,@#HADTMP ;TEMPORARY HARD ERROR COUNTER
5332 035610 005037 050464    CLR     @#GECC1     ;ECC LOW ORDER TO BE GENERATED
5333 035614 005037 050466    CLR     @#GECC2     ;ECC HIGH ORDER TO BE GENERATED
5334 035620 005037 050502    CLR     @#DATENV    ;CLEAR DATA ENVELOPE CLOCK COUNT
5335 035624 005037 050504    CLR     @#ZCODE     ;CLEAR LEADING ZEROS CLOCK COUNT
5336
5337
5338          ;*THESE ARE TO SETUP FOR DISKLESS USE ONLY
5339
5340 035630 012737 010000 052654    MOV     #FMT22,@#CYL ;16 BITS PER WORD
5341
5342 035636 112737 000000 052657    MOV     #0,@#SECTR+1 ;CYLINDER 0, FORMAT 16 BITS
5343 035644 112737 000000 052656    MOV     #0,@#SECTR   ;SECTOR 0
5344 035652 012737 000000 052660    MOV     #0,@#KEY1    ;KEY1=0
5345 035660 012737 000000 052662    MOV     #0,@#KEY2    ;KEY2=0
5346 035666 012737 000400 052734    MOV     #256.,@#DAWORD ;NO. OF DATA WORDS
5347 035674 005037 052664    CLR     @#X          ;THIS IS A READ COMMAND
5348 035700 004537 047176    JSR     R5,@#CRC     ;GO TO CALCULATE CRC
5349 035704 052654          CYL
5350 035706 054554          WCRC
5351
5352
5353          ;*THIS IS TO INSERT ERROR
  
```

```
5354 ;*THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'  
5355 ;*THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING  
5356 ;*THIS MOVE  
5357 035710 012737 177760 054574 MOV #177760,@#DISK+2 ;FORCE ERROR ON BIT NUMBER 21 THRU 32  
5358 ;SO ERROR POSITION REGISTER WILL SHOW  
5359 ;22  
5360 035716 012737 010040 036072 MOV #4128.,@#8$ ;INSERT POSITION REG.  
5361  
5362 ;*THESE ARE REGULAR SETUPS  
5363  
5364  
5365 035724 004737 045770 JSR PC,@#CLDISK ;SETUP GENERAL REGISTERS  
5366 035730 012777 177374 157012 MOV #-256.-4.,@RHWC ;256. DATA 4 HEADER WORDS  
5367 035736 012777 016264 157006 MOV #REINTO,@RHBA ;STARTING ADDRESS OF READ BUFFER  
5368 035744 112746 000000 MOVB #0,-(SP) ;IN LOWER BYTE GET SECTOR  
5369 035750 112766 000000 000001 MOVB #0,1(SP) ;GET TRACK IN HIGHER BYTE  
5370 035756 012677 157000 MOV (SP)+,@RHDST ;TRACK/SECTOR IN RHDST  
5371 035762 012777 010000 156776 MOV #FMT22,@RHOF ;16 BITS PER WORD  
5372 ;ECC CORRECTION NOT INHIBIT  
5373 ;BECAUSE ECC IS NOT GOING  
5374 ;TO BE CHECKED  
5375 035770 005077 156774 CLR @RHCA ;CYLINDER 0  
5376 035774 004737 046024 JSR PC,@#CHECKT ;CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DON'T  
5377 036000 104401 005123 TYPE ,CPHALT ;CANNOT CONTINUE TESTING IF ANY OF THE  
5378 036004 000000 HALT ;STOP THE TEST  
5379 036006 013711 015176 MOV @#REFOR,@R1 ;READ HEADER AND DATA=72  
5380 036012 005037 015124 CLR @#ERFLG$ ;CLEAR ERROR FLAG  
5381 036016 004737 052514 JSR PC,@#COMHD ;READ HEADER AND DATA  
5382 ;IF THERE ARE READ ERRORS THEN  
5383 ;ECC WILL NOT BE CHECKED  
5384
```

```
5385
5386
5387
5388
5389
5390
5391
5392
5393
5394
5395
5396 036022 005737 015124
5397 036026 001106
5398 036030 004737 045470
5399 036034 022737 100000 015034
5400 036042 001401
5401 036044 104032
5402
5403
5404
5405
5406 036046 013746 050464 6$:
5407 036052 042716 174000
5408 036056 022637 015064
5409 036062 001401
5410 036064 104032
5411
5412 036066 004037 051106 7$:
5413 036072 000000 8$:
5414
5415
5416
5417
5418 036074 004737 045470
5419 036100 022737 100100 015034
5420
5421 036106 001401
5422 036110 104036
5423
5424
5425
5426
5427 036112 9$:
5428 036112 004737 046214
5429 036116 104401 005123
5430 036122 000000
5431 036124 012700 015220
5432 036130 012720 010000
5433 036134 112746 000000
5434 036140 112766 000000 000001
5435 036146 012620
5436 036150 012720 000000
5437 036154 012720 000000
5438 036160 012701 000400
5439 036164 012702 000000
5440 036170 010220 3$:
```

```

;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
;*FROM THE 'COMHD' ROUTINE THAT MEANS SECTOR GAP,
;*FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
;*SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
;*DETECTED
;*HEADER AND DATA ARE TO BE CHECKED.
;*IN CHECKING READ DATA THE WRITE FROM BUFFER
;*'WRFROM' IS FILLED WITH EXPECTED DATA AND
;*COMPARISONS ARE MADE

TST @MERFLG$ ;ANY ERRORS ALREADY THERE
BNE TST57 ;BRANCH IF YES
JSR PC,@PUTREG ;SAVE REGISTERS
CMP #DCK,@R1 ;ONLY DATA CHECK ERROR SHOULD BE SET
BEQ 6$ ;BRANCH IF YES
ERROR 32 ;32 BIT ECC REGISTER SHOULD BE NON
;ZERO
;ONLY 11 OF THE 32 BITS CAN BE SEEN
;IN THE PATTERN REGISTER
;DCK SHOULD BE SET IN RHER1
6$: MOV @GECC1,-(SP) ;GET PATTERN REGISTER
BIC #174000,(SP) ;KEEP ONLY 11 BITS
CMP (SP)+,@R2 ;COMPARE PATTERN REGISTER
BEQ 7$ ;BRANCH IF GOOD
ERROR 32 ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT

7$: JSR R0,@R2 ;GO TO ECC CORRECTION PROCESS
8$: .WORD ;EXPECTED POSITION REG. WHEN CORRECTION
;IS COMPLETE

JSR PC,@PUTREG ;SAVE REGISTERS
CMP #DCK.ECH,@R1 ;WITH ERRORS INSERTED IN BIT POSITION 21
;THRU 32 HARD ERROR BIT SHOULD SET
BEQ 9$ ;BRANCH IF GOOD
ERROR 36 ;WITH ERROR INSERTED IN BIT POSITION 21 THRU
;32 ECH SHOULD SET

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


CZRJHCO.RP04/5/6 DSKLS CTRLR2
CZRJHC.P12 10-NOV-77 11:09

MACY11 30A(1052) 27-JUL-78 11:16 PAGE 129
T56 READ ECC ENABLED 1C

K 10

SEQ 0127

5441 036172 005301
5442 036174 001375
5443

DEC R1
BNE 3\$

:COUNT
:BRANCH IF 256 NOT DONE

CZ
CZ

```
5444 :ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'  
5445 :*NOW THE INSERTED ERROR WILL BE PUT IN  
5446  
5447 036176 012737 177760 015232 MOV #177760,@WRFROM+<5*2> ;INSERTED ERROR  
5448  
5449  
5450  
5451 036204 005037 015124 CLR @WERFLG$ ;CLEAR ERROR FLAG  
5452 036210 004737 045470 JSR PC,@PUTREG ;SAVE REGISTERS  
5453  
5454  
5455 :*NOW READ DATA BUFFER WILL BE CHECKED  
5456  
5457 036214 004037 046664 JSR RO,@COMPAR ;CHECK  
5458 036220 015220 WRFROM ;GOOD BUFFER  
5459 036222 016264 REINTO ;TEST BUFFER  
5460 036224 000404 4+256. ;NUMBER OF WORDS CHECKED  
5461 036226 036234 4$ ;RETURN POINT FOR ERROR HEADER  
5462 036230 036240 5$ ;RETURN POINT FOR ERROR DATA  
5463 036232 036244 TST57 ;RETURN FOR GOOD COMPARISON  
5464 036234 104004 4$: ERROR 4 ;READ NEXT ERROR  
5465 036236 000207 RTS PC ;RETURN TO 'COMPAR'  
5466 036240 104005 5$: ERROR 5 ;WORD NOS 1 TO 4 ARE  
5467 ;HEADER WORDS  
5468 ;5 TO 260 ARE DATA WORDS  
5469 036242 000207 RTS PC ;RETURN TO 'COMPAR'  
5470  
5471  
5472  
5473  
5474  
5475
```

```
5476
5477 036244 000004          TST57: SCOPE
5478 036246 012706 001000  MOV      #STACK,SP      ;RESET STACK
5479
5480 036252 012737 000057 017330  MOV      #TTNO,@TSTNM    ;THIS SAVES TEST NUMBER
5481 036260 012700 054474      MOV      #SECGAP,R0      ;POINTER
5482 036264 012701 000460      MOV      #304.,R1        ;COUNTER
5483 036270 005020          1$: CLR      (R0)+           ;CLEAR SIMULATED DISK AREA
5484 036272 005301          DEC      R1
5485 036274 001375          BNE     1$
5486 036276 004737 045770      JSR     PC,CLDISK        ;THIS IS USED TO SET GENERAL REGISTERS
5487
5488                          ;*THESE ARE FOR ECC TEST ONLY
5489
5490 036302 012737 177777 015142  MOV      #-1,@TSECC      ;THIS IS AN ECC TEST
5491 036310 005037 050476      CLR     @POSITI         ;CLEAR ERROR POSITION COUNTER
5492 036314 013737 050472 050474  MOV      @NCODE,@NCOUNT ;TEMPORARY N-CODE COUNTER
5493 036322 013737 050500 050506  MOV      @HARDER,@HADTMP ;TEMPORARY HARD ERROR COUNTER
5494 036330 005037 050464      CLR     @GECC1         ;ECC LOW ORDER TO BE GENERATED
5495 036334 005037 050466      CLR     @GECC2         ;ECC HIGH ORDER TO BE GENERATED
5496 036340 005037 050502      CLR     @DATENV        ;CLEAR DATA ENVELOPE CLOCK COUNT
5497 036344 005037 050504      CLR     @ZCODE         ;CLEAR LEADING ZEROS CLOCK COUNT
5498
5499
5500
5501
5502                          ;*THESE ARE TO BE SETUP FOR DISKLESS USE ONLY
5503
5504 036350 012737 010000 056012  MOV      #FMT22,@WCYL    ;FORMAT22=16BIT WORDS AND
5505                          ;CYLINDER 0
5506 036356 012737 000001 056014  MOV      #1,@WSECTR      ;TRACK=0, SECTOR 1
5507 036364 005037 056016      CLR     @WKEY1         ;KEY1=0
5508 036370 005037 056020      CLR     @WKEY2         ;KEY2=0
5509 036374 012737 000400 056052  MOV      #256.,@FNWORD   ;256 DATA WORDS
5510 036402 004537 047176      JSR     R5,@CRC         ;GO TO CALCULATE CRC
5511 036406 056012          WCYL
5512 036410 056022          GCRC
5513
5514                          ;*THESE ARE REGULAR SETUPS
5515
5516 036412 012777 177374 156330  MOV      #-260.,@RHWC    ;256 DATA WORDS 4 HEADER WORDS
5517 036420 012700 015220      MOV     #WRFROM,R0      ;THESE TWO INSTRUCTIONS GETS
5518 036424 010077 156322      MOV     R0,@RHBA        ;ADDR. OF WRFROM INTO R0 AND
5519                          ;BUS ADDRESS REGISTER
5520 036430 012720 010000      MOV     #FMT22,(R0)+    ;FORMAT=16 BIT WORDS
5521                          ;CYLINDER=0
5522 036434 012720 000001 2$: MOV     #1,(R0)+        ;TRACK=0, SECTOR=1, KEYS 0
5523 036440 005020          CLR     (R0)+          ;KEY1 0
5524 036442 005020          CLR     (R0)+          ;KEY2=0
5525 036444 012705 000400      MOV     #256.,R5        ;COUNTER
5526 036450 012720 177777 3$: MOV     #-1,(R0)+      ;MOVE ALL ONES FOR DATA
5527 036454 005305          DEC     R5
5528 036456 001374          BNE     3$             ;BRANCH IF DATA NOT COMPLETE
5529 036460 012777 000001 1562'4  MOV     #1,@RHDS1      ;TRACK 0 SECTOR 1
5530
5531 036466 004737 046024      JSR     PC,@CHECKT     ;CHECK DVA,RDY,DPR,DRY 1 AND OTHERS DON'T
```

```
5532 036472 104401 005123 TYPE ,CPHALT ;CANNOT CONTINUE TESTING IF ANY OF THE
5533 036476 000000 HALT ;STOP THE TEST
5534
5535 036500 013711 015172 MOV @WRIFOR,@R1 ;GET READY FOR WRITE HEADER AND
5536 ;DATA WITH 62 IN RHCS1
5537 036504 005037 015124 CLR @ERFLGS ;CLEAR ERROR FLAG
5538 036510 012777 010000 156250 MOV #FMT22,@RHOF ;FORMAT BIT=1 (16 BIT WORDS)
5539 036516 005077 156246 CLR @RHCA ;CYLINDER 0
5540 036522 004737 055636 JSR PC,@COMWHD ;WRITE HEADER AND DATA
5541
5542 ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
5543 ;*FROM THE 'COMWHD' ROUTINE THAT MEANS ALL HEADER ON DISK
5544 ;*IS GOOD IE. ONLY DATA IS TO BE CHECKED TO SEE IF THEY ARE
5545 ;*ALL ONES AND WRITE DATA GAP AND TOLERANCE GAP TO SEE IF
5546 ;*THEY ARE ALL ZEROS
5547
5548 036526 005737 015124 TST @ERFLGS ;HAS ANY ERRORS OCCURED?
5549 ;IF WRITE ERROR OCCURS ECC IS NOT CHECKED
5550 036532 001056 BNE TST60 ;BRANCH IF YES
5551
5552
5553 ;*COMPARE SOFTWARE GENERATED ECC WITH THAT GENERATED BY HARDWARE
5554
5555 036534 023737 050464 055572 CMP @GECC1,@WECC1;COMPARE SOFTWARE ECC WITH HARDWARE ECC
5556 036542 001402 BEQ 6$ ;BRANCH IF GOOD
5557 036544 104031 ERROR 31 ;LOW ORDER ECC IN ERROR
5558 036546 000405 BR 7$ ;BRANCH TO CONTINUE
5559 036550 023737 050466 055574 6$: CMP @GECC2,@WECC2;COMPARE SOFTWARE ECC WITH HARDWARE ECC
5560 036556 001401 BEQ 7$ ;BRANCH IF GOOD
5561 036560 104031 ERROR 71 ;HIGH ORDER ECC IN ERROR
5562
5563
5564 036562 7$:
5565 036562 004737 046214 JSR PC,@CHECKE ;CHECK THAT DVA,RDY,DPR,DRY 1
5566 036566 104401 005123 TYPE ,CPHALT ;CANNOT CONTINUE IF THEY DON'T
5567 036572 000000 HALT ;STOP THE TEST AND RESTART PROGRAM
5568
5569
5570
5571
5572 ;*FILL 'REINTO' BUFFER WITH EXPECTED DATA
5573
5574 036574 004037 045706 JSR R0,@CLAREA ;FILL REINTO BUFFER
5575 036600 016264 REINTO ;FROM
5576 036602 017262 REINTO+<255.*2> ;TO
5577 036604 177777 .WORD -1 ;DATA
5578
5579 036606 013737 050464 017264 MOV @GECC1,@REINTO+<256.*2>;FILL ECC1
5580 036614 013737 050466 017266 MOV @GECC2,@REINTO+<257.*2>;FILL ECC2
5581 036622 004037 045706 JSR R0,@CLAREA ;FILL REST
5582 036626 017270 REINTO+<258.*2> ;FROM
5583 036630 017324 REINTO+<272.*2> ;TO
5584 036632 000000 0 ;DATA
5585
5586
5587 036634 005037 015124 CLR @ERFLGS ;CLEAR ERROR FLAG
```

```
5588
5589
5590      ;*NOW COMPARE 'DISK' BUFFER WITH 'REINTO'
5591
5592 036640 004037 036664 JSR RO, @#COMPAR ;CHECK
5593 036644 016264 REINTO ;GOOD BUFFER
5594 036646 054572 DISK ;TEST BUFFER
5595 036650 000402 258. ;NUMBER OF WORDS CHECKED
5596 036652 036660 4$ ;RETURN POINT FOR ERROR HEADER
5597 036654 036664 5$ ;RETURN POINT FOR ERROR DATA
5598 036656 036670 TST60 ;RETURN FOR GOOD COMPARISON
5599 036660 104007 4$: ERROR 7 ;READ ERROR 10 NEXT
5600 036662 000207 RTS PC ;RETURN TO COMPARE
5601 036664 104010 5$: ERROR 10 ;WORD NOS 1 TO 256 ARE
5602 ;DATA WORDS
5603 ;WORD NOS 257 AND 258
5604 ;ARE ECC WHICH ARE CHECKED
5605 ;WORD NOS 259
5606 ;IS DATA GAP
5607 ;WORD NOS 260 TO 273
5608 ;ARE TOLERANCE GAP
5609 036666 000207 RTS PC ;RETURN TO COMPARE
5610
5611
5612
5613
5614
```

```
5615
5616 036670 000004          TST60: SCOPE
5617 036672 012706 001000  MOV      #STACK,SP      ;RESET STACK
5618
5619 036676 012737 000060 017330  MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
5620
5621
5622      ;      SETUP FOR WHAT IS TO BE READ
5623      ;      HEADER CRC IS RESTORED FROM A SUBROUTINE
5624
5625 036704 012746 177777      MOV      #-1,-(SP)      ;DATA TO BE READ
5626 036710 012705 000400      MOV      #256.,R5      ;COUNTER
5627 036714 012700 054572      MOV      #DISK,R0      ;START OF SIMULATED DISK DATA
5628 036720 011620          1$:  MOV      (SP),(R0)+      ;MOVE IN DATA ON TO SIMULATED DISK
5629 036722 005305          DEC      R5              ;COUNT
5630 036724 001375          BNE     1$              ;BRANCH IF 256 NOT COMPLETE
5631 036726 005726          TST     (SP)+          ;UNDO -(SP)
5632 036730 022020          CMP     (R0)+,(R0)+    ;JUMP OVER THE TWO ECC WORDS
5633 036732 012705 000017      MOV      #15.,R5      ;1 DATA GAP
5634                                ;14 TOLERANCE GAP
5635 036736 005020          2$:  CLR     (R0)+          ;CLEAR DATA GAP, AND
5636 036740 005305          DEC     R5              ;TOLERANCE GAP
5637 036742 001375          BNE     2$              ;BRANCH IF NOT COMPLETE
5638
5639
5640 036744 004737 051260      JSR     PC,@#FILLEC    ;INSERT THE TWO ECC WORDS ON THE DISK
5641                                ;IN THE CORRECT PLACE
5642
5643      ;*THESE ARE FOR ECC TEST ONLY
5644
5645 036750 012737 177777 015142  MOV      #-1,@#TSECC    ;THIS IS AN ECC TEST
5646 036756 005037 050476      CLR     @#POSITI      ;CLEAR ERROR POSITION COUNTER
5647 036762 013737 050472 050474  MOV      @#NCODE,@#NCOUNT ;TEMPORARY N-CODE COUNTER
5648 036770 013737 050500 050506  MOV      @#HARDER,@#HADTMP ;TEMPORARY HARD ERROR COUNTER
5649 036776 005037 050464      CLR     @#GECC1      ;ECC LOW ORDER TO BE GENERATED
5650 037002 005037 050466      CLR     @#GECC2      ;ECC HIGH ORDER TO BE GENERATED
5651 037006 005037 050502      CLR     @#DATENV      ;CLEAR DATA ENVELOPE CLOCK COUNT
5652 037012 005037 050504      CLR     @#ZCODE      ;CLEAR LEADING ZEROS CLOCK COUNT
5653
5654
5655      ;*THESE ARE TO SETUP FOR DISKLESS USE ONLY
5656
5657 037016 012737 010000 052654  MOV      #FMT22,@#CYL  ;16 BITS PER WORD
5658                                ;CYLINDER 0, FORMAT 16 BITS
5659 037024 112737 000000 052657  MOVB    #0,@#SECOTR+1  ;TRACK 0
5660 037032 112737 000000 052656  MOVB    #0,@#SECOTR    ;SECTOR 0
5661 037040 012737 000000 052660  MOV     #0,@#KEY1      ;KEY1=0
5662 037046 012737 000000 052662  MOV     #0,@#KEY2      ;KEY2=0
5663 037054 012737 000400 052734  MOV     #256.,@#DAWORD ;NO. OF DATA WORDS
5664 037062 005037 052664      CLR     @#X            ;THIS IS A READ COMMAND
5665 037066 004537 047176      JSR     R5,@#CRC      ;GO TO CALCULATE CRC
5666 037072 052654      CYL
5667 037074 054554      WCRC
5668
5669
5670
```

```
5671
5672
5673
5674 037076 004737 045770 JSR PC,@#CLDISK ;SETUP GENERAL REGISTERS
5675 037102 012777 177374 155640 MOV #-256,-4,@RHWC ;256. DATA 4 HEADER WORDS
5676 037110 012777 016264 155634 MOV #REINTO,@RHBA ;STARTING ADDRESS OF READ BUFFER
5677 037116 112746 000000 MOVB #0,-(SP) ;IN LOWER BYTE GET SECTOR
5678 037122 112766 000000 000001 MOVB #0,1(SP) ;GET TRACK IN HIGHER BYTE
5679 037130 012677 155626 MOV (SP)+,@RHDST ;TRACK/SECTOR IN RHDST
5680 037134 012777 010000 155624 MOV #FMT22,@RHOF ;16 BITS PER WORD
5681 ;ECC CORRECTION NOT INHIBIT
5682 ;BECAUSE ECC IS NOT GOING
5683 ;TO BE CHECKED
5684 037142 005077 155622 CLR @RHCA ;CYLINDER 0
5685
5686 037146 004737 046024 JSR PC,@#CHECKT ;CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DGN'T
5687 037152 104401 005123 TYPE ,CPHALT ;CANNOT CONTINUE TESTING IF ANY OF THE
5688 037156 000000 HALT ;STOP THE TEST
5689
5690 037160 013711 015176 MOV @#REFOR,@P1 ;READ HEADER AND DATA=72
5691 037164 005037 015124 CLR @#ERFLG$ ;CLEAR ERROR FLAG
5692 037170 004737 052514 JSR PC,@#COMHD ;READ HEADER AND DATA
5693 ;IF THERE ARE READ ERRORS THEN
5694 ;ECC WILL NOT BE CHECKED
5695
5696
5697
5698 ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
5699 ;*FROM THE 'COMHD' ROUTINE THAT MEANS SECTOR GAP,
5700 ;*FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
5701 ;*SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
5702 ;*DETECTED
5703 ;*HEADER AND DATA ARE TO BE CHECKED.
5704 ;*IN CHECKING READ DATA THE WRITE FROM BUFFER
5705 ;*'WRFROM' IS FILLED WITH EXPECTED DATA AND
5706 ;*COMPARISONS ARE MADE
5707 037174 005737 015124 TST @#ERFLG$ ;ANY ERRORS ALREADY THERE
5708 037200 001102 BNE TST61 ;BRANCH IF YES
5709 037202 004737 045470 JSR PC,@#PUTREG ;SAVE REGISTERS
5710 037206 005737 015034 TST @#ER1 ;NO ERRORS SHOULD BE SET
5711 037212 001401 BEQ 6$ ;BRANCH IF NO ERRORS SET
5712 037214 104032 ERROR 32 ;32 BIT ECC REGISTER SHOULD BE ZERO
5713 ;ONLY 11 OF THE 32 BITS CAN BE SEEN
5714 ;IN THE PATTERN REGISTER
5715 ;DCK SHOULD BE SET IN RHER1
5716 037216 013746 050464 6$: MOV @#GECC1,-(SP) ;GET PATTERN REGISTER
5717 037222 042716 174000 BIC #174000,(SP) ;KEEP ONLY 11 BITS
5718 037226 022637 015064 CMP (SP)+,@#EC2 ;COMPARE PATTERN REGISTER
5719 037232 001401 BEQ 7$ ;BRANCH IF GOOD
5720 037234 104032 ERROR 32 ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
5721
5722
5723
5724
5725
5726 ;*ADD 16 MAINTENANCE CLOCKS TO
;*BRING EBL DOWN
```

```

5727
5728 037236 012700 000020      7$:  MOV    #16.,R0      ;COUNTER
5729 037242 052777 000002      8$:  BIS    #MCLK,@RHMR  ;SET CLOCK
5730 037250 042777 000002      155526 8$:  BIC    #MCLK,@RHMR  ;CLEAR CLOCK
5731 037256 005300                DEC    R0            ;COUNT
5732 037260 001370                BNE    8$           ;BRANCH IF 16 CLOCKS NOT DONE
5733 037262 004737 046214      JSR    PC,@#CHECKE  ;CHECK THAT DVA,RDY,DPR,DRY = 1
5734 037266 104401 005123      TYPE  ,CPHALT      ;CANNOT CONTINUE IF THEY DON'T
5735 037272 000000                HALT                ;STOP THE TEST AND RESTART PROGRAM
5736 037274 012700 015220      MOV    #WRFROM,R0  ;GETTING READY TO FILL EXPECTED DATA
5737 037300 012720 010000      MOV    #0.FMT22,(R0)+ ;CYLINDER 0
5738 037304 112746 000000      MOV    #0, -(SP)   ;IN LOWER BYTE GET SECTOR
5739 037310 112766 000000      000001 MOV    #0, 1(SP)   ;GET TRACK IN HIGHER BYTE
5740 037316 012620                MOV    (SP)+, (R0)+ ;GET TRACK/SECTOR IN BUFFER
5741 037320 012720 000000      MOV    #0, (R0)+  ;KEY1 IN BUFFER
5742 037324 012720 000000      MOV    #0, (R0)+  ;KEY2 IN BUFFER
5743 037330 012701 000400      MOV    #256., R1   ;DATA WORD COUNTER
5744 037334 012702 177777      MOV    #-1, R2    ;DATA
5745
5746 037340 010220                3$:  MOV    R2, (R0)+  ;DATA INTO BUFFER
5747 037342 005301                DEC    R1          ;COUNT
5748 037344 001375                BNE    3$         ;BRANCH IF 256 NOT DONE
5749 037346 005037 015124      CLR    @#ERFLG$   ;CLEAR ERROR FLAG
5750 037352 004737 045470      JSR    PC,@#PIJTREG ;SAVE REGISTERS
5751
5752                ;NOW READ DATA BUFFER WILL BE CHECKED
5753
5754 037356 004037 046664      JSR    R0,@#COMPAR ;CHECK
5755 037362 015220                WRFROM            ;GOOD BUFFER
5756 037364 016264                REINTO           ;TEST BUFFER
5757 037366 000404                4+256.          ;NUMBER OF WORDS CHECKED
5758 037370 037376                4$              ;RETURN POINT FOR ERROR HEADER
5759 037372 037402                5$              ;RETURN POINT FOR ERROR DATA
5760 037374 037406                TST61           ;RETURN FOR GOOD COMPARISON
5761 037376 104004                4$:  ERROR  4        ;READ NEXT ERROR
5762 037400 000207                RTS    PC        ;RETURN TO 'COMPAR'
5763 037402 104005                5$:  ERROR  5        ;WORD NOS 1 TO 4 ARE
5764                ;HEADER WORDS
5765                ;5 TO 260 ARE DATA WORDS
5766 037404 000207                RTS    PC        ;RETURN TO 'COMPAR'
5767
5768
5769
5770

```



```
5771
5772 037406 000004          TST61: SCOPE
5773 037410 012706 001000  MOV      #STACK,SP      ;RESET STACK
5774
5775 037414 012737 000061 017330  MOV      #TINO,@#TSTNM  ;THIS SAVES TEST NUMBER
5776
5777
5778          ;*SETUP FOR WHAT IS TO BE READ
5779          ;*HEADER CRC IS RESTORED FROM A SUBROUTINE
5780
5781 037422 012746 177777      MOV      #-1,-(SP)      ;DATA TO BE READ
5782 037426 012705 000400      MOV      #256.,R5      ;COUNTER
5783 037432 012700 054572      MOV      #DISK,R0      ;START OF SIMULATED DISK DATA
5784 037436 011620          1$:  MOV      (SP),(R0)+     ;MOVE IN DATA ON TO SIMULATED DISK
5785 037440 005305          DEC      R5             ;COUNT
5786 037442 001375          BNE     1$             ;BRANCH IF 256 NOT COMPLETE
5787 037444 005726          TST     (SP)+         ;UNDO -(SP)
5788 037446 022020          CMP     (R0)+,(R0)+   ;JUMP OVER THE TWO ECC WORDS
5789 037450 012705 000017      MOV      #15., R5     ;1 DATA GAP
5790
5791 037454 005020          2$:  CLR     (R0)+         ;CLEAR DATA GAP, AND
5792 037456 005305          DEC     R5            ;TOLERANCE GAP
5793 037460 001375          BNE     2$           ;BRANCH IF NOT COMPLETE
5794
5795
5796 037462 004737 051260      JSR     PC,@#FILLEC   ;INSERT ECC IN PROPER PLACE ON DISK
5797
5798
5799
5800          ;*THESE ARE FOR ECC TEST ONLY
5801
5802 037466 012737 177777 015142  MOV      #-1,@#TSECC   ;THIS IS AN ECC TEST
5803 037474 005037 050476      CLR     @#POSITI     ;CLEAR ERROR POSITION COUNTER
5804 037500 013737 050472 050474  MOV      @#NCODE,@#NCOUNT ;TEMPORARY N-CODE COUNTER
5805 037506 013737 050500 050506  MOV      @#HARDER,@#HADTMP ;TEMPORARY HARD ERROR COUNTER
5806 037514 005037 050464      CLR     @#GECC1      ;ECC LOW ORDER TO BE GENERATED
5807 037520 005037 050466      CLR     @#GECC2      ;ECC HIGH ORDER TO BE GENERATED
5808 037524 005037 050502      CLR     @#DATENV     ;CLEAR DATA ENVELOPE CLOCK COUNT
5809 037530 005037 050504      CLR     @#ZCODE      ;CLEAR LEADING ZEROS CLOCK COUNT
5810
5811
5812          ;*THESE ARE TO SETUP FOR DISKLESS USE ONLY
5813
5814 037534 012737 010000 052654  MOV      #FMT22,@#CYL  ;16 BITS PER WORD
5815
5816 037542 112737 000000 052657  MOVVB   #0,@#SECOTR+1 ;CYLINDER 0, FORMAT 16 BITS
5817 037550 112737 000000 052656  MOVVB   #0,@#SECOTR   ;TRACK 0
5818 037556 012737 000000 052660  MOV     #0,@#KEY1     ;SECTOR 0
5819 037564 012737 000000 052662  MOV     #0,@#KEY2     ;KEY1=0
5820 037572 012737 000400 052734  MOV     #256.,@#DAWORD ;KEY2=0
5821 037600 005037 052664      MOV     #256.,@#DAWORD ;NO. OF DATA WORDS
5822 037604 004537 047176      CLR     @#X           ;THIS IS A READ COMMAND
5823 037610 052654          JSR     R5,@#CRC      ;GO TO CALCULATE CRC
5824 037612 054554          CYL
5825
5826          WCRC
```

```

5827 ;*THIS IS TO INSERT ERROR
5828 ;*THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
5829 ;*THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
5830 ;*THIS MOVE
5831
5832 037614 012737 077777 054574 MOV #77777,@#DISK+2 ;FORCE ERROR ON BIT NUMBER 32
5833 ;SO ERROR POSITION REGISTER WILL SHOW
5834 ;22
5835 037622 012737 000026 037776 MOV #22.,@#8$ ;INSERT POSITION REG.
5836
5837 ;*THESE ARE REGULAR SETUPS
5838
5839
5840 037630 004737 045770 JSR PC,@#CLDISK ;SETUP GENERAL REGISTERS
5841 037634 012777 177374 155106 MOV #-256.-4.,@#RHW ;256. DATA 4 HEADER WORDS
5842 037642 012777 016264 155102 MOV #REINTO,@#RHB ;STARTING ADDRESS OF READ BUFFER
5843 037650 112746 000000 MOVB #0,-(SP) ;IN LOWER BYTE GET SECTOR
5844 037654 112766 000000 000001 MOVB #0,1(SP) ;GET TRACK IN HIGHER BYTE
5845 037662 012677 155074 MOV (SP)+,@#RHDST ;TRACK/SECTOR IN RHDST
5846 037666 012777 010000 155072 MOV #FMT22,@#RHF ;16 BITS PER WORD
5847 ;ECC CORRECTION NOT INHIBIT
5848 ;BECAUSE ECC IS NOT GOING
5849 ;TO BE CHECKED
5850 037674 005077 155070 CLR @#RHC ;CYLINDER 0
5851 037700 004737 046024 JSR PC,@#CHECKT ;CHECK DVA,RDY,DPR,DRY - 1 AND OTHERS DON'T
5852 037704 104401 005123 TYPE ,CPHALT ;CANNOT CONTINUE TESTING IF ANY OF THE
5853 037710 000000 HALT ;STOP THE TEST
5854 037712 013711 015176 MOV @#REFOR,@#R1 ;READ HEADER AND DATA=72
5855 037716 005037 015124 CLR @#ERFLG$ ;CLEAR ERROR FLAG
5856 037722 004737 052514 JSR PC,@#COMHD ;READ HEADER AND DATA
5857 ;IF THERE ARE READ ERRORS THEN
5858 ;ECC WILL NOT BE CHECKED
5859
5860
5861 ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
5862 ;*FROM THE 'COMHD' ROUTINE THAT MEANS SECTOR GAP,
5863 ;*FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
5864 ;*SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
5865 ;*DETECTED
5866 ;*HEADER AND DATA ARE TO BE CHECKED.
5867 ;*IN CHECKING READ DATA THE WRITE FROM BUFFER
5868 ;*'WRFROM' IS FILLED WITH EXPECTED DATA AND
5869 ;*COMPARISONS ARE MADE
5870
5871 037726 005737 015124 TST @#ERFLG$ ;ANY ERRORS ALREADY THERE
5872 037732 001077 BNE TST62 ;BRANCH IF YES
5873 037734 004737 045470 JSR PC,@#PUTREG ;SAVE REGISTERS
5874 037740 022737 100000 015034 CMP #DCK,@#R1 ;ONLY DATA CHECK ERROR SHOULD BE SET
5875 037746 001401 BEQ 6$ ;BRANCH IF YES
5876 037750 104032 ERROR 32 ;32 BIT ECC REGISTER SHOULD BE NON
5877 ;ZERO
5878 ;ONLY 11 OF THE 32 BITS CAN BE SEEN
5879 ;IN THE PATTERN REGISTER
5880 ;DCK SHOULD BE SET IN RHER1
5881 037752 013746 050464 6$: MOV @#GECC1,-(SP) ;GET PATTERN REGISTER
5882 037756 042716 174000 BIC #174000,(SP) ;KEEP ONLY 11 BITS
    
```

CZ
CZ

```

5883 037762 022637 015064      CMP      (SP)+,@#ECC      ;COMPARE PATTERN REGISTER
5884 037766 001401      BEQ      7$              ;BRANCH IF GOOD
5885 037770 104032      ERROR   32              ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
5886
5887 037772 004037 051106      7$:      JSR      RO,@#ECORR  ;GO TO ECC CORRECTION PROCESS
5888 037776 000026      8$:      22.              ;EXPECTED POSITION REG. WHEN CORRECTION
5889                                     ;IS COMPLETE
5890
5891
5892
5893 040000 004737 046214      JSR      PC,@#CHECKE     ;CHECK THAT DVA,RDY,DPR,DRY = 1
5894 040004 104401 005123      TYPE    ,CPHALT        ;CANNOT CONTINUE IF THEY DON'T
5895 040010 000000      HALT                                     ;STOP THE TEST AND RESTART PROGRAM
5896 040012 012700 015220      MOV     @#WRFROM,RO     ;GETTING READY TO FILL EXPECTED DATA
5897 040016 012720 010000      MOV     #0,FMT22,(RO)+ ;CYLINDER 0
5898 040022 112746 000000      MOV     #0,-(SP)        ;IN LOWER BYTE GET SECTOR
5899 040026 112766 000000 000001      MOV     #0,1(SP)        ;GET TRACK IN HIGHER BYTE
5900 040034 012620      MOV     (SP)+,(RO)+     ;GET TRACK/SECTOR IN BUFFER
5901 040036 012720 000000      MOV     #0,(RO)+        ;KEY1 IN BUFFER
5902 040042 012720 000000      MOV     #0,(RO)+        ;KEY2 IN BUFFER
5903 040046 012701 000400      MOV     #256.,R1        ;DATA WORD COUNTER
5904 040052 012702 177777      MOV     #-1,R2          ;DATA
5905 040056 010220      3$:      MOV     R2,(RO)+        ;DATA INTO BUFFER
5906 040060 005301      DEC     R1              ;COUNT
5907 040062 001375      BNE     3$              ;BRANCH IF 256 NOT DONE
5908
5909                                     ;*ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
5910                                     ;*NOW THE INSERTED ERROR WILL BE PUT IN
5911
5912 040064 012737 077777 015232      MOV     #77777,@#WRFROM+<5*2> ;INSERTED ERROR
5913 040072 004737 045470      JSR     PC,@#PUTREG     ;SAVE REGISTERS
5914 040076 005037 015124      CLR     @#ERFLG$       ;CLEAR ERROR FLAG
5915
5916
5917                                     ;*NOW READ DATA BUFFER WILL BE CHECKED
5918
5919 040102 004037 046664      JSR     RO,@#COMPAR     ;CHECK
5920 040106 015220      WRFROM                    ;GOOD BUFFER
5921 040110 016264      REINTO                    ;TEST BUFFER
5922 040112 000404      4+256.                    ;NUMBER OF WORDS CHECKED
5923 040114 040122      4$                          ;RETURN POINT FOR ERROR HEADER
5924 040116 040126      5$                          ;RETURN POINT FOR ERROR DATA
5925 040120 040132      TST62                      ;RETURN FOR GOOD COMPARISON
5926 040122 104004      4$:      ERROR   4              ;READ NEXT ERROR
5927 040124 000207      RTS     PC                ;RETURN TO 'COMPAR'
5928 040126 104005      5$:      ERROR   5              ;WORD NOS 1 TO 4 ARE
5929                                     ;HEADER WORDS
5930                                     ;5 TO 260 ARE DATA WORDS
5931 040130 000207      RTS     PC                ;RETURN TO 'COMPAR'
5932

```

CZ
CZ

```

5933
5934 040132 000004          TST62: SCOPE
5935 040134 012706 001000  MOV      #STACK,SP      ;RESET STACK
5936
5937 040140 012737 000062 017330  MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
5938
5939
5940      :      SETUP FOR WHAT IS TO BE READ
5941      :      HEADER CRC IS RESTORED FROM A SUBROUTINE
5942
5943 040146 012746 77777      MOV      #-1,-(SP)      ;DATA TO BE READ
5944 040152 012705 000400      MOV      #256.,R5      ;COUNTER
5945 040156 012700 054572      MOV      #DISK,R0      ;START OF SIMULATED DISK DATA
5946 040162 011620          1$:  MOV      (SP),(R0)+     ;MOVE IN DATA ON TO SIMULATED DISK
5947 040164 005305          DEC      R5             ;COUNT
5948 040166 001375          BNE     1$             ;BRANCH IF 256 NOT COMPLETE
5949 040170 005726          TST     (SP)+         ;UNDO -(SP)
5950 040172 022020          CMP     (R0)+,(R0)+   ;JUMP OVER THE TWO ECC WORDS
5951 040174 012705 000017      MOV      #15.,R5      ;1 DATA GAP
5952
5953 040200 005020          2$:  CLR     (R0)+         ;CLEAR DATA GAP, AND
5954 040202 005305          DEC     R5            ;TOLERANCE GAP
5955 040204 001375          BNE     2$           ;BRANCH IF NOT COMPLETE
5956
5957
5958 040206 004737 051260      JSR     PC,@#FILLEC   ;INSERT THE TWO ECC WORDS ON THE DISK
5959
5960
5961
5962      ;*THESE ARE FOR ECC TEST ONLY
5963 040212 012737 177777 015142  MOV      #-1,@#TSECC   ;THIS IS AN ECC TEST
5964 040220 005037 050476      CLR     @#POSITI     ;CLEAR ERROR POSITION COUNTER
5965 040224 013737 050472 050474  MOV      @#NCODE,@#NCOUNT ;TEMPORARY N-CODE COUNTER
5966 040232 013737 050500 050506  MOV      @#HARDER,@#HADTMP ;TEMPORARY HARD ERROR COUNTER
5967 040240 005037 050464      CLR     @#GECC1      ;ECC LOW ORDER TO BE GENERATED
5968 040244 005037 050466      CLR     @#GECC2      ;ECC HIGH ORDER TO BE GENERATED
5969 040250 005037 050502      CLR     @#DATENV     ;CLEAR DATA ENVELOPE CLOCK COUNT
5970 040254 005037 050504      CLR     @#ZCODE      ;CLEAR LEADING ZEROS CLOCK COUNT
5971
5972
5973      ;*THESE ARE TO SETUP FOR DISKLESS USE ONLY
5974
5975 040260 012737 010000 052654  MOV      #FMT22,@#CYL  ;16 BITS PER WORD
5976
5977 040266 112737 000000 052657  MOV     #0,@#SECOTR+1  ;CYLINDER 0, FORMAT 16 BITS
5978 040274 112737 000000 052656  MOV     #0,@#SECOTR    ;TRACK 0
5979 040302 012737 000000 052660  MOV     #0,@#KEY1     ;SECTOR 0
5980 040310 012737 000000 052662  MCV    #0,@#KEY2     ;KEY1=0
5981 040316 012737 000400 052734  MOV     #256.,@#DAWORD ;KEY2=0
5982 040324 005037 052664      CLR     @#X          ;NO. OF DATA WORDS
5983 040330 004537 047176      JSR     R5,@#CRC     ;THIS IS A READ COMMAND
5984 040334 052654          CYL     ;GO TO CALCULATE CRC
5985 040336 054554          WCRC
5986
5987
5988      ;*THIS IS TO INSERT ERROR

```

```
5989 ;*THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'  
5990 ;*THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING  
5991 ;*THIS MOVE  
5992  
5993 040340 012737 077757 054574 MOV #77757,@#DISK+2 ;FORCE ERROR ON BIT NUMBER 32 AND 21  
5994 ;SO ERROR POSITION REGISTER WILL SHOW  
5995 ;22  
5996 040346 012737 010040 040522 MOV #4128.,@#8$ ;INSERT POSITION REG.  
5997  
5998  
5999 ;*THESE ARE REGULAR SETUPS  
6000  
6001 040354 004737 045770 JSR PC,@#CLDISK ;SETUP GENERAL REGISTERS  
6002 040360 012777 177374 154362 MOV #-256.-4.,@RHWC ;256. DATA 4 HEADER WORDS  
6003 040366 012777 016264 154356 MOV #REINTO,@RHBA ;STARTING ADDRESS OF READ BUFFER  
6004 040374 112746 000000 MOVB #0,-(SP) ;IN LOWER BYTE GET SECTOR  
6005 040400 112766 000000 000001 MOVB #0,1(SP) ;GET TRACK IN HIGHER BYTE  
6006 040406 012677 154350 MOV (SP)+,@RHDST ;TRACK/SECTOR IN RHDST  
6007 040412 012777 010000 154346 MOV #FMT22,@RHOF ;16 BITS PER WORD  
6008 ;ECC CORRECTION NOT INHIBIT  
6009 ;BECAUSE ECC IS NOT GOING  
6010 ;TO BE CHECKED  
6011 040420 005077 154344 CLR @RHCA ;CYLINDER 0  
6012  
6013 040424 004737 046024 JSR PC,@#CHECKT ;CHECK DVA,RDY,DPR,DRY - 1 AND OTHERS DON'T  
6014 040430 104401 005123 TYPE ,CPHALT ;CANNOT CONTINUE TESTING IF ANY OF THE  
6015 040434 000000 HALT ;STOP THE TEST  
6016  
6017 040436 013711 015176 MOV @#REFOR,@R1 ;READ HEADER AND DATA-72  
6018 040442 005037 015124 CLR @#ERFLG$ ;CLEAR ERROR FLAG  
6019 040446 004737 052514 JSR PC,@#COMHD ;READ HEADER AND DATA  
6020 ;IF THERE ARE READ ERRORS THEN  
6021 ;ECC WILL NOT BE CHECKED  
6022  
6023  
6024 ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS  
6025 ;*FROM THE 'COMHD' ROUTINE THAT MEANS SECTOR GAP,  
6026 ;*FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND  
6027 ;*SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY  
6028 ;*DETECTED  
6029 ;*HEADER AND DATA ARE TO BE CHECKED.  
6030 ;*IN CHECKING READ DATA THE WRITE FROM BUFFER  
6031 ;*'WRFROM' IS FILLED WITH EXPECTED DATA AND  
6032 ;*COMPARISONS ARE MADE  
6033  
6034 040452 005737 015124 TST @#ERFLG$ ;ANY ERRORS ALREADY THERE  
6035 040456 001106 BNE TST63 ;BRANCH IF YES  
6036 040460 004737 045470 JSR PC,@#PUTREG ;SAVE REGISTERS  
6037 040464 022737 100000 015034 CMP #DCK,@#ER1 ;ONLY DATA CHECK ERROR SHOULD BE SET  
6038 040472 001401 BEQ 6$ ;BRANCH IF YES  
6039 040474 104032 ERROR 32 ;32 BIT ECC REGISTER SHOULD BE NON  
6040 ;ZERO  
6041 ;ONLY 11 OF THE 32 BITS CAN BE SEEN  
6042 ;IN THE PATTERN REGISTER  
6043 ;DCK SHOULD BE SET IN RHER1  
6044 040476 013746 050464 6$: MOV @#GECC1,-(SP) ;GET PATTERN REGISTER
```

```
6045 040502 042716 174000 BIC #174000,(SP) ;KEEP ONLY 11 BITS
6046 040506 022637 015064 CMP (SP)+,@#EC2 ;COMPARE PATTERN REGISTER
6047 040512 001401 BEQ 7$ ;BRANCH IF GOOD
6048 040514 104032 ERROR 32 ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
6049
6050 040516 004037 051106 7$: JSR R0,@#ECORR ;GO TO ECC CORRECTION PROCESS
6051 040522 000700 8$: .WORD ;EXPECTED POSITION REG. WHEN CORRECTION
6052 ;IS COMPLETE
6053
6054 040524 004737 045470 JSR PC,@#PUTREG ;SAVE REGISTERS
6055
6056
6057 040530 022737 100100 015034 CMP #DCK.ECH,@#ER1 ;WITH ERRORS INSERTED IN BIT POSITION 21
6058 ;AND 32 HARD ERROR BIT SHOULD SET
6059 040536 001401 BEQ 9$ ;BRANCH IF GOOD
6060 040540 104036 ERROR 36 ;WITH ERROR INSERTED IN BIT POSITION 21 THRU
6061 ;32 HCE SHOULD SET
6062
6063
6064
6065
6066 040542 9$:
6067 040542 004737 046214 JSR PC,@#CHECKE ;CHECK THAT DVA,RDY,DPR,DRY 1
6068 040546 104401 005123 TYPE ,CPHALT ;CANNOT CONTINUE IF THEY DON'T
6069 040552 000000 HALT ;STOP THE TEST AND RESTART PROGRAM
6070 040554 012700 015220 MOV #WRFROM,R0 ;GETTING READY TO FILL EXPECTED DATA
6071 040560 012720 010000 MOV #0!FMT22,(R0)+ ;CYLINDER 0
6072 040564 112746 000000 MOVB #0,-(SP) ;IN LOWER BYTE GET SECTOR
6073 040570 112766 000000 000001 MOVB #0,1(SP) ;GET TRACK IN HIGHER BYTE
6074 040576 012620 MOV (SP)+,(R0)+ ;GFT TRACK/SECTOR IN BUFFER
6075 040600 012720 000000 MOV #0,(R0)+ ;KEY1 IN BUFFER
6076 040604 012720 000000 MOV #0,(R0)+ ;KEY2 IN BUFFER
6077 040610 012701 000400 MOV #256,R1 ;DATA WORD COUNTER
6078 040614 012702 177777 MOV #-1,R2 ;DATA
6079 040620 010220 3$: MOV R2,(R0)+ ;DATA INTO BUFFER
6080 040622 005301 DEC R1 ;COUNT
6081 040624 001375 BNE 3$ ;BRANCH IF 256 NOT DONE
6082
6083 ;*ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
6084 ;*NOW THE INSERTED ERROR WILL BE PUT IN
6085
6086 040626 012737 077757 015232 MOV #77757,@#WRFROM+<5*2> ;INSERTED ERROR
6087 040634 004737 045470 JSR PC,@#PUTREG ;SAVE REGISTERS
6088 040640 005037 015124 CLR @#ERFLG$ ;CLEAR ERROR FLAG
6089
6090
6091 ;*NOW READ DATA BUFFER WILL BE CHECKED
6092
6093 040644 004037 046664 JSR R0,@#COMPAR ;CHECK
6094 040650 015220 WRFROM ;GOOD BUFFER
6095 040652 016264 REINTO ;TEST BUFFER
6096 040654 000404 4+256. ;NUMBER OF WORDS CHECKED
6097 040656 040664 4$ ;RETURN POINT FOR ERROR HEADER
6098 040660 040670 5$ ;RETURN POINT FOR ERROR DATA
6099 040662 040674 ;RETURN FOR GOOD COMPARISON
6100 040664 104004 4$: ERROR 4 ;READ NEXT ERROR
```



```
6106
6107 040674 000004          TST63: SCOPE
6108 040676 012706 001000  MOV      #STACK,SP      ;RESET STACK
6109
6110 040702 012737 000063 017330  MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
6111 040710 012700 054474      MOV      #SECGAP,R0     ;POINTER
6112 040714 012701 000460      MOV      #304.,R1      ;COUNTER
6113 040720 005020          1$: CLR      (R0)+          ;CLEAR SIMULATED DISK AREA
6114 040722 005301          DEC      R1
6115 040724 001375          BNE     1$
6116 040726 004737 045770      JSR     PC,CLDISK      ;THIS IS USED TO SET GENERAL REGISTERS
6117
6118                          ;*THESE ARE FOR ECC TEST ONLY
6119
6120 040732 012737 177777 015142  MOV      #-1,@#TSECC   ;THIS IS AN ECC TEST
6121 040740 005037 050476      CLR     @#POSITI      ;CLEAR ERROR POSITION COUNTER
6122 040744 013737 050472 050474  MOV      @#NCODE,@#NCOUNT ;TEMPORARY N-CODE COUNTER
6123 040752 013737 050500 050506  MOV      @#HARDER,@#HADTMP ;TEMPORARY HARD ERROR COUNTER
6124 040760 005037 050464      CLR     @#GECC1      ;ECC LOW ORDER TO BE GENERATED
6125 040764 005037 050466      CLR     @#GECC2      ;ECC HIGH ORDER TO BE GENERATED
6126 040770 005037 050502      CLR     @#DATENV     ;CLEAR DATA ENVELOPE CLOCK COUNT
6127 040774 005037 050504      CLR     @#ZCODE      ;CLEAR LEADING ZEROS CLOCK COUNT
6128
6129
6130
6131
6132                          ;*THESE ARE TO BE SETUP FOR DISKLESS USE ONLY
6133
6134 041000 012737 010000 056012  MOV      #FMT22,@#WCYL ;FORMAT22=16BIT WORDS AND
6135                          ;CYLINDER 0
6136 041006 012737 000001 056014  MOV      #1,@#WSECTR   ;TRACK=0, SECTOR 1
6137 041014 005037 056016      CLR     @#WKEY1      ;KEY1=0
6138 041020 005037 056020      CLR     @#WKEY2      ;KEY2=0
6139 041024 012737 000400 056052  MOV      #256.,@#FNWORD ;256 DATA WORDS
6140 041032 004537 047176      JSR     R5,@#CRC     ;GO TO CALCULATE CRC
6141 041036 056012          WCYL
6142 041040 056022          GCRC
6143
6144                          ;*THESE ARE REGULAR SETUPS
6145
6146 041042 012777 177374 153700  MOV      #-260.,@#RHWC ;256 DATA WORDS 4 HEADER WORDS
6147 041050 012700 015220      MOV      #WRFROM,R0    ;THESE TWO INSTRUCTIONS GETS
6148 041054 010077 153672      MOV      R0,@#RHBA    ;ADDR. OF WRFROM INTO R0 AND
6149                          ;BUS ADDRESS REGISTER
6150 041060 012720 010000      MOV      #FMT22,(R0)+ ;FORMAT=16 BIT WORDS
6151                          ;CYLINDER=0
6152 041064 012720 000001 2$: MOV      #1,(R0)+      ;TRACK=0, SECTOR-1, KEYS 0
6153 041070 005020          CLR     (R0)+          ;KEY1=0
6154 041072 005020          CLR     (R0)+          ;KEY2=0
6155 041074 012705 000400      MOV      #256.,R5     ;COUNTER
6156 041100 012720 052525 3$: MOV      #52525,(R0)+ ;MOVE ALL 52525 FOR DATA
6157 041104 005305          DEC     R5
6158 041106 001374          BNE     3$            ;BRANCH IF DATA NOT COMPLETE
6159 041110 012777 000001 153644  MOV      #1,@#RHDST   ;TRACK 0 SECTOR=1
6160
6161 041116 004737 046024      JSR     PC,@#CHECKT   ;CHECK DVA,RDY,DPR,DRY 1 AND OTHERS DON'T
```



```

6162 041122 104401 005123 TYPE ,CPHALT ;CANNOT CONTINUE TESTING IF ANY OF THE
6163 041126 000000 HALT ;STOP THE TEST
6164
6165 041130 013711 015172 MOV @WRIFOR,@R1 ;GET READY FOR WRITE HEADER AND
6166 ;DATA WITH 62 IN RHCS1
6167 041134 005037 015124 CLR @WERFLG$ ;CLEAR ERROR FLAG
6168 041140 012777 010000 153620 MOV #FMT22,@RHOF ;FORMAT BIT-1 (16 BIT WORDS)
6169 041146 005077 153616 CLR @RHCA ;CYLINDER =0
6170 041152 004737 055636 JSR PC,@COMWHD ;WRITE HEADER AND DATA
6171
6172 ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
6173 ;*FROM THE 'COMWHD' ROUTINE THAT MEANS ALL HEADER ON DISK
6174 ;*IS GOOD IE. ONLY DATA IS TO BE CHECKED TO SEE IF THEY ARE
6175 ;*ALL 52525 AND WRITE DATA GAP AND TOLERANCE GAP TO SEE IF
6176 ;*THEY ARE ALL ZEROS
6177
6178 041156 005737 015124 TST @WERFLG$ ;HAS ANY ERRORS OCCURED?
6179 ;IF WRITE ERROR OCCURS ECC IS NOT CHECKED
6180 041162 001056 BNE TST64 ;:BRANCH IF YES
6181
6182
6183 ;COMPARE SOFTWARE GENERATED ECC WITH THAT GENERATED BY HARDWARE
6184 041164 023737 050464 055572 CMP @GECC1,@WECC1;COMPARE SOFTWARE ECC WITH HARDWARE ECC
6185 041172 001402 BEQ 6$ ;BRANCH IF GOOD
6186 041174 104031 ERROR 31 ;LOW ORDER ECC IN ERROR
6187 041176 000405 BR 7$ ;BRANCH TO CONTINUE
6188 041200 023737 050466 055574 6$: CMP @GECC2,@WECC2 ;COMPARE SOFTWARE ECC WITH HARDWARE ECC
6189 041206 001401 BEQ 7$ ;BRANCH IF GOOD
6190 041210 104031 ERROR 31 ;HIGH ORDER ECC IN ERROR
6191
6192
6193 041212 7$:
6194 041212 004737 046214 JSR PC,@CHECKE ;CHECK THAT DVA,RDY,DPR,DRY - 1
6195 041216 104401 005123 TYPE ,CPHALT ;CANNOT CONTINUE IF THEY DON'T
6196 041222 000000 HALT ;STOP THE TEST AND RESTART PROGRAM
6197
6198
6199
6200
6201 ;*FILL 'REINTO' BUFFER WITH EXPECTED DATA
6202
6203 041224 004037 045706 JSR RO,@CLAREA ;FILL REINTO BUFFER
6204 041230 016264 REINTO ;FROM
6205 041232 017262 REINTO+<255.*2> ;TO
6206 041234 052525 .WORD 52525 ;DATA
6207
6208 041236 013737 050464 017264 MOV @GECC1,@REINTO+<256.*2>;FILL ECC1
6209 041244 013737 050466 017266 MOV @GECC2,@REINTO+<257.*2>;FILL ECC2
6210 041252 004037 045706 JSR RO,@CLAREA ;FILL REST
6211 041256 017270 REINTO+<258.*2> ;FROM
6212 041260 017324 REINTO+<272.*2> ;TO
6213 041262 000000 0 ;DATA
6214
6215
6216 041264 005037 015124 CLR @WERFLG$ ;CLEAR ERROR FLAG
6217
  
```

```
6218  
6219  
6220  
6221 041270 004037 046564 JSR RO, @COMPAR :CHECK  
6222 041274 016264 REINTO :GOOD BUFFER  
6223 041276 054572 DISK :TEST BUFFER  
6224 041300 000402 258. :NUMBER OF WORDS CHECKED  
6225 041302 041310 4$ :RETURN POINT FOR ERROR HEADER  
6226 041304 041314 5$ :RETURN POINT FOR ERROR DATA  
6227 041306 041320 TST64 :RETURN FOR GOOD COMPARISON  
6228 041310 104007 4$: ERROR 7 :READ ERROR 10 NEXT  
6229 041312 000207 RTS PC :RETURN TO COMPARE  
6230 041314 104010 5$: ERROR 10 :WORD NOS 1 TO 256 ARE  
6231 :DATA WORDS  
6232 :WORD NOS 257 AND 258  
6233 :ARE ECC WHICH ARE CHECKED  
6234 :WORD NOS 259  
6235 :IS DATA GAP  
6236 :WORD NOS 260 TO 273  
6237 :ARE TOLERANCE GAP  
6238 041316 000207 RTS PC :RETURN TO COMPARE
```

```

6239
6240 041320 000004          TST64: SCOPE
6241 041322 012706 001000  MOV      #STACK,SP      ;RESET STACK
6242 041326 012737 000064 017330  MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
6243
6244
6245          ;*SETUP FOR WHAT IS TO BE READ
6246          ;*HEADER CRC IS RESTORED FROM A SUBROUTINE
6247
6248 041334 012746 052525      MOV      #52525,-(SP)    ;DATA TO BE READ
6249 041340 012705 000400      MOV      #256.,R5       ;COUNTER
6250 041344 012700 054572      MOV      #DISK,R0       ;START OF SIMULATED DISK DATA
6251 041350 011620          1$:  MOV      (SP),(R0)+      ;MOVE IN DATA ON TO SIMULATED DISK
6252 041352 005305          DEC      R5              ;COUNT
6253 041354 001375          BNE     1$              ;BRANCH IF 256 NOT COMPLETE
6254 041356 005726          TST     (SP)+           ;UNDO -(SP)
6255 041360 022020          CMP     (R0)+,(R0)+    ;JUMP OVER THE TWO ECC WORDS
6256 041362 012705 000017      MOV      #15.,R5       ;1 DATA GAP
6257          ;14 TOLERANCE GAP
6258 041366 005020          2$:  CLR     (R0)+           ;CLEAR DATA GAP, AND
6259 041370 005305          DEC     R5              ;TOLERANCE GAP
6260 041372 001375          BNE     2$              ;BRANCH IF NOT COMPLETE
6261
6262 041374 004737 051260      JSR     PC,@#FILLEC    ;INSERT THE TWO ECC WORDS ON THE DISK
6263          ;IN THE CORRECT PLACE
6264
6265          ;*THESE ARE FOR ECC TEST ONLY
6266
6267 041400 012737 177777 015142  MOV      #-1,@#TSECC    ;THIS IS AN ECC TEST
6268 041406 005037 050476      CLR     @#POSITI       ;CLEAR ERROR POSITION COUNTER
6269 041412 013737 050472 050474  MOV      @#NCODE,@#NCOUNT ;TEMPORARY N-CODE COUNTER
6270 041420 013737 050500 050506  MOV      @#HARDER,@#HADTMP ;TEMPORARY HARD ERROR COUNTER
6271 041426 005037 050464      CLR     @#GECC1        ;ECC LOW ORDER TO BE GENERATED
6272 041432 005037 050466      CLR     @#GECC2        ;ECC HIGH ORDER TO BE GENERATED
6273 041436 005037 050502      CLR     @#DATENV       ;CLEAR DATA ENVELOPE CLOCK COUNT
6274 041442 005037 050504      CLR     @#ZCODE        ;CLEAR LEADING ZEROS CLOCK COUNT
6275
6276
6277          ;*THESE ARE TO SETUP FOR DISKLESS USE ONLY
6278
6279 041446 012737 010000 052654  MOV      #FMT22,@#CYL   ;16 BITS PER WORD
6280          ;CYLINDER 0, FORMAT 16 BITS
6281 041454 112737 000000 052657  MOV     #0,@#SECOTR+1   ;TRACK 0
6282 041462 112737 000000 052656  MOV     #0,@#SECOTR     ;SECTOR 0
6283 041470 012737 000000 052660  MOV     #0,@#KEY1       ;KEY1=0
6284 041476 012737 000000 052662  MOV     #0,@#KEY2       ;KEY2=0
6285 041504 012737 000400 052734  MOV     #256.,@#DAWORD  ;NO. OF DATA WORDS
6286 041512 005037 052664      CLR     @#X             ;THIS IS A READ COMMAND
6287 041516 004537 047176      JSR     R5,@#CRC        ;GO TO CALCULATE CRC
6288 041522 052654      CYL
6289 041524 054554      WCRC
6290
6291
6292
6293
6294          ;*THESE ARE REGULAR SETUPS

```

```

6295
6296 041526 004737 045770      JSR    PC,@#CLDISK      ;SETUP GENERAL REGISTERS
6297 041532 012777 177374 153210  MOV    #-256.-4.,@RHWC  ;256. DATA 4 HEADER WORDS
6298 041540 012777 016264 153204  MOV    #REINTO,@RHBA   ;STARTING ADDRESS OF READ BUFFER
6299 041546 112746 000000      MOV    #0,-(SP)        ;IN LOWER BYTE GET SECTOR
6300 041552 112766 000000 000001  MOV    #0,1(SP)        ;GET TRACK IN HIGHER BYTE
6301 041560 012677 153176      MOV    (SP)+,@RHDST    ;TRACK/SECTOR IN RHDST
6302 041564 012777 010000 153174  MOV    #FMT22,@RHOF ;16 BITS PER WORD
6303                                     ;ECC CORRECTION NOT INHIBIT
6304                                     ;BECAUSE ECC IS NOT GOING
6305                                     ;TO BE CHECKED
6306 041572 005077 153172      CLR    @RHCA          ;CYLINDER 0
6307
6308 041576 004737 046024      JSR    PC,@#CHECKT    ;CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DON'T
6309 041602 104401 005123      TYPE  ,CPHALT        ;CANNOT CONTINUE TESTING IF ANY OF THE
6310 041606 000000      HALT                ;STOP THE TEST
6311
6312 041610 013711 015176      MOV    @#REFOR,@R1    ;READ HEADER AND DATA-72
6313 041614 005037 015124      CLR    @#ERFLG$      ;CLEAR ERROR FLAG
6314 041620 004737 052514      JSR    PC,@#COMHD    ;READ HEADER AND DATA
6315                                     ;IF THERE ARE READ ERRORS THEN
6316                                     ;ECC WILL NOT BE CHECKED
6317
6318
6319                                     ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
6320                                     ;*FROM THE 'COMHD' ROUTINE THAT MEANS SECTOR GAP,
6321                                     ;*FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
6322                                     ;*SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
6323                                     ;*DETECTED
6324                                     ;*HEADER AND DATA ARE TO BE CHECKED.
6325                                     ;*IN CHECKING READ DATA THE WRITE FROM BUFFER
6326                                     ;*'WRFROM' IS FILLED WITH EXPECTED DATA AND
6327                                     ;*COMPARISONS ARE MADE
6328
6329 041624 005737 015124      TST    @#ERFLG$      ;ANY ERRORS ALREADY THERE
6330 041630 001102      BNE    TST65         ;BRANCH IF YES
6331 041632 004737 045470      JSR    PC,@#PUTREG   ;SAVE REGISTERS
6332 041636 005737 015034      TST    @#ER1        ;NO ERRORS SHOULD BE SET
6333 041642 001401      BEQ    6$           ;BRANCH IF NO ERRORS SET
6334 041644 104032      ERROR  32          ;32 BIT ECC REGISTER SHOULD BE ZERO
6335                                     ;ONLY 11 OF THE 32 BITS CAN BE SEEN
6336                                     ;IN THE PATERN REGISTER
6337                                     ;DCK SHOULD BE SET IN RHER1
6338 041646 013746 050464      6$:  MOV    @#GECC1,-(SP) ;GET PATTERN REGISTER
6339 041652 042716 174000      BIC    #174000,(SP)  ;KEEP ONLY 11 BITS
6340 041656 022637 015064      CMP    (SP)+,@#EC2  ;COMPARE PATTERN REGISTER
6341 041662 001401      BEQ    7$           ;BRANCH IF GOOD
6342 041664 104032      ERROR  32          ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
6343
6344
6345
6346
6347                                     ;*ADD 16 MAINTENANCE CLOCKS TO
6348                                     ;*BRING EBL DOWN
6349
6350 041666 012700 000020      7$:  MOV    #16.,R0      ;COUNTER

```

```

6351 041672 052777 000002 153076 8$: BIS #MCLK,@RHMR ;SET CLOCK
6352 041700 042777 000002 153070 BIC #MCLK,@RHMR ;CLEAR CLOCK
6353 041706 005300 DEC R0 ;COUNT
6354 041710 001370 BNE 8$ ;BRANCH IF 16 CLOCKS NOT DONE
6355 041712 004737 046214 JSR PC,@CHECKE ;CHECK THAT DVA,RDY,DPR,DRY = 1
6356 041716 104401 005123 TYPE ,CPHALT ;CANNOT CONTINUE IF THEY DON'T
6357 041722 000000 HALT ;STOP THE TEST AND RESTART PROGRAM
6358 041724 012700 015220 MOV #WRFROM,R0 ;GETTING READY TO FILL EXPECTED DATA
6359 041730 012720 010000 MOV #0,FMT22,(R0)+ ;CYLINDER 0
6360 041734 112746 000000 MOVB #0,-(SP) ;IN LOWER BYTE GET SECTOR
6361 041740 112766 000000 000001 MOVB #0,1(SP) ;GET TRACK IN HIGHER BYTE
6362 041746 012620 MOV (SP)+,(R0)+ ;GET TRACK/SECTOR IN BUFFER
6363 041750 012720 000000 MOV #0,(R0)+ ;KEY1 IN BUFFER
6364 041754 012720 000000 MOV #0,(R0)+ ;KEY2 IN BUFFER
6365 041760 012701 000400 MOV #256,R1 ;DATA WORD COUNTER
6366 041764 012702 052525 MOV #52525,R2 ;DATA
6367 041770 010220 3$: MOV R2,(R0)+ ;DATA INTO BUFFER
6368 041772 005301 DEC R1 ;COUNT
6369 041774 001375 BNE 3$ ;BRANCH IF 256 NOT DONE
6370 041776 005037 015124 CLR @ERFLG$ ;CLEAR ERROR FLAG
6371 042002 004737 045470 JSR PC,@PUTREG ;SAVE REGISTERS
6372
6373 ;*NOW READ DATA BUFFER WILL BE CHECKED
6374
6375 042006 004037 046664 JSR R0,@COMPAR ;CHECK
6376 042012 015220 WRFROM ;GOOD BUFFER
6377 042014 016264 REINTO ;TEST BUFFER
6378 042016 000404 4+256. ;NUMBER OF WORDS CHECKED
6379 042020 042026 4$ ;RETURN POINT FOR ERROR HEADER
6380 042022 042032 5$ ;RETURN POINT FOR ERROR DATA
6381 042024 042036 TST65 ;RETURN FOR GOOD COMPARISON
6382 042026 104004 4$: ERROR 4 ;READ NEXT ERROR
6383 042030 000207 RTS PC ;RETURN TO 'COMPAR'
6384 042032 104005 5$: ERROR 5 ;WORD NOS 1 TO 4 ARE
6385 ;HEADER WORDS
6386 ;5 TO 260 ARE DATA WORDS
6387 042034 000207 RTS PC ;RETURN TO 'COMPAR'
    
```

```

6388
6389 042036 000004          TST65: SCOPE
6390 042040 012706 001000  MOV      #STACK,SP      ;RESET STACK
6391 042044 012737 000065 017330  MOV      #TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
6392
6393          ;*SETUP FOR WHAT IS TO BE READ
6394          ;*HEADER CRC IS RESTORED FROM A SUBROUTINE
6395
6396 042052 012746 052525  MOV      #52525, -(SP)  ;DATA TO BE READ
6397 042056 012705 000400  MOV      #256.,R5      ;COUNTER
6398 042062 012700 054572  MOV      #DISK,R0      ;START OF SIMULATED DISK DATA
6399 042066 011620          1$:  MOV      (SP),(R0)+     ;MOVE IN DATA ON TO SIMULATED DISK
6400 042070 005305          DEC      R5             ;COUNT
6401 042072 001375          BNE     1$             ;BRANCH IF 256 NOT COMPLETE
6402 042074 005726          TST     (SP)+         ;UNDO -(SP)
6403 042076 022020          CMP     (R0)+,(R0)+   ;JUMP OVER THE TWO ECC WORDS
6404 042100 012705 000017  MOV      #15., R5     ;1 DATA GAP
6405          ;14 TOLERANCE GAP
6406 042104 005020          2$:  CLR     (R0)+         ;CLEAR DATA GAP, AND
6407 042106 005305          DEC     R5            ;TOLERANCE GAP
6408 042110 001375          BNE     2$            ;BRANCH IF NOT COMPLETE
6409
6410
6411 042112 004737 051260  JSR     PC,@#FILLEC   ;INSERT ECC IN PROPER PLACE ON DISK
6412
6413
6414
6415          ;*THESE ARE FOR ECC TEST ONLY
6416
6417 042116 012737 177777 015142  MOV      #-1,@#TSECC   ;THIS IS AN ECC TEST
6418 042124 005037 050476  CLR     @#POSITI      ;CLEAR ERROR POSITION COUNTER
6419 042130 013737 050472 050474  MOV      @#NCODE,@#NCOUNT ;TEMPORARY N-CODE COUNTER
6420 042136 013737 050500 050506  MOV      @#HARDER,@#HADTMP ;TEMPORARY HARD ERROR COUNTER
6421 042144 005037 050464  CLR     @#GECC1       ;ECC LOW ORDER TO BE GENERATED
6422 042150 005037 050466  CLR     @#GECC2       ;ECC HIGH ORDER TO BE GENERATED
6423 042154 005037 050502  CLR     @#DATENV      ;CLEAR DATA ENVELOPE CLOCK COUNT
6424 042160 005037 050504  CLR     @#ZCODE       ;CLEAR LEADING ZEROS CLOCK COUNT
6425
6426
6427          ;*THESE ARE TO SETUP FOR DISKLESS USE ONLY
6428
6429 042164 012737 010000 052654  MOV      #FMT22,@#CYL  ;16 BITS PER WORD
6430          ;CYLINDER 0, FORMAT 16 BITS
6431 042172 112737 000000 052657  MOV     #0,@#SECOTR+1 ;TRACK 0
6432 042200 112737 000000 052656  MOV     #0,@#SECOTR   ;SECTOR 0
6433 042206 012737 000000 052660  MOV     #0,@#KEY1     ;KEY1=0
6434 042214 012737 000000 052662  MOV     #0,@#KEY2     ;KEY2=0
6435 042222 012737 000400 052734  MOV     #256.,@#DAWORD ;NO. OF DATA WORDS
6436 042230 005037 052664  CLR     @#X           ;THIS IS A READ COMMAND
6437 042234 004537 047176  JSR     R5,@#CRC      ;GO TO CALCULATE CRC
6438 042240 052654  CYL
6439 042242 054554  WCRC
6440
6441
6442          ;*THIS IS TO INSERT ERROR
6443          ;*THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
    
```

```
6444      ;*THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
6445      ;*THIS MOVE
6446      ;*THIS CHANGES THE LAST BIT OF THE ECC
6447
6448 042244 013746 055574      MOV    @WECC2,-(SP)      ;GET LAST ECC
6449 042250 005116              COM    (SP)             ;INVERT ALL BITS OF WECC2
6450 042252 042716 077777      BIC    #*(100000,(SP)   ;KEEP BIT 16
6451 042256 042737 100000 055574 BIC    #100000,@WECC2   ;CLEAR BIT 16 IN ECC
6452 042264 052637 055574      BIS    (SP)+,@WECC2     ;THIS WILL SET BIT 16 IF IT WAS 0
6453                          ;OR WILL SET NOTHING IF IT WAS A 1
6454
6455
6456 042270 012737 010026 042444 MOV    #4118.,@#8$      ;INSERT POSITION REG.
6457
6458
6459      ;*THESE ARE REGULAR SETUPS
6460
6461 042276 004737 045770      JSR    PC,@#CLDISK     ;SETUP GENERAL REGISTERS
6462 042302 012777 177374 152440 MOV    #-256.-4.,@RHWC  ;256. DATA 4 HEADER WORDS
6463 042310 012777 016264 152434 MOV    #REINTO,@RHBA    ;STARTING ADDRESS OF READ BUFFER
6464 042316 112746 000000      MOVB   #0,-(SP)        ;IN LOWER BYTE GET SECTOR
6465 042322 112766 000000 000001 MOVB   #0,1(SP)         ;GET TRACK IN HIGHER BYTE
6466 042330 012677 152426      MOV    (SP)+,@RHDST    ;TRACK/SECTOR IN RHDST
6467 042334 012777 010000 152424 MOV    #FMT22,@RHOF ;16 BITS PER WORD
6468                          ;ECC CORRECTION NOT INHIBIT
6469                          ;BECAUSE ECC IS NOT GOING
6470                          ;TO BE CHECKED
6471 042342 005077 152422      CLR    @RHCA           ;CYLINDER 0
6472
6473 042346 004737 046024      JSR    PC,@#CHECKT     ;CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DON'T
6474 042352 104401 005123      TYPE  ,CPHALT         ;CANNOT CONTINUE TESTING IF ANY OF THE
6475 042356 000000              HALT                    ;STOP THE TEST
6476
6477 042360 013711 015176      MOV    @#REFOR,@R1     ;READ HEADER AND DATA=72
6478 042364 005037 015124      CLR    @#ERFLG$        ;CLEAR ERROR FLAG
6479 042370 004737 052514      JSR    PC,@#COMHD      ;READ HEADER AND DATA
6480                          ;IF THERE ARE READ ERRORS THEN
6481                          ;ECC WILL NOT BE CHECKED
6482
6483
6484      ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
6485      ;*FROM THE 'COMHD' ROUTINE THAT MEANS SECTOR GAP,
6486      ;*FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
6487      ;*SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
6488      ;*DETECTED
6489      ;*HEADER AND DATA ARE TO BE CHECKED.
6490      ;*IN CHECKING READ DATA THE WRITE FROM BUFFER
6491      ;*'WRFROM' IS FILLED WITH EXPECTED DATA AND
6492      ;*COMPARISONS ARE MADE
6493
6494 042374 005737 015124      TST    @#ERFLG$        ;ANY ERRORS ALREADY THERE
6495 042400 001074              BNE    TST66           ;BRANCH IF YES
6496 042402 004737 045470      JSR    PC,@#PUTREG     ;SAVE REGISTERS
6497 042406 022737 100000 015034 CMP    #DCK,@#ERI      ;ONLY DATA CHECK ERROR SHOULD BE SET
6498 042414 001401              BEQ    6$              ;BRANCH IF YES
6499 042416 104032              ERROR 32              ;32 BIT ECC REGISTER SHOULD BE NON
```

```
6500                                     :ZERO
6501                                     :ONLY 11 OF THE 32 BITS CAN BE SEEN
6502                                     :IN THE PATTERN REGISTER
6503                                     :DCK SHOULD BE SET IN RHER'
6504 042420 013746 050464 6$: MOV @#GECC1,-(SP) :GET PATTERN REGISTER
6505 042424 042716 174000 BIC #174000,(SP) :KEEP ONLY 11 BITS
6506 042430 022637 015064 CMP (SP)+,@#EC2 :COMPARE PATTERN REGISTER
6507 042434 001401 BEQ 7$ :BRANCH IF GOOD
6508 042436 104032 ERROR 32 :11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
6509
6510 042440 004037 051106 7$: JSR R0,@#ECORR :GO TO ECC CORRECTION PROCESS
6511 042444 010026 8$: 4118. :EXPECTED POSITION REG. WHEN CORRECTION
6512                                     :IS COMPLETE
6513
6514
6515
6516 042446 004737 046214 JSR PC,@#CHECKE :CHECK THAT DVA,RDY,DPR,DRY 1
6517 042452 104401 005123 TYPE ,CPHALT :CANNOT CONTINUE IF THEY DON'T
6518 042456 000000 HALT :STOP THE TEST AND RESTART PROGRAM
6519 042460 012700 015220 MOV #WRFROM,R0 :GETTING READY TO FILL EXPECTED DATA
6520 042464 012720 010000 MOV #0!FMT22,(R0)+ :CYLINDER 0
6521 042470 112746 000000 MOVB #0,-(SP) :IN LOWER BYTE GET SECTOR
6522 042474 112766 000000 000001 MOVB #0,1(SP) :GET TRACK IN HIGHER BYTE
6523 042502 012620 MOV (SP)+,(R0)+ :GET TRACK/SECTOR IN BUFFER
6524 042504 012720 000000 MOV #0,(R0)+ :KEY1 IN BUFFER
6525 042510 012720 000000 MOV #0,(R0)+ :KEY2 IN BUFFER
6526 042514 012701 000400 MOV #256,R1 :DATA WORD COUNTER
6527 042520 012702 052525 MOV #52525,R2 :DATA
6528 042524 010220 3$: MOV R2,(R0)+ :DATA INTO BUFFER
6529 042526 005301 DEC R1 :COUNT
6530 042530 001375 BNE 3$ :BRANCH IF 256 NOT DONE
6531
6532 :*ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
6533 :*NOW THE INSERTED ERROR WILL BE PUT IN
6534 :*BUT INSERTED ERROR IS IN ECC SO DATA IS NOT WRONG
6535
6536
6537 042532 004737 045470 JSR PC,@#PUTREG :SAVE REGISTERS
6538 042536 005037 015124 CLR @#ERFLG$ :CLEAR ERROR FLAG
6539
6540
6541 :*NOW READ DATA BUFFER WILL BE CHECKED
6542
6543 042542 004037 046664 JSR R0,@#COMPAR :CHECK
6544 042546 015220 WRFROM :GOOD BUFFER
6545 042550 016264 REINTO :TEST BUFFER
6546 042552 000404 4+256. :NUMBER OF WORDS CHECKED
6547 042554 042562 4$ :RETURN POINT FOR ERROR HEADER
6548 042556 042566 5$ :RETURN POINT FOR ERROR DATA
6549 042560 042572 TST66 :RETURN FOR GOOD COMPARISON
6550 042562 104004 4$: ERROR 4 :READ NEXT ERROR
6551 042564 000207 RTS PC :RETURN TO 'COMPAR'
6552 042566 104005 5$: ERROR 5 :WORD NOS 1 TO 4 ARE
6553 :HEADER WORDS
6554 :5 TO 260 ARE DATA WORDS
6555 042570 000207 RTS PC :RETURN TO 'COMPAR'
```



```
6788 ;*THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'  
6789 ;*THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING  
6790 ;*THIS MOVE  
6791  
6792 043556 012737 152525 054574 MOV #152525,@#DISK+2 ;FORCE ERROR ON BIT NUMBER 32  
6793 043564 012737 152525 055570 MOV #152525,@#DISK+<255.*2>;FORCE ERROR IN BIT 4096  
6794 043572 012737 010040 043746 MOV #4128.,@#8$ ;INSERT POSITION REG.  
6795  
6796  
6797 ;*THESE ARE REGULAR SETUPS  
6798  
6799 043600 004737 045770 JSR PC,@#CLDISK ;SETUP GENERAL REGISTERS  
6800 043604 012777 177374 151136 MOV #-256.-4.,@RHWC ;256. DATA 4 HEADER WORDS  
6801 043612 012777 016264 151132 MOV #REINTO,@RHBA ;STARTING ADDRESS OF READ BUFFER  
6802 043620 112746 000000 MOVB #0,-(SP) ;IN LOWER BYTE GET SECTOR  
6803 043624 112766 000000 000001 MOVB #0,1(SP) ;GET TRACK IN HIGHER BYTE  
6804 043632 012677 151124 MOV (SP)+,@RHDST ;TRACK/SECTOR IN RHDST  
6805 043636 012777 010000 151122 MOV #FMT22,@RHOF ;16 BITS PER WORD  
6806 ;ECC CORRECTION NOT INHIBIT  
6807 ;BECAUSE ECC IS NOT GOING  
6808 ;TO BE CHECKED  
6809 043644 005077 151120 CLR @RHCA ;CYLINDER 0  
6810  
6811 043650 004737 046024 JSR PC,@#CHECKT ;CHECK DVA,RDY,DPR,DRY - 1 AND OTHERS DON'T  
6812 043654 104401 005123 TYPE ,CPHALT ;CANNOT CONTINUE TESTING IF ANY OF THE  
6813 043660 000000 HALT ;STOP THE TEST  
6814  
6815 043662 013711 015176 MOV @#REFOR,@R1 ;READ HEADER AND DATA=72  
6816 043666 005037 015124 CLR @#ERFLG$ ;CLEAR ERROR FLAG  
6817 043672 004737 052514 JSR PC,@#COMHD ;READ HEADER AND DATA  
6818 ;IF THERE ARE READ ERRORS THEN  
6819 ;ECC WILL NOT BE CHECKED  
6820  
6821  
6822 ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS  
6823 ;*FROM THE 'COMHD' ROUTINE THAT MEANS SECTOR GAP,  
6824 ;*FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND  
6825 ;*SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY  
6826 ;*DETECTED  
6827 ;*HEADER AND DATA ARE TO BE CHECKED.  
6828 ;*IN CHECKING READ DATA THE WRITE FROM BUFFER  
6829 ;*'WRFROM' IS FILLED WITH EXPECTED DATA AND  
6830 ;*COMPARISONS ARE MADE  
6831  
6832 043676 005737 015124 TST @#ERFLG$ ;ANY ERRORS ALREADY THERE  
6833 043702 001111 BNE TST70 ; BRANCH IF YES  
6834 043704 004737 045470 JSR PC,@#PUTREG ;SAVE REGISTERS  
6835 043710 022737 100000 015034 CMP #DCK,@#ER1 ;ONLY DATA CHECK ERROR SHOULD BE SET  
6836 043716 001401 BEQ 6$ ;BRANCH IF YES  
6837 043720 104032 ERROR 32 ;32 BIT ECC REGISTER SHOULD BE NON  
6838 ;ZERO  
6839 ;ONLY 11 OF THE 32 BITS CAN BE SEEN  
6840 ;IN THE PATTERN REGISTER  
6841 ;DCK SHOULD BE SET IN RHER1  
6842 043722 013746 050464 6$: MOV @#GECC1,-(SP) ;GET PATTERN REGISTER  
6843 043726 042716 174000 BIC #174000,(SP) ;KEEP ONLY 11 BITS
```

```

6844 043732 022637 015064      CMP      (SP)+,@#ECC      ;COMPARE PATTERN REGISTER
6845 043736 001401      BEQ      7$              ;BRANCH IF GOOD
6846 043740 104032      ERROR    32              ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
6847
6848 043742 004037 051106      7$:      JSR      RO,@#ECORR    ;GO TO ECC CORRECTION PROCESS
6849 043746 000000      8$:      .WORD
6850
6851
6852
6853
6854 043750 004737 045470      JSR      PC,@#PUTREG     ;SAVE REGISTERS
6855 043754 022737 100100 015034      CMP      #DCK.ECH,@#ER1 ;WITH ERRORS INSERTED IN BIT POSITION 32
6856                                ;AND 4096 HARD ERROR BIT SHOULD SET
6857 043762 001401      BEQ      9$              ;BRANCH IF GOOD
6858 043764 104036      ERROR    36              ;WITH ERROR INSERTED IN BIT POSITION 21 THRU
6859                                ;32 HCE SHOULD SET
6860
6861
6862
6863
6864 043766                                9$:
6865 043766 004737 046214      JSR      PC,@#CHECKE    ;CHECK THAT DVA,RDY,DPR,DRY = 1
6866 043772 104401 005123      TYPE     ,CPHALT        ;CANNOT CONTINUE IF THEY DON'T
6867 043776 000000      HALT
6868 044000 012700 015220      MOV      #WRFROM,RO     ;GETTING READY TO FILL EXPECTED DATA
6869 044004 012720 010000      MOV      #0!FMT22,(RO)+ ;CYLINDER 0
6870 044010 112746 000000      MOV      #0,-(SP)       ;IN LOWER BYTE GET SECTOR
6871 044014 112766 000000 000001      MOV      #0,1(SP)       ;GET TRACK IN HIGHER BYTE
6872 044022 012620      MOV      (SP)+,(RO)+    ;GET TRACK/SECTOR IN BUFFER
6873 044024 012720 000000      MOV      #0,(RO)+       ;KEY1 IN BUFFER
6874 044030 012720 000000      MOV      #0,(RO)+       ;KEY2 IN BUFFER
6875 044034 012701 000400      MOV      #256,R1        ;DATA WORD COUNTER
6876 044040 012702 052525      MOV      #52525,R2      ;DATA
6877 044044 010220 3$:      MOV      R2,(RO)+       ;DATA INTO BUFFER
6878 044046 005301      DEC      R1              ;COUNT
6879 044050 001375      BNE     3$              ;BRANCH IF 256 NOT DONE
6880
6881                                ;*ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
6882                                ;*NOW THE INSERTED ERROR WILL BE PUT IN
6883
6884 044052 012737 152525 015232      MOV      #152525,@#WRFROM+<5*2> ;INSERTED ERROR IN BIT 32
6885 044060 012737 152525 016226      MOV      #152525,@#WRFROM+<259.*2> ;INSERT ERROR IN BIT 4096
6886
6887 044066 005037 015124      CLR      @#ERFLG$       ;CLEAR ERROR FLAG
6888 044072 004737 045470      JSR      PC,@#PUTREG     ;SAVE REGISTERS
6889
6890                                ;*NOW READ DATA BUFFER WILL BE CHECKED
6891
6892 044076 004037 046664      JSR      RO,@#COMPAR     ;CHECK
6893 044102 015220      WRFROM                    ;GOOD BUFFER (CHANGED)
6894 044104 016264      REINTO                    ;TEST BUFFER
6895 044106 000404      4+256.                    ;NUMBER OF WORDS CHECKED
6896 044110 044116      4$                          ;RETURN POINT FOR ERROR HEADER
6897 044112 044122      5$                          ;RETURN POINT FOR ERROR DATA
6898 044114 044126      TST70                      ;RETURN FOR GOOD COMPARISON
6899 044116 1040C4      4$:      ERROR    4              ;READ NEXT ERROR
    
```


CZRJHCO, RPO4/5/6 DSKLS CTRLR2
CZRJHC.P12 10-NOV-77 11:09

MACY11 30A(1052) 27-JUL-78 11:16 PAGE 161
T67 READ ECC ENABLED 3D

D 13

SEQ 0159

6900 044120 000207
6901 044122 104005
6902
6903
6904 044124 000207

58: RTS PC
ERROR 5
RTS PC

:RETURN TO 'COMPAR'
:WORD NOS 1 TO 4 ARE
:HEADER WORDS
:5 TO 260 ARE DATA WORDS
:RETURN TO 'COMPAR'

CZ
CZ


```
6965 044334 000004          TST72: SCOPE
6966 044336 012737 000001 001212  MOV    #1,$TIMES      ;;DO 1 ITERATION
6967 044344 004737 045770          JSR    PC,@#CLDISK
6968
6969 044350 012737 000000 177776  MOV    #0,PS          ;REINSTATE PS TO 0
6970 044356 104401 044364          TYPE  ,65$           ;;TYPE ASCIZ STRING
6971 044362 000425          BR     64$           ;;GET OVER THE ASCIZ
6972 044436 013746 015112  MOV    @#UNIT,-(SP)   ;GET READY TO TYPE UNIT NUMBER
6973 044442 104405          TYPDS
6974 044444 104401 044452  TYPE  ,67$           ;;TYPE ASCIZ STRING
6975 044450 000402          BR     66$           ;;GET OVER THE ASCIZ
6976 044456 013746 001112  MOV    @#SERTTL,-(SP) ;GET READY TO TYPE NUMBER OF ERRORS
6977 044462 104405          TYPDS
6978 044464 005037 001112  CLR    @#SERTTL       ;CLEAR TOTAL NUMBER OF ERRORS
6979 044470 005037 001102  CLR    @#STSTNM       ;CLEAR TEST NUMBER
6980 044474 005737 015120  TST   @#SELECT        ;STARTING FROM 200 ?
6981 044500 001413          BEQ    3$            ;TEST NEXT DRIVE IF SO
6982
6983 044502 005237 001100          INC    @#$PASS        ;INCREASE PASS COUNT
6984 044506 104401 044671          TYPE  ,SENDMG        ;TYPE END PASS #
6985 044512 013746 001100  MOV    @#$PASS,-(SP)
6986 044516 104405          TYPDS
6987 044520 104401 044666  TYPE  ,SENULL
6988 044524 000137 022642  JMP    @#TST5        ;CONTINUE TESTING THIS DRIVE ----->
6989
6990 044530 005337 015114          3$:  DEC    @#NOUNITS    ;NO. OF UNITS PRESENT DECREMENTED
6991 044534 001413          BEQ    $EOP          ;BRANCH IF ALL DRIVES COMPLETE
6992 044536 013700 015112  MOV    @#UNIT,R0      ;UNIT UNDER TEST
6993 044542 012701 015072  MOV    #UNITS,R1      ;TABLE
6994 044546 022100          1$:  CMP    (R1)+,R0      ;IS THIS UNIT JUST TESTED
6995 044550 001401          BEQ    2$            ;BRANCH IF YES
6996 044552 000775          BR     1$            ;BRANCH IF NO
6997 044554 011137 015112  2$:  MOV    (R1),@#UNIT   ;THIS IS NEXT UNIT
6998 044560 000137 022642  JMP    @#TST5        ;TEST NEXT DRIVE ----->
```

```
6999
7000          .SBTTL
7001          .SBTTL  ***SUBROUTINES***
7002          .SBTTL
7003
7004
7005
7006
7007 044564 000004          SCOPE
7008 044566 005037 001102  CLR      $TSTNM      ;;ZERO THE TEST NUMBER
7009 044572 005037 001212  CLR      $TIMES      ;;ZERO THE NUMBER OF ITERATIONS
7010 044576 005237 001100  INC      $PASS       ;;INCREMENT THE PASS NUMBER
7011 044602 042737 100000 001100  BIC      #100000,$PASS ;;DON'T ALLOW A NEG. NUMBER
7012 044610 005327          DEC      (PC)+      ;;LOOP?
7013 044612 000001          $EOPCT: .WORD 1
7014 044614 003022          BGT      $DOAGN     ;;YES
7015 044616 012737          MOV      (PC)+,@(PC)+ ;;RESTORE COUNTER
7016 044620 000001          $ENDCT: .WORD 1
7017 044622 044612          $EOPCT
7018 044624 104401 044671  TYPE     $ENDMG     ;;TYPE 'END PASS #'
7019 044630 013746 001100  MOV      $PASS,-(SP) ;;SAVE $PASS FOR TYPEOUT
7020 044634 104405          TYPDS     ;;GO TYPE--DECIMAL ASCII WITH SIGN
7021 044636 104401 044666  TYPE     $ENULL     ;;TYPE A NULL CHARACTER
7022 044642 013700 000042  $GET42: MOV     @#42,R0 ;;GET MONITOR ADDRESS
7023 044646 001405          BEQ     $DOAGN     ;;BRANCH IF NO MONITOR
7024 044650 000005          RESET    ;;CLEAR THE WORLD
7025 044652 004710          $ENDAD: JSR    PC,(R0) ;;GO TO MONITOR
7026 044654 000240          NOP
7027 044656 000240          NOP      ;;SAVE ROOM
7028 044660 000240          NOP      ;;FOR
7029 044662 000137          NOP      ;;ACT11
7030 044664 021034          JMP     @(PC)+    ;;RETURN
7031 044666      377      377      000  $RTNAD: .WORD  TST1
7032 044671      015     042412 042116 $ENULL: .BYTE  -1,-1,0 ;;NULL CHARACTER STRING
7033 044676 050040 051501 020123 $ENDMG: .ASCIIZ <15><12>/END PASS #/
7034 044704 000043
7035
7036
7037
7038
7039
7040          ;*HERE IS A DETAILED EXPLANATION OF HOW THE LOOP ON ERROR WORKS.
7041          ;*ON HITTING AN ERROR IF THE LOOP ON ERROR SWITCH IS SET, THE
7042          ;*PROGRAM GOES BACK - USUALLY BACK TO THE BEGINNING OF THE TEST.
7043
7044          ;*WHEN THIS OPERATOR SELECTABLE SCOPE LOOP IS USED THEN THE POINT
7045          ;*THE PROGRAM GOES BACK TO CAN BE CHANGED.
7046          ;*THE RESTRICTIONS TO THE POINT WHERE THE PROGRAM CAN GO ARE: -
7047          ;*1. IT MUST BE WITHIN THE TEST UNDER CONSIDERATION
7048          ;*2. LOOP ON ERROR SWITCH MUST BE SET
7049          ;*3. THE ERROR MUST OCCUR WITHIN THE TEST UNDER CONSIDERATION
7050          ;*IF THE ERROR DOES NOT OCCUR WITHIN THE TEST UNDER CONSIDERATION
7051          ;*THE PROGRAM WILL REVERT TO NORMAL OPERATION. HOWEVER, IF LOOP ON
7052          ;*TEST SWITCH IS SET AND THIS OPERATOR SELECTABLE SCOPE LOOP IS USED
7053          ;*THEN THE PROGRAM WILL LOOP BACK TO THE SELECTED POINT WHEN IT
7054          ;*COMES TO THE END OF THE TEST UNDER CONSIDERATION.
```

7055
7056
7057
7058
7059
7060 044706 000000
7061
7062 044710
7063 044710 005037 177776
7064 044714 104401 044722
7065 044720 000421
7066 044764 013746 017330
7067 044770 104402
7068 044772 104401 045000
7069 044776 000414
7070 045030 013746 001110
7071 045034 104402
7072 045036 104401 001223
7073 045042 104401 045050
7074 045046 000426
7075 045124 104401 045132
7076 045130 000420
7077 045172 104401 045200
7078 045176 000423
7079 045246 104412
7080 045250 062716 000002
7081 045254 012637 001106
7082 045260 104401 045266
7083 045264 000417
7084 045324 104401 045332
7085 045330 000441
7086 045434 104412
7087 045436 012637 001110
7088 045442 013746 001106
7089
7090 045446 005037 052770
7091 045452 005037 015142
7092
7093
7094
7095 045456 005037 050470
7096
7097
7098
7099 045462 005037 015144
7100 045466 000002

;*
;*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 @#SLPERR,-(SP) ;:GET READY TO TYPE LOOP BACK PC
TYPOC
TYPE ,SCLRF
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)+,@#SLPADR
TYPE ,75\$;:TYPE ASCIZ STRING
BR 74\$;:GET OVER THE ASCIZ
TYPE ,77\$;:TYPE ASCIZ STRING
BR 76\$;:GET OVER THE ASCIZ
RDOCT
MOV (SP)+,@#SLPERR ;GET LPERR
MOV @#SLPADR,-(SP)
;THIS CLEARS UP GARBAGE
CLR @#NOSYNC ;CLEAR FLAG FOR HEADER ERROR COMMANDS
CLR @#TSECC ;CLEAR FLAG FOR ECC TEST
;WHEN =-177777 IT IS AN ECC TEST
;WHEN =0 IT IS NOT AN ECC TEST
CLR @#TSECCG ;EVEN IN AN ECC TEST EVERY CLOCK
;IS NOT TO GENERATE ECC
;IF =177777 GENERATE ECC
;IF =0 DO NOT GENERATE ECC
CLR @#TESDTE ;DRIVE TIMING ERROR TEST
RTI

```
7101
7102          .SBYTL  SAVE REGISTERS ROUTINE
7103
7104          ;THIS SAVES THE CONTENTS OF ALL HARDWARE REGISTERS
7105          ;IN MEMORY LOCATIONS TAGED FROM 'WC' TO 'EC2'
7106          ;
7107          ;
7108          ;THIS IS DONE SO THAT COMPARES ARE DONE WITH SAVED LOCATIONS
7109          ;AND NOT THE REGISTERS THEMSELVES.  THIS WILL MAKE
7110          ;ERROR PRINTOUTS FOR GOOD AND BAD DATA ALWAYS DIFFRENT
7111
7112          PUTREG:
7113          045470 010046      MOV     R0,-(SP)          ;;PUSH R0 ON STACK
7114          045472 010146      MOV     R1,-(SP)          ;;PUSH R1 ON STACK
7115          045474 010246      MOV     R2,-(SP)          ;;PUSH R2 ON STACK
7116          045476 01270C 014750  MOV     #RHC,R0          ;STARTING ADDRESS OF REG
7117          045502 012701 015024  MOV     #WC,R1          ;STARTING ADDRESS OF SAVE LOCATIONS
7118          045506 012702 000023  MOV     #RHC-RHC+2/2,R2 ;NUMBER OF REG. INTO R2
7119          045512 013021      10$:  MOV     @R0+,R1+        ;SAVE HARDWARE REG.
7120          045514 005302      DEC     R2
7121          045516 001375      BNE    10$
7122          045520 012602      MOV     (SP)+,R2        ;;POP STACK INTO R2
7123          045522 012601      MOV     (SP)+,R1        ;;POP STACK INTO R1
7124          045524 012600      MOV     (SP)+,R0        ;;POP STACK INTO R0
7125          045526 000207      RTS     PC
```



```
7165
7166          .SBTTL  CLEAR MEMORY ROUTINE
7167
7168
7169          ;*      THIS CLEARS ANY BLOCK OF MEMORY
7170          ;*      FILLING IT WITH ANY DATA
7171          ;*
7172          ;*      CALL
7173          ;*      JSR      R0,CLAREA
7174          ;*      X              ;STARTING ADDRESS OF BLOCK
7175          ;*      Y
7176          ;*      Z              ;DATA TO BE FILLED
7177          ;*R1 WILL HAVE STARTING ADDRESS OF BLOCK TO BE FILLED
7178          ;*R2 AFTER SUBTRACTION WILL HAVE TWICE NUMBER OF LOCATIONS
7179          ;*R3 WILL HAVE DATA TO BE FILLED
7180          ;*TO AVOID DIVIDE ROUTINE TWO DECREMENT R2 WILL BE USED
7181
7182          CLAREA:
7183          045706 010146      MOV      R1,-(SP)          ;;PUSH R1 ON STACK
7184          045710 010246      MOV      R2,-(SP)          ;;PUSH R2 ON STACK
7185          045712 010346      MOV      R3,-(SP)          ;;PUSH R3 ON STACK
7186          045714 012001      MOV      (R0)+,R1          ;FROM
7187          045716 012002      MOV      (R0)+,R2          ;TO
7188          045720 012003      MOV      (R0)+,R3          ;DATA
7189          045722 160102      SUB      R1,R2          ;NO. OF LOCATIONS MINUS TWO
7190          045724 062702 000002 ADD      #2,R2          ;GET TWICE NO OF LOCATIONS
7191          045730 010321      1$:    MOV      R3,(R1)+      ;MOVE IN DATA
7192          045732 005302      DEC      R2
7193          045734 005302      DEC      R2
7194          045736 001374      BNE      1$          ;BRANCH IF NOT COMPLETE
7195          045740 012603      MOV      (SP)+,R3      ;;POP STACK INTO R3
7196          045742 012602      MOV      (SP)+,R2      ;;POP STACK INTO R2
7197          045744 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
7198          045746 000200      RTS      R0          ;RETURN
```



```
7284 046212 000207          RTS      PC          ;RETURN TO TEST AND CONTINUE TESTING
7285
7286
7287          ;*THIS CHECKS THAT DEVICE AVAILABLE (DVA) AND READY (RDY) IN RHCS1 = 1
7288          ;*AND CHECKS THAT DEVICE PRESENT (DPR), DEVICE READY (DRY) IN RHDS1 = 1
7289          ;*IT DOES NOT CHECK THAT OTHER BITS IN THESE REGISTERS = 0
7290
7291 046214 011637 015132  CHECKE: MOV      (SP),@#PCJSR      ;SAVE PC OF JSR+4
7292 046220 162737 000004 015132  SUB      #4,@#PCJSR      ;GET PC OF JSR
7293 046226 004737 045470          JSR      PC,@#PUTREG      ;SAVE REGISTERS
7294 046232 032737 000200 015032  BIT      #RDY,@#CS1      ;RHCS1 SHOULD HAVE DEVICE AVAILABLE
7295          ;AND BE READY
7296 046240 001004          BNE      1$              ;BRANCH IF GOOD
7297 046242 010137 001122  MOV      R1,@#$BDADR      ;FAILING REGISTER
7298 046246 104026          ERROR    26              ;RHCS1 IS IN ERROR
7299          ;DOES NOT HAVE DVA, RDY
7300 046250 000427          BR       4$              ;BRANCH
7301
7302 046252 032737 004000 015032  1$: BIT      #DVA,@#CS1      ;RHCS1 SHOULD HAVE DEVICE AVAILABLE
7303          ;AND BE READY
7304 046260 001004          BNE      2$              ;BRANCH IF GOOD
7305 046262 010137 001122  MOV      R1,@#$BDADR      ;FAILING REGISTER
7306 046266 104026          ERROR    26              ;RHCS1 IS IN ERROR
7307          ;DOES NOT HAVE DVA, RDY
7308 046270 000417          BR       4$              ;BRANCH OUT
7309 046272 032737 000200 015054  2$: BIT      #DRY,@#DS1      ;RHDS1 SHOULD HAVE DPR,DRY
7310 046300 001004          BNE      3$              ;BRANCH IF THERE
7311 046302 010337 001122  MOV      R3,@#$BDADR      ;FAILING REGISTER RHDS1
7312 046306 104026          ERROR    26              ;RHDS1 DOES NOT HAVE DPR,DRY
7313 046310 000407          BR       4$              ;BRANCH OUT
7314 046312 032737 000400 015054  3$: BIT      #DPR,@#DS1      ;RHDS1 SHOULD HAVE DPR,DRY
7315 046320 001004          BNE      5$              ;BRANCH IF THERE
7316 046322 010337 001122  MOV      R3,@#$BDADR      ;FAILING REGISTER RHDS1
7317 046326 104026          ERROR    26              ;RHDS1 DOES NOT HAVE DPR,DRY
7318 046330 000207          4$: RTS      PC          ;RETURN TO TEST AND HALT - FATAL ERROR
7319
7320 046332 062716 000006  5$: ADD      #6,(SP)        ;ADJUST STACK TO GET OVER HALT IN TEST
7321 046336 000207          RTS      PC          ;RETURN TO TEST AND CONTINUE TESTING
7322
7323          .SBTTL WAIT LOOP
7324          ;*
7325          ;* WAIT LCOP
7326          ;* ONE LOOP OR ONE COUNT = 5.15 MICROSEC WITH BIPOLAR MEMORY (MIN)
7327          ;* ONE LOOP OR ONE COUNT = 11.86 MICROSEC WITH CORE (MIN)
7328          ;* WITH CORE ERROR IS INDICATED AFTER ABOUT 650 MILLISEC (MIN)
7329 046340 177777          TIMCNT: 177777          ;WAITING COUNT
7330
7331 046342 010046          WAIT.T: MOV      R0,-(SP)      ;SAVE R0
7332 046344 016600 000002  MOV      2(SP),R0        ;GET ADRESS OF REG. ADDRESS
7333 046350 010037 001204  MOV      R0,@#$TMP3      ;WAT PC+2 IN $TMP3
7334 046354 162737 000002 001204  SUB      #2,@#$TMP3      ;WAT PC FOR TYPEOUT
7335 046362 012037 001176  MOV      (R0)+,@#$TMP0    ;WAIT REGISTER ADDRESS
7336 046366 012037 001200  MOV      (R0)+,@#$TMP1    ;WAIT ON BIT
7337 046372 010066 000002  MOV      R0,2(SP)        ;RESTORE RETURN ON STACK
7338 046376 012600          MOV      (SP)+,R0        ;RESTORE R0
7339 046400 013737 046340 001202  MOV      @#TIMCNT,@#$TMP2;TEMPORARY COUNT
```

```
7340
7341 046406 033777 001200 132562 1$: BIT @#STMP1,@#STMP0 ;IS REQUIRED BIT THERE?
7342 046414 001021 BNE 2$ ;BRANCH IF YES
7343 046416 005337 001202 DEC @#STMP2 ;COUNT
7344 046422 001371 BNE 1$ ;BRANCH IF NOT TIME UP
7345 046424 013737 046340 001202 MOV @#TIMCNT,@#STMP2 ;TEMPORARY COUNT
7346 046432 033777 001200 132536 3$: BIT @#STMP1,@#STMP0 ;IS REQUIRED BIT THERE?
7347 046440 001007 BNE 2$ ;BRANCH IF YES
7348 046442 005337 001202 DEC @#STMP2 ;COUNT
7349 046446 001371 BNE 3$ ;BRANCH IF NOT TIME UP
7350 046450 017737 132522 001126 MOV @#STMP0,@#SBDDAT ;REGISTER CONTENTS
7351 046456 104016 ERROR 16 ;WAITED ON BIT FAILED TO SET
7352 046460 000002 2$: RTI
7353
7354
7355 ;* CALL FOR THE ABOVE WAITLOOP IS
7356 ;*
7357 ;* MOV @A,@#X$ ;A CONTAINS REGISTER ADDRESS
7358 ;* - - - ;HENCE X$ WILL HAVE ABSOLUTE REG. ADR.
7359 ;* - - -
7360 ;* - - -
7361 ;* WAT
7362 ;*X$: 0 ;ABSOLUTE REG. ADDRESS UNDER WAIT
7363 ;* .WORD 0 ;BIT WAITED FOR
7364 ;* ;CONTINUE
```

```
7365
7366 .SBTTL SAVE ROUTINE
7367
7368 ;THIS IS A SUBROUTINE TO READ & SAVE REGISTERS
7369 ;IN THE REGISTER TABLE TO ANY LOCATION
7370 ;THE CALL IS
7371 ;JSR R0,@SAVER
7372 ;FROM
7373 ;TO
7374 ;NUMBER OF WORDS SAVED
7375
7376 SAVER:
7377 046462 010146 MOV R1,-(SP) ;:PUSH R1 ON STACK
7378 046464 010246 MOV R2,-(SP) ;:PUSH R2 ON STACK
7379 046466 010346 MOV R3,-(SP) ;:PUSH R3 ON STACK
7380 046470 012001 MOV (R0)+,R1 ;FROM
7381 046472 012002 MOV (R0)+,R2 ;TO
7382 046474 012003 MOV (R0)+,R3 ;NUMBER
7383 046476 013122 1$: MOV @(R1)+,(R2)+ ;SAVE REGISTER CONTENTS
7384 046500 005303 DEC R3 ;COUNT
7385 046502 001375 BNE 1$ ;BRANCH IF NOT DONE
7386 046504 012603 MOV (SP)+,R3 ;:POP STACK INTO R3
7387 046506 012602 MOV (SP)+,R2 ;:POP STACK INTO R2
7388 046510 012601 MOV (SP)+,R1 ;:POP STACK INTO R1
7389 046512 000200 RTS R0
7390
7391
7392
7393
7394 .SBTTL WRITE CHECK ROUTINE
7395
7396 ;THIS IS A SUBROUTINE TO DO WRITE CHECK HEADER AND DATA
7397 ;CYLINDER 0, TRACK 1, SECTOR 1, KEYS 0
7398
7399 ;THESE ARE TO SET UP FOR DISKLESS USE ONLY
7400
7401 046514 012737 010000 052654 WRCHHD: MOV #FMT22,@#CYL ;CYLINDER 0 FORMAT 16 BIT WORDS
7402 046522 112737 000001 052657 MOV #1,@#SECOTR+1 ;TRACK=1
7403 046530 112737 000001 052656 MOV #1,@#SECOTR ;SECTOR=1
7404 046536 005037 052660 CLR @#KEY1 ;KEY1=0
7405 046542 005037 052662 CLR @#KEY2 ;KEY2=0
7406 046546 012737 000044 052734 MOV #36.,DAWORD ;NO OF DATA WORDS
7407 046554 005037 052664 CLR @#X ;THIS IS A READ OPERATION
7408 046560 004537 047176 JSR R5,@#CRC ;GO TO CALCULATE CRC
7409 046564 052654 CYL
7410 046566 054554 WCRC
7411
7412 ;THESE ARE REGULAR SETUPS
7413
7414 046570 004737 045770 JSR PC,@#CLDISK ;SET UP GENERAL REGISTERS
7415 ;AND CLEAR DISK REGISTERS
7416 046574 012777 177730 146146 MOV #-40.,@RHWC ;36 DATA WORDS 4 HEADER WORDS
7417 046602 012777 016264 146142 MOV #REINTO,@RHBA ;STARTING ADDRESS OF READ BUFFER
7418 046610 112746 000001 MOV #1,-(SP) ;SECTOR=1
7419 046614 112766 000001 000001 MOV #1,1(SP) ;TRACK=1 IN UPPER BYTE
7420 046622 012677 146134 MOV (SP)+,@RHDST ;TRACK=1, SECTOR=1 IN RHDST
```

```
7421 046626 012777 014000 146132      MOV      #FMT22!ECI,@RHOF ;16 BIT WORDS
7422                                     ;ECC CORRECTION INHIBIT BECAUSE
7423                                     ;ECC LOGIC IS NOT CHECKED YET
7424 046634 005077 146130      CLR      @RHCA ;CYLINDER=0
7425 046640 004737 046024      JSR      PC,@#CHECKT ;CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DON'T
7426 046644 104401 005123      TYPE    ,CPHALT ;CANNOT CONTINUE TESTING IF ANY OF THE
7427 046650 000000              HALT    ;STOP THE TEST
7428 046652 013711 015166      MOV      @#WRCHDT,@R1 ;WRITE CHECK HEADER AND DATA=52
7429                                     ;INTO RHCS1
7430 046656 004737 052514      JSR      PC,@#COMHD ;WRITE CHECK HEADER AND DATA
7431                                     ;SAME AS READ HEADER AND DATA
7432
7433 046662 000207              RTS     PC ;RETURN TO WRITE CHECK TEST
```

CZ
CZ

```

7434
7435
7436
7437
7438
7439
7440
7441
7442
7443
7444
7445 046664
7446 046664 010146
7447 046666 010246
7448 046670 010346
7449 046672 010446
7450 046674 010546
7451 046676 012001
7452 046700 012002
7453 046702 012003
7454 046704 012037 001176
7455 046710 012037 001200
7456 046714 011000
7457 046716 010304
7458 046720 005204
7459 046722 010437 052774
7460 046726 022122
7461 046730 001426
7462
7463 046732 014137 001124
7464 046736 014237 001126
7465 046742 160337 052774
7466 046746 005737 015124
7467 046752 001003
7468 046754 004777 132216
7469 046760 000402
7470 046762 004777 132212
7471 046766 022122
7472 046770 017746 132144
7473 046774 042716 177177
7474 047000 022726 000200
7475 047004 001402
7476 047006 005303
7477 047010 001344
7478 047012
7479 047012 012605
7480 047014 012604
7481 047016 012603
7482 047020 012602
7483 047022 012601
7484 047024 000200

```

.SBTTL COMPARE ROUTINE

```

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

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

MOV -(R1),@#$GDDAT ;GOOD DATA
MOV -(R2),@#$BDDAT ;BAD DATA
SUB R3,@#ERWORD    ;ERROR WORD NO.
TST @#ERFLG$       ;ANY ERRORS ALREAY THERE
BNE 2$            ;BRANCH IF YES
JSR PC,@$TMP0      ;RETURN TO PRINT HEADER
BR 5$             ;BRANCH TO AVOID PRINTING NEXT ERROR
2$: JSR PC,@$TMP1   ;RETURN TO PRINT DATA
5$: CMP (R1)+,(R2)+ ;UNDO -(R1) AND -(R2) FOR ERRORS
MOV @SWR,-(SP)     ;GET SWITCH SETTING
BIC #^C600,(SP)   ;KEEP ONLY SWITCH 7 AND 8
CMP #SW07,(SP)+   ;IS 7 SET AND 8 RESET
BEQ 4$            ;BRANCH OUT IF YES
3$: DEC R3         ;COUNT
BNE 1$           ;BRANCH IF ALL NOT DEVICE
4$: MOV (SP)+,R5   ;;POP STACK INTO R5
MOV (SP)+,R4     ;;POP STACK INTO R4
MOV (SP)+,R3     ;;POP STACK INTO R3
MOV (SP)+,R2     ;;POP STACK INTO R2
MOV (SP)+,R1     ;;POP STACK INTO R1
RTS R0           ;RETURN TO MAIN PROGRAM

```



```
7485 .SBTTL WRITE CHECK DATA
7486
7487 ;THIS IS A SUBROUTINE TO DO WRITE CHECK DATA
7488 ;CYLINDER 0, TRACK 1, SECTOR 1, KEYS 0
7489
7490
7491 ;THESE ARE TO SET UP FOR DISKLESS USE ONLY
7492
7493 047026 012737 010000 052654 WRCHDA: MOV #FMT22,@#CYL ;CYLINDER 0 FORMAT 16 BIT WORDS
7494 047034 112737 000001 052657 MOVB #1,@#SECOTR+1 ;TRACK=1
7495 047042 112737 000001 052656 MOVB #1,@#SECOTR ;SECTOR=1
7496 047050 005037 052660 CLR @#KEY1 ;KEY1=0
7497 047054 005037 052662 CLR @#KEY2 ;KEY2=0
7498 047060 012737 000040 052734 MOV #32.,@#DAWORD ;NO OF DATA WORDS
7499 047066 005037 052664 CLR @#X ;THIS IS A READ OPERATION
7500
7501 047072 004537 047176 JSR R5,@#CRC ;GO TO CALCULATE CRC
7502 047076 052654 CYL
7503 047100 054554 WCRC
7504
7505 ;THESE ARE REGULAR SETUPS
7506
7507 047102 004737 045770 JSR PC,@#CLDISK ;SET UP GENERAL REGISTERS
7508 ;AND CLEAR DISK REGISTERS
7509
7510 047106 012777 177740 145634 MOV #-32.,@RHWC ;36 DATA WORDS 4 HEADER WORDS
7511 047114 012777 016264 145630 MOV #REINTO,@RHBA ;STARTING ADDRESS OF READ BUFFER
7512 047122 112746 000001 MOVB #1,-(SP) ;SECTOR=1
7513 047126 112766 000001 000001 MOVB #1,1(SP) ;TRACK=1 IN UPPER BYTE
7514 047134 012677 145622 MOV (SP)+,@RHDST ;TRACK=1, SECTOR=1 IN RHDST
7515 047140 012777 014000 145620 MOV #FMT22!ECI,@RHOF ;16 BIT WORDS
7516 ;ECC CORRECTION INHIBIT BECAUSE
7517 ;ECC LOGIC IS NOT CHECKED YET
7518 047146 005077 145616 CLR @RHCA ;CYLINDER=0
7519 047152 004737 046024 JSR PC,@#CHECKT ;CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DON'T
7520 047156 104401 005123 TYPE ,CPHALT ;CANNOT CONTINUE TESTING IF ANY OF THE
7521 047162 000000 HALT ;STOP THE TEST
7522 047164 013711 015164 MOV @#WRCHKEK,@R1 ;WRITE CHECK DATA=50 INTO RHCS1
7523 047170 004737 052514 JSR PC,@#COMHD ;WRITE CHECK HEADER AND DATA
7524 ;SAME AS READ HEADER AND DATA
7525
7526 047174 000207 RTS PC ;RETURN TO WRITE CHECK TEST
7527
```

```

7528          .SBTTL  CRC GENERATION ROUTINE
7529
7530          ;THIS IS A SUBROUTINE TO CALCULATE CRC FOR THE FOUR
7531          ;HEADER WORDS AND STORE THEM IN 'WCRC' AND 'GCRC'
7532          ;R1 - REGISTER FOR CRC, INCREMENTED CRC VALUE IS HERE
7533          ;R2 - THIS HAS BIT POSITION 2 VALUE C
7534          ;R3 - THIS HAS BIT POSITION 16 I.E. OUTPUT BIT VALUE B
7535          ;R4 - THIS HAS BIT POSITION 15 VALUE E
7536          ;$TMP0 - NUMBER OF WORDS
7537          ;$TMP2 - NUMBER OF BITS PER WORD 16
7538          ;$TMP3 - TEMPORARY REG.
7539          ;$TMP4 - TEMPORARY REG TO TRANSFER CARRY
7540          ;$TMP5 - THIS HAS DATA BIT VALUE D
7541
7542          ;FETCH DATA BIT D
7543          ;B - D XOR 16
7544          ;C - B XOR 2
7545          ;E = B XOR 15
7546          ;ROTATE RIGHT ONE POSITION
7547          ;B GOES TO POSITION 1
7548          ;C GOES TO POSITION 3
7549          ;E GOES TO POSITION 16
7550          ;REPEAT 64 TIMES
7551
7552          ;CALL JSR R5,@#CRC
7553          ;X ;FIRST LOCATION AT
7554          ;Y ;PUT CRC IN WCRC FOR READ GCRC FOR WRITE
7555
7556          CRC:
7557          047176 010046      MOV R0,-(SP)      ;;PUSH R0 ON STACK
7558          047200 012500      MOV (R5)+,R0     ;GET POINTER TO CYL NO.
7559          047202 010146      MOV R1,-(SP)     ;;PUSH R1 ON STACK
7560          047204 010246      MOV R2,-(SP)     ;;PUSH R2 ON STACK
7561          047206 010346      MOV R3,-(SP)     ;;PUSH R3 ON STACK
7562          047210 010446      MOV R4,-(SP)     ;;PUSH R4 ON STACK
7563          047212 005001      CLR R1           ;CLEAR WORKING LOCATION
7564          047214 005037 001210  CLR @#$TMP5
7565          047220 012737 000004 001176  MOV #4,@#$TMP0   ;WORD COUNT
7566          047226 012037 001204 16$:  MOV (R0)+,@#$TMP3 ;TEMPORARY WORD STORAGE
7567          047232 012737 000020 001202  MOV #16,$TMP2    ;BIT COUNT
7568          047240 013737 001204 001206  MOV @#$TMP3,@#$TMP4 ;TEMPORARY WORD STORAGE
7569          047246 006037 001204 15$:  ROR @#$TMP3      ;GET LSB INTO 'C'
7570          047252 006037 001210      ROR @#$TMP5     ;GET ABOVE 'C' INTO $TMP5
7571          047256 032701 000001      BIT #BIT0,R1   ;IS POSITION 15 HIGH
7572          047262 001403      BEQ 1$         ;BRANCH IF POSITION 16 LOW
7573          047264 012703 100000      MOV #BIT15,R3 ;GET POSITION 16
7574          047270 000401      BR 2$
7575          047272 005003 1$:  CLR R3         ;GET POSITION 16
7576          047274 063703 001210 2$:  ADD @#$TMP5,R3 ;XOR POSITION 16 WITH D
7577          ;TO GIVE B
7578          047300 032701 040000      BIT #BIT14,R1 ;IS POSITION 2 HIGH
7579          047304 001403      BEQ 3$         ;BRANCH IF POSITION 2 LOW
7580          047306 012702 100000      MOV #BIT15,R2 ;GET POSITION 2
7581          047312 000401      BR 4$
7582          047314 005002 3$:  CLR R2         ;GET POSITION 2
7583          047316 060302 4$:  ADD R3,R2     ;XOR B WITH POSITION 2

```



```
7620 .SBTTL SIMULATED DISK SETUP
7621
7622 ;THIS IS A SUBROUTINE TO SET UP THE SIMULATOR DISK FOR
7623 ;CYLINDER 0 (16 BITS PER WORD)
7624 ;TRACK 1, SECTOR 1
7625 ;KEY1 1
7626 ;KEY2 1
7627 ;CRC THROUGH THE JSR R5,@#CRC
7628 ;256 WORDS OF 177400
7629
7630 ;CALL JSR PC,@#SETDSK
7631
7632 SETDSK:
7633 047452 010046 MOV R0,-(SP) ;:PUSH R0 ON STACK
7634 047454 010146 MOV R1,-(SP) ;:PUSH R1 ON STACK
7635 047456 010246 MOV R2,-(SP) ;:PUSH R2 ON STACK
7636 047460 012700 177400 MOV #177400,R0 ;:DATA IN THE DISK
7637 047464 012701 000400 MOV #256.,R1 ;:COUNTER
7638 047470 012702 054572 MOV #DISK,R2 ;:START OF SIMULATOR DISK
7639 047474 010022 1$: MOV R0,(R2)+ ;:MOVE IN DATA
7640 047476 005301 DEC R1 ;:COUNT FOR 256
7641 047500 001375 BNE 1$ ;:BRANCH IF 256 NOT COMPLETE
7642 047502 012701 000021 MOV #17.,R1 ;:2 ECC WORDS, 1 DATA GAP
7643 ;:14 TOLERANCE GAP
7644 047506 005022 2$: CLR (R2)+ ;:CLEAR ECC,DATA GAP AND
7645 ;:TOLERANCE GAP
7646 047510 005301 DEC R1 ;:COUNT
7647 047512 001375 BNE 2$ ;:BRANCH IF NOT COMPLETE
7648
7649 ;NOW SET UP FOR DISKLESS USE
7650
7651 047514 012737 010000 052654 MOV #FMT22,@#CYL ;:CYLINDER 0 (16 BIT WORDS)
7652 047522 112737 000001 052657 MOVB #1,@#SECOTR+1 ;:TRACK=1
7653 047530 112737 000001 052656 MOVB #1,@#SECOTR ;:SECTOR=1
7654 047536 012737 000001 052660 MOV #1,@#KEY1 ;:KEY1=1
7655 047544 012737 000001 052662 MOV #1,@#KEY2 ;:KEY2=1
7656 047552 013737 000400 052734 MOV 256.,@#DAWORD ;:NO. OF DATA WORDS
7657 047560 004537 047176 JSR R5,@#CRC ;:GO TO CALCULATE CRC
7658 047564 052654 CYL ;:FIRST CRC WORD
7659 047566 054554 WCRC ;:PUT CALCULATED CRC
7660 047570 012602 MOV (SP)+,R2 ;:POP STACK INTO R2
7661 047572 012601 MOV (SP)+,R1 ;:POP STACK INTO R1
7662 047574 012600 MOV (SP)+,R0 ;:POP STACK INTO R0
7663 047576 000207 RTS PC
7664
```

```

7665          .SBTTL CHECK HCE ROUTINE
7666
7667          ;THIS IS A SUBROUTINE TO CHECK HEADER COMPARE ERROR
7668          ;(BIT #7) AND CRC ERROR (BIT #8)
7669
7670          ;CALL JSR RO,@#HCCRCE
7671
7672          ;      COM      ;COMMAND-READ HEADER AND DATA
7673          ;              ;      -WRITE DATA
7674          ;      C      ;CYLINDER
7675          ;      S      ;SECTOR
7676          ;      T      ;TRACK
7677          ;      -N.    ;WORD COUNT
7678          ;      B      ;RHBA BUFFER START
7679          ;      X      ;1-WRITE DATA 0=READ
7680          ;      H      ;H=1 HEADER CHECK, H=0 CRC CHECK
7681
7682 047600 010037 015132 HCCRCE: MOV    RO,@#PCJSR      ;SAVE PC OF JSR+4
7683 047604 162737 000004 015132 SUB    #4,@#PCJSR      ;GET PC OF JSR
7684 047612 004737 045770 JSR    PC,@#CLDISK    ;INIT AND SETUP GENERAL REG.
7685 047616 004737 046024 JSR    PC,@#CHECKT    ;CHECK DVA,RDY,DPR,DRY - 1 AND OTHERS DON'T
7686 047622 104401 005123 TYPE   ,CPHALT        ;CANNOT CONTINUE TESTING IF ANY OF THE
7687 047626 000000 HALT                          ;STOP THE TEST
7688 047630 011037 001210 MOV    (RO),@#STMP5    ;SAVE COMMAND
7689 047634 012011 MOV    (RO)+,@R1       ;COMMAND
7690 047636 012077 145126 MOV    (RO)+,@RHCA    ;CYLINDER
7691 047642 112046 MOV    (RO)+,-(SP)    ;SECTOR
7692 047644 105720 TSTB  (RO)+           ;UP DATE RO
7693 047646 112066 000001 MOV    (RO)+,1(SP)    ;TRACK
7694 047652 105720 TSTB  (RO)+           ;UPDATE RO
7695 047654 012677 145102 MOV    (SP)+,@RHDST   ;TRACK SECTOR
7696 047660 012077 145064 MOV    (RO)+,@RHWC    ;NO. OF DATA WORDS +4 HEADER
7697                                ;IF A READ HEADER AND DATA
7698 047664 012077 145062 MOV    (RO)+,@RHBA    ;STARTING ADDRESS OF BUFFER
7699 047670 012037 052664 MOV    (RO)+,@#X      ;X=0 READ HEADER AND DATA
7700                                ;X=1 WRITE DATA
7701 047674 012777 014000 145064 MOV    #FMT22!ECI,@RHOF ;16 BITS PER WORD
7702                                ;ECC CORRECTION INHIBIT
7703 047702 005037 015124 CLR    @#ERFLG$       ;CLEAR ERROR FLAG
7704 047706 004737 052514 JSR    PC,@#COMHD     ;COMMAND
7705
7706          ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
7707          ;FROM THE 'COMHD' ROUTINE THAT MEANS SECTOR GAP,
7708          ;FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
7709          ;SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
7710          ;DETECTED
7711          ;HEADER AND DATA ARE TO BE CHECKED.
7712
7713 047712 004737 045470 JSR    PC,@#PUTREG    ;SAVE REGISTERS
7714 047716 005737 015124 TS*   @#ERFLG$       ;ANY ERRORS ALREADY THERE
7715 047722 001034 BNE   10$           ;BRANCH IF YES
7716 047724 005737 052664 TS*   @#X            ;IS THIS A READ
7717 047730 001015 BNE   3$           ;IF A WRITE DATA BRANCH
7718
7719          ;NOW THE READ BUFFER WILL BE CHECKED
7720          ;HEADER SHOULD BE COMPLETELY READ AS WRITTEN

```

```
7721 ;NO DATA WORDS SHOULD BE READ
7722 ;REINTO BUFFER HAS BEEN FILLED WITH 0
7723 ;WRFROM BUFFER HAS BEEN FILLED WITH EXPECTED DATA
7724
7725 047732 004037 046664 JSR RO,@#COMPAR ;CHECK
7726 047736 015220 WRFROM ;GOOD DATA
7727 047740 016264 REINTO ;TEST BUFFER
7728 047742 000400 256. ;4 HEADER 252 DATA
7729 047744 047752 1$ ;RETURN POINT FOR ERROR HEADER
7730 047746 047756 2$ ;RETURN POINT FOR ERROR DATA
7731 047750 050014 10$ ;RETURN FOR GOOD COMPARISON
7732 047752 104004 1$: ERROR 4 ;READ NEXT ERROR 5
7733 047754 000207 RTS PC ;RETURN TO COMPARISON SUBROUTINE
7734 047756 104005 2$: ERROR 5 ;WORD NO 1 THRU 4 ARE
7735 ;HEADER WORDS AND HENCE
7736 ;SHOULD BE READ AS WRITTEN ON
7737 ;DISK, WORD NOS. 5 ONWARDS
7738 ;SHOULD NOT BE READ AND HENCE
7739 ;READ INTO BUFFER
7740 ;SHOULD BE UNCHANGED
7741 047760 000207 RTS PC ;RETURN TO COMPARISON
7742
7743 047762 000414 BR 10$ ;JUMP OUT
7744
7745 ;NOW THE DISK WILL BE CHECKED
7746 ;NO DATA SHOULD BE WRITTEN
7747 ;REINTO BUFFER HAS BEEN FILLED WITH EXPECTED DATA
7748 ;DISK HAS BEEN FILLED WITH 177400
7749 ;WRFROM HAS BEEN FILLED WITH 125252
7750
7751 047764 004037 046664 3$: JSR RO,@#COMPAR ;CHECK
7752 047770 016264 REINTO ;GOOD DATA BUFFER
7753 047772 054572 DISK ;TEST BUFFER
7754 047774 000400 256.
7755 047776 050004 4$ ;RETURN POINT FOR ERROR HEADER
7756 050000 050010 5$ ;RETURN POINT FOR ERROR DATA
7757 050002 050014 10$ ;RETURN POINT FOR GOOD COMPARISON
7758 050004 104004 4$: ERROR 4 ;READ NEXT ERROR 5
7759 050006 000207 RTS PC ;RETURN TO COMPARISON SUBROUTINE
7760 050010 104005 5$: ERROR 5 ;WORD NO ARE ALL DATA
7761 ;WORDS THE SHOULD NOT
7762 ;HAVE BEEN CHANGED BY THE
7763 ;WRITE COMMAND
7764 050012 000207 RTS PC ;RETURN TO COMPARISON SUBROUTINE
7765 050014 005720 10$: TST (R0)+ ;IS THIS A HCRC ON HCE CHECK?
7766 050016 001442 BEQ 6$ ;BRANCH IF HCRC
7767 050020 022737 000072 001210 CMP #72,@#STMP5 ;IS THIS A READ COMMAND
7768 050026 001417 BEQ 11$ ;BRANCH IF YES
7769 050030 017737 144724 001126 MOV @RHER1,@#SBDDAT ;TEST DATA
7770 050036 022737 000200 001126 CMP #HCE,@#SBDDAT ;ONLY HEADER COMPARE BIT?
7771 ;SHOULD BE SET
7772 050044 001470 BEQ 7$ ;BRANCH IF GOOD
7773 050046 013737 014760 045534 MOV @RHER1,@#REGADR ;REGISTER ADDRESS RHER1
7774 050054 012737 000200 001124 MOV #HCE,@#SGDDAT ;GOOD DATA
7775 050062 104027 ERROR 27 ;AFTER AN ERROR ON THE
7776 ;HEADER ONLY HCE SHOULD
```

```
7777 050064 000460          BR      7$          ;BE SET
7778 050066                11$:    MOV      @RHER1,@#SBDDAT ;TEST DATA
7779 050066 017737 144666 001126    CMP      #DCK!HCE,@#SBDDAT ;ONLY HEADER COMPARE BIT?
7780 050074 022737 100200 001126                ;SHOULD BE SET
7781                ;DCK IS SET BECAUSE ECC IS NOT READ
7782                ;BRANCH IF GOOD
7783 050102 001451          BEQ      7$          ;REGISTER ADDRESS RHER1
7784 050104 013737 014760 045534    MOV      @RHER1,@#REGADR ;GOOD DATA
7785 050112 012737 100200 001124    MOV      #DCK!HCE,@#SGDDAT ;AFTER AN ERROR ON THE
7786 050120 104027          ERROR    27      ;HEADER ONLY HCE SHOULD
7787                ;BE SET
7788 050122 000441          BR      7$          ;IS THIS A READ COMMAND?
7789 050124 022737 000072 001210 6$:    CMP      #72,@#STMP5    ;BRANCH IF A READ
7790 050132 001417          BEQ      12$         ;TEST DATA
7791 050134 017737 144620 001126    MOV      @RHER1,@#SBDDAT ;ONLY CRC ERROR SHOULD BE THERE
7792 050142 022737 000400 001126    CMP      #HCRC,@#SBDDAT ;REG. ADDR = RHER1
7793 050150 001426          BEQ      7$          ;GOOD DATA
7794 050152 013737 014760 045534    MOV      @RHER1,@#REGADR ;AFTER A CRC ERROR ONLY CRC
7795 050160 012737 000400 001124    MOV      #HCRC,@#SGDDAT ;SHOULD BE SET
7796 050166 104027          ERROR    27      ;BRANCH OUT
7797                ;TEST DATA
7798 050170 000416          BR      7$          ;HCRC AND DCK SHOULD BE SET
7799 050172 017737 144562 001126 12$:    MOV      @RHER1,@#SBDDAT ;DCK IS SET BECAUSE ECC IS NOT READ
7800                ;BRANCH IF GOOD
7801 050200 022737 100400 001126    CMP      #DCK!HCRC,@#SBDDAT ;GOOD DATA
7802                ;FAILING REGISTER RHER1
7803 050206 001407          BEQ      7$          ;AFTER A CRC ERROR ON A READ
7804 050210 012737 100400 001124    MOV      #DCK!HCRC,@#SGDDAT ;DCK AND HCRC SHOULD BE SET
7805 050216 013737 014760 045534    MOV      @RHER1,@#REGADR ;DCK IS SET BECAUSE ECC IS NOT READ
7806 050224 104027          ERROR    27      ;RETURN TO MAIN TEST
7807                ;
7808                ;
7809 050226 000200          7$:    RTS      R0
```


7841
7842
7843
7844
7845
7846
7847
7848
7849
7850
7851
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

.SBTTL JAM CURRENT CYLINDER ROUTINE

;*THIS SUBROUTINE WILL CHANGE THE CURRENT CYLINDER REGISTER
;*THIS IS DONE BY GIVING A SEEK COMMAND THEN AN INIT
;*WHICH WILL LOAD THE CURRENT CYLINDER WITH THE DESIRED CYLINDER VALUE

;*CALL IS:
;* JSR RO,@MAKECYL ;DESIRED VALUE OF CURRENT CYLINDER?
;* XC

MAKECYL:

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

1\$:

MOV (SP)+,R5 ;:POP STACK INTO R5
RTS RO

050342
050342 010546
050344 010037 015132
050350 162737 000004 015132
050356 012005
050360 010577 144404
050364 005077 144372
050370 013777 015200 144360
050376 012777 000001 144372
050404 052777 000001 144344
050412 000240
050414 000240
050416 000240
050420 000240
050422 004737 045770
050426 017737 144362 001126
050434 020537 001126
050440 001406
050442 010537 001124
050446 013737 015014 045534
050454 104030

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
7908
7909
7910
7911
7912
7913
7914
7915
7916
7917
7918
7919
7920
7921
7922
7923
7924
7925
7926
7927
7928
7929
7930
7931
7932
7933

.SBTTL ECC GENERATION AND COMPARISON ROUTINE

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

100000	PIE1	=100000
040000	PIE2	=40000
020000	PIE3	=20000
010000	PIE4	=10000
004000	PIE5	=4000
002000	PIE6	=2000
001000	PIE7	=1000
000400	PIE8	=400
000200	PIE9	=200
000100	PIE10	=100
000040	PIE11	=40
000020	PIE12	=20
000010	PIE13	=10
000004	PIE14	=4
000002	PIE15	=2
000001	PIE16	=1
100000	PIE17	=100000
040000	PIE18	=40000
020000	PIE19	=20000
010000	PIE20	=10000
004000	PIE21	=4000
002000	PIE22	=2000
001000	PIE23	=1000
000400	PIE24	=400
000200	PIE25	=200
000100	PIE26	=100
000040	PIE27	=40
000020	PIE28	=20
000010	PIE29	=10
000004	PIE30	=4
000002	PIE31	=2
000001	PIE32	=1

050462	000000	ECDATA: 0	:DATA BIT FOR ECC :IF ALL ONES THEN CURRENT BIT IS A ONE :IF ZERO THEN CURRENT BIT IS A ZERO
050464	000000	GECC1: 0	:LOW ORDER ECC WORD TO BE GENERATED HERE :=R1
050466	000000	GECC2: 0	:HIGH ORDER ECC WORD TO BE GENERATED HERE :=R2
050470	000000	TSECCG: 0	:IF =177777 GENERATE AND TEST ECC FOR THIS BIT :IF =0 DO NOT GENERATE AND TEST ECC FOR THIS BIT
050472	113713	NCODE: 38859.	:N-CODE WORD

```
7934 050474 000000      NCOUNT: 0          ;TEMPORARY N CODE
7935 050476 000000      POSITI: 0           ;POSITION REGISTER
7936 050500 010041      HARDER: 4129.       ;HARD ERROR COUNT
7937                                     ;TRUE COUNT IS 4128 BUT AS COMPARES ARE
7938                                     ;DONE ONE STAGE LATER SO 4129
7939 050502 000000      DATENV: 0          ;DATA ENVELOPE FOR TYPE OUT
7940                                     ;MAX FOR WRITE IS 4096
7941                                     ;MAX FOR READ IS 4128
7942 050504 000000      ZCODE: 0          ;LEADING ZEROS ENVELOPE FOR TYPE OUT
7943                                     ;THIS IS SHUT OFF WHEN POSITION COUNTER
7944                                     ;IN ENABLED
7945                                     ;MAX COUNT IS 38859
7946
7947
7948
7949 050506 000000      HADTMP: 0          ;TEMPORARY HARD ERROR COUNT
7950 050510 000000      P3: 0
7951 050512 000000      P12: 0
7952 050514 000000      P22: 0
7953 050516 000000      P24: 0
7954
7955
7956
7957
7958
7959 050520      ECTEST:
7960 050520 010046      MOV R0,-(SP)        ;;PUSH R0 ON STACK
7961 050522 010146      MOV R1,-(SP)        ;;PUSH R1 ON STACK
7962 050524 010246      MOV R2,-(SP)        ;;PUSH R2 ON STACK
7963 050526 010346      MOV R3,-(SP)        ;;PUSH R3 ON STACK
7964 050530 010446      MOV R4,-(SP)        ;;PUSH R4 ON STACK
7965 050532 010546      MOV R5,-(SP)        ;;PUSH R5 ON STACK
7966 050534 013701 050464  MOV @#GECC1,R1      ;ECC1 WORD
7967 050540 013702 050466  MOV @#GECC2,R2      ;ECC2 WORD
7968 050544 005737 050462  TST @#ECCDATA      ;IS CURRENT BIT A ONE
7969 050550 001406      BEQ 2$             ;BRANCH IF CURRENT DATA D=0
7970
7971                                     ;IF CARRY IS NOT ZERO THEN D=1
7972                                     ;INVERT X32 TO GIVE R0
7973
7974 050552 010103      1$: MOV R1,R3
7975 050554 052703 177776  BIS #^CPIE32,R3
7976 050560 005103      COM R3
7977 050562 010300      MOV R3,R0
7978 050564 000404      BR 3$
7979
7980                                     ;IF CARRY IS ZERO THEN D=0
7981                                     ;X32 BECOMES R0
7982 050566 010103      2$: MOV R1,R3
7983 050570 042703 177776  BIC #^CPIE32,R3
7984 050574 010300      MOV R3,R0
7985
7986 050576 000241      3$: CLC
7987 050600 006000      ROR R0
7988 050602 006000      ROR R0
7989 050604 005700      TST R0
```

```
7990 050606 001462      BEQ    10$          ;BRANCH IF R0=0
7991                      ;INVERT X2
7992
7993 050610 010203      MOV    R2,R3
7994 050612 052703 137777  BIS    #^CPIE2,R3
7995 050616 005103      COM    R3
7996 050620 010337 050510  MOV    R3,@#P3
7997 050624 006237 050510  ASR    @#P3
7998
7999                      ;INVERT X11
8000
8001
8002 050630 010203      MOV    R2,R3
8003 050632 052703 177737  BIS    #^CPIE11,R3
8004 050636 005103      COM    R3
8005 050640 010337 050512  MOV    R3,@#P12
8006 050644 006237 050512  ASR    @#P12
8007
8008                      ;INVERT X21
8009
8010 050650 010103      MOV    R1,R3
8011 050652 052703 173777  BIS    #^CPIE21,R3
8012 050656 005103      COM    R3
8013 050660 010337 050514  MOV    R3,@#P22
8014 050664 006237 050514  ASR    @#P22
8015
8016                      ;INVERT X23
8017
8018 050670 010103      MOV    R1,R3
8019 050672 052703 176777  BIS    #^CPIE23,R3
8020 050676 005103      COM    R3
8021 050700 010337 050516  MOV    R3,@#P24
8022 050704 006237 050516  ASR    @#P24
8023
8024                      ;NOW THAT R0 FOR POSITION 1
8025                      ;      P3 FOR POSITION 3
8026                      ;      P12 FOR POSITION 12
8027                      ;      P22 FOR POSITION 22
8028                      ;      P24 FOR POSITION 24
8029                      ;ARE KNOWN THE ROTATE WILL BE DONE AND
8030                      ;THESE BITS JAMED IN
8031
8032 050710 006002      ROR    R2
8033 050712 006001      ROR    R1
8034 050714 053700 050510  BIS    @#P3,R0
8035 050720 053700 050512  BIS    @#P12,R0
8036 050724 042702 120020  BIC    #PIE1!PIE3!PIE12,R2
8037 050730 050002      BIS    R0,R2
8038
8039 050732 005000      CLR    R0
8040 050734 053700 050514  BIS    @#P22,R0
8041 050740 053700 050516  BIS    @#P24,R0
8042 050744 042701 002400  BIC    #PIE22!PIE24,R1
8043 050750 050001      BIS    R0,R1
8044 050752 000404      BR     12$
8045
```

```

8046                                     ;THE PROGRAM COMES HERE IF R0=0
8047                                     ;SO AFTER ROTATE R0 GETS PUT INTO POSITION 1
8048
8049 050754 006002          10$:  ROR    R2
8050 050756 006001          ROR    R1
8051 050760 042702 100000    BIC    #PIE1,R2
8052 050764 010137 050464    12$:  MOV    R1,@#GECC1      ;SAVE ECC1
8053 050770 010237 050466    MOV    R2,@#GECC2      ;SAVE ECC2
8054 050774 005737 050470    TST    @#TSECCG        ;IS HARDWARE TO BE CHECKED
8055                                     ;IF =1777777 TEST HARDWARE
8056                                     ;IF = 0 DO NOT TEST HARDWARE
8057 051000 001432          BEQ    14$      ;BRANCH IF HARDWARE NOT TO BE CHECKED
8058
8059
8060                                     ;*CHECK HARDWARE
8061 051002 032777 000400 130130 BIT    #SW8,@SWR        ;IS SWITCH 8 SET
8062 051010 001005          BNE    15$      ;BRANCH IF SW8 IS SET
8063 051012 032777 000100 130120 BIT    #SW6,@SWR        ;IS SWITCH 6 SET
8064 051020 001401          BEQ    15$      ;BRANCH IF SW6 IS NOT SET
8065 051022 000421          BR     14$      ;IF SWITCH 8 IS NOT SET AND
8066                                     ;SWITCH 6 IS SET THEN
8067                                     ;DO NOT DO COMPARES
8068 051024 010146          15$:  MOV    R1,-(SP)      ;GOOD PATTERN REGISTER
8069 051026 042716 174000    BIC    #174000,(SP)    ;GET ONLY PATTERN BITS
8070 051032 022677 143752    CMP    (SP)+,@RHEC2    ;COMPARE PATTERN REGISTER
8071 051036 001404          BEQ    13$      ;BRANCH IF GOOD
8072                                     ;TO SAVE TIME
8073 051040 004737 045470    JSR    PC,@#PUTREG     ;SAVE REGISTERS
8074 051044 104035          ERROR 35      ;PATTERN REGISTER IN 11 BITS IN ERROR
8075 051046 000407          BR     14$      ;BRANCH OUT
8076 051050 023777 050476 143730 13$:  CMP    @#POSIT1,@RHEC1 ;COMPARE POSITION REGISTER
8077 051056 001403          BEQ    14$      ;BRANCH IF GOOD
8078                                     ;TO SAVE TIME
8079 051060 004737 045470    JSR    PC,@#PUTREG     ;SAVE REGISTERS
8080 051064 104035          ERROR 35      ;POSITION REGISTER IN ERROR
8081                                     ;'DATA ENVELOP' GIVES NUMBER OF CLOCK
8082                                     ;PULSES FROM BEGINING OF COMMAND
8083                                     ;THAT IS THE CLOCKS IN THE R/W DATA FIELD ENVELOPE
8084                                     ;
8085                                     ;IN A WRITE THERE ARE 10000 OCTAL CLOCKS
8086                                     ;IN A READ THERE ARE 10040 OCTAL CLOCKS
8087                                     ;
8088                                     ;
8089                                     ;'N-CODE ZEROS' GIVE THE NUMBER OF CLOCKS
8090                                     ;GIVEN FOR THE LEADING ZEROS FIELD
8091                                     ;MAX COUNT IS 113713 OCTAL
8092                                     ;
8093                                     ;'GOOD POSITION' GIVES NUMBER OF CLOCKS
8094                                     ;GIVEN AFTER LEADING ZEROS WHICH IS FOR THE DATA
8095                                     ;FIELD
8096                                     ;MAX COUNT IS 10040 OR 10041 OCTAL
8097
8098
8099 051066          14$:  MOV    (SP)+,R5      ;;POP STACK INTO R5
8100 051066 012605          MOV    (SP)+,R4      ;;POP STACK INTO R4
8101 051070 012604

```



```
8158  
8159 051244 052711 000002      5S:   BIS   #MCLK,@R1      ;SET CLOCK  
8160 051250 042711 000002      BIC   #MCLK,@R1      ;CLEAR CLOCK  
8161  
8162 051254                      4S:  
8163 051254 012601      MOV   (SP)+,R1      ;;POP STACK INTO R1  
8164 051256 000200      RTS   R0
```

.SBTTL SOFTWARE DISK DATA ECC GEN. ROUTINE

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

```
8165
8166
8167
8168
8169
8170
8171 051260          FILLEC:
8172 051260 010046   MOV     R0,-(SP)      ;;PUSH R0 ON STACK
8173 051262 010146   MOV     R1,-(SP)      ;;PUSH R1 ON STACK
8174 051264 010246   MOV     R2,-(SP)      ;;PUSH R2 ON STACK
8175 051266 010346   MOV     R3,-(SP)      ;;PUSH R3 ON STACK
8176 051270 010446   MOV     R4,-(SP)      ;;PUSH R4 ON STACK
8177 051272 010546   MOV     R5,-(SP)      ;;PUSH R5 ON STACK
8178 051274 005037 050476 CLR     @#POSITI    ;CLEAR POSITION
8179 051300 005037 050464 CLR     @#GECC1     ;CLEAR GECC1
8180 051304 005037 050466 CLR     @#GECC2     ;CLEAR
8181 051310 012701 054572 MOV     #DISK,R1    ;POINTER TO DATA FOR ECC GENERATION
8182 051314 012702 000400 MOV     #256.,R2    ;COUNTER FOR NUMBER OF DATA WORDS
8183 051320 012703 000020 9$: MOV     #16.,R3    ;COUNTER FOR NUMBER OF BITS PER WORD
8184 051324 012104   MOV     (R1)+,R4    ;DATA IN R4
8185 051326 006004   10$: ROR     R4      ;GET ONE DATA BIT IN CARRY
8186 051330 103004   BCC     11$        ;BRANCH IF DATA BIT IS ZERO
8187 051332 012737 177777 050462 MOV     #-1,@#ECCDATA ;ECC DATA BIT IS A ONE
8188 051340 000402   BR     12$        ;BRANCH TO GENERATE ECC
8189 051342 005037 050462 11$: CLR     @#ECCDATA  ;ECC DATA BIT IS A ZERO
8190 051346 004737 050520 12$: JSR     PC,@#ECTEST ;GO TO GENERATE ECC
8191 051352 005303   DEC     R3        ;DECREMENT BIT COUNT
8192 051354 001364   BNE    10$        ;BRANCH IF 16 BITS NOT DONE
8193 051356 005302   DEC     R2        ;DECREMENT WORD COUNT
8194 051360 001357   BNE    9$         ;BRANCH IF 256 WORDS NOT DONE
8195 051362 013737 050464 055572 MOV     @#GECC1,@#DISK+<256.*2>;INSERT ECC1 ON DISK
8196 051370 013737 050466 055574 MOV     @#GECC2,@#DISK+<257.*2>;INSERT ECC2 ON DISK
8197 051376 012605   MOV     (SP)+,R5    ;;POP STACK INTO R5
8198 051400 012604   MOV     (SP)+,R4    ;;POP STACK INTO R4
8199 051402 012603   MOV     (SP)+,R3    ;;POP STACK INTO R3
8200 051404 012602   MOV     (SP)+,R2    ;;POP STACK INTO R2
8201 051406 012601   MOV     (SP)+,R1    ;;POP STACK INTO R1
8202 051410 012600   MOV     (SP)+,R0    ;;POP STACK INTO R0
8203 051412 000207   RTS     PC
```



```

8204
8205          .SBTTL  RH BASE ADDRESS CHANGE ROUTINE
8206
8207          :*      THIS ROUTINE WILL ALLOW THE CHANGE OF THE BASE
8208          :*      ADDRESS FROM 176700 TO ANY TYPED VALUE
8209
8210          BASECH:
8211 051414 104401 051422          TYPE      ,65$          ;;TYPE ASCIZ STRING
8212 051420 000425          BR      64$          ;;GET OVER THE ASCIZ
8213 051474 013746 014756          MOV     @#RHCS1,-(SP)  ;;GET READY TO TYPE OLD BASE
8214 051500 104402          TYPOC
8215 051502 104401 051510          TYPE     ,67$          ;;TYPE ASCIZ STRING
8216 051506 000425          BR      66$          ;;GET OVER THE ASCIZ
8217 051562 004737 060352          JSR     PC,@#STKINT  ;;INITIALIZE THE TTY KEYBOARD
8218 051566 104412          RDOCT
8219 051570 012700 014746          MOV     #RHDB,R0      ;;GET STARTING ADDRESS OF REGISTERS
8220 051574 012701 000026          MOV     #22,R1       ;;NUMBER OF REGISTERS
8221 051600 012737 052400 000004          MOV     #ADTIMO,@#4  ;;SET UP TO TEST THIS ADDRESS
8222 051606 021637 014756          CMP     @SP,@#RHCS1  ;;NEW CSR?
8223 051612 001431          BEQ     1$          ;;NO, THE OLD ONE WAS RETYPED
8224 051614 005776 000000          TST     @0(SP)       ;;ACCESS THE NEW ADDRESS
8225 051620 163716 014756          SUB     @#RHCS1,@SP  ;;GET THE ADDRESS OFFSET
8226 051624 061620          2$: ADD     @SP,(R0)+    ;;AND PLUG IT IN
8227 051626 005301          DEC     R1           ;;ONE LESS ADDRESS TO CHANGE
8228 051630 001375          BNE     2$          ;;BUT DO SOME MORE
8229 051632 104401 051640          TYPE     ,69$          ;;TYPE ASCIZ STRING
8230 051636 000417          BR      68$          ;;GET OVER THE ASCIZ
8231 051676 013746 014744          1$: MOV     @#RPVEC,-(SP) ;;GET READY TO TYPE OLD VECTOR ADDRESS
8232 051702 104402          TYPOC
8233 051704 104401 051712          TYPE     ,71$          ;;TYPE ASCIZ STRING
8234 051710 000437          BR      70$          ;;GET OVER THE ASCIZ
8235 052010 104412          RDOCT
8236 052012 012637 014744          MOV     (SP)+,@#RPVEC ;;SETUP VECTOR ADDRESS
8237 052016 104401 052024          TYPE     ,73$          ;;TYPE ASCIZ STRING
8238 052022 000416          BR      72$          ;;GET OVER THE ASCIZ
8239 052060 013746 014756          MOV     @#RHCS1,-(SP)
8240 052064 104402          TYPOC
8241 052066 104401 052074          TYPE     ,75$          ;;TYPE ASCIZ STRING
8242 052072 000416          BR      74$          ;;GET OVER THE ASCIZ
8243 052130 013746 014744          MOV     @#RPVEC,-(SP)
8244 052134 104402          TYPOC
8245 052136 104401 052144          TYPE     ,77$          ;;TYPE ASCIZ STRING
8246 052142 000417          BR      76$          ;;GET OVER THE ASCIZ
8247 052202 104401 052210          TYPE     ,79$          ;;TYPE ASCIZ STRING
8248 052206 000402          BR      78$          ;;GET OVER THE ASCIZ
8249 052214 104401 052222          TYPE     ,81$          ;;TYPE ASCIZ STRING
8250 052220 000424          BR      80$          ;;GET OVER THE ASCIZ
8251 052272 104401 052300          TYPE     ,83$          ;;TYPE ASCIZ STRING
8252 052276 000426          BR      82$          ;;GET OVER THE ASCIZ
8253 052354 012746 000200          MOV     #RA,-(SP)
8254 052360 104402          TYPOC
8255 052362 104401 052370          TYPE     ,85$          ;;TYPE ASCIZ STRING
8256 052366 000402          BR      84$          ;;GET OVER THE ASCIZ
8257 052374 000137 017356          JMP     @#BEGIN      ;;ALL DONE, NOW START OVER!
8258 052400 022626          ADTIMO: CMP     (SP)+,(SP)+
8259 052402 104401 052410          TYPE     ,65$          ;;TYPE ASCIZ STRING

```

8260 052406 000424
8261 052460 000137 051414
8262
8263
8264
8265
8266
8267
8268
8269 052464 000000
8270 052466 004737 045770
8271 052472 013712 052464
8272 052476 005714
8273 052500 032712 010000
8274 052504 001401
8275 052506 000773
8276 052510 000772

BR 64\$;;GET OVER THE ASCIZ
JMP @#BASECH

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

8277
8278
8279
8280
8281
8282
8283
8284
8285
8286
8287
8288
8289
8290
8291
8292
8293
8294
8295
8296
8297
8298
8299
8300
8301
8302
8303
8304
8305
8306
8307
8308
8309
8310
8311
8312
8313
8314
8315
8316
8317
8318
8319
8320
8321
8322
8323
8324
8325
8326
8327
8328
8329

```
.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
*:*
```

```
8330
8331
8332
8333
8334
8335
8336      ;*WRITE DATA COMMAND
8337      ;*OR READ COMMAND, I.E DATA ONLY, OR HEADER AND DATA
8338
8339
8340
8341
8342
8343      ;*THIS SUBROUTINE IS THE FIRST IN A SERIES OF NESTED SUBROUTINES
8344
8345      ;*IT ISSUES DIAGNOSTIC MODE, AND EXTRA DIAGNOSTIC INDEX, AND THE
8346      ;*'GO' BIT
8347
8348      ;*IT THEN CALLS THREE OTHER SUBROUTINES, WHICH IN TURN CALL
8349      ;*OTHER SUBROUTINES. THE SUBROUTINES CALLED HERE ARE:
8350
8351      ;*      SEARCH      ;ISSUES SECTOR CLOCKS TO SET SECTOR FOUND FLOP
8352      ;*      RDHEAD     ;READS THE SECTOR HEADER
8353      ;*      WRDATA     ;WRITES THE SECTOR DATA (WRITE OPERATION)
8354      ;*      REDATA     ;READS THE SECTOR DATA (READ OPERATION)
8355
8356
8357
8358
8359
8360 052512 000000      RUNCTR: .WORD 0
8361 052514 011637 015132 COMHD: MOV (SP),@#PCJSR ;SAVE PC OF JSR + 4
8362 052520 162737 000004 015132 SUB #4,@#PCJSR ;SAVE PC OF JSR
8363 052526 010046      MOV R0,-(SP) ;:PUSH R0 ON STACK
8364 052530 010146      MOV R1,-(SP) ;:PUSH R1 ON STACK
8365 052532 010246      MOV R2,-(SP) ;:PUSH R2 ON STACK
8366 052534 010346      MOV R3,-(SP) ;:PUSH R3 ON STACK
8367 052536 010446      MOV R4,-(SP) ;:PUSH R4 ON STACK
8368 052540 010546      MOV R5,-(SP) ;:PUSH R5 ON STACK
8369
8370 052542 012777 000001 142226      MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE
8371 052550 052777 000004 142220      BIS #MINX,@RHMR ;SET DIAGNOSTIC INDEX
8372 052556 042777 000004 142212      BIC #MINX,@RHMR ;CLEAR DIAGNOSTIC INDEX
8373 052564 052777 000001 142164      BIS #GO,@RHCS1 ;ISSUE 'GO' BIT & STALL 'TILL 'RUN'
8374 ;FUNCTION CODE WAS ISSUED BY THE TEST
8375 052572 012737 000113 052512 RUNWAT: MOV #75,@#RUNCTR ;LOAD STALL COUNT = APPROX. 450US FOR 11/50 CPU
8376 052600 005337 052512 1$: DEC @#RUNCTR ;COUNT DOWN ONE
8377 052604 001375      BNE 1$ ;CONTINUE UNTIL = 0
8378
8379 052606 013746 052656      MOV SECTR,-(SP) ;GET DESIRED SECTOR/TRACK
8380 052612 042716 177740      BIC #177740,(SP) ;MAKE ONLY SECTOR
8381 052616 012637 052626      MOV (SP)+,@#TRK ;SAVE SECTOR
8382 052622 004137 056722      JSR R1,@#SEARCH ;ISSUE SECTOR CLOCKS <----->
8383
8384 052626 000000      TRK: .WORD 0
8385 052630 012701 000240      MOV #*NOP,R1 ;GOING TO MOVE NOPS
```

```
8386 052634 010137 052666      MOV      R1,@#SSYN      ;NOP INTO SSYN
8387 052640 010137 052670      MOV      R1,@#HEDGAP   ;NOP INTO HEDGAP
8388 052644 010137 052672      MOV      R1,@#HEDSYN   ;NOP INTO HEDSYN
8389 052650 004137 052776      JSR      R1,@#RDHEAD   ;READ THE HEADER <----->
8390
8391 052654 000000      CYL      .WORD 0      ;CYLINDER ADDRESS
8392 052656 000000      SECOTR: .WORD 0      ;SECTOR/TRACK ADDRESS
8393 052660 000000      KEY1:   .WORD 0      ;KEY1 WORD
8394 052662 000000      KEY2:   .WORD 0      ;KEY2 WORD
8395 052664 000000      X:      .WORD 0      ;X=1 WRITE COMMAND
8396                                     ;X=0 READ COMMAND
8397
8398
8399                                     ;DUMMY ERROR CALL LOCATIONS FOR THE READ HEADER OPERATION
8400
8401 052666 000240      SSYN:   NOP          ;IF 'ERROR 2' INSERTED BY RDHEAD
8402                                     ;SUBROUTINE, THEN THE FIRST SYNC
8403                                     ;IS NOT DETECTED. NO BAD DATA
8404                                     ;IS GIVEN BECAUSE SYNC=144000
8405                                     ;CANNOT BE READ. WORD NUMBER
8406                                     ;IS '1' BECAUSE THIS IS THE FIRST
8407                                     ;WORD TESTED.
8408
8409 052670 000240      HEDGAP: NOP        ;IF 'ERROR 3' INSERTED BY
8410                                     ;RDHEAD SUBROUTINE, THEN THE
8411                                     ;HEADER GAP 0'S WERE NOT
8412                                     ;WRITTEN RIGHT.
8413
8414                                     ;IF 'WORD NO' CONTAINS, SAY
8415                                     ;3(8), THEN IT IS THE THIRD
8416                                     ;WORD OF A 5 WORD HEADER
8417                                     ;GAP THAT IS WRONG.
8418
8419                                     ;'BAD DATA' CONTAINS WHAT IS
8420                                     ;GOING ON THE DISK.
8421
8422 052672 000240      HEDSYN: NOP       ;IF 'ERROR 3' INSERTED BY RDHEAD
8423                                     ;SUBROUTINE, THEN THE HEADER SYNC
8424                                     ;GENERATED BY DCL IS WRONG,
8425                                     ;OR THE LAST BYTE
8426                                     ;OF THE HEADER GAP 0'S IS WRONG.
8427
8428                                     ;IN EITHER CASE WORD NO=6,
8429                                     ;RIGHT BYTE IS HEADER 0,
8430                                     ;LEFT BYTE IS SYNC.
8431
8432                                     ;'BAD DATA' HAS WHAT IS GOING
8433                                     ;ON DISK.
8434
8435
8436 052674 005737 015124      TST      @#ERFLGS     ;WERE ANY ERRORS DETECTED ?
8437 052700 001017      BNE      OUT          ;IF YES, EXIT ----->
8438
8439 052702 005737 052664      TST      @#X          ;IS IT A DATA WRITE OPERATION ?
8440 052706 001410      BEQ      DAREAD       ;NO...THEN DO A DATA READ
8441 052710 005737 052770      TST      @#NOSYNC    ;IS THIS FORCED HEADER ERROR COMMAND ?
```

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

8465
8466
8467
8468
8469
8470
8471
8472
8473 052756 014400
8474 052760 000000
8475 052762 000000
8476 052764 000000
8477 052766 000000
8478
8479
8480
8481
8482
8483
8484
8485
8486
8487
8488
8489
8490
8491
8492
8493
8494

;*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'S, 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'S FOR DATA GAP AND TOLERANCE GAP

```

8495          ;*READ DISK HEADER
8496
8497
8498
8499
8500
8501 052770 000000      NOSYNC: 0          ;FORCED HEADER ERROR = -1
8502                                     ;NORMAL = 0
8503 052772 000000      TY: 0          ;ERROR TYPE NO.
8504 052774 000000      ERWORD: 0       ;ERROR WORD NO.
8505
8506
8507
8508
8509 052776 012137 052760  RDHEAD: MOV (R1)+,@#RCYL ;STORE CYLINDER ADDRESS
8510 053002 012137 052762      MOV (R1)+,@#RSETR ;STORE SECTOR AND TRACK ADDRESS
8511 053006 012137 052764      MOV (R1)+,@#RKEY1 ;STORE KEY1
8512 053012 012137 052766      MOV (R1)+,@#RKEY2 ;STORE KEY2
8513 053016 012137 053566      MOV (R1)+,@#COMPA ;STORE COMPARE OR NOT
8514 053022 010146      MOV R1,-(SP) ;:PUSH R1 ON STACK
8515
8516 053024 013700 014776      MOV @#RHMR,R0 ;R0 CONTAINS MAINTANENCE REG.
8517 053030 012705 000002      MOV #2,R5 ;R5 IS A COUNTER FOR WORDS
8518 053034 012710 000001      MOV #DMD,@RO ;SET DIAG. MODE
8519 053040 052710 000010      BIS #MSTCK,@RO ;SET SECTOR CLOCK FOR FIRST WORD
8520 053044 052710 000002      BIS #MCLK,@RO ;SEI MAINT.CLOCK FOR FIRST WORD
8521 053050 042710 000012      BIC #MSTCK!MCLK,@RO ;RESET THEM
8522
8523 053054 000404          BR 2$ ;DON'T GIVE SECTOR CLOCK FIRST TIME
8524 053056 012710 000013 1$: MOV #MSTCK!MCLK!DMD,@RO ;SET SECTOR, CLOCK, DIAG. MODE, RESET INDEX
8525 053062 042710 000012      BIC #MSTCK!MCLK,@RO ;RESET SECTOR & MAINT.CLOCK
8526
8527 053066 012702 000007 2$: MOV #7,R2 ;LOAD BYTE COUNTER
8528 053072 052710 000002 3$: BIS #MCLK,@RO ;SET MAINT. CLOCK
8529 053076 042710 000002      BIC #MCLK,@RO ;RESET IT
8530 053102 005302          DEC R2 ;BYTE COUNTER
8531 053104 001372          BNE 3$ ;CONTINUE IF BYTE NOT COMPLETE
8532 053106 005305          DEC R5 ;WORD COUNTER
8533 053110 001362          BNE 1$ ;CONTINUE IF WORD NOT COMPLETE
8534
8535 053112 012702 000022          MOV #18.,R2 ;LOAD NO OF WORDS OF ZEROS
8536 053116 005037 053564 4$: CLR @#WORD ;DO 0'S
8537 053122 004737 053570      JSR PC,@#READ ;READ 0'S <----->
8538 053126 005302          DEC R2 ;COUNT DOWN WORDS
8539 053130 001372          BNE 4$ ;CONTINUE
8540
8541 053132 013737 052756 053564 MOV @#RSYNC,@#WORD ;SYNC. WORD
8542 053140 004737 053570      JSR PC,@#READ ;READ IT <----->
8543 053144 032710 001000      BIT #DTSY,@RO ;SYNC. BYTE DETECTED?
8544 053150 001012          BNE 5$ ;CONTINUE IF SYNC DETECTED
8545 053152 012737 000001 052774 MOV #1,@#ERWORD ;ERROR WORD NO
8546 053160 013737 052756 001124 MOV @#RSYNC,@#SGDDAT ;SYNC WORD
8547 053166 012737 104002 052666 MOV #104002,@#SSYN ;INSERT 'ERROR 2' IN SSYN
8548 053174 000571          BR 13$ ;BRANCH OUT <----->
8549
8550 053176 013737 052760 053564 5$: MOV @#RCYL,@#WORD ;SETUP CYLINDER

```



```

8607 053460 012702 000005      11$:  MOV    #5,R2
8608 053464 005037 053564      12$:  CLR    WORD
8609 053470 004737 053570          JSR    PC,@#READ ;READ HEADER GAP <----->
8610 053474 005302          DEC    R2 ;ARE 5 HEADER GAP ZEROS COMPLETE ?
8611 053476 001372          BNE    12$ ;IF NOT CONTINUE
8612 053500 013737 052756 053564    MOV    @#RSYNC,@#WORD ;SYNC WORD
8613 053506 004737 053570          JSR    PC,@#READ ;READ HEADER (DATA) SYNC)
8614 053512 005737 052770          TST    @#NOSYNC ;FORCED SYNC ERROR ?
8615 053516 001404          BEQ    16$ ;IF NOT ERROR COMMAND CONTINUE
8616 053520 032710 001000          BIT    #DTSY,@R0 ;SYNC. DETECTED
8617 053524 001415          BEQ    13$ ;IF ZERO BRANCH OUT ----->
8618 053526 000403          BR    17$ ;IF NOT ZERO BRANCH TO ERROR
8619
8620 053530 032710 001000      16$:  BIT    #DTSY, @R0 ;SYNC. DETECTED ?
8621 053534 001011          BNE    13$ ;EXIT IF YES ----->
8622 053536 012737 000006 052774 17$:  MOV    #6,@#ERWORD ;ERROR WORD NO.
8623 053544 013737 052756 001124    MOV    @#RSYNC,@#SGDDAT;SYNC WORD
8624 053552 012737 104002 052672    MOV    #104002,@#HEDSYN;MOVE 'ERROR 2'
8625 053560          13$:
8626 053560 012601          MOV    (SP)+,R1 ;:POP STACK INTO R1
8627 053562 000201          RTS    R1 ;EXIT ----->
8628
8629
8630
8631
8632
8633
8634
8635
  
```

```
8636 ;*READ ONE WORD IN 'WORD'  
8637  
8638  
8639  
8640  
8641 053564 000000 WORD: 0  
8642 053566 000000 COMPA: 0  
8643  
8644  
8645  
8646  
8647 053570 READ:  
8648 053570 010246 MOV R2,-(SP) ;:PUSH R2 ON STACK  
8649 053572 012705 000002 MOV #2,R5 ;:WORD COUNTER  
8650 053576 012710 000001 MOV #DMD,@RO ;:SET DIAG. MODE  
8651 053602 006037 053564 ROR @WORD ;:CHECKING IF THERE IS A ONE  
8652 053606 103002 BCC 1$ ;:IF NO ONE BRANCH  
8653 053610 052710 000020 BIC #MRD,@RO ;:SET BIT 4 IF DATA HAS ONE  
8654 053614 012702 000007 1$: MOV #7,R2 ;:BYTE COUNTER  
8655 053620 052710 000012 BIS #MSTCK!MCLK,@RO ;:SET CLOCK,DATA IF ANY, SECTOR  
8656 053624 005737 050470 TST @TSECCG ;:IS THIS BIT TO GENERATE AND TEST ECC ?  
8657 053630 001411 BEQ 6$ ;:BRANCH IF NO  
8658 053632 032710 000020 BIT #MRD,@RO ;:IS DATA BIT A ONE ?  
8659 053636 001404 BEQ 5$ ;:BRANCH IF DATA BIT IS 0  
8660 053640 012737 177777 050462 MOV #-1,@ECCDATA ;:ECC DATA BIT IS A ONE  
8661 053646 000402 BR 6$ ;:BRANCH  
8662 053650 005037 050462 5$: CLR @ECCDATA ;:ECC DATA BIT IS A 0  
8663 053654 012746 000001 6$: MOV #DMD,-(SP) ;:KEEP ONLY DIAG. MODE  
8664 053660 006037 053564 ROR @WORD ;:CHECKING IF THERE IS A ONE  
8665 053664 103002 BCC 2$ ;:IF NO ONE BRANCH  
8666 053666 012716 000021 MOV #MRD!DMD,(SP) ;:KEEP DATA AND DIAG. MODE  
8667 053672 012610 2$: MOV (SP)+,@RO ;:PUT IN DATA,RESET CLOCK, SECTOR  
8668 053674 005737 050470 TST @TSECCG ;:IS ECC TO BE GENERATED FOR THIS BIT  
8669 053700 001404 BEQ 3$ ;:BRANCH IF NO  
8670 053702 005237 050502 INC @DATENV ;:NUMBER OF CLOCKS GIVEN FOR DATA ENVELOPE  
8671 053706 004737 050520 JSR PC,@ECTEST ;:GO TO GENERATE AND TEST ECC  
8672 053712 052710 000002 3$: BIS #MCLK,@RO ;:SET CLOCK  
8673 053716 005737 050470 TST @TSECCG ;:IS THIS BIT TO GENERATE ECC  
8674 053722 001411 BEQ 8$ ;:BRANCH IF NO  
8675 053724 032710 000020 BIT #MRD,@RO ;:IS DATA BIT A ONE  
8676 053730 001404 BEQ 7$ ;:BRANCH IF DATA BIT IS - 0  
8677 053732 012737 177777 050462 MOV #-1,@ECCDATA ;:ECC DATA BIT IS A ONE  
8678 053740 000402 BR 8$ ;:BRANCH  
8679 053742 005037 050462 7$: CLR @ECCDATA ;:ECC DATA BIT IS = 0  
8680 053746 012746 000001 8$: MOV #DMD,-(SP) ;:KEEP DIAG. MODE  
8681 053752 006037 053564 ROR @WORD ;:CHECKING IF THERE IS A ONE  
8682 053756 103002 BCC 4$ ;:BRANCH IF NO ONE  
8683 053760 012716 000021 MOV #MRD.DMD,(SP) ;:KEEP DIAG. MODE AND DATA  
8684 053764 012610 4$: MOV (SP)+,@RO ;:SET DATA, DIAG. MODE, CLEAR CLOCK  
8685 053766 005737 050470 TST @TSECCG ;:IS THIS BIT TO GENERATE ECC  
8686 053772 001404 BEQ 9$ ;:BRANCH IF NO  
8687 053774 005237 050502 INC @DATENV ;:NUMBER OF CLOCKS FOR DATA ENVELOPE  
8688 054000 004737 050520 JSR PC,@ECTEST ;:GO TO GENERATE AND TEST ECC  
8689  
8690 054004 005302 9$: DEC R2 ;:BYTE COUNTER  
8691 054006 001341 BNE 3$ ;:CONTINUE IF ONE BYTE NOT COMPLETE
```

8692	054010	005305	DEC	R5	:WORD COUNTER
8693	054012	001300	BNE	1\$:CONTINUE IF ONE WORD NOT COMPLETE
8694	054014	012602	MOV	(SP)+,R2	::POP STACK INTO R2
8695	054016	000207	RTS	PC	:EXIT ----->
8696					
8697					
8698					
8699					
8700					
8701					

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

```
8702
8703
8704
8705
8706
8707
8708
8709 054020 000000      WWORD: 0
8710
8711
8712
8713
8714 054022      WRITE:
8715 054022 010046      MOV R0,-(SP)      ;;PUSH R0 ON STACK
8716 054024 010246      MOV R2,-(SP)      ;;PUSH R2 ON STACK
8717 054026 010346      MOV R3,-(SP)      ;;PUSH R3 ON STACK
8718 054030 010546      MOV R5,-(SP)      ;;PUSH R5 ON STACK
8719 054032 012705 000002  MOV #2,R5          ;WORD COUNTER
8720 054036 012710 000001  MOV #1,@R0        ;SET DIAG. MODE
8721 054042 012702 000007  1$: MOV #7,R2        ;BYTE COUNTER
8722 054046 012710 000013  MOV #MSTCK!MCLK!DMD,@R0 ;SET SECTOR & MANT. CLOCKS
8723 054052 032710 000040  BIT #MWR,@R0      ;CHECK WRITEBIT IN MAINT. REG.
8724 054056 001406      BEQ 2$           ;BRANCH IF ZERO
8725 054060 012737 177777 050462  MOV #-1,@#FCDATA ;ECC DATA BIT IS A ONE
8726 054066 000261      SEC             ;SET CARRY
8727 054070 006003      ROR R3         ;MOVE 1 FORWARD
8728 054072 000404      BR 3$         ;
8729 054074 005037 050462  2$: CLR @#ECDATA    ;ECC DATA BIT IS = 0
8730 054100 000241      CLC           ;CLEAR CARRY
8731 054102 006003      ROR R3         ;MOVE 0 FOR WWORD
8732 054104 012710 000001  3$: MOV #DMD,@R0    ;CLEAR SECTOR AND CLOCK
8733 054110 005737 050470  TST @#TSECCG    ;IS THIS BIT TO GENERATE ECC ?
8734 054114 001404      BEQ 4$         ;BRANCH IF NO
8735 054116 005237 050502  INC @#DATENV    ;NUMBER OF CLOCKS FOR DATA ENVELOPE
8736 054122 004737 050520  JSR PC,@#ECTEST ;GO TO GENERATE AND TEST ECC <----->
8737
8738 054126 052710 000002  4$: BIS #MCLK,@R0 ;SET CLOCK
8739 054132 032710 000040  BIT #MWR,@R0    ;CHECK WRITE BIT IN MAINT. REG.
8740 054136 001406      BEQ 5$         ;BRANCH IF ZERO
8741 054140 012737 177777 050462  MOV #-1,@#ECDATA ;ECC DATA BIT IS A ONE
8742 054146 000261      SEC             ;SET CARRY
8743 054150 006003      ROR R3         ;MOVE 1 FOR WWORD
8744 054152 000404      BR 6$         ;
8745 054154 005037 050462  5$: CLR @#ECDATA    ;ECC DATA BIT IS ZERO
8746 054160 000241      CLC           ;CLEAR CARRY
8747 054162 006003      ROR R3         ;MOVE 0 FOR WWORD
8748 054164 012710 000001  6$: MOV #DMD,@R0    ;CLEAR CLOCK
8749 054170 005737 050470  TST @#TSECCG    ;IS THIS BIT TO GENERATE ECC ?
8750 054174 001404      BEQ 7$         ;BRANCH IF NO
8751 054176 005237 050502  INC @#DATENV    ;NUMBER OF CLOCKS FOR DATA ENVELOPE
8752 054202 004737 050520  JSR PC,@#ECTEST ;GO TO GENERATE AND TEST ECC <----->
8753
8754 054206 005302  7$: DEC R2         ;COUNT FOR BYTE END
8755 054210 001346      BNE 4$         ;IF NOT BYTE END CONTINUE
8756 054212 005305      DEC R5         ;COUNT FOR WORD END
8757 054214 001312      BNE 1$         ;IF NOT WORD END CONTINUE
```

```
8758 054216 010337 054020      MOV    R3,@WORD      ;STORE THE WORD
8759 054222 012605              MOV    (SP)+,R5      ;:POP STACK INTO R5
8760 054224 012603              MOV    (SP)+,R3      ;:POP STACK INTO R3
8761 054226 012602              MOV    (SP)+,R2      ;:POP STACK INTO R2
8762 054230 012600              MOV    (SP)+,R0      ;:POP STACK INTO R0
8763 054232 000207              RTS     PC           ;EXIT ----->
8764
8765
8766
8767
8768
8769
```

```

8770                                     ;*WRITE DATA HOUSEKEEPING ROUTINE
8771
8772
8773
8774
8775
8776
8777
8778 054234 000000          COUNTD: 0
8779 054236 000400          FORMAT: 256.
8780 054240 000000          ZWORDS: 0
8781 054242
8782 054242 011137 054234          WRDATA:
8783 054246 012102          MOV      (R1),@#COUNTD ;STORE NO. OF WORDS TO BE WRITTEN
8784 054250 012137 053566          MOV      (R1)+,R2 ;SAME IN R2
8785 054254 010046          MOV      (R1)+,@#COMPA ;COMPARE OR NOT
8786 054256 010146          MOV      R0,-(SP) ;:PUSH R0 ON STACK
8787 054260 010246          MOV      R1,-(SP) ;:PUSH R1 ON STACK
8788 054262 010346          MOV      R2,-(SP) ;:PUSH R2 ON STACK
8789 054264 010446          MOV      R3,-(SP) ;:PUSH R3 ON STACK
8790 054266 012701 000016          MOV      R4,-(SP) ;:PUSH R4 ON STACK
8791 054272 012703 055600          MOV      #14.,R1 ;NO. OF TOLERANCE GAP WORDS
8792 054276 012723 177777          MOV      #TOLGAP,R3 ;START OF TOLERANCE GAP TABLE
8793 054302 005301          1$: MOV      #-1,(R3)+ ;MAKE IT 177777
8794 054304 001374          DEC      R1 ;IS 14 COMPLETED ?
8795
8796 054306 013700 014776          BNE     1$ ;IF NOT, CONTINUE
8797 054312 013746 054236          MOV      @#RHMR,R0 ;R0 CONTAINS MAINTANENCE REG.
8798 054316 163716 054234          MOV      @#FORMAT,-(SP)
8799 054322 011637 054240          SUB     @#COUNTD,(SP)
8800 054326 012604          MOV      (SP),@#ZWORDS ;NO. OF ZERO WORDS TO BE WRITTEN
8801 054330 005737 015142          MOV      (SP)+,R4
8802 054334 001403          TST     @#TSECC ;IS THIS AN ECC TEST
8803 054336 012737 177777 050470          BEQ     7$ ;BRANCH IF NO
8804 054344 012703 054572          MOV      #-1,@#TSECCG ;THESE BITS ARE TO GENERATE ECC
8805 054350 004737 054022          7$: MOV      #DISK,R3 ;SIMULATED DISK AREA
8806 054354 013723 054020          2$: JSR     PC,@#WRITE ;WRITE ON SIMULATED DISK
8807 054360 005302          MOV      @#WORD,(R3)+ ;STORE ON SIMULATED DISK
8808 054362 001372          DEC      R2
8809 054364 005704          BNE     2$
8810 054366 001406          TST     R4 ;ANY ZEROS TO BE WRITTEN ?
8811
8812 054370 004737 054022          BEQ     4$ ;BRANCH IF NONE TO BE WRITTEN
8813 054374 013723 054020          3$: JSR     PC,@#WRITE ;WRITE ZEROS ON SIMULATED DISK
8814 054400 005304          MOV      @#WORD,(R3)+ ;STORE THEM
8815 054402 001372          DEC      R4
8816 054404 005037 050470          BNE     3$
8817 054410 012701 000002          CLR     @#TSECCG ;NO MORE ECC TO BE GENERATED
8818 054414 004737 054022          4$: MOV      #2,R1
8819 054420 013723 054020          5$: JSR     PC,@#WRITE ;WRITE ECC1 AND ECC2 ON SIMULATED DISK
8820 054424 005301          MOV      @#WORD,(R3)+ ;STORE IN WEEC1 AND WEEC2
8821 054426 001372          DEC      R1
8822 054430 004737 054022          BNE     5$
8823 054434 013723 054020          JSR     PC,@#WRITE ;WRITE DATA GAP
8824 054440 012701 000016          MOV      @#WORD,(R3)+ ;STORE IT
8825 054444 004737 054022          6$: MOV      #14.,R1
          JSR     PC,@#WRITE ;WRITE TOLERANCE GAP ZEROS

```

```
8826 054450 013723 054020      MOV  @WORD,(R3)+  ;STORE THEM
8827 054454 005301      DEC  R1
8828 054456 001372      BNE  6$
8829 054460 012604      MOV  (SP)+,R4    ;;POP STACK INTO R4
8830 054462 012603      MOV  (SP)+,R3    ;;POP STACK INTO R3
8831 054464 012602      MOV  (SP)+,R2    ;;POP STACK INTO R2
8832 054466 012601      MOV  (SP)+,R1    ;;POP STACK INTO R1
8833 054470 012600      MOV  (SP)+,R0    ;;POP STACK INTO R0
8834 054472 000201      RTS  R1          ;EXIT ----->
8835
8836
8837
8838
8839
8840
8841
8842
```


8843
8844
8845
8846
8847
8848
8849
8850
8851
8852
8853
8854
8855
8856
8857
8858
8859
8860
8861
8862
8863
8864
8865
8866
8867
8868
8869
8870

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

054474 000023
054542 000001
054544 000004
054554 000001
054556 000005
054570 000001
054572
054572 000400
055572 000001
055574 000001
055576 000001
055600 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
;(ALSO USED IN SILO TEST AS SILO TABLE)
;DATA SPACE
;ECC1
;ECC2
;DATA GAP 2 BYTES OF 0
;TOLERANCE GAP 28 BYTES OF 0

```
8871 ;*WRITE HEADER AND DATA
8872
8873
8874
8875
8876
8877 ;*THIS SUBROUTINE IS THE FIRST IN A SERIES OF NESTED SUBROUTINES
8878
8879 ;*IT ISSUES DIAGNOSTIC MODE, AN EXTRA DIAGNOSTIC INDEX, AND THE
8880 ;*'GO' BIT
8881
8882 ;*IT THEN CALLS THREE OTHER SUBROUTINES, WHICH IN TURN CALL OTHER
8883 ;*SUBROUTINES. THE THREE SUBROUTINES CALLED HERE ARE:
8884
8885 ;* SEARCH ;ISSUES SECTOR CLOCKS TO SET SECTOR FOUND FLOP
8886 ;* WRHEAD ;WRITES THE SECTOR HEADER
8887 ;* WRDATA ;WRITES THE ACTUAL SECTOR DATA
8888
8889 ;*ALL OF THE ABOVE MENTIONED 'WRITING' IS ACTUALLY DONE INTO A CORE
8890 ;*BUFFER AREA CALLED 'DISK' VIA THE MAINTENANCE REGISTER (RHMR)
8891
8892
8893
8894
8895
8896
8897 055634 000000 RNCTR1: .WORD 0 ;'RUN' LINE STALL COUNTER
8898
8899 055636 011637 015132 COMWHD: MOV (SP),@#PCJSR ;SAVE PC OF JSR + 4
8900 055642 162737 000004 015132 SUB #4,@#PCJSR ;SAVE PC OF JSR
8901 055650 010046 MOV R0,-(SP) ;:PUSH R0 ON STACK
8902 055652 010146 MOV R1,-(SP) ;:PUSH R1 ON STACK
8903 055654 010246 MOV R2,-(SP) ;:PUSH R2 ON STACK
8904 055656 010346 MOV R3,-(SP) ;:PUSH R3 ON STACK
8905 055660 010446 MOV R4,-(SP) ;:PUSH R4 ON STACK
8906 055662 010546 MOV R5,-(SP) ;:PUSH R5 ON STACK
8907
8908 055664 012777 000001 137104 MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE
8909 055672 052777 000004 137076 BIS #MINX,@RHMR ;SET DIAGNOSTIC INDEX
8910 055700 042777 000004 137070 BIC #MINX,@RHMR ;CLEAR DIAGNOSTIC INDEX
8911 055706 052777 000001 137042 BIS #GO,@RHCS1 ;SET 'GO' BIT & STALL 'TILL 'RUN'
8912 ;(FUNCTION CODE WAS SET UP BY THE TEST)
8913 055714 012737 000113 055634 RNWAT1: MOV #75.,@#RNCTR1 ;LOAD STALL COUNTER = APPROX 450US
8914 ;FOR 11/50 CPU
8915 055722 005337 055634 1$: DEC @#RNCTR1 ;COUNT DOWN 1 TIME
8916 055726 001375 BNE 1$ ;CONTINUE UNTIL = 0
8917
8918 055730 013746 056014 MOV @#WSECTR,-(SP) ;GET DESIRED SECTOR/TRACK
8919 055734 042716 177740 BIC #177740,(SP) ;MAKE ONLY SECTOR
8920 055740 012637 055750 MOV (SP)+,@#WTRK ;SAVE SECTOR
8921 055744 004137 056722 2$: JSR R1,@#SEARCH ;ISSUE SECTOR CLOCKS TO GET TO
8922 ;DESIRED SECTOR <----->
8923
8924 055750 000000 WTRK: .WORD 0 ;SECTOR NO.
8925 055752 012701 000240 MOV #+NOP,R1 ;GOING TO MOVE NOPS
8926 055756 010137 056024 MOV R1,@#SEGPER ;NOP INTO SEGAP
```

```
8927 055762 010137 056026      MOV R1,@FSYNER      ;NOP INTO FSYNER
8928 055766 010137 056030      MOV R1,@ERHEAD     ;NOP INTO ERHEAD
8929 055772 010137 056032      MOV R1,@ERCRC      ;NOP INTO ERCRC
8930 055776 010137 056034      MOV R1,@ERHDGP     ;NOP INTO ERHDGAP
8931 056002 010137 056036      MOV R1,@HDESYN     ;NOP INTO HDESYN
8932
8933 056006 004137 056106      JSR R1,@WRHEAD     ;WRITE THE HEADER <----->
8934
8935 056012 000000      WCYL: 0            ;CYLINDER
8936 056014 000000      WSECTR: 0         ;SECTOR AND TRACK
8937 056016 000000      WKEY1: 0          ;KEY1
8938 056020 000000      WKEY2: 0          ;KEY2
8939 056022 000000      GCRC: 0           ;GOOD CRC
8940
8941
8942      ;DUMMY ERROR CALL LOCATIONS FOR THE WRITE HEADER OPERATION
8943
8944 056024 000240      SEGPER: NOP       ;IF 'ERROR 6' INSERTED BY
8945                                     ;WRHEAD SUBROUTINE, THEN
8946                                     ;SECTOR GAP GOING ON DISK
8947                                     ;IS NOT RIGHT.
8948
8949                                     ;WORD NO. CONTAINS WHICH
8950                                     ;WORD IS WRONG, THAT IS
8951                                     ;FIRST OF TENTH, OR WHAT EVER.
8952
8953                                     ;BAD WORD IS WHAT IS GOING ON DISK
8954
8955 056026 000240      FSYNER: NOP       ;IF 'ERROR 6' INSERTED BY
8956                                     ;WRHEAD SUBROUTINE, THEN
8957                                     ;THE LAST 0 BYTE OF SECTOR
8958                                     ;GAP OF FIRST SYNC. BYTE
8959                                     ;AFTER SECTOR GAP IS IN
8960                                     ;ERROR.
8961
8962                                     ;WORD NO. CONTAINS 20
8963                                     ;RIGHT BYTE IS SECTOR GAP
8964                                     ;LEFT BYTE IS SYNC. BYTE.
8965
8966                                     ;BAD WORD IS WHAT IS GOING ON
8967                                     ;DISK
8968
8969 056030 000240      ERHEAD: NOP      ;IF 'ERROR 6' INSERTED BY
8970                                     ;WRHEAD SUBROUTINE, THEN
8971                                     ;HEADER GOING ON DISK
8972                                     ;IS WRONG.
8973
8974                                     ;WORD NO 1 = CYLINDER NO
8975                                     ;WORD NO 2 = SECTOR/TRACK
8976                                     ;WORD NO 3 = KEY1
8977                                     ;WORD NO 4 = KEY2
8978
8979                                     ;BAD WORD IS WHAT IS GOING ON
8980                                     ;DISK
8981
8982
```

```
8983 056032 000240          ERCRC: NOP
8984
8985                          ;IF 'ERROR 6' INSERTED BY
8986                          ;WRHEAD SUBROUTINE, THEN CRC WRITTEN
8987                          ;ON DISK IS IN ERROR.
8988
8989                          ;GOOD DATA IS WHAT SHOULD BE ON DISK
8990                          ;BAD DATA IS WHAT IS GOING ON DISK.
8991
8992                          ;WORD NO IS 5
8993 056034 000240          ERHDGP: NOP
8994
8995                          ;IF 'ERROR 6' INSERTED BY
8996                          ;WRHEAD SUBROUTINE, THEN HEADER
8997                          ;GAP GOING ON DISK IS WRONG.
8998
8999                          ;WORD NO GIVES WHICH OF
9000                          ;THE HEADER GAP WORDS
9001                          ;ARE WRONG. FOR EXAMPLE:
9002
9003                          ;WORD NO 1 = FIRST HEADER
9004                          ;          GAP WORD
9005                          ;BAD WORD IS WHAT IS GOING ON DISK
9006 056036 000240          HDESYN: NOP
9007
9008                          ;IF 'ERROR 6' INSERTED BY
9009                          ;WRHEAD SUBROUTINE, THEN LAST
9010                          ;HEADER GAP BYTE OR HEADER
9011                          ;SYNC BYTE GOING ON DISK IS WRONG.
9012
9013                          ;WORD NO = 5
9014
9015                          ;BAD DATA IS WHAT IS GOING
9016                          ;ON DISK, RIGHT BYTE IS HEADER
9017                          ;GAP 0'S BYTE, LEFT BYTE IS HEADER
9018                          ;GAP SYNC.
9019
9020 056040 005737 015124          TST    @#ERFLG$      ;ARE ANY ERRORS DETECTED ?
9021 056044 001004          BNE    FOUT          ;IF YES BRANCH
9022 056046 004137 054242          JSR    R1,@#WRDATA    ;WRITE THE DATA
9023
9024 056052 000000          FNWORD: .WORD 0      ;FORMAT COMMAND NO. OF DATA
9025 056054 000000          .WORD 0
9026
9027 056056          FOUT:
9028 056056 012605          MOV    (SP)+,R5      ;;POP STACK INTO R5
9029 056060 012604          MOV    (SP)+,R4      ;;POP STACK INTO R4
9030 056062 012603          MOV    (SP)+,R3      ;;POP STACK INTO R3
9031 056064 012602          MOV    (SP)+,R2      ;;POP STACK INTO R2
9032 056066 012601          MOV    (SP)+,R1      ;;POP STACK INTO R1
9033 056070 012600          MOV    (SP)+,R0      ;;POP STACK INTO R0
9034 056072 000207          RTS    PC          ;EXIT
```

```
9035
9036      ;*WRITE HEADER
9037
9038
9039
9040
9041
9042      ;*R0 = MAINT.REG.
9043      ;*R1 = SIMULATED DISK
9044      ;*R2 = BYTE COUNT
9045      ;*R3 = WRITE WORD
9046      ;*R5 = WORD COUNT
9047
9048
9049
9050
9051      056074 000000      SCYL: 0
9052      056076 000000      SSECTR: 0
9053      056100 000000      SKEY1: 0
9054      056102 000000      SKEY2: 0
9055      056104 000000      SCRC: 0
9056
9057
9058      056106 012137 056074      WRHEAD: MOV (R1)+,@#SCYL
9059      056112 012137 056076      MOV (R1)+,@#SSECTR
9060      056116 012137 056100      MOV (R1)+,@#SKEY1
9061      056122 012137 056102      MOV (R1)+,@#SKEY2
9062      056126 012137 056104      MOV (R1)+,@#SCRC
9063      056132 010146      MOV R1,-(SP)      ;;PUSH R1 ON STACK
9064
9065      056134 012701 054474      MOV #SECGAP,R1      ;SIMULATED DISK INDICATOR
9066      056140 013700 014776      MOV @#RHMR,R0      ;R0 NOW HAS MAINT. REG. ADDR.
9067      056144 012710 000001      MOV #DMD,@R0      ;SET DIAG. MODE
9068      056150 012705 000002      MOV #2,R5      ;WORD COUNTER
9069      056154 052710 000010      BIS #MSTCK,@R0      ;SET SECTOR FOR FIRST BYTE
9070      056160 012710 000013      1$: MOV #MSTCK!MCLK!DMD,@R0;SET SECTOR, CLOCK, DIAG.
9071      ;MODE, RESET INDEX
9072      056164 032710 000040      BIT #MWR,@R0      ;CHECK WRITE BIT IN MAINT. REG.
9073      056170 001403      BEQ 2$
9074      056172 000261      SEC      ;SET CARRY
9075      056174 006003      ROR R3      ;MOVE ONE FORWARD
9076      056176 000402      BR 3$
9077      056200 000241      2$: CLC      ;CLEAR CARRY
9078      056202 006003      ROR R3      ;MOVE ZERO FORWARD
9079      056204 012710 000001      3$: MOV #DMD,@R0      ;CLEAR CLOCK, SECTOR
9080      056210 012702 000007      MOV #7,R2      ;BYTE COUNTER
9081      056214 052710 000002      4$: BIS #MCLK,@R0      ;SET BIT CLOCK
9082      056220 032710 000040      BIT #MWR,@R0      ;CHECK WRITE BIT IN MAINT.REG.
9083      056224 001403      BEQ 5$      ;BRANCH IF ZERO
9084
9085      056226 000261      SEC      ;SET CARRY
9086      056230 006003      ROR R3      ;MOVE ONE FORWARD
9087      056232 000402      BR 6$
9088      056234 000241      5$: CLC
9089      056236 006003      ROR R3
9090      056240 012710 000001      6$: MOV #DMD,@R0
```

```
9091 056244 005302      DEC      R2
9092 056246 001362      BNE      4$
9093 056250 005305      DEC      R5
9094 056252 001342      BNE      1$
9095 056254 010321      MOV      R3,(R1)+
9096 056256 005703      TST      R3
9097 056260 001414      BEQ      7$
9098
9099 056262 012737 000001 052774      MOV      #1,@#ERWORD
9100 056270 005037 001124      CLR      @#$GDDAT
9101 056274 010337 001126      MOV      R3,@#$BDDAT
9102 056300 012737 104006 056024      MOV      #104006,@#SEGP
9103 056306 000137 056714      JMP      @#17$ ;BRANCH OUT ----->
9104
9105 056312 012702 000022      MOV      #18,R2 ;COUNT NO. OF SECTOR GAP
9106 056316 012737 000024 052774 10$: MOV      #20,@#ERWORD ;COUNT TO GIVE ERROR WORD
9107 056324 004737 054022      JSR      PC,@#WRITE ;WRITE SECTOR GAP
9108 056330 013721 054020      MOV      @#WORD,(R1)+ ;STORE SECTOR GAP WORD
9109 056334 001413      BEQ      11$
9110 056336 160237 052774      SUB      R2,@#ERWORD ;IF NOT GET ERROR WORD NO.
9111 056342 005037 001124      CLR      @#$GDDAT ;GOOD WORD
9112 056346 013737 054020 001126      MOV      @#WORD,@#$BDDAT ;BAD WORD
9113 056354 012737 104006 056024      MOV      #104006,@#SEGP ;STORE 'ERROR 6' IN SEGP
9114 056362 000554      BR       17$ ;BRANCH OUT
9115 056364 005302      11$: DEC      R2 ;HAVE 18 WORDS OF ZEROS BEEN WRITTEN ?
9116 056366 001353      BNE      10$ ;IF NOT CONTINUE
9117
9118 ;AT THIS POINT THE SECTOR FOUND FLOP SHOULD
9119 ;BE HIGH, SO THAT THE HEADER SYNC BYTE CAN BE GIVEN.
9120
9121 ;HOWEVER, IN THE DRIVE TIMING ERROR TEST THE REST OF THE ROUTINE
9122 ;IS ABORTED
9123
9124 056370 005737 015144      TST      @#TESDTE ;IS THIS A DRIVE TIMING ERROR TEST ?
9125 056374 001147      BNE      17$ ;BRANCH OUT IF YES ----->
9126
9127 056376 004737 054022      JSR      PC,@#WRITE ;WRITE ONE SECTOR GAP 0 BYTE
9128 ;AND ONE SYNC. BYTE = 230
9129 056402 013711 054020      MOV      @#WORD,(R1) ;SAVE 0 BYTE AND SYNC BYTE
9130 056406 023721 052756      CMP      @#RSYNC,(R1)+ ;IF SYNC. BYTE RIGHT
9131 056412 001414      BEQ      12$ ;IF YES CONTINUE OPERATION
9132 056414 012737 000024 052774      MOV      #20,@#ERWORD ;IF NOT GET READY FOR ERROR PRINT
9133
9134 056422 013737 052756 001124      MOV      @#RSYNC,@#$GDDAT ;GOOD WORD
9135 056430 014137 001126      MOV      -(R1),@#$BDDAT ;BAD WORD
9136 056434 012737 104006 056026      MOV      #104006,@#FSYNER ;INSERT 'ERROR 6' IN FSYNER
9137 056442 000524      BR       17$ ;BRANCH OUT ----->
9138
9139 056444 012702 000004      12$: MOV      #4,R2 ;FOUR HEADER WORDS
9140 056450 012703 056074      MOV      #SCYL,R3 ;POINTER FOR HEADER TABLE
9141 056454 012737 000005 052774 13$: MOV      #5,@#ERWORD ;ERROR WORD NO SET
9142 056462 004737 054022      JSR      PC,@#WRITE ;WRITE 4 HEADER WORDS
9143 056466 013711 054020      MOV      @#WORD,(R1) ;STORE WRITTEN WORD
9144 056472 022321      CMP      (R3)+,(R1)+ ;IS IT CORRECT ?
9145 056474 001412      BEQ      14$ ;IF GOOD CONTINUE OPERATION
9146 ;IF NOT GET READY FOR ERROR PRINT
```

```
9147
9148 056476 160237 052774      SUB      R2,@#ERWORD      :WORD NO
9149 056502 014337 001124      MOV      -(R3),@#SGDDAT  :GOOD DATA
9150 056506 014137 001126      MOV      -(R1),@#SBDDAT  :BAD DATA
9151 056512 012737 104006 056030  MOV      #104006,@#ERHEAD:INSERT 'ERROR 6'
9152 056520 000475              BR       17$             :BRANCH OUT ----->
9153
9154 056522 005302              14$:   DEC      R2             :ARE 4 HEADER WORDS DONE?
9155 056524 001353              BNE     13$             :IF NOT CONTINUE
9156 056526 004737 054022      JSR     PC,@#WRITE     :WRITE CRC
9157 056532 013711 054020      MOV     @#WORD,(R1)    :STORE CRC
9158 056536 022137 056022      CMP     (R1)+,@#GCRC   :COMPARE GOOD CRC
9159 056542 001414              BEQ     20$             :IF GOOD CONTINUE OPERATION
9160
9161
9162 056544 014137 001126      MOV     -(R1),@#SBDDATA:BAD CRC WRITTEN
9163 056550 013737 056022 001124  MOV     @#GCRC,@#SGDDAT:GOOD CRC
9164 056556 012737 000005 052774  MOV     #5,@#ERWORD    :ERROR WORD NO
9165 056564 012737 104006 056032  MOV     #104006,@#ERCRC:INSERT ERROR 6
9166 056572 000450              BR       17$             :EXIT ----->
9167
9168 056574 012702 000005      20$:   MOV     #5,R2           :NO OF HEADER GAP
9169 056600 012737 000006 052774  15$:   MOV     #6,@#ERWORD    :ERROR WORD NO SET
9170 056606 004737 054022      JSR     PC,@#WRITE     :WRITE HEADER GAP
9171 056612 013721 054020      MOV     @#WORD,(R1)+  :STORE
9172 056616 001412              BEQ     16$             :IF GOOD CONTINUE OPERATION
9173
9174
9175 056620 160237 052774      SUB     R2,@#ERWORD    :ERROR WORD NO
9176 056624 005037 001124      CLR     @#SGDDAT       :GOOD DATA
9177 056630 014137 001126      MOV     -(R1),@#SBDDAT:BAD DATA
9178 056634 012737 104006 056034  MOV     #104006,@#ERHDP:STORE 'ERROR 6'
9179 056642 000424              BR       17$             :BRANCH OUT ----->
9180
9181 056644 005302              16$:   DEC      R2             :ARE 5 HEADER GAP ZEROS DONE ?
9182 056646 001354              BNE     15$             :IF NOT CONTINUE
9183 056650 004737 054022      JSR     PC,@#WRITE     :WRITE CRC
9184 056654 013711 054020      MOV     @#WORD,(R1)   :STORE CRC
9185 056660 023721 052756      CMP     @#RSYNC,(R1)+ :COMPARE GOOD CRC
9186 056664 001413              BEQ     17$             :A-OK, EXIT ----->
9187
9188 056666 012737 000005 052774  MOV     #5,@#ERWORD    :IF NOT GET READY FOR ERROR PRINT
9189 056674 014137 001126      MOV     -(R1),@#SBDDAT:BAD DATA
9190 056700 013737 052756 001124  MOV     @#RSYNC,@#SGDDAT:GOOD DATA
9191 056706 012737 104006 056036  MOV     #104006,@#HDESYN:STORE 'ERROR 6'
9192
9193 056714              17$:   MOV     (SP)+,R1       :POP STACK INTO R1
9194 056714 012601              RTS     R1             :EXIT ----->
9195 056716 000201
9196
9197
9198
9199
9200
9201
9202
```



```
9204          ;*SEARCH SECTOR
9205
9206
9207
9208
9209
9210
9211          ;*      R0=RHMR ADDRESS
9212          ;*      R1=PASSED ARGUMENT (SECTOR SEARCHED FOR)
9213          ;*      R2=CLOCK COUNT (PER BYTE)
9214          ;*      R3=SECTOR COUNTER FROM R1
9215          ;*      R5=BYTES PER WORD COUNT
9216
9217          ;*BEFORE INDEX IS GIVEN TWO SECTOR CLOCKS ARE GIVEN TO RESET
9218          ;*SECTOR PULSE IN CASE IT IS SET.
9219
9220          ;*AT THE BEGINNING OF EACH SECTOR, ONE SECTOR CLOCK HAS TO RISE
9221          ;*BEFORE MAINT. CLOCK, THEN EVERY EIGHT MAINT.CLOCKS, ONE SECTOR CLOCK
9222          ;*IS IDENTICAL WITH THE MAINT. CLOCK
9223          ;*THE SECTOR CLOCKS ARE NUMBERED AS FOLLOWS:
9224
9225          ;*THE SECTOR CLOCK UNDER INDEX - 0
9226          ;*THE NEXT - 1
9227          ;*THE NEXT - 2
9228          ;*ETC.
9229          ;*THEN THE LAST SECTOR CLOCK IN ONE SECTOR HAS NUMBER - 608
9230          ;*THE NEXT SECTOR WILL HAVE 608 SECTOR CLOCKS
9231          ;*THE NEXT SECTOR THEN HAS ANOTHER 608 SECTOR CLOCKS
9232          ;*AND SO ON
9233
9234
9235
9236 056720 000000          SECTR: 0          ;SECTOR SEARCHED FOR
9237
9238 056722 012137 056720 SEARCH: MOV      (R1)+,@#SECTR ;SAVE SECTOR SEARCHED FOR
9239 056726 010046          MOV      R0,-(SP)          ;;PUSH R0 ON STACK
9240 056730 010146          MOV      R1,-(SP)          ;;PUSH R1 ON STACK
9241 056732 010246          MOV      R2,-(SP)          ;;PUSH R2 ON STACK
9242 056734 010346          MOV      R3,-(SP)          ;;PUSH R3 ON STACK
9243 056736 010446          MOV      R4,-(SP)          ;;PUSH R4 ON STACK
9244 056740 010546          MOV      R5,-(SP)          ;;PUSH R5 ON STACK
9245 056742 013700 014776 MOV      @#RHMR, R0          ;NOW R0 HAS MAINTENANCE REG. ADR.
9246 056746 013703 056720 MOV      @#SECTR,R3          ;LOAD SECTOR COUNTER
9247
9248 056752 012710 000001 MOV      #DMD,@R0          ;SET DIAGNOSTIC MODE
9249 056756 052710 000010 BIS      #MSTCK,@R0          ;SET SECTOR CLOCK
9250 056762 042710 000010 BIC      #MSTCK,@R0          ;CLEAR SECTOR CLOCK
9251 056766 052710 000010 BIS      #MSTCK,@R0          ;SET SECTOR CLOCK
9252 056772 042710 000010 BIC      #MSTCK,@R0          ;CLEAR SECTOR CLOCK
9253          ;THE ABOVE TWO SECTOR CLOCKS ARE GIVEN FOR
9254          ;RESETTING SECTOR PULSE
9255          ;IN CASE IT STARTS SET
9256 056776 052710 000014 BIS      #MINX!MSTCK,@R0    ;SET INDEX AND SECTOR CLOCK
9257 057002 042710 000014 BIC      #MINX!MSTCK,@R0    ;RESET INDEX AND SECTOR CLOCK
9258 057006 005703          TST      R3                  ;TEST FOR SECTOR 0
9259 057010 001461          BEQ      7$                  ;IF SECTOR 0 IS REQUIRED, EXIT ----->
```

9260
9261
9262
9263
9264
9265
9266
9267
9268
9269
9270
9271
9272
9273
9274
9275
9276
9277
9278
9279
9280
9281
9282
9283
9284
9285
9286
9287
9288
9289
9290
9291
9292
9293
9294
9295
9296
9297
9298
9299
9300
9301
9302
9303
9304
9305
9306
9307
9308
9309
9310
9311
9312
9313
9314
9315

057012 012702 000010
057016 012705 000002
057022 052710 000010
057026 052710 000002
057032 000402
057034 052710 000012
057040 042710 000012
057044 052710 000002
057050 042710 000002
057054 005302
057056 001372

057060 012702 000007
057064 005305
057066 001362

057070 012701 000457
057074 012705 000002
057100 012702 000007
057104 052710 000012
057110 042710 000012
057114 052710 000002
057120 042710 000002
057124 005302
057126 001372

057130 005305
057132 001362

057134 005301
057136 001356

057140 052710 000010
057144 042710 000010
057150 005303
057152 001317

057154
057154 012605
057156 012604
057160 012603

:NOW 304 WORDS WILL START (608 BYTES)

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

:ONE WORD ONLY

1\$: MOV #8., R2 ;BYTE COUNTER
MOV #2., R5 ;BYTES PER WORD
BIS #MSTCK,@R0 ;SET SECTOR CLOCK
BIS #MCLK,@R0 ;SET MAINT. CLOCK
BR 3\$;BRANCH TO CLEAR SECTOR AND CLOCK
2\$: BIS #MSTCK!MCLK,@R0 ;SET BOTH CLOCKS
3\$: BIC #MSTCK!MCLK,@R0 ;CLEAR BOTH CLOCKS
8\$: BIS #MCLK,@R0 ;SET MAINT. CLOCK
BIC #MCLK,@R0 ;CLEAR MAINT. CLOCK
DEC R2 ;BYTE COUNTER
BNE 8\$;CONTINUE IF BYTE NOT COMPLETE

MOV #7,R2 ;SETUP FOR SECOND BYTE
DEC R5 ;IS WORD COMPLETE?
BNE 2\$;CONTINUE IF NOT COMPLETE
;TO GIVE SECTOR CLOCK AND MAINT. CLOCK

:NOW 303 WORDS ARE LEFT AND ALL ARE IDENTICAL

4\$: MOV #303.,R1 ;WORDS PER SECTOR COUNTER
MOV #2.,R5 ;BYTES PER WORD COUNTER
5\$: MOV #7,R2 ;BIT COUNTER (MAINT.CLOCK COUNTER)
BIS #MSTCK!MCLK,@R0 ;SET SECTOR CLOCK AND MAINT. CLOCK
BIC #MSTCK!MCLK,@R0 ;CLEAR CLOCKS
6\$: BIS #MCLK,@R0 ;SET MAINT. CLOCK
BIC #MCLK,@R0 ;RESET MAINT. CLOCK
DEC R2 ;IS BYTE COMPLETE ?
BNE 6\$;CONTINUE IF NOT COMPLETE

DEC R5 ;IS WORD COMPLETE ?
BNE 5\$;CONTINUE IF NOT

DEC R1 ;IS SECTOR COMPLETE ?
BNE 4\$;CONTINUE IF NOT

BIS #MSTCK,@R0 ;SET SECTOR CLOCK 1 MORE TIME (FOR 609)
BIC #MSTCK,@R0 ;CLEAR SECTOR CLOCK
DEC R3 ;IS REQUIRED NO OF SECTORS COMPLETE ?
BNE 1\$;CONTINUE IF NOT

7\$: MOV (SP)+,R5 ;;POP STACK INTO R5
MOV (SP)+,R4 ;;POP STACK INTO R4
MOV (SP)+,R3 ;;POP STACK INTO R3

```
9316 057162 012602      MOV      (SP)+,R2      ;;POP STACK INTO R2
9317 057164 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
9318 057166 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
9319 057170 000201      RTS      R1
9320
9321
9322      ;*READ ONE SECTOR OF DATA
9323
9324 057172 000000      RNO:      0      ;NO. OF WORDS READ
9325 057174 000000      RCOM:     0      ;EXTRA STORAGE
9326
9327
9328
9329 057176 012137 057172      REDATA:  MOV      (R1)+,@#RNO      ;SAVE NO. OF WORDS ONLY FOR INFORMATION
9330 057202 012137 057174      MOV      (R1)+,@#RCOM     ;EXTRA WORD ONLY FOR INFORMATION
9331 057206 010146      MOV      R1,-(SP)        ;PUSH R1 ON STACK
9332 057210 005737 015142      TST      @#TSECC         ;IS THIS AN ECC TEST
9333 057214 001403      BEQ      1$              ;BRANCH IF NO
9334 057216 012737 177777 050470      MOV      #-1,@#TSECCG     ;THESE BITS ARE TO GENERATE ECC
9335 057224 012702 000402      1$:      MOV      #258.,R2        ;256 WORDS PER SECTOR
9336                                     ;PLUS 2 ECC WORDS
9337 057230 012703 054572      MOV      #DISK,R3        ;POINTE TO DISK SIMULATION
9338 057234 012337 053564      2$:      MOV      (R3)+,@#WORD     ;READY TO READ CONTENTS
9339 057240 004737 053570      JSR      PC,@#READ       ;READ
9340 057244 005302      DEC      R2              ;IS 256 WORDS DONE?
9341 057246 001372      BNE      2$              ;IF NOT BRANCH
9342 057250 005737 015142      TST      @#TSECC         ;IS THIS AN ECC TEST
9343 057254 001012      BNE      4$              ;BRANCH OUT IF YES
9344 057256 005037 050470      CLR      @#TSECCG        ;NO MORE ECC BITS ARE TO BE GENERATED
9345 057262 012702 000017      MOV      #15.,R2         ;ONE DATA GAP, 14 TOLERANCE GAP
9346 057266 012337 053564      3$:      MOV      (R3)+,@#WORD     ;READY TO READ CONTENTS OF WORD
9347 057272 004737 053570      JSR      PC,@#READ       ;READ
9348 057276 005302      DEC      R2              ;COUNT
9349 057300 001372      BNE      3$              ;BRANCH IF 14 NOT DONE
9350      4$:
9351 057302 012601      MOV      (SP)+,R1        ;;POP STACK INTO R1
9352 057304 000201      RTS      R1              ;RETURN
```



```
9360 .SBTTL
9361 .SBTTL ***SYS/AC LIBRARY ROUTINES***
9362 .SBTTL
9363
9364
9365
9366 057374 104407 CKSWR ;;TEST FOR CHANGE IN SOFT-SWR
9367 057376 005037 052770 CLR @#NOSYNC ;CLEAR FLAG FOR HEADER ERROR COMMANDS
9368 057402 005037 015142 CLR @#TSECC ;CLEAR FLAG FOR ECC TEST
9369 057406 005037 050470 CLR @#TSECCG ;EVEN IN AN ECC TEST EVERY CLOCK
9370 057412 005037 015144 CLR @#TESDTE ;DRIVE TIMING ERROR TEST
9371 057416 032777 040000 121514 1$: BIT #BIT14,@SWR ;;LOOP ON PRESENT TEST?
9372 057424 001111 BNE $OVER ;;YES IF SW14=1
9373 057426 000416 $XTSTR: BR 6$ ;;IF RUNNING ON THE 'XOR' TEST? CHANGE
9374 057430 013746 000004 MOV @#ERRVEC,-(SP) ;;SAVE THE CONTENTS OF THE ERROR VECTOR
9375 057434 012737 057454 000004 MOV #5$,@#ERRVEC ;;SET FOR TIMEOUT
9376 057442 005737 177060 TST @#177060 ;;TIME OUT ON XOR?
9377 057446 012637 000004 MOV (SP)+,@#ERRVEC ;;RESTORE THE ERROR VECTOR
9378 057452 000463 BR $SVLAD ;;GO TO THE NEXT TEST
9379 057454 022626 5$: CMP (SP)+,(SP)+ ;;CLEAR THE STACK AFTER A TIME OUT
9380 057456 012637 000004 MOV (SP)+,@#ERRVEC ;;RESTORE THE ERROR VECTOR
9381 057462 000423 BR 7$ ;;LOOP ON THE PRESENT TEST
9382 057464 032777 000400 121446 BIT #BIT08,@SWR ;;LOOP ON SPEC. TEST?
9383 057472 001404 BEQ 2$ ;;BR IF NO
9384 057474 127737 121440 001102 CMPB @SWR,$STSTM ;;ON THE RIGHT TEST? SWR<7:0>
9385 057502 001462 BEQ $OVER ;;BR IF YES
9386 057504 105737 001103 2$: TSTB $ERFLG ;;HAS AN ERROR OCCURRED?
9387 057510 001421 BEQ 3$ ;;BR IF NO
9388 057512 123737 001115 001103 CMPB $ERMAX,$ERFLG ;;MAX. ERRORS FOR THIS TEST OCCURRED?
9389 057520 101015 BHI 3$ ;;BR IF NO
9390 057522 032777 001000 121410 BIT #BIT09,@SWR ;;LOOP ON ERROR?
9391 057530 001404 BEQ 4$ ;;BR IF NO
9392 057532 013737 001110 001106 7$: MOV $LPERR,$LPADR ;;SET LOOP ADDRESS TO LAST SCOPE
9393 057540 000443 BR $OVER
9394 057542 105037 001103 4$: CLRB $ERFLG ;;ZERO THE ERROR FLAG
9395 057546 005037 001212 CLR $TIMES ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
9396 057552 000415 BR 1$ ;;ESCAPE TO THE NEXT TEST
9397 057554 032777 004000 121356 3$: BIT #BIT11,@SWR ;;INHIBIT ITERATIONS?
9398 057562 001011 BNE 1$ ;;BR IF YES
9399 057564 005737 001100 TST $PASS ;;IF FIRST PASS OF PROGRAM
9400 057570 001406 BEQ 1$ ;; INHIBIT ITERATIONS
9401 057572 005237 001104 INC $ICNT ;;INCREMENT ITERATION COUNT
9402 057576 023737 001212 001104 CMP $TIMES,$ICNT ;;CHECK THE NUMBER OF ITERATIONS MADE
9403 057604 002021 BGE $OVER ;;BR IF MORE ITERATION REQUIRED
9404 057606 012737 000001 001104 1$: MOV #1,$ICNT ;;REINITIALIZE THE ITERATION COUNTER
9405 057614 013737 057664 001212 MOV $MXCNT,$TIMES ;;SET NUMBER OF ITERATIONS TO DO
9406 057622 105237 001102 $SVLAD: INCB $STSTM ;;COUNT TEST NUMBERS
9407 057626 011637 001106 MOV (SP),$LPADR ;;SAVE SCOPE LOOP ADDRESS
9408 057632 011637 001110 MOV (SP),$LPERR ;;SAVE ERROR LOOP ADDRESS
9409 057636 005037 001214 CLR $ESCAPE ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
9410 057642 112737 000001 001115 MOVB #1,$ERMAX ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
9411 057650 013777 001102 121264 $OVER: MOV $STSTM,@DISPLAY ;;DISPLAY TEST NUMBER
9412 057656 013716 001106 MOV $LPADR,(SP) ;;FUDGE RETURN ADDRESS
9413 057662 000002 RTI ;;FIXES PS
9414 057664 000004 $MXCNT: 4 ;;MAX. NUMBER OF ITERATIONS
```

CZF
CZF
CC
CHE
CHE
CHE
CKS
CLA
CLD
CLF
COM
COM
COM
COM
COM
COL
CPH
CR
CR
CRI
CSI
CSU
CS
CSI
CYI
DAI
DAI
DAI
DB
DCI
DCI
DD
DEI
DE
DFI
DF
DF
DF
DF
DF

9572	060656	117746	120264			MOVB	@\$TKB,-(SP)	::YES
9573	060662	042716	177600			BIC	#^C177,(SP)	::MAKE IT 7-BIT ASCII
9574	060666	021627	000007			CMP	(SP),#7	::IS IT A CONTROL-G?
9575	060672	001300				BNE	2\$::IF NOT, PUT IT IN THE TTY QUEUE
9576	060674	123727	001134	000001	6\$:	CMPB	\$AUTOB,#1	::ARE WE RUNNING IN AUTO-MODE?
9577	060702	001674				BEQ	2\$::BRANCH IF YES
9578	060704	005726				TST	(SP)+	::CLEAR CONTROL-G OFF STACK
9579	060706	004737	060352			JSR	PC,\$TKINT	::FLUSH THE TTY INPUT QUEUE
9580	060712	005077	120226			CLR	@\$TKS	::DISABLE TTY KEYBOARD INTERRUPTS
9581	060716	112737	000001	001135		MOVB	#,\$INTAG	::SET INTERRUPT MODE INDICATOR
9582	060724	104401	061423			TYPE	,\$CNTLG	::ECHO THE CONTROL-G (^G)
9583	060730	104401	061430			\$GTSWR: TYPE	,\$MSWR	::TYPE CURRENT CONTENTS
9584	060734	013746	000176			MOV	SWREG,-(SP)	::SAVE SWREG FOR TYPEOUT
9585	060740	104402				TYPOC		::GO TYPE--OCTAL ASCII(ALL DIGITS)
9586	060742	104401	061441			TYPE	,\$MNEW	::PROMPT FOR NEW SWR
9587	060746	005046			19\$:	CLR	-(SP)	::CLEAR COUNTER
9588	060750	005046				CLR	-(SP)	::THE NEW SWR
9589	060752	105777	120166		7\$:	TSTB	@\$TKS	::CHAR THERE?
9590	060756	100375				BPL	7\$::IF NOT TRY AGAIN
9591	060760	117746	120162			MOVB	@\$TKB,-(SP)	::PICK UP CHAR
9592	060764	042716	177600			BIC	#^C177,(SP)	::MAKE IT 7-BIT ASCII
9593	060770	021627	000003			CMP	(SP),#3	::IS IT A CONTROL-C?
9594	060774	001015				BNE	9\$::BRANCH IF NOT
9595	060776	104401	061411			TYPE	,\$CNTLC	::YES, ECHO CONTROL-C (^C)
9596	061002	062706	000006			ADD	#6,SP	::CLEAN UP STACK
9597	061006	123727	001135	000001		CMPB	\$INTAG,#1	::REENABLE TTY KEYBOARD INTERRUPTS?
9598	061014	001003				BNE	8\$::BRANCH IF NO
9599	061016	012777	000100	120120		MOV	#100,@\$TKS	::ALLOW TTY KEYBOARD INTERRUPTS
9600	061024	000137	044710		8\$:	JMP	OPERSEL	::CONTROL-C RESTART
9601	061030	021627	000025		9\$:	CMP	(SP),#25	::IS IT A CONTROL-U?
9602	061034	001005				BNE	10\$::BRANCH IF NOT
9603	061036	104401	061416			TYPE	,\$CNTLU	::YES, ECHO CONTROL-U (^U)
9604	061042	062706	000006		20\$:	ADD	#6,SP	::IGNORE PREVIOUS INPUT
9605	061046	000737				BR	19\$::LET'S TRY IT AGAIN
9606	061050	021627	000015		10\$:	CMP	(SP),#15	::IS IT A <CR>?
9607	061054	001022				BNE	16\$::BRANCH IF NO
9608	061056	005766	000004			TST	4(SP)	::YES, IS IT THE FIRST CHAR?
9609	061062	001403				BEQ	11\$::BRANCH IF YES
9610	061064	016677	000002	120046		MOV	2(SP),@SWR	::SAVE NEW SWR
9611	061072	062706	000006		11\$:	ADD	#6,SP	::CLEAR UP STACK
9612	061076	104401	001223		14\$:	TYPE	,\$CRLF	::ECHO <CR> AND <LF>
9613	061102	123727	001135	000001		CMPB	\$INTAG,#1	::RE-ENABLE TTY KBD INTERRUPTS?
9614	061110	001003				BNE	15\$::BRANCH IF NOT
9615	061112	012777	000100	120024		MOV	#100,@\$TKS	::RE-ENABLE TTY KBD INTERRUPTS
9616	061120	000002			15\$:	RTI		::RETURN
9617	061122	004737	060262		16\$:	JSR	PC,\$TYPEC	::ECHO CHAR
9618	061126	021627	000060			CMP	(SP),#60	::CHAR < 0?
9619	061132	002420				BLT	18\$::BRANCH IF YES
9620	061134	021627	000067			CMP	(SP),#67	::CHAR > 7?
9621	061140	003015				BGT	18\$::BRANCH IF YES
9622	061142	042726	000060			BIC	#60,(SP)+	::STRIP-OFF ASCII
9623	061146	005766	000002			TST	2(SP)	::IS THIS THE FIRST CHAR
9624	061152	001403				BEQ	17\$::BRANCH IF YES
9625	061154	006316				ASL	(SP)	::NO, SHIFT PRESENT
9626	061156	006316				ASL	(SP)	:: CHAR OVER TO MAKE
9627	061160	006316				ASL	(SP)	:: ROOM FOR NEW ONE.

CZ
CZ
ER
ER
EX
EX
EX
EX
EO
E1
E1
E4
FE
FE
FI
FI
FM
FN
FO
FO
FS
FU
GC
GC
GE
GE
GN
GC
GR
GT
HA
HA
HC
HC
HC
HD
HD
HE


```
9840 062362 010046          $TRAP:  MOV    R0,-(SP)      ;;SAVE R0
9841 062364 016600 000002    MOV    2(SP),R0          ;;GET TRAP ADDRESS
9842 062370 005740          TST    -(R0)             ;;BACKUP BY 2
9843 062372 111000          MOVVB  (R0),R0           ;;GET RIGHT BYTE OF TRAP
9844 062374 006300          ASI    R0                ;;POSITION FOR INDEXING
9845 062376 016000 062416    MOV    $TRPAD(R0),R0     ;;INDEX TO TABLE
9846 062402 000200          RTS    R0                ;;GO TO ROUTINE
9847 062404 011646          $TRAP2: MOV   (SP),-(SP)   ;;MOVE THE PC DOWN
9848 062406 016666 000004 000002  MOV   4(SP),2(SP)       ;;MOVE THE PSW DOWN
9849 062414 000002          RTI                     ;;RESTORE THE PSW
9850 062416 062404          $TRPAD: .WORD  $TRAP2
9851 062420 060112          $TYPE  ;;CALL=TYPE      TRAP+1(104401) TTY TYPEOUT ROUTINE
9852 062422 062160          $TYPOC ;;CALL=TYPOC     TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
9853 062424 062134          $TYPOS ;;CALL=TYPOS     TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
9854 062426 062174          $TYPON ;;CALL=TYPON        TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
9855 062430 057666          $TYPDS ;;CALL=TYPDS      TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
9856 062432 060730          $GTSWR ;;CALL=GTSWR     TRAP+6(104406) GET SOFT-SWR SETTING
9857 062434 060640          $CKSWR ;;CALL=CKSWR    TRAP+7(104407) TEST FOR CHANGE IN SOFT-SWR
9858 062436 061202          $RDCHR ;;CALL=RDCHR   TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
9859 062440 061272          $RDLIN ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
9860 062442 061452          $RDOCT ;;CALL=RDOCT    TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
9861 062444 045752          T.SCOPI;;CALL=SCOPI   TRAP+13(104413) MY LOCAL SCOPES
9862 062446 046024          CHECKT ;;CALL=CHECKD   TRAP+14(104414) CHECK DVA,RDY,DPR,DRY
9863 062450 046342          WAIT.T  ;;CALL=WAT        TRAP+15(104415) WAIT LOOP
```

CZF
CZF
RTM
RUN
RUN
R11
R15
R16
R44
SAV
SAV
SAV
SC
SCC
SCF
SCY
SC1
SC1
SC2
SC2
SC4
SEA
SEC
SEC

SEC
SEE
SEC
SEL
SEL
SEF
SE1
SE1
SE1
SE1
SIL
SKE
SKE
SK.
SN
SNE
SRC
SR.
SR.
SR.
SS.
SS.
SS.
SS.
SS.
SS.
SS.
SS.

```

9864 062452 012737 062616 000024 $PWRDN: MOV    #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
9865 062460 012737 000340 000026      MOV    #340,@#PWRVEC+2 ;;PRIO:7
9866 062466 010046          MOV    R0,-(SP)      ;;PUSH R0 ON STACK
9867 062470 010146          MOV    R1,-(SP)      ;;PUSH R1 ON STACK
9868 062472 010246          MOV    R2,-(SP)      ;;PUSH R2 ON STACK
9869 062474 010346          MOV    R3,-(SP)      ;;PUSH R3 ON STACK
9870 062476 010446          MOV    R4,-(SP)      ;;PUSH R4 ON STACK
9871 062500 010546          MOV    R5,-(SP)      ;;PUSH R5 ON STACK
9872 062502 017746 116432      MOV    @SWR,-(SP)    ;;PUSH @SWR ON STACK
9873 062506 010637 062622      MOV    SP,$SAVR6    ;;SAVE SP
9874 062512 012737 062524 000024      MOV    #$PWRUP,@#PWRVEC ;;SET UP VECTOR
9875 062520 000000          HALT
9876 062522 000776          BR     .-2          ;;HANG UP
9877 062524 012737 062616 000024 $PWRUP: MOV    #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
9878 062532 013706 062622      MOV    $SAVR6,SP    ;;GET SP
9879 062536 005037 062622      CLR    $SAVR6      ;;WAIT LOOP FOR THE TTY
9880 062542 005237 062622      1$:   INC    $SAVR6  ;;WAIT FOR THE INC
9881 062546 001375          BNE    1$          ;;OF WORD
9882 062550 012677 116364      MOV    (SP)+,@SWR   ;;POP STACK INTO @SWR
9883 062554 012605          MOV    (SP)+,R5    ;;POP STACK INTO R5
9884 062556 012604          MOV    (SP)+,R4    ;;POP STACK INTO R4
9885 062560 012603          MOV    (SP)+,R3    ;;POP STACK INTO R3
9886 062562 012602          MOV    (SP)+,R2    ;;POP STACK INTO R2
9887 062564 012601          MOV    (SP)+,R1    ;;POP STACK INTO R1
9888 062566 012600          MOV    (SP)+,R0    ;;POP STACK INTO R0
9889 062570 012737 062452 000024      MOV    #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
9890 062576 012737 000340 000026      MOV    #340,@#PWRVEC+2 ;;PRIO:7
9891 062604 104401          TYPE   $POWER      ;;REPORT THE POWER FAILURE
9892 062606 062624      $PWRMG: .WORD $POWER ;;POWER FAIL MESSAGE POINTER
9893 062610 012716          MOV    (PC)+,(SP)  ;;RESTART AT BEGIN
9894 062612 017356      $PWRAD: .WORD BEGIN ;;RESTART ADDRESS
9895 062614 000002          RTI
9896 062616 000000      $ILLUP: HALT      ;;THE POWER UP SEQUENCE WAS STARTED
9897 062620 000776          BR     .-2          ;;BEFORE THE POWER DOWN WAS COMPLETE
9898 062622 000000      $SAVR6: 0          ;;PUT THE SP HERE
9899 062624 005015 047520 042527 $POWER: .ASCIZ <15><12>'POWER''
9900 062632 000122
9901
9902          000001      .END
  
```

CZR
CZR
SS7
STA
STA
STA
STK
SWR
SWR
SWC
SWC
SWC
SWC
SWC
SWC
SWC
SWC
SWC
SWC
SW1
SW1
SW1
SW1
SW1
SW1
SW1
SW2
SW3
SW4
SW5
SW6
SW7
SW8
SW9
S11
S12
S30
TAC
TBJ
TDF
TES
TES
TIM
TIM
TKV
TMF
TOU
TOV
TPV
TRA
TRE

HEDGAP	052670	8387*	8409#	8577*														
HEDSYN	052672	8388*	8422#	8599*	8624*													
HEGAP	054556	8568	8857#															
HT	= 000011	494#	9480	9516														
IAE	= 002000	2361#																
IE	= 000100	2321#	3719	3749	4046	4074	4081	6919	6948									
ILF	= 000001	2351#																
ILLEGL	015212	2645#																
ILR	= 000002	2352#																
IOTVEC	= 000020	494#	2689*	2690*														
IR	= 000100	2301#	4490	4495	4633													
IXE	= 004000	2441#																
I00	024156	3178#	3184															
I02	024262	3218#																
I03	024356	3241#																
I04	024456	3267#																
I05	024556	3293#																
I06	024656	3320#																
I07	024756	3346#																
I14	026130	3560#																
I15	026332	3619#																
I16	026474	3665#																
I24	027060	3772#																
KEY1	052660	4159*	4232*	4392*	5019*	5179*	5344*	5661*	5818*	5979*	6283*	6433*	6602*	6778*				
		7404*	7496*	7654*	8393#													
KEY2	052662	4160*	4233*	4393*	5020*	5180*	5345*	5662*	5819*	5980*	6284*	6434*	6603*	6779*				
		7405*	7497*	7655*	8394#													
KIPAR0-	172340	520#																
KIPAR1=	172342	520#																
KIPAR2=	172344	520#																
KIPAR3=	172346	520#																
KIPAR4=	172350	520#																
KIPAR5=	172352	520#																
KIPAR6=	172354	520#																
KIPAR7=	172356	520#																
KIPDR0=	172300	520#																
KIPDR1=	172302	520#																
KIPDR2=	172304	520#																
KIPDR3=	172306	520#																
KIPDR4=	172310	520#																
KIPDR5=	172312	520#																
KIPDR6=	172314	520#																
KIPDR7=	172316	520#																
LA	015066	2576#	4328*	4502*	4638*													
LAD	045750	7150*	7200#	7204														
LERR	045532	7133#	7139*	7162														
LF	= 000012	494#	9511	9516														
LST	= 002000	2343#																
LST14A	026160	3556	3570#															
LST15A	026362	3613	3632#															
LST16A	026532	3657	3682#															
L00	024152	3176#																
L02	024260	3216#																
L03	024352	3237#																
L04	024450	3264#																
L05	024546	3288#																

CZ
 CZ
 ST
 ST
 ST
 ST
 SX
 SS
 SO
 SA
 .
 .
 E
 C
 R
 C
 D

PIE6 = 002000	7892#													
PIE7 = 001000	7893#													
PIE8 = 000400	7894#													
PIE9 = 000200	7895#													
PIP = 020000	2346#													
PIRQ = 177772	494#													
PIRQVE = 000240	494#													
PKACK 015206	2643#	3098												
PLU = 020000	2443#													
POSITI 050476	1977	1983	4850*	5004*	5164*	5329*	5491*	5646*	5803*	5964*	6121*	6268*	6418*	
	6587*	6763*	7935#	8076	8129	8137*	8138	8140	8178*					
PRE = 000020	2490#													
PROG = 001000	2342#	7263												
PRO = 000000	494#													
PR1 = 000040	494#													
PR2 = 000100	494#													
PR3 = 000140	494#													
PR4 = 000200	494#													
PR5 = 000240	494#													
PR6 = 000300	494#													
PR7 = 000340	494#													
PS = 177776	494#	2717*	4013*	4044*	4072*	6918*	6947*	6969*	7063*					
PSEL = 002000	2325#	3745	3957	3962										
PSU = 000001	2487#													
PSW = 177776	494#													
PUTREG 045470	4194	4279	4438	4596	5067	5109	5236	5279	5398	5418	5452	5709	5750	
	5873	5913	6036	6054	6087	6331	6371	6496	6537	6658	6678	6712	6834	
	6854	6888	6925	6931	6957	7112#	7238	7293	7713	8073	8079	8146		
PWRVEC = 000024	494#	2695*	2696*	9864*	9865*	9874*	9877*	9889*	9890*					
P12 050512	7951#	8005*	8006*	8035										
P22 050514	7952#	8013*	8014*	8040										
P24 050516	7953#	8021*	8022*	8041										
P3 050510	7950#	7996*	7997*	8034										
RA 000200	504#	8253												
RCOM 057174	9325#	9330*												
RCYL 052760	8474#	8509*	8550											
RDCHR - 104410	9652	9858#												
RDHEAD 052776	8389	8509#												
RDLIN 104411	9684	9859#												
RDOCT = 104412	2768	7079	7086	8218	8235	9860#								
RDY = 000200	2322#	4046	6919	6932	6948	7239	7250	7294						
READ 053570	8537	8542	8551	8553	8555	8557	8559	8609	8613	8647#	9339	9347		
READAT 015174	2638#													
READIN 015210	2644#													
RECALI 015154	2630#													
REDATA 057176	8452	9329#												
REFOR 015176	2639#	4252	5048	5217	5379	5690	5854	6017	6312	6477	6639	6815		
REGADR 045534	1922	1957	1960	2266	2268	2270	2852*	3137*	3179*	3213*	3234*	3261*	3285*	
	3313*	3338*	3382*	3438*	3478*	3531*	3555*	3612*	3656*	3712*	3741*	3769*	3799*	
	3827*	3857*	3896*	3927*	3954*	3980*	4010*	4036*	4065*	4116*	4373*	4531*	4665*	
	7134#	7138*	7773*	7784*	7794*	7805*	7871*							
REGSA1 061622	2780	9720#												
REINTO 016264	2653#	3103	3110*	3123	3129	4243	4323	4348	4511	4515*	4523	4646	4650*	
	4657	4933	4934	4937*	4938*	4940	4941	4951	5034	5117	5203	5286	5367	
	5459	5575	5576	5579*	5580*	5582	5583	5593	5676	5756	5842	5921	6003	
	6095	6204	6205	6208*	6209*	6211	6212	6222	6298	6377	6463	6545	6627	

SS7	033224	4688*	4727#	4818*										
STACK =	001000	494#	2688	2778	2824	3091	3147	3176	3210	3231	3258	3282	3310	3335
		3389	3435	3475	3516	3552	3609	3653	3709	3744	3767	3796	3824	3854
		3888	3915	3951	3977	4007	4043	4071	4100	4129	4143	4223	4384	4542
		4675	4837	4975	5134	5300	5478	5617	5773	5935	6108	6241	6390	6559
		6734	6912	6941										
START	017362	2678	2682#											
STARTA	017604	2717#												
STKLMT=	177774	494#												
SWR	001140	541#	2686	2704*	2706	2711*	2754	2760	2793	2862	2883	4215	7202	7472
		8061	8063	9371	9382	9384	9390	9397	9537	9568	9610*	9723	9730	9734
		9738	9872	9882*										
SWREG	000176	496#	2711	2754	9537	9568	9584							
SW0 =	000001	494#												
SW00 =	000001	494#												
SW01 =	000002	494#												
SW02 =	000004	494#												
SW03 =	000010	494#												
SW04 =	000020	494#												
SW05 =	000040	494#												
SW06 =	000100	494#												
SW07 =	000200	494#	4217	7474										
SW08 =	000400	494#												
SW09 =	001000	494#	7202											
SW1 =	000002	494#												
SW10 =	002000	494#												
SW11 =	004000	494#												
SW12 =	010000	494#	2760											
SW13 =	020000	494#	2793	2862	2883									
SW14 =	040000	494#												
SW15 =	100000	494#												
SW2 =	000004	494#												
SW3 =	000010	494#												
SW4 =	000020	494#												
SW5 =	000040	494#												
SW6 =	000100	494#	8063											
SW7 =	000200	494#												
SW8 =	000400	494#	8061											
SW9 =	001000	494#												
S11	025524	3442#	3464											
S12	025660	3481#	3502											
S36	027620	3922#	3939											
TAGDTE	015146	2617#	4679*	4825*										
TBITVE=	000014	494#												
TDF =	000040	2435#												
TESDTE	015144	2613#	4268*	4427*	4585*	7099*	8561	9124	9370*					
TESTAD	044706	7060#												
TIMCNT	046340	6920	6949	7329#	7339	7345								
TIMOT =	000004	3157#	3173	3174*	3189*	3386	3387*	3397*						
TKVEC =	000060	494#	9522*	9523*										
TMPILL	015140	2607#	4173*	4197	4203									
TOLGAP	055600	4151	8791	8864#										
TOTALA	015136	2605#	2876*	2878*	3416	3440	3479	3920						
TPVEC =	000064	494#												
TRAPVE=	000034	494#	2693*	2694*										
TRE -	040000	2328#	3891	3899	3918	3930	4324	4498	4632					

VUF = 000002	2488#													
VU30 = 010000	2442#													
VV = 000100	2339#	3110	7263											
WAIT.T 046342	7331#	9863												
WAT = 104415	9863#													
WC = 015024	1940	2557#	4258	4322*	4364	4418	4488*	4493*	4522	4576	4625*	4629*	4656	
	7117													
WCE = 040000	2309#													
WCF = 000040	2356#													
WCRC = 054554	4165	4237	4397	5025	5185	5350	5667	5824	5985	6289	6439	6608	6784	
	7410	7503	7659	8558	8856#									
WCU = 000001	2430#													
WCYL = 056012	4547*	4554	4763*	4770	4863*	4870	5504*	5511	6134*	6141	8935#			
WECC1 = 055572	1974	4914	5555	6184	8861#									
WECC2 = 055574	1974	4918	5559	6188	6448	6451*	6452*	8862#						
WKEY1 = 056016	4550*	4766*	4866*	5507*	6137*	8937#								
WKEY2 = 056020	4551*	4767*	4867*	5508*	6138*	8938#								
WLE = 004000	2362#													
WORD = 053564	8536*	8541*	8550*	8552*	8554*	8556*	8558*	8608*	8612*	8641#	8651*	8664*	8681*	
	9338*	9346*												
WRCHDA = 047026	7493#													
WRCHDT = 015166	2635#	7428												
WRCHEK = 015164	2634#	7522												
WRCHHD = 046514	7401#													
WRDATA = 054242	8447	8781#	9022											
WRFROM = 015220	2652#	3117	3123*	3130	4338	4342*	4365	4403	4489	4494	4561	4626	4630	
	4705	4731	4732	4776	4801	4876	5094	5116	5259	5274*	5285	5431	5447*	
	5458	5517	5736	5755	5896	5912*	5920	6070	6086*	6094	6147	6358	6376	
	6519	6544	6691	6707*	6709*	6719	6868	6884*	6885*	6893	7726	7825		
WRHEAD = 056106	8933	9058#												
WRIDAT = 015170	2636#	4180	4412											
WRIFOR = 015172	2637#	4570	4778	4894	5535	6165	7822							
WRITE = 054022	8571	8583	8714#	8805	8812	8818	8822	8825	9107	9127	9142	9156	9170	
	9183													
WRL = 004000	2344#													
WRU = 000400	2438#													
WSECTR = 056014	4549*	4765*	4865*	5506*	6136*	8918	8936#							
WSSYNC = 054542	8854#													
WSU = 000004	2432#													
WTRK = 055750	8920*	8924#												
WWORD = 054020	8572	8576	8580	8584	8592	8597	8602	8709#	8758*	8806	8813	8819	8823	
	8826	9108	9112	9129	9143	9157	9171	9184						
X = 052664	4162*	4234*	4394*	5022*	5182*	5347*	5664*	5821*	5982*	6286*	6436*	6605*	6781*	
	7407*	7499*	7699*	7716	8395#	8439								
XE2 = 022270	2881	2903#												
X11 = 025604	3423	3462#												
Y = 052724	8449#													
Y11 = 025176	3422	3425#												
ZCODE = 050504	1983	4856*	5010*	5170*	5335*	5497*	5652*	5809*	5970*	6127*	6274*	6424*	6593*	
	6769*	7942#	8134*											
ZER = 000400	2377#	8143												
ZWORDS = 054240	8780#	8799*												
\$AUTOB = 001134	538#	2758*	9576	9678										
\$BDADR = 001122	533#	1929	1932	1954	1987	7245*	7252*	7256*	7269*	7274*	7277*	7297*	7305*	
	7311*	7316*												
\$BDDAT = 001126	535#	1922	1926	1932	1937	1949	1951	1957	1960	1964	1967	2268	2848*	

