

RL11,RLV11

RL01/02 DRIVE COMPAT
CZRLLB0

AH-F130B-MC
FICHE 1 OF 1

MAR 1980
COPYRIGHT 77 80
MADE IN USA



Microfiche grid containing multiple frames of data, including text and tables.



IDENTIFICATION

PRODUCT CODE: AC-F131B-MC
PRODUCT NAME: CZRLB0 RL01/02 DRIVE COMPATABILITY
DATE CREATED: 5-JAN-79
REVISED: 7-DEC-79
MAINTAINER: DIAGNOSTIC ENGINEERING
AUTHORS: D. DEKNIS, C. CAMPBELL

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) 1979, DIGITAL EQUIPMENT CORPORATION

TABLE OF CONTENTS

1.0	GENERAL INFORMATION
1.1	PROGRAM ABSTRACT
1.1.1	STRUCTURE OF PROGRAM
1.1.2	DIAGNOSTIC INFORMATION
1.2	SYSTEM REQUIREMENTS
1.2.1	HARDWARE REQUIREMENTS
1.2.2	SOFTWARE REQUIREMENTS
1.3	RELATED DOCUMENTS AND STANDARDS
1.4	DIAGNOSTIC HIERARCHY PREREQUISITES
1.5	ASSUMPTIONS
2.0	OPERATING INSTRUCTIONS
2.1	HOW TO RUN THIS DIAGNOSTIC
2.1.1	THE FIVE STEPS OF EXECUTION
2.1.2	SAMPLE RUN-THROUGH
2.2	CHAIN MODE OPERATION
2.3	DETAILS OF COMMANDS AND SYNTAX
2.3.1	TABLE OF COMMAND VALIDITY
2.3.2	COMMAND SYNTAX
2.4	EXTENDED P-TABLE DIALOGUE
2.5	HARDWARE PARAMETERS
2.6	SOFTWARE PARAMETERS
3.0	ERROR INFORMATION
3.1	ERROR REPORTING
3.2	ERROR HALTS
4.0	PERFORMANCE AND PROGRESS REPORTS
4.1	PERFORMANCE REPORTS
4.2	PROGRESS REPORTS
5.0	DEVICE INFORMATION TABLES
6.0	TEST SUMMARIES

1.0 GENERAL INFORMATION1.1 PROGRAM ABSTRACT1.1.1 STRUCTURE OF PROGRAM

THIS DIAGNOSTIC IS COMPATIBLE WITH BOTH XXDP+ AND ACT. IT CAN BE RUN STANDALONE UNDER XXDP+, AND CAN BE CHAINED UNDER XXDP+, ACT AND APT IN ACT MODE (SEE 2.2 "CHAIN MODE OPERATION" FOR DETAILS OF CHAINING PROCEDURE). IT IS A SINGLE PROGRAM FROM THE STANDPOINT OF THE DIAGNOSTIC USER, WHICH AT RUN TIME IS APPENDED TO A COMMON FRONT-END PIECE OF SUPERVISOR SOFTWARE THROUGH WHICH THE DIAGNOSTIC PROGRAM INTERFACES TO THE ENVIRONMENT AS IT EXECUTES.

WHEN THIS DIAGNOSTIC IS STARTED, CONTROL GOES FIRST TO THE SUPERVISOR PORTION, WHICH WILL ASK CERTAIN "HARD CORE" QUESTIONS ABOUT THE ENVIRONMENT. THEN IT WILL ENTER COMMAND MODE, INDICATED BY A PROMPT CHARACTER (DR>). AT COMMAND MODE THE OPERATOR MAY ENTER ANY OF SEVERAL COMMANDS AS DESCRIBED IN 2.0 "OPERATING INSTRUCTIONS".

THE DIAGNOSTIC PROGRAM IS LOADED IN THE LOWER 8K OF MEMORY. THE DIAGNOSTIC SUPERVISOR CODING OCCUPIES 6.25K OF THE UPPER PART OF MEMORY JUST BELOW THE XXDP+ MONITOR WHICH RESIDES IN THE UPPERMOST 1.5K OF MEMORY SPACE.

1.1.2 DIAGNOSTIC INFORMATION

THE RLO1 DRIVE COMPATABILITY TEST IS A PDP-11 (LSI-11) BASED PROGRAM THAT WILL TEST INTERCHANGEABILITY OF CARTRIDGES BETWEEN DRIVES. THE TEST PERFORMS WRITES, READS, OVERWRITES, ADJACENT CYLINDER WRITES TO PROVE COMPATABILITY.

1.2 SYSTEM REQUIREMENTS1.2.1 HARDWARE REQUIREMENTS

- * PDP-11/LSI-11 PROCESSOR WITH 16K OR MORE OF MEMORY
- * CONSOLE DEVICE (LA30,LA36,VT50,ETC.)

* 1 OR 2 RL11/RLV11 CONTROLLER(S) WITH:

- 1 - 8 RL01 DRIVES WITH RL01K CARTRIDGES CONTAINING A 'BAD
SEC. OR FILE'
- 1 - 8 RL02 DRIVES WITH RL02K CARTRIDGES CONTAINING A 'BAD
SECTOR FILE'

* LINE PRINTER (OPTIONAL)

1.2.2 SOFTWARE REQUIREMENTS

CZRLLAO RL01/02 DRIVE COMPATABILITY
(FORMERLY CZRLF8)

1.3 RELATED DOCUMENTS AND STANDARDS

RL01 DISK SUBSYSTEM USER'S GUIDE (EK-RL01-UG-002)
XXDP+/SUPERVISOR USER'S MANUAL

1.4 DIAGNOSTIC HIERARCHY PREREQUISITES

THE RL01/02 SUBSYSTEM SHOULD HAVE SUCCESSFULLY RUN THE FOLLOWING
PROGRAMS:

CVRLABO	RLV11 RL01/02 DISKLESS TEST (RLV11 ONLY)
CZRLGBO	RL11/RLV11 RL01/02 CONTROLLER TEST (PART 1)
CZRLHBO	RL11/RLV11 RL01/02 CONTROLLER TEST (PART 2)
CZRLIBO	RL01/02 DRIVE TEST (PART 1)
CZRLJBO	RL01/02 DRIVE TEST (PART 2)
CZRLKBO	RL11/RLV11 RL01/02 PERFORMANCE EXERCISER
CZRLNAO	RL01/02 DRIVE TEST (PART 3)

1.5 ASSUMPTIONS

THE HARDWARE OTHER THAN THE RL01/02 SUBSYSTEM IS ASSUMED TO WORK
PROPERLY. FALSE ERRORS MAY BE REPORTED IF THE PROCESSOR, ETC., DO
NOT FUNCTION PROPERLY.

2.0 OPERATING INSTRUCTIONS

2.1 HOW TO RUN THIS DIAGNOSTIC2.1.1 THE FIVE STEPS OF EXECUTION

THIS DIAGNOSTIC PROGRAM SHOULD BE LOADED AND STARTED USING NORMAL XXDP+ PROCEDURES. START THE EXECUTION OF THE XXDP+ MONITOR BY USING THE APPROPRIATE BOOTSTRAP PROGRAM. THE MONITOR WILL PRINT A MESSAGE IDENTIFYING ITSELF AND REQUESTING THAT THE CURRENT DATE BE ENTERED. AN EXAMPLE OF THIS MESSAGE IS GIVEN BELOW FOR THE XXDP+ MONITOR:

```
CHMDKAO XXDP+ MONITOR
BOOTED VIA UNIT 0
ENTER DATE (DD-MMM-YY):
```

AFTER THE DATE HAS BEEN ACCEPTED BY THE MONITOR THE RESTART ADDRESS OF THE MONITOR IS PRINTED. THEN THE FOLLOWING TWO QUESTIONS ARE ASKED:

```
50 HZ ? N
LSI ? N
```

THE DEFAULTS ARE BOTH 'NO'. TYPE 'R' AND THE PROGRAM NAME TO RUN THE PROGRAM. DO NOT TYPE THE EXTENSION.

WHEN THIS DIAGNOSTIC IS STARTED THE FOLLOWING 5 STEPS WILL OCCUR:

```
*****
* STEP 1 *
*****
```

THE DIAGNOSTIC WILL ISSUE THE PROMPT 'DR>'. FROM THIS POINT UNTIL THE TIME WHEN YOU RESTART XXDP+, YOU WILL BE TALKING TO THE DIAGNOSTIC, NOT XXDP+. WE WILL REFER TO THE PRESENCE OF THIS PROMPT AS BEING IN DIAGNOSTIC COMMAND MODE, AS OPPOSED TO XXDP+ COMMAND MODE.

AT THIS POINT YOU WILL ENTER A 'START' COMMAND. THIS IS NOT THE SAME AS THE XXDP+ 'START' COMMAND, WHICH YOU ALREADY ISSUED IN RESPONSE TO THE XXDP+ DOT PROMPT. THIS 'START' COMMAND CAN TAKE A NUMBER OF SWITCHES AND FLAGS (ALL OPTIONAL) AND THE DETAILS OF THESE ARE SET FORTH IN 2.3 'DETAILS OF COMMANDS AND SYNTAX'. HOWEVER, IN ORDER TO USE THE PROGRAM, ALL YOU NEED TO SAY IS SOMETHING LIKE THIS:

```
STA/PASS:1/FLAGS:HOE
```


THINGS TO NOTE HERE:

1. ONLY THE FIRST THREE CHARACTERS OF THIS OR ANY COMMAND AT THE 'DR>' LEVEL NEED TO BE TYPED.
2. THE 'PASS' SWITCH SPECIFIES HOW MANY PASSES YOU DESIRE. A PASS CONSISTS OF RUNNING THE FULL DIAGNOSTIC AGAINST ALL UNITS BEING TESTED (THIS WILL BE EXPLAINED SHORTLY). ONE PASS IS SPECIFIED IN THE ABOVE EXAMPLE.
3. THE 'FLAGS' SWITCH MAY SPECIFY ANY OF A NUMBER OF FLAGS, BUT THE MAIN USEFUL ONES ARE:

PNT	PRINT NUMBER OF TEST BEING EXECUTED
LOE	LOOP ON ERROR
HOE	HALT ON ERROR
IER	INHIBIT ERROR PRINTOUT

THE HOE FLAG IS SPECIFIED IN THE ABOVE EXAMPLE (WE'LL SEE WHY SHORTLY).

* STEP 2 *

WHEN YOU HAVE TYPED IN A 'START' COMMAND, THE DIAGNOSTIC WILL COME BACK WITH THE QUESTION '# UNITS?' TO WHICH YOU SHOULD RESPOND BY TYPING IN THE NUMBER OF DEVICES YOU WISH TO TEST.

A WORD OF WARNING HERE: THE NUMBER OF UNITS DEPENDS ON THE TARGET DEVICE OF THE DIAGNOSTIC. FOR EXAMPLE, IF THE DIAGNOSTIC IS DIRECTED AT A DISK DRIVE, THEN THE NUMBER OF UNITS WOULD BE THE NUMBER OF DRIVES TO BE TESTED. WHEREAS IF THE DIAGNOSTIC WAS DIRECTED AT THE DISK CONTROLLER, THEN THE NUMBER OF UNITS WOULD BE THE NUMBER OF CONTROLLERS. THE TARGET DEVICE OF A DIAGNOSTIC CAN ALWAYS BE DETERMINED BY INSPECTING THE 'HEADER' STATEMENT NEAR THE BEGINNING OF THE SOURCE CODE. ONE OF THE OPERANDS OF THIS 'HEADER' STATEMENT SHOULD BE THE DEVICE TYPE OF THE DIAGNOSTIC.

* STEP 3 *

WHEN YOU HAVE TYPED IN THE NUMBER OF UNITS TO BE TESTED, THE DIAGNOSTIC WILL ASK YOU THE 'HARDWARE QUESTIONS'. THE ANSWERS TO THESE QUESTIONS ARE USED TO BUILD TABLES IN CORE, CALLED 'HARDWARE P-TABLES'. ONE HARDWARE P-TABLE WILL BE BUILT FOR EACH UNIT TO BE TESTED.

THERE ARE SEVERAL HARDWARE QUESTIONS AND THE ENTIRE SERIES WILL BE POSED N TIMES, WHERE N IS THE NUMBER OF UNITS.

THIS REPRESENTS A NEW PHILOSOPHY IN DIAGNOSTIC ENGINEERING. DIAGNOSTICS IN THE FUTURE WILL NOT BE WRITTEN TO AUTOSIZE OR ASSUME STANDARD ADDRESSES: INSTEAD, THEY WILL ASK THE OPERATOR FOR ALL THE INFORMATION THEY NEED TO TEST THE DEVICE.

* STEP 4 *

AFTER YOU HAVE ANSWERED ALL THE HARDWARE QUESTIONS (SEC 2.5) FOR ALL THE UNITS, YOU WILL BE ASKED "CHANGE SW?" IF YOU WANT TO BE ASKED THE SOFTWARE QUESTIONS THAT DETERMINE THE BEHAVIOR OF THIS PROGRAM, TYPE 'Y'. IF YOU WANT TO TAKE ALL THE DEFAULTS TO THESE QUESTIONS, TYPE 'N'. IF YOU TYPE 'Y' YOU WILL BE ASKED THE SOFTWARE QUESTIONS (SEC 2.6), AND THE ANSWERS WILL BE PUT INTO THE SOFTWARE P-TABLE IN THE PROGRAM. THE SERIES OF QUESTIONS WILL BE ASKED JUST ONCE, REGARDLESS OF THE NUMBER OF UNITS TO BE TESTED.

* STEP 5 *

AFTER YOU HAVE ANSWERED THE SOFTWARE QUESTIONS, THE DIAGNOSTIC WILL BEGIN TO EXECUTE THE HARDWARE TEST CODE. THERE ARE SEVERAL THINGS THAT CAN HAPPEN NEXT, DEPENDING ON WHETHER A HARDWARE ERROR IS ENCOUNTERED AND ALSO ON WHAT SWITCH VALUES YOU SELECTED ON THE START COMMAND. CONSIDER THE POSSIBILITIES:

1. IF NO ERROR IS ENCOUNTERED, THEN THE DIAGNOSTIC WILL SIMPLY EXECUTE THE DESIRED NUMBER OF PASSES AND RETURN TO COMMAND MODE (PROMPT DR>).
2. IF AN ERROR IS ENCOUNTERED, THEN ONE OF THREE THINGS HAPPENS, DEPENDING ON THE SETTINGS OF THE HOE AND LOE FLAGS.

HOE SET: THE ERROR WILL BE REPORTED ON THE CONSOLE AND THE DIAGNOSTIC WILL RETURN TO COMMAND MODE.

LOE SET: THE DIAGNOSTIC WILL LOOP ENDLESSLY ON THE BLOCK OF CODE THAT DETECTED THE ERROR.

NEITHER HOE NOR LOE SET: THE ERROR WILL BE REPORTED ON THE CONSOLE AND NORMAL EXECUTION WILL RESUME AS IF NO ERROR HAD OCCURRED.

2.1.2 SAMPLE RUN-THROUGH

LET'S SEE HOW ALL THIS WORKS IN A REAL SITUATION. RECALL THAT WE ENTERED THE COMMAND "STA/PASS:1/FLAGS:HOE". THIS WOULD BE A VERY TYPICAL WAY TO RUN THE DIAGNOSTIC. IF NO ERRORS ARE ENCOUNTERED, THE SINGLE REQUESTED PASS WILL BE EXECUTED AND THE PROMPT WILL BE RE-ISSUED.

IF AN ERROR IS ENCOUNTERED, THE ERROR WILL BE REPORTED AND THE PROMPT WILL BE REISSUED (BECAUSE THE HOE FLAG IS SET). AT THIS POINT THERE ARE FOUR DIFFERENT WAYS YOU CAN GET THE PROGRAM GOING AGAIN:

1. ISSUE ANOTHER "START" COMMAND (THUS GOING THRU ALL OF STEPS 1, 2, 3, 4, AND 5 AGAIN)
2. ISSUE A "RESTART" COMMAND (SAME AS START COMMAND EXCEPT THAT THE HARDWARE QUESTIONS ARE NOT ASKED)
3. ISSUE A "CONTINUE" COMMAND (EXECUTION WILL RESUME AT THE BEGINNING OF THE PARTICULAR HARDWARE TEST (MOST DIAGNOSTICS CONSIST OF A NUMBER OF THESE) THAT IT WAS IN WHEN THE ERROR HALT OCCURRED. NO QUESTIONS ASKED.)
4. ISSUE A "PROCEED" COMMAND: EXECUTION WILL RESUME AT THE INSTRUCTION FOLLOWING THE ERROR REPORT (THIS IS A SPECIAL COMMAND AND CAN BE ISSUED ONLY AT A HALT ON ERROR).

THE MOST TYPICAL THING TO DO HERE IS TO ISSUE THE PROCEED, BUT WITH DIFFERENT FLAG SETTINGS. PROBABLY YOU WOULD WANT TO SAY:

```
PRO/FLAGS:IER:LOE:HOE=0
```

THIS WILL DO THE FOLLOWING:

1. TURN ON THE IER (INHIBIT ERROR PRINTOUT) FLAG
2. TURN ON THE LOE FLAG
3. TURN OFF THE HOE FLAG
4. RESUME EXECUTION AT INSTRUCTION AFTER ERROR REPORT

THE DIAGNOSTIC WILL NOW LOOP ON THE BLOCK OF CODE THAT DETECTED AND REPORTED THE ERROR, BUT NO ERROR PRINTOUT WILL OCCUR. THUS YOU CAN STUDY THE ERROR OR SCOPE IT OR WHATEVER.

WHEN YOU'VE SEEN ENOUGH, YOU MAY HIT CONTROL/C. THIS WILL TAKE YOU OUT OF THE LOOP AND PUT YOU BACK INTO COMMAND MODE. YOU NOW HAVE THREE CHOICES:

1. START
2. RESTART
3. CONTINUE

LET'S SAY YOU'VE REPAIRED THE DEFECT FOUND ABOVE AND WANT TO FINISH RUNNING THE DIAGNOSTIC. YOU WOULD TYPE

CON/FLAGS:HOE:IER=0:LOE=0

THIS WILL RESTORE THE FLAGS TO THEIR ORIGINAL VALUES AND RESUME EXECUTION AT THE BEGINNING OF THE HARDWARE TEST YOU WERE IN. IF THE ERROR DOES NOT RECUR, THE EXECUTION WILL FLOW RIGHT ON THRU TO THE NEXT ERROR OR TO END OF PASS.

IF AT END OF PASS YOU WANT TO RUN THE DIAGNOSTIC AGAIN, YOU HAVE TWO CHOICES:

1. START
2. RESTART

YOU WOULD CHOOSE ONE, DEPENDING ON WHETHER YOU WANTED TO ANSWER THE HARDWARE QUESTIONS AGAIN.

THE FULL PRINT-OUT FROM THE ABOVE DIALOGUE MIGHT LOOK LIKE THIS
(O=OPERATOR, D=DIAGNOSTIC):

	BY WHOM ENTERED: -----
.R CZRLLB	O
DRS LOADED	D
DIAG. RUN-TIME SERVICES REV. C APR-79	D
CZRLB-B-0	D
CZRLB VERIFIES INTERCHANGEABILITY OF CARTRIDGES BETWEEN DRIVES	D
UNIT IS RL01, RL02	D
DR>STA/PASS:1/FLAGS:HOE	D,O
CHANGE HW (L) ? Y	D,O
# UNITS (D) ? 2	D,O
UNIT 0	D
BUS ADDRESS (O) 174400 ?	D,O
VECTOR (O) 160 ?	D,O
DRIVE (O) 0 ?	D,O
DRIVE TYPE = RL01 (L) Y ?	D,O
UNIT 1	D
BUS ADDRESS (O) 174400 ?	D,O
VECTOR (O) 160 ?	D,O
DRIVE (O) 0 ? 1	D,O
DRIVE TYPE = RL01 (L) ? N	D,O (N=RL02)
CZRLB HRD ERR 00004 TST 003 SUB 002 PC:004130 ERR HLT	
DR>PRO/FLAGS:IER:LOE:HOE=0	D,O
***** AT THIS POINT THE DIAGNOSTIC IS LOOPING ON THE ERROR WITHOUT PRINTING ANYTHING. YOU CAN SCOPE THE ERROR UNTIL YOU HAVE LOCATED IT, THEN ^C OUT. *****	
^C	O
DR>CON/FLAGS:HOE:IER:LOE=0	D,O


```
CZRL EOP 1          D
^C
DR>RESTART/PASS:1   D,0
```

```
-----
-----
-----
-----
```

2.2 CHAIN MODE OPERATION

CHAIN MODE OPERATION CONSISTS OF THE SEQUENTIAL EXECUTION OF PROGRAMS WITHOUT OPERATOR INTERVENTION. ONLY PROGRAMS THAT HAVE BEEN MODIFIED TO RUN IN CHAIN MODE CAN BE CHAINED. CHAINABLE PROGRAMS ARE IDENTIFIED IN THE DIRECTORY BY A BIC EXTENSION.

TO RUN CHAIN MODE, THE XXDP+ MONITOR USES AN ASCII FILE (KNOWN AS A CHAIN FILE) LISTING THE PROGRAMS TO BE RUN AND THE NUMBER OF PASSES EACH PROGRAM SHOULD RUN. THIS FILE MUST BE ON THE SYSTEM DEVICE.

A CHAIN FILE MAY BE GENERATED BY USE OF THE XTECO TEXT EDITOR. THIS FILE MUST HAVE A CCC EXTENSION. THE CHAIN FILE MAY CONTAIN ANY OF THE COMMANDS SUPPORTED BY THE XXDP+ MONITOR. THE COMMANDS IN THE ASCII FILE ARE EXECUTED IN THE ORDER IN WHICH THEY ARE ENCOUNTERED.

TO EXECUTE A CHAIN FILE THE USER TYPES:

```
C FILNAM <CR> OR
C FILNAM/QV <CR>
```

IN THE FIRST CASE THE PASS COUNT SPECIFIED IN THE CHAIN FILE IS USED BY THE XXDP+ MONITOR TO DETERMINE THE NUMBER OF PASSES TO EXECUTE EACH PROGRAM. IN THE SECOND CASE THE PASS COUNT IS NOT USED AND EACH PROGRAM IS EXECUTED ONLY ONCE. THE /QV SWITCH PROVIDES A SINGLE EXECUTION MODE OF OPERATION OF QUICK VERIFY.

WHEN PROGRAMS ARE RUN IN CHAIN MODE, THE SOFTWARE SWITCH REGISTER SHOULD BE SET TO 00000. THE XXDP+ MONITOR PRINTS EACH COMMAND TAKEN FROM THE CHAIN FILE AND THEN EXECUTES THE COMMAND. WHEN THE LAST COMMAND OTHER THAN ANOTHER C COMMAND HAS BEEN EXECUTED THE XXDP+ MONITOR TERMINATES CHAIN MODE AND TYPES A PROMPT (.). IT IS READY TO ACCEPT ANOTHER COMMAND FROM THE CONSOLE. IF THE LAST COMMAND IS ANOTHER C COMMAND, THE CHAIN MODE WILL CONTINUE AND THE CHAIN FILE SPECIFIED BY THIS NEW C COMMAND WILL BE USED.

IF THE USER WISHES TO TERMINATE CHAIN MODE BEFORE ITS NORMAL TERMINATION HE MAY DO SO BY TYPING A CONTROL/C. HOWEVER, THE MONITOR WILL NOT ABORT THE CHAIN MODE UNTIL IT RECEIVES PROGRAM CONTROL FROM THE PROGRAM CURRENTLY RUNNING.

2.3 DETAILS OF COMMANDS AND SYNTAX2.3.1 TABLE OF COMMAND VALIDITY

THERE ARE FOUR WAYS OF ENTERING DIAGNOSTIC COMMAND MODE, AND DIFFERENT SUBSETS OF THE DIAG COMMAND SET ARE AVAILABLE WITH EACH:

<u>HOW ENTERED</u>	<u>LEGAL COMMANDS</u>
1. OPERATOR ENTERED 'RUN DIAG'	START PRINT DISPLAY FLAGS ZFLAGS EXIT
2. DIAGNOSTIC HAS FINISHED ALL ITS REQUESTED PASSES	START RESTART PRINT DISPLAY FLAGS ZFLAGS EXIT
3. OPERATOR INTERRUPTED THE DIAGNOSTIC WITH CTRL/C	START RESTART CONTINUE PRINT DISPLAY FLAGS ZFLAGS EXIT
4. AN ERROR WAS ENCOUNTERED WITH THE HOE FLAG SET SET	START RESTART CONTINUE PROCEED PRINT DISPLAY FLAGS ZFLAGS EXIT

2.3.2 COMMAND SYNTAX

STA(RT)/TESTS:TEST-LIST/PASS:PASS-CNT/FLAGS:FLAG-LIST/EOP:EOP-INCR

THE DIAGNOSTIC IN CORE IS EXECUTED IN ACCORDANCE WITH THE SWITCHES SPECIFIED. THE MESSAGE "# UNITS?" IS PRINTED. THE START COMMAND MAY BE ISSUED WHEN DIAGNOSTIC COMMAND MODE HAS BEEN ENTERED VIA ONE OF THE FOLLOWING: A) OPERATOR TYPED "RUN DIAGNOSTIC" B) DIAGNOSTIC FINISHED EXECUTING C) ERROR WAS ENCOUNTERED WITH HOE FLAG SET D) OPERATOR ENTERED CONTROL/C. AFTER THE OPERATOR RESPONDS TO "# UNITS?", THE HARDWARE DIALOGUE IS INITIATED. WHEN IT IS COMPLETED, THE QUESTIONS "CHANGE SW?" IS ISSUED, AND THE ANSWERS, IF GIVEN, BECOME THE NEW DEFAULTS. THEREFORE IT IS NECESSARY TO RELOAD THE PROGRAM IN ORDER TO RETURN TO THE LOAD DEFAULTS.

THE SWITCH ARGUMENTS ARE AS FOLLOWS:

"TEST-LIST" IS A SEQUENCE OF DECIMAL NUMBERS (1:2 ETC.) OR RANGES OF DECIMAL NUMBERS (1-5:8-10 ETC.) THAT SPECIFY THE TESTS TO BE EXECUTED. THE NUMBERS ARE SEPARATED BY COLONS. THE NUMBERS RANGE FROM 1 TO THE LARGEST TEST NUMBER IN THE DIAGNOSTIC. THEY MAY BE SPECIFIED IN ANY ORDER. TESTS WILL BE EXECUTED IN NUMERICAL ORDER REGARDLESS OF THE ORDER OF SPECIFICATION. THE DEFAULT IS TO EXECUTE ALL TESTS.

"PASS-CNT" IS A DECIMAL NUMBER INDICATING THE DESIRED NUMBER OF PASSES. A PASS IS DEFINED AS THE EXECUTION OF THE FULL DIAGNOSTIC (ALL SELECTED TESTS) AGAINST ALL UNITS SUBMITTED. THE DEFAULT IS NON-ENDING TEST EXECUTION. "FLAG-LIST" IS A SEQUENCE OF ELEMENTS OF THE FORM <FLAG>, <FLAG=1>, OR <FLAG=0>, SEPARATED BY COLONS, WHERE <FLAG> HAS ONE OF THE FOLLOWING VALUES:

HOE HALT ON ERROR, CAUSING COMMAND MODE TO BE ENTERED WHEN AN ERROR IS ENCOUNTERED

LOE LOOP ON ERROR, CAUSING THE DIAGNOSTIC TO LOOP CONTINUOUSLY WITHIN THE SMALLEST DEFINED BLOCK OF CODING (SEGMENT, SUBTEST, OR TEST) CONTAINING THE ERROR

IER INHIBIT ERROR REPORTING

IBE INHIBIT BASIC ERROR REPORTS

IXE INHIBIT EXTENDED ERROR REPORTS

PRI DIRECT ALL MESSAGES TO A LINE PRINTER

PNT PRINT NUMBER OF TEST BEING EXECUTED

BOE BELL ON ERROR
UAM RUN IN UNATTENDED MODE, BYPASSING MANUAL INTERVENTION TESTS
ISR INHIBIT STATISTICAL REPORTS
IDU INHIBIT DROPPING OF UNITS BY DIAGNOSTIC
ADR EXECUTE AUTODROP CODE
LOT LOOP ON TEST
EVL EVALUATE

THE FLAGS NAMED OR EQUATED TO 1 ARE SET, THOSE EQUATED TO 0 ARE CLEARED. A FLAG NOT SPECIFIED IS CLEARED. IF THE FLAGS SWITCH IS NOT GIVEN ALL FLAGS ARE CLEARED.

'EOP-INCR' IS A DECIMAL NUMBER INDICATING HOW OFTEN (IN TERMS OF PASSES) IT IS DESIRED THAT THE END OF PASS MESSAGE BE PRINTED. THE DEFAULT IS AT THE END OF EVERY PASS.

RES(TART)/TEST:TEST-LIST/PASS:PASS-CNT/FLAGS:FLAG-LIST/EOP:EOP-INCR/UNITS:UNIT-LIST

THE DIAGNOSTIC IN CORE IS EXECUTED IN ACCORDANCE WITH THE SWITCHES SPECIFIED. HOWEVER, NEW 'P-TABLES' ARE NOT BUILT. INSTEAD, THE ONES IN CORE ARE USED.

THE QUESTION 'CHANGE SW?' IS ASKED AND THE ANSWERS GIVEN BECOME THE NEW DEFAULTS. THE COMMAND MAY BE ISSUED WHEN COMMAND MODE HAS BEEN ENTERED VIA A) DIAGNOSTIC IS FINISHED B) HALT ON ERROR C) CONTROL/C.

THE SWITCH ARGUMENTS ARE AS IN THE START COMMAND EXCEPT:

1. 'UNIT-LIST' IS A SEQUENCE OF LOGICAL UNIT NUMBERS RANGING FROM 1 THRU N (N = NUMBER OF UNITS BEING TESTED) SPECIFYING WHICH UNITS ARE TO BE TESTED. THE LOGICAL UNIT NUMBER DESIGNATES THE POSITION OF THE P-TABLE IN CORE, ACCORDING TO THE ORDER IN WHICH THEY WERE BUILT. THE UNITS SPECIFIED MUST NOT HAVE BEEN DROPPED BY THE OPERATOR DROP COMMAND. THE UNIT-LIST DEFAULTS TO 'ALL THAT HAVE NOT BEEN DROPPED BY OPERATOR COMMAND'. THE EFFECT OF THE UNIT-LIST LASTS UNTIL THE NEXT START (WHERE IT IS AUTOMATICALLY RESET TO 'ALL') OR THE NEXT RESTART.
2. ALL UNSPECIFIED FLAG SETTINGS ARE UNCHANGED.

CON(TINUE)/PASS:<PASS-CNT/FLAGS:<FLAG-LIST>

COMMAND MODE MUST HAVE BEEN ENTERED DUE TO A HALT ON ERROR OR A CONTROL/C. THE EFFECT OF THE COMMAND IS TO GO TO THE BEGINNING OF THE TEST THAT WAS BEING EXECUTED WHEN THE HALT OR CONTROL/C TOOK PLACE. SOFTWARE DIALOGUE MAY OPTIONALLY BE RE-EXECUTED. HARDWARE PARAMETERS MAY NOT BE CHANGED.

THE SWITCH ARGUMENTS ARE AS IN THE START COMMAND EXCEPT:

1. DEFAULT FOR PASS-CNT IS THE UNSATISFIED PASS-CNT FROM THE PREVIOUS START OR RESTART
2. UNSPECIFIED FLAG SETTINGS ARE UNCHANGED

PRO(CCEED)/FLAGS:<FLAG-LIST>

COMMAND MODE MUST HAVE BEEN ENTERED VIA A HALT ON ERROR. THE EFFECT OF THE COMMAND IS TO BEGIN EXECUTION AT THE LOCATION FOLLOWING THE ERROR CALL. NEITHER HARDWARE NOR SOFTWARE PARAMETERS MAY BE ALTERED.

THE SWITCH ARGUMENTS ARE THE SAME AS THE START COMMAND EXCEPT:

1. UNSPECIFIED FLAG SETTINGS ARE UNCHANGED

EXIT

RETURN TO XXDP+ PROMPT MODE.

DRO(P)/UNITS:UNIT-LIST

THE UNITS SPECIFIED ARE DROPPED FROM TESTING UNTIL THEY ARE ADDED BACK OR UNTIL A START COMMAND IS GIVEN. A DROP CANNOT BE FOLLOWED BY A PROCEED.

THERE IS ALSO A 'DROP' MACRO INTERNAL TO THE DIAGNOSTIC, WHICH GIVES THE FACILITY OF AUTO-DROPPING. THE DURATION OF A PROGRAM DROP, HOWEVER, IS ONLY UNTIL THE NEXT START OR RESTART.

ADD/UNITS:UNIT-LIST

THE UNITS SPECIFIED ARE ADDED BACK (THEY MUST HAVE BEEN PREVIOUSLY DROPPED BY THE DROP COMMAND) TO THE TEST SEQUENCE. AN ADD CANNOT BE FOLLOWED BY A PROCEED.

PRI(NT)

ALL STATISTICS TABLES ACCUMULATED BY THE DIAGNOSTIC ARE PRINTED. THE ISR (INHIBIT STATISTICAL REPORTING) FLAG IS CLEARED.

DIS(PLAY)/UNITS:<UNIT-LIST>

THE HARDWARE P-TABLES FOR ALL UNITS UNDER TEST ARE PRINTED OUT IN THE FORMAT IN WHICH THEY WERE ENTERED. ANY UNITS THAT WERE DROPPED BY THE OPERATOR 'DROP' COMMAND ARE SO DESIGNATED.

FLA(GS)

THE CURRENT SETTINGS OF ALL FLAGS ARE PRINTED.

ZFL(AGS)

ALL FLAGS ARE CLEARED.

2.4 EXTENDED P-TABLE DIALOGUE

THE FULL CAPABILITY OF THE HARDWARE DIALOGUE IS REVEALED BY THE FOLLOWING DISCUSSION OF WHAT HAPPENS INTERNALLY.

AS SOON AS THE QUESTION '# UNITS?' IS ANSWERED (WITH THE NUMBER N), SPACE IN CORE IS ALLOCATED FOR 'N' P-TABLES. ALL OF THE P-TABLES ARE OF THE SAME FORMAT, AND THERE IS A ONE-TO-ONE CORRESPONDENCE BETWEEN THE HARDWARE PARAMETER QUESTIONS AND THE SLOTS IN THE P-TABLE FORMAT.

IN GIVING A STRING OF VALUES, COMMAS WITHOUT INTERVENING VALUES MAY BE USED TO INDICATE A REPETITION OF THE LAST NAMED VALUE.

A STRING OF VALUES MAY BE GIVEN AS A RANGE (6-10 FOR EXAMPLE). IF THE VALUES REPRESENT PURE NUMERICAL DATA, THIS SAMPLE RANGE TRANSLATES TO THE STRING 6,7,8,9,10 (AN INCREMENT OF 1). IF THE VALUES ARE ADDRESSES, THE SAMPLE RANGE TRANSLATES TO THE STRING 6,8,10 (AN INCREMENT OF 2).

NOW LET US SEE HOW WE COULD USE THESE CAPABILITIES TO CONSTRUCT A SET OF P-TABLES. ASSUME THAT WE HAVE 8 RL UNITS, AND THAT THERE ARE FIVE (5) HARDWARE PARAMETERS FOR EACH (5 SLOTS IN THE P-TABLE, 5 HARDWARE QUESTIONS IN THE DIALOGUE).

FOLLOWING IS THE DIALOGUE FOR THIS 8 RLOX DRIVE SYSTEM. THIS SYSTEM HAS TWO (2) RL11 TYPE CONTROLLERS ALL TO BE SET AT 'BR LEVEL' 5. THE FIRST 4 DRIVES ARE RLO1'S AND THE LAST 4 DRIVES ARE RLO2'S (ON THE SECOND CONTROLLER):

UNITS (D) ? 8

UNIT 0
BUS ADDRESS (O) 174400 ?
VECTOR (O) 160 ?
DRIVE (O) 0 ? 0-3
DRIVE TYPE = RLO1 (L) Y ?

UNIT 4
BUS ADDRESS (O) 174400 ? 175400
VECTOR (O) 160 ? 164
DRIVE (O) 0 ? 0-3
DRIVE TYPE = RLO1 (L) Y ? N

THE FIRST TIME THRU THE P-TABLE QUESTIONS THE DEFAULT VALUES ARE USED FOR THE CSR ADDRESS OF THE CONTROLLER (QUESTION #1), THE CONTROLLER VECTOR ASSIGNMENT (QUESTION #2), AND THE DRIVE TYPE (QUESTION #4). THE ACTUAL UNIT NUMBERS OF THE RLO1'S FOR QUESTION #3 WAS ASSIGNED 0 THRU 3 FOR THE FIRST 4 P-TABLE SLOTS.

THE SECOND TIME THRU THE P-TABLE QUESTIONS THE FIRST QUESTION WAS ANSWERED TO REFLECT THE CHANGE IN CSR ADDRESS FOR THE RLO2 CONTROLLER (175400). THE SECOND CONTROLLER'S VECTOR WAS ALSO CHANGED TO 164 IN QUESTION #2. THE RLO2 TEST UNIT NUMBERS WERE ASSIGNED VALUES 0 TO 3 IN QUESTION #3 AND THE DRIVE TYPE WAS SET FOR RLO2'S FOR THE REMAINING 4 UNITS IN QUESTION #4.

2.5 HARDWARE PARAMETERS

THE FOLLOWING QUESTIONS WILL BE ASKED ON A START COMMAND. THE VALUE LOCATED TO THE LEFT OF THE QUESTION MARK IS THE DEFAULT VALUE THAT WILL BE TAKEN ON A CARRIAGE RETURN RESPONSE.

BUS ADDRESS (O) 174400?

ANSWER WITH THE BUS ADDRESS OF THE CONTROLLER.

VECTOR (O) 160?

ANSWER WITH THE INTERRUPT VECTOR OF THE CONTROLLER.

DRIVE (O) 0?

ANSWER WITH THE DRIVE(S) CONNECTED TO THE CONTROLLER

DRIVE TYPE = RL01 (L) ?

ANSWER NO (N) IF DRIVE IS AN RL02

2.6 SOFTWARE PARAMETERS

THE FOLLOWING QUESTIONS ARE ASKED IF REQUESTED ON A START, RESTART, OR CONTINUE. THEY ALLOW FLEXABILITY IN THE WAY THE PROGRAM BEHAVES. THE SOFTWARE PARAMETERS GIVE THE PROGRAM FLEXIBILITY IN THE WAY IT RUNS. THE PARAMETERS CAN BE MODIFIED ON A START, RESTART, OR CONTINUE BY ANSWERING (Y)ES TO THE FOLLOWING QUESTION:

'CHANGE S.W. ?'

A YES ANSWER WILL ASK THE FOLLOWING SOFTWARE PARAMETER QUESTIONS, WITH THE PRESENT DEFAULT VALUE PRINTED TO THE LEFT OF THE QUESTION MARK. (THE LAST ANSWER GIVEN IS THE DEFAULT) THE DEFAULT IS TAKEN ON A <CR>. CONTROL Z (^Z) WILL DEFAULT ALL REMAINING QUESTIONS AND START THE TEST.

THERE ARE NO SOFTWARE PARAMETERS.

3.0 ERROR INFORMATION

ERROR INFORMATION IS COMPLETE IN GIVING ALL INFORMATION NECESSARY. ALL REGISTERS ARE GIVEN AS WELL AT TRACK, SECTOR AND DRIVES INVOLVED IN ERROR.

3.1 ERROR REPORTING

ALL ERROR INFORMATION IS PRINTED ON THE CONSOLE DEVICE. ERROR REPORTS ARE AIMED AT BEING SELF EXPLANATORY. THE GENERAL FORMAT IS:

DZRLL XXX ERR YYYYY TST ZZZ SUB PPP PC: RRRRRR

WHERE:

? IS PROGRAM LETTER
XXX IS SFT - SOFT ERROR
HRD - HARD ERROR
DV FAT - DEVICE FATAL ERROR
SYS FAT - SYSTEM FATAL ERROR
YYYYY IS THE ERROR NUMBER
ZZZ IS THE TEST NUMBER
PPP IS THE SUBTEST NUMBER
RRRRRR IS THE PROGRAM LISTING LOCATION

ERRORS GIVE THE REGISTER CONTENTS BEFORE AND AFTER THE ERROR ALONG WITH A ONE LINE DESCRIPTION AND RELEVANT DATA.

EXAMPLE:

ONE LINE DESCRIPTION
(OPTIONAL SECOND LINE)
(OPTIONAL THIRD LINE)
BEFORE CS:XXXXXX BA:XXXXXX DA:XXXXXX MP:XXXXXX
AFTER CS:XXXXXX BA:XXXXXX DA:XXXXXX MP:XXXXXX
OTHER PERTINENT INFORMATION IS GIVEN AT THIS TIME.

REGISTER DESCRIPTIONS CAN BE FOUND IN SECTION 5.0.

ERROR DESCRIPTIONS:

'ERROR READING SECTOR'

ERROR WAS ENCOUNTERED WHILE TRYING TO READ VERIFY THE SECTOR AFTER IT WAS WRITTEN BY THE SAME DRIVE.

'MINIMUM OF TWO DRIVES REQUIRED'

THE PROGRAM REQUIRES AT LEAST TWO DRIVES TO PROVE COMPATABILITY.

'MAXIMUM OF FOUR DRIVES ALLOWED''

THE PROGRAM ONLY ALLOWS A MAXIMUM OF FOUR DRIVES.

'CAN'T FIND FIVE ADJACENT TRACKS''

THE PROGRAM REQUIRES TEN SETS OF FIVE ADJACENT TRACKS AT
PREDETERMINED SPOTS ACROSS THE PACK. IT WAS UNABLE TO FIND FIVE
COMPLETELY GOOD ADJACENT TRACKS IN THE LIMITS GIVEN.

'ERROR WRITING SECTOR''

AN ERROR WAS ENCOUNTERED WHILE TRYING TO WRITE THE GIVEN SECTOR.

'OVERWRITE ERROR''

AN ERROR WAS ENCOUNTERED WHILE TRYING TO READ DATA AFTER AN
OVERWRITE BY ONE DRIVE. BOTH DRIVES INVOLVED ARE GIVEN.

'READ RECOVERY ERROR''

AN ERROR WAS ENCOUNTERED WHILE TRYING TO RECOVER ANOTHER DRIVES
DATA.

'ADJACENT TRACK TEST''

AN ERROR WAS ENCOUNTERED WHILE IN THE ADJACENT TEST PART, A FURTHER
DESCRIPTION IS GIVEN.

3.2 ERROR HALTS

ERROR HALTS ARE SUPPORTED PER DESCRIBED IN THE PREVIOUS SECTION
WITH /FLAG:HOE. THERE ARE NO OTHER HALTS.

4.0 PERFORMANCE AND PROGRESS REPORTS

4.1 PERFORMANCE REPORTS

THIS PROGRAM WILL NOT GIVE ANY PERFORMANCE REPORTS.

4.2 PROGRESS REPORTS

THIS PROGRAM WILL NOT GIVE ANY PROGRESS REPORTS.

5.0 DEVICE INFORMATION TABLES

THE RL11/RLV11 CONTROLLER HAS THE FOLLOWING FOUR(4) REGISTERS FOR CONTROL OF THE SUBSYSTEM.

RLCS - CONTROL AND STATUS REGISTER (XXXXX0)

- BIT 15 - COMPOSITE ERROR
- BIT 14 - DRIVE ERROR
- BIT 13 - NON EXISTANT MEMORY ERROR
- BIT 12 - HEADER NOT FOUND (WITH BIT 10 SET)
- DATA LATE (WITH BIT 10 CLEAR)
- BIT 11 - HEADER CRC (WITH BIT 10 SET)
- DATA CRC (WITH BIT 10 CLEAR)
- BIT 10 - OPERATION INCOMPLETE
- BIT 9/8 - DRIVE SELECT (0-3)
- BIT 7 - CONTROLLER READY
- BIT 6 - INTERRUPT ENABLE
- BIT 5 - EXTENDED BUS ADDRESS (BIT 17)
- BIT 4 - EXTENDED BUS ADDRESS (BIT 16)
- BIT 3-1 - FUNCTION CODE
 - 0 - NOP (PDP-11) MAINT (LSI-11)
 - 1 - WRITE CHECK
 - 2 - GET DRIVE STATUS
 - 3 - SEEK
 - 4 - READ HEADER
 - 5 - WRITE DATA
 - 6 - READ DATA
 - 7 - READ WITHOUT HEADER COMPARE

BIT 0 - DRIVE READY

RLBA - BUS ADDRESS REGISTER (XXXXX2)

BITS 15-1 BUS ADDRESS OF DATA TRANSFER
BIT 0 SHOULD BE 0

RLDA - DISK ADDRESS REGISTER (XXXXX4)

FOR READ/WRITE FUNCTIONS

BIT 15-7 - CYLINDER ADDRESS FOR TRANSFER
BIT 6 - SURFACE FOR TRANSFER
BIT 5-0 - SECTOR FOR TRANSFER (1-40.)

FOR SEEK FUNCTION

BIT 15-7 - DIFFERENCE TO NEW CYLINDER
BIT 6-5 - MUST BE ZERO (0)
BIT 4 - SURFACE (0=UPPER, 1=LOWER)
BIT 3 - MUST BE ZERO (0)
BIT 2 - SEEK DIRECTION(1=IN / 0=OUT)
BIT 1 - MUST BE ZERO (0)
BIT 0 - MUST BE ONE (1)

FOR GET STATUS FUNCTION

BIT 15-4 - IGNORED SHOULD BE ZERO (0)
BIT 3 - DRIVE RESET
BIT 2 - MUST BE ZERO (0)
BIT 1 - MUST BE ONE (1)
BIT 0 - MUST BE ONE (1)

RLMP - MULTIPURPOSE REGISTER

FOR READ/WRITE FUNCTION

BIT 15 - 0 - WORD COUNT (TWO'S COMPLIMENT)

FOR READ HEADER FUNCTION

BIT 15-0 - DISK HEADER OF SECTOR (FIRST READ)
- ZERO WORD (SECOND READ)
- HEADER CRC (THIRD READ)

FOR GET STATUS FUNCTION

HAS DRIVE STATUS

BIT 15 - WRITE DATA ERROR
BIT 14 - CURRENT HEAD ERROR (CHE)
BIT 13 - WRITE LOCK STATUS (WL)
BIT 12 - SEEK TIME OUT (SKTO)
BIT 11 - SPIN ERROR (SPE)
BIT 10 - WRITE GATE ERROR (WGE)

BIT 9 - VOLUME CHECK (VC)
BIT 8 - DRIVE SELECT ERROR (DSE)
BIT 7 - DRIVE TYPE IS RL02 IF SET
BIT 6 - SURFACE (0=UPPPER, 1=LOWER)
BIT 5 - COVER OPEN
BIT 4 - HEADS HOME
BIT 3 - BRUSHES HOME
BIT 2-0 - STATE BITS
0 - LOAD STATE
1 - SPIN UP
2 - BRUSH CYCLE
3 - LOAD HEADS
4 - SEEK - TRACK COUNTING
5 - SEEK - LINEAR MODE
6 - UNLOAD HEADS
7 - SPIN DOWN

6.0 TEST SUMMARIES

THE FOLLOWING IS A BRIEF DESCRIPTION OF THE WAY THE PROGRAM EXECUTES. THE PROGRAM WILL CHECK COMPATIBILITY BETWEEN 2 - 4 DRIVES USING THE SAME RL01K CARTRIDGE OR SAME RL02K CARTRIDGE. THE PROGRAM WILL ASK THE OPERATOR TO SEQUENCE THE PACK BETWEEN THE DRIVES GIVEN IN THE FOLLOWING MANNER.

PLACE PACK IN DRIVE N ON CONTROLLER X AND LOAD
UNLOAD DRIVE N ON CONTROLLER X
PLACE PACK IN DRIVE N+1 ON CONTROLLER X AND LOAD
UNLOAD DRIVE N+1 ON CONTROLLER X
ETC.....

THE PROGRAM WILL SEQUENCE IN THE ORDER THAT WAS GIVEN IN THE HARDWARE QUESTIONS. I.E.

DRIVE ? 0,1,2,3
PROGRAM WILL SEQUENCE 0,1,2,3,2,1,0
DRIVE ? 1,0,3,2
PROGRAM WILL SEQUENCE 1,0,3,2,3,0,1

WHEN THE FIRST DRIVE IS LOADED THE PROGRAM WILL ATTEMPT TO FIND TEN SETS OF FIVE ADJACENT TRACKS AT PREDETERMINED SPOTS THAT CONTAIN NO BAD SECTORS USING THE BAD SECTOR FILE. THE 10 SPOTS ARE: ON BOTH SURFACES, INNER, OUTER, MIDDLE, ONE QUARTER AND THREE QUARTERS. AFTER THIS IS DONE THE OVERWRITE TEST IS PREPARED (FIRST DRIVE CAN'T OVERWRITE) AS WELL AS THE ADJACENT TEST. AS THE PACK IS CYCLED BETWEEN DRIVES THE FOLLOWING CHECKS ARE MADE:

EACH DRIVE CAN OVERWRITE EACH OTHER DRIVE

EACH DRIVE CAN RECOVER EACH OTHERS DATA

EACH DRIVE CAN WRITE ADJACENT TO EVERY OTHER DRIVE WITHOUT DISTURBING THE OTHER'S DATA.

READS AND WRITES TAKE PLACE AFTER SEEKS FROM BOTH DIRECTIONS.

ADJACENT WRITES TAKE PLACE TO BOTH SIDES OF EACH WRITE

TESTS ARE PERFORMED AT ALL TEN SPOTS ACROSS THE PACK.

@

8	MACRO DEFINITIONS
36	GLOBAL EQUATES SECTION
96	GLOBAL DATA SECTION
268	GLOBAL TEXT SECTION
300	GLOBAL ERROR REPORT SECTION
411	INITIALIZATION SECTION
734	GLOBAL SUBROUTINES SECTION


```
1 .TITLE CZRLB0 RL01/02 DRIVE COMPAT
2 .ENABLE AMA
3 .ENABLE ABS
4 .MCALL SVC
5 002000
6
7
8 .SBTTL MACRO DEFINITIONS
9
10 .MACRO WAITUS ARG ;MACRO MICRO-SECOND WAIT
11 MOV ARG,XDELAY ;SAVE ARGUMENT
12 JSR PC,TIME ;CALL TIMING ROUTING
13 .ENDM
14
15 .MACRO WAITMS ARG ;MACRO MILLI-SECOND WAIT
16 MOV ARG,YDELAY ;SAVE ARGUMENT
17 JSR PC,XTIME ;CALL TIMING ROUTINE
18 .ENDM
19
20 .NLIST CND,MD,ME
21
22 002000
23 SVC
24 SVCINS=0
25 SVCTAG=0
26 002000
27 POINTER NONE
28 002000
29 002000
(4) 002000 103
(4) 002001 132
(4) 002002 122
(4) 002003 114
(4) 002004 114
(6) 002005 000
(6) 002006 000
(5) 002007 000
(4) 002010 102
(4) 002011 060
(4) 002012 000000
(4) 002014 000000
(4) 002016 033250
(4) 002020 000000
(4) 002022 022252
(4) 002024 000000
(4) 002026 033370
(4) 002030 000000
(4) 002032 000000
(4) 002034 000001
(4) 002036 000000
(4) 002040 022264
(4) 002042 000000
(4) 002044 000000
(4) 002046 000000
(4) 002050 003
(3) 002051 003
BGNMOD MDHEDR
HEADER CZRLB,0,0,1
.ASCII /C/
.ASCII /Z/
.ASCII /R/
.ASCII /L/
.BYTE 0
.BYTE 0
.BYTE 0
.ASCII /B/
.ASCII /O/
.WORD 0
.WORD 0
.WORD L$HARD
.WORD 0
.WORD L$HW
.WORD 0
.WORD L$LAST
.WORD 0
.WORD 0
.WORD 1
.WORD 0
.WORD L$DISPATCH
.WORD 0
.WORD 0
.WORD 0
.BYTE C$REVISION
.BYTE C$EDIT
```



```
(4) 002052 000000 .WORD 0
(5) 002054 000000 .WORD 0
(4) 002056 000000 .WORD 0
(4) 002060 002222 .WORD L$DVTYP
(4) 002062 000000 .WORD 0
(4) 002064 000000 .WORD 0
(4) 002066 000000 .WORD 0
(4) 002070 000000 .WORD 0
(4) 002072 000000 .WORD 0
(4) 002074 000000 .WORD 0
(4) 002076 002122 .WORD L$DESC
(4) 002100 104035 EMT E$LOAD
(4) 002102 000000 .WORD 0
(4) 002104 022266 .WORD L$INIT
(4) 002106 023714 .WORD L$CLEAN
(4) 002110 023710 .WORD L$AUTO
(4) 002112 022242 .WORD L$PROT
(4) 002114 000000 .WORD 0
(4) 002116 000000 .WORD 0
(4) 002120 000000 .WORD 0
30 002122 ENDMOD
```

```
31
32 002122 DESCRIPT <CZRL VERIFIES INTERCHANGEABILITY OF CARTRIDGES BETWEEN DRIVES>
(3) 002122 055103 046122 020114 .ASCIZ /CZRL VERIFIES INTERCHANGEABILITY OF CARTRIDGES BETWEEN DRIVES/
(3) 002130 042526 044522 044506
(3) 002136 051505 044440 052116
(3) 002144 051105 044103 047101
(3) 002152 042507 041101 046111
(3) 002160 052111 020131 043117
(3) 002166 041440 051101 051124
(3) 002174 042111 042507 020123
(3) 002202 042502 053524 042505
(3) 002210 020116 051104 053111
(3) 002216 051505 000
```

```
(2) 002222 .EVEN
33
34 002222 DEVTYP <RL01,RL02>
(3) 002222 046122 030460 051054 .ASCIZ /RL01,RL02/
(3) 002230 030114 000062
(2) .EVEN
```

```
35
36 .SBTTL GLOBAL EQUATES SECTION
37
```

```
38 ;DEFINITIONS
39
```

```
40 002234 BGNMOD GLBEQAT
41
```

```
42 002234 EQUALS
```

```
(1) ; BIT DIFINITIONS
(1)
(1) BIT15== 100000
(1) 100000 BIT14== 40000
(1) 040000 BIT13== 20000
(1) 020000 BIT12== 10000
(1) 010000 BIT11== 4000
(1) 004000
```



```
(1) 002000 BIT10== 2000
(1) 001000 BIT09== 1000
(1) 000400 BIT08== 400
(1) 000200 BIT07== 200
(1) 000100 BIT06== 100
(1) 000040 BIT05== 40
(1) 000020 BIT04== 20
(1) 000010 BIT03== 10
(1) 000004 BIT02== 4
(1) 000002 BIT01== 2
(1) 000001 BIT00== 1
(1)
(1) 001000 ; BIT9== BIT09
(1) 000400 ; BIT8== BIT08
(1) 000200 ; BIT7== BIT07
(1) 000100 ; BIT6== BIT06
(1) 000040 ; BIT5== BIT05
(1) 000020 ; BIT4== BIT04
(1) 000010 ; BIT3== BIT03
(1) 000004 ; BIT2== BIT02
(1) 000002 ; BIT1== BIT01
(1) 000001 ; BIT0== BIT00
(1)
(1) ;
(1) ; EVENT FLAG DEFINITIONS
(1) ; EF32:EF17 RESERVED FOR SUPERVISOR TO PROGRAM COMMUNICATION
(1) ;
(1) 000040 EF.START== 32. ; START COMMAND WAS ISSUED
(1) 000037 EF.RESTART== 31. ; RESTART COMMAND WAS ISSUED
(1) 000036 EF.CONTINUE== 30. ; CONTINUE COMMAND WAS ISSUED
(1) 000035 EF.NEW== 29. ; A NEW PASS HAS BEEN STARTED
(1) 000034 EF.PWR== 28. ; A POWER-FAIL/POWER-UP OCCURRED
(1) ;
(1) ;
(1) ; PRIORITY LEVEL DEFINITIONS
(1) ;
(1) 000340 PRI07== 340
(1) 000300 PRI06== 300
(1) 000240 PRI05== 240
(1) 000200 PRI04== 200
(1) 000140 PRI03== 140
(1) 000100 PRI02== 100
(1) 000040 PRI01== 40
(1) 000000 PRI00== 0
(1) ;
(1) ; OPERATOR FLAG BITS
(1) ;
(1) 000004 EVL== 4
(1) 000010 LOT== 10
(1) 000020 ADR== 20
(1) 000040 IDU== 40
(1) 000100 ISR== 100
(1) 000200 UAM== 200
(1) 000400 BOE== 400
(1) 001000 PNT== 1000
(1) 002000 PRI== 2000
(1) 004000 IXE== 4000
```



```
(1) 010000 IBE== 10000
(1) 020000 IER== 20000
(1) 040000 LOE== 40000
(1) 100000 HOE== 100000
43
44 000000 CS=0 ;CONTROL AND STATUS OFFSET
45 000002 BA=2 ;BUSS ADDRESS OFFSET
46 000004 DA=4 ;DISK ADDRESS OFFSET
47 000006 MP=6 ;MULTI PURPOSE OFFSET
48
49 ;CONSTANT OFFSETS FOR INDIVIDUAL DRIVE BUFFERS
50
51 000000 CSR=0 ;CONTROLLER ADDRESS
52 000002 VEC=2 ;VECTOR OF CONTROLLER
53 000004 DSB=4 ;DRIVE SELECT
54 000006 PAT=6 ;PATTERN UNIQUE TO DRIVE
55
56
57 000001 DRDY=BIT0 ;DRIVE READY
58 000100 INTEN=BIT6 ;INTERRUPT ENABLE
59 100000 ERR=BIT15 ;COMPOSITE ERROR
60 040000 DERR=BIT14 ;DRIVE ERROR
61 020000 NXM=BIT13 ;NON-EXISTANT MEMORY ERROR
62 010000 DLT=BIT12 ;DATA LATE
63 004000 DCRC=BIT11 ;DATA CRC ERROR
64 004000 HCRC=BIT11 ;HEADER CRC ERROR
65 010000 HNF=BIT12 ;HEADER NOT FOUND ERROR
66 002000 OPI=BIT10 ;OPERATION INCOMPLETE ERROR
67 000200 CRDY=BIT7 ;CONTROLLER READY
68 000040 BA17=BIT5 ;EXTENDED BUS ADDRESS BIT 17
69 000020 BA16=BIT4 ;EXTENDED BUS ADDRESS BIT 16
70 000002 CRSET=BIT1 ;CONTROLLER RESET FUNCTION CODE
71 000004 GSTAT=BIT2 ;GET DRIVE STATUS FUNCTION CODE
72 000006 SEEK=BIT1!BIT2 ;SEEK FUNCTION CODE
73 000010 RDHDR=BIT3 ;READ HEADER FUNCTION CODE
74 000012 WRITE=BIT3!BIT1 ;WRITE FUNCTION CODE
75 000014 READ=BIT3!BIT2 ;READ FUNCTION CODE
76 000013 DRST=BIT3!BIT1!BIT0 ;DRIVE RESET COMMAND CODE FOR DRIVE COMMAD WORD
77 000003 GSBIT=BIT1!BIT0 ;GET STATUS COMMAND CODE FOR DRIVE COMMAND WORD
78 000001 MK=BIT0 ;MARKER BIT FOR DRIVE COMMAND WORD(SEEK,GET STATUS)
79 000004 SIGN=BIT2 ;DIRECTION FOR SEEK(0=AWAY FROM SPINDLE)
80 000020 SKHS=BIT4 ;HEAD SELECT FOR SEEK
81 000100 HEAD=BIT6 ;HEAD SELECT FOR READ,WRITE,GET STATUS
82
83 ;OFFSET FOR HARDWARE P-TABLE
84
85 000000 CSR= 0
86 000002 VECT= 2
87 000004 TYPDR= 4
88 000006 DRBT= 6
89
90 002234 ENDMOD
91
92
93
```



```
95
96 .SBTTL GLOBAL DATA SECTION
97
98 002234 BGNMOD GLBDAT
99
100 002234 000000 HDRFND: .WORD 0 ;1=HEADER IN BAD SECTOR LIST
101
102 ;HERE IS THE LIST OF TRACKS TO USE FOR THIS TEST
103 ;TRACKS ARE ENTERED BY 'FNDTRK' ROUTINE & 'FIXTRK' ROUTINE
104
105 002236 000000 OUT10: .WORD 0 ;OUTER TRK HEAD 0
106 002240 000000 OUT20: .WORD 0
107 002242 000000 OUT30: .WORD 0
108 002244 000000 OUT40: .WORD 0
109 002246 000000 OUT50: .WORD 0
110 002250 000000 OUT11: .WORD 0 ;OUTER TRK HEAD 1
111 002252 000000 OUT21: .WORD 0
112 002254 000000 OUT31: .WORD 0
113 002256 000000 OUT41: .WORD 0
114 002260 000000 OUT51: .WORD 0
115 002262 000000 OQU10: .WORD 0 ;1ST QUARTER TRK HEAD 0
116 002264 000000 OQU20: .WORD 0
117 002266 000000 OQU30: .WORD 0
118 002270 000000 OQU40: .WORD 0
119 002272 000000 OQU50: .WORD 0
120 002274 000000 OQU11: .WORD 0 ;1ST QUARTER TRK HEAD 1
121 002276 000000 OQU21: .WORD 0
122 002300 000000 OQU31: .WORD 0
123 002302 000000 OQU41: .WORD 0
124 002304 000000 OQU51: .WORD 0
125 002306 000000 MID10: .WORD 0 ;MIDDLE TRK HEAD 0
126 002310 000000 MID20: .WORD 0
127 002312 000000 MID30: .WORD 0
128 002314 000000 MID40: .WORD 0
129 002316 000000 MID50: .WORD 0
130 002320 000000 MID11: .WORD 0 ;MIDDLE TRK HEAD 1
131 002322 000000 MID21: .WORD 0
132 002324 000000 MID31: .WORD 0
133 002326 000000 MID41: .WORD 0
134 002330 000000 MID51: .WORD 0
135 002332 000000 TQU10: .WORD 0 ;3RD QUARTER TRK HEAD 0
136 002334 000000 TQU20: .WORD 0
137 002336 000000 TQU30: .WORD 0
138 002340 000000 TQU40: .WORD 0
139 002342 000000 TQU50: .WORD 0
140 002344 000000 TQU11: .WORD 0 ;3RD QUARTER TRK HEAD 1
141 002346 000000 TQU21: .WORD 0
142 002350 000000 TQU31: .WORD 0
143 002352 000000 TQU41: .WORD 0
144 002354 000000 TQU51: .WORD 0
145 002356 000000 INN10: .WORD 0 ;INNER TRK HEAD 0
146 002360 000000 INN20: .WORD 0
147 002362 000000 INN30: .WORD 0
148 002364 000000 INN40: .WORD 0
149 002366 000000 INN50: .WORD 0
150 002370 000000 INN11: .WORD 0 ;INNER TRK HEAD 1
```


151 002372 000000
152 002374 000000
153 002376 000000
154 002400 000000
155
156
157
158
159
160 002402 000020
161
162
163
164 002442 000170
165
166
167
168
169
170 003022 002242
171 003024 002266
172 003026 002312
173 003030 002336
174 003032 002362
175 003034 002254
176 003036 002300
177 003040 002324
178 003042 002350
179 003044 002374
180
181 003046 152525
182 003050 133333
183 003052 066666
184 003054 155555
185

INN21: .WORD 0
INN31: .WORD 0
INN41: .WORD 0
INN51: .WORD 0
.EVEN

;SECTOR LIST FOR LAST DRIVE WRITTEN
;MAP OF 16 SECTOR DRIVE BITS

SECLST: .BLKW 16.

;BUFFER TABLE FOR 24 X 5 MATRIX USED FOR ADJACENT CYLINDER TESTING.

SECBUF: .BLKW 5*24.

;LIST OF TRACKS USED TO OVERWRITE TEST.
;FIRST FIVE ARE CYLINDER ADDRESSES OF TOP SURFACE.
;LAST FIVE ARE CYLINDER ADDRESSES OF BOTTOM SURFACE.

OVWTRK: OUT30
OQU30
MID30
TQU30
INN30
OUT31
OQU31
MID31
TQU31
INN31

PATLST: .WORD 152525
.WORD 133333
.WORD 066666
.WORD 155555

187					
188	003056	000000	TEM:	.WORD	0
189	003060	000000	T.DRIVE:	.WORD	0
190	003062	000000	FOWR:	.WORD	0
191	003064	000000	FADJ:	.WORD	0
192	003066	000000	TEMP:	.WORD	0
193	003070	000000	LSTCLR:	.WORD	0
194	003072	000000	REASON:	.WORD	0
195	003074	000000	ERFLG:	.WORD	0
196	003076	000000	STFLG:	.WORD	0
197	003100	000000	ADJLOC:	.WORD	0
198	003102	000000	ADJFLG:	.WORD	0
199	003104	000000	ADJDIR:	.WORD	0
200	003106	000000	DRSTAT:	.WORD	0
201	003110	000000	HSFLG:	.WORD	0
202	003112	000000	OSECT:	.WORD	0
203	003114	000000	HEADC1:	.WORD	0
204	003116	000000	DIRC:	.WORD	0
205	003120	000000	SURF:	.WORD	0
206	003122	000000	CYL:	.WORD	0
207	003124	000000	REVSK:	.WORD	0
208	003126	000000	FORSK:	.WORD	0
209	003130	000000	UUT:	.WORD	0
210	003132	000000	SECT:	.WORD	0
211	003134	000000	LSTDRV:	.WORD	0
212	003136	000000	GDATA:	.WORD	0
213	003140	000000	BDATA:	.WORD	0
214	003142	000000	WCOUNT:	.WORD	0
215	003144	000000	SECWRD:	.WORD	0
216	003146	000000	OFFSET:	.WORD	0
217	003150	000000	LSTTRK:	.WORD	0
218	003152	000000	FRTTRK:	.WORD	0
219	003154	000000	PRSTRK:	.WORD	0
220	003156	000000	SURFACE:	.WORD	0
221	003160	000000	TRKFND:	.WORD	0
222	003162	000000	TRKCNT:	.WORD	0
223	003164	000000	E.CS:	.WORD	0
224	003166	000000	E.BA:	.WORD	0
225	003170	000000	E.DA:	.WORD	0
226	003172	000000	E.MP:	.WORD	0
227	003174	000000	E.MP1:	.WORD	0
228	003176	000000	E.MP2:	.WORD	0
229	003200	000000	BCS:	.WORD	0
230	003202	000000	BBA:	.WORD	0
231	003204	000000	BDA:	.WORD	0
232	003206	000000	BMP:	.WORD	0
233	003210	000000	SERNM1:	.WORD	0
234	003212	000000	SERNM2:	.WORD	0
235	003214	000000	ADJTRK:	.WORD	0
236	003216	000000	ADJUUT:	.WORD	0
237	003220	000000	ADJLC2:	.WORD	0
238	003222	000000	ADJLC3:	.WORD	0
239	003224	000000	ADJLC4:	.WORD	0
240	003226	000000	STSEC1:	.WORD	0
241	003230	000000	STSEC:	.WORD	0
242	003232	006000	BUF:	.BLKW	3072.

:LAST CONTROLLER
:DRIVE ERROR REASON
:ERROR FLAG
:PROGRAM START UP FLAG
:TRACK INDEX FOR ADJ. CYL TEST
:FLAG FOR ADJ. STORE OR RETRIEVE
:ADJACENT SEEK DIRECTION

:SURFACE FLAG
:DIRECTION OF SEEK

:REVERSE SEEK
:FORWARD SEEK
:UNIT UNDER TEST
:SECTOR
:LAST DRIVE
:GOOD DATA
:BAD DATA
:WORD COUNT
:SECTOR WORD
:INCREMENT
:LAST TRACK OF SEARCH
:FIRST TRACK OF SEARCH
:PRESENT TRACK
:SURFACE
:TRACK FOUND
:TRACK COUNT
:IMAGE OF CSR
:IMAGE OF BUS ADDRESS
:IMAGE OF DISK ADDRESS
:IMAGE OF MULTI-PURPOSE WORD 1
: " " " " " " 2
: " " " " " " 3

:COMMAND LOADED
:BUS ADDRESS LOADED
:DISK ADDRESS LOADED
:WORD COUNT LOADED
:SERIAL NUMBER OF CATRIDGE
:INSIDE/OUTSIDE FLAG
:UUT FOR "ADJCYL"
:TEMP LOC FOR "ADJCYL"
: " " " " "
:SECTORS TO WRITE "ADJCYL"
:BUFFER FOR 24 SECTOR READS


```
243 017232 000000 XDELAY: .WORD 0 ;DELAY FOR WAIT MICRO-SECOND MACRO
244 017234 000000 YDELAY: .WORD 0 ;DELAY FOR WAIT MILLI-SECOND MACRO
245
246
247 017236 DRBUF: ;DRIVE INFORMATION BUFFERS
248
252
260
(1) 017236 000000 CSR ;CONTROLLER ADDRESS
(1) 017240 000002 VEC ;VECTOR
(1) 017242 000004 DSB ;DRIVE SELECT BITS
(1) 017244 000006 PAT ;PATTERN UNIQUE TO DRIVE
(1)
(1)
(1) 017246 000000 CSR ;CONTROLLER ADDRESS
(1) 017250 000002 VEC ;VECTOR
(1) 017252 000004 DSB ;DRIVE SELECT BITS
(1) 017254 000006 PAT ;PATTERN UNIQUE TO DRIVE
(1)
(1)
(1) 017256 000000 CSR ;CONTROLLER ADDRESS
(1) 017260 000002 VEC ;VECTOR
(1) 017262 000004 DSB ;DRIVE SELECT BITS
(1) 017264 000006 PAT ;PATTERN UNIQUE TO DRIVE
(1)
(1)
(1) 017266 000000 CSR ;CONTROLLER ADDRESS
(1) 017270 000002 VEC ;VECTOR
(1) 017272 000004 DSB ;DRIVE SELECT BITS
(1) 017274 000006 PAT ;PATTERN UNIQUE TO DRIVE
(1)
261
265 017276 000000 ENDBUF: .WORD 0 ;END OF DRIVE BUFFERS
266 017300 ENDMOD
```

```
268 .SBTTL GLOBAL TEXT SECTION
269 017300 BGNMOD GLBTXT
270
271 ;GLOBAL TEXT
272
273
277
278 017300 047503 052116 047522 CNTTOT: .ASCIZ /CONTROLLER TIMED OUT/
279 017325 105 051122 051117 INITWR: .ASCIZ /ERROR ON RECOVERING INITIAL WRITE BY FIRST DRIVE /
280 017407 105 051122 051117 DCKER: .ASCIZ /ERROR ON READ/
281 017425 115 047111 046511 FEW: .ASCIZ /MINIMUM OF TWO DRIVES REQUIRED/
282 017464 040515 044530 052515 MANY: .ASCIZ /MAXIMUM OF FOUR DRIVES ALLOWED/
283 017523 124 051505 020124 NONE: .ASCIZ /TEST ABORTED - CAN'T FIND ANY GOOD SPOTS/
284 017574 051124 044531 043516 OVMES: .ASCIZ /TRYING TO OVERWRITE DRIVE /
285 017627 124 054522 047111 RECMS: .ASCIZ /TRYING TO READ DATA WRITTEN BY DRIVE /
286 017675 103 047101 052047 ERRFND: .ASCIZ /CAN'T FIND FIVE ADJACENT TRACKS/
287 017735 117 042526 053522 OVWER: .ASCIZ /OVERWRITE ERROR/
288 017755 122 040505 020104 RECER: .ASCIZ /READ RECOVERY ERROR/
289 020001 105 051122 051117 FUNERR: .ASCIZ /ERROR IN SEEK OPERATION/
290 020031 115 051511 051440 SKER: .ASCIZ /MIS SEEK ERROR/
291 020050 047506 053522 051101 FWD: .ASCIZ /FORWARD/
292 020060 042522 042526 051522 REV: .ASCIZ /REVERSE/
293 020070 051105 047522 020122 WRIT1: .ASCIZ /ERROR WRITING SECTOR/
294 020115 105 051122 051117 READ1: .ASCIZ /ERROR READING SECTOR/
295 020142 042101 040512 042503 ADJTXT: .ASCIZ /ADJACENT CYLINDER TEST/
296 020172 .EVEN
297
298 020172 ENDMOD
299
300 .SBTTL GLOBAL ERROR REPORT SECTION
301
302 020172 BGNMOD GLBERR
303
304 020172 BGNMSG ERR1
305
306 020172 PRINTB #FRM10,FRTRK,LSTTRK,SURFACE
(10) 020172 013746 003156 MOV SURFACE,-(SP)
(9) 020176 013746 003150 MOV LSTTRK,-(SP)
(8) 020202 013746 003152 MOV FRTRK,-(SP)
(7) 020206 012746 021477 MOV #FRM10,-(SP)
(6) 020212 012746 000004 MOV #4,-(SP)
(3) 020216 010600 MOV SP,R0
(4) 020220 104414 TRAP C$PNTB
(4) 020222 062706 000012 ADD #12,SP
307
308 020226 ENDMMSG
(3) 020226 L10000:
(3) 020226 104423 TRAP C$MSG
309
310 020230 BGNMSG ERR2
311 020230 PRINTB #FRM4,CSR(R4),<B,DSB+1(R4)>
(9) 020230 005046 CLR -(SP)
(9) 020232 156416 000005 BISB DSB+1(R4),(SP)
(8) 020236 016446 000000 MOV CSR(R4),-(SP)
(7) 020242 012746 021200 MOV #FRM4,-(SP)
(6) 020246 012746 000003 MOV #3,-(SP)
```


(3)	020252	010600		MOV	SP,R0	
(4)	020254	104414		TRAP	C\$PNTB	
(4)	020256	062706	000010	ADD	#10,SP	
312	020262	004737	026120	JSR	PC,REGDMP	;REGISTER DUMP ROUTINE
313	020266			ENDMSG		
(3)	020266			L10001:		
(3)	020266	104423		TRAP	C\$MESSG	
314				BGNMSG	ERR3	
315	020270			PRINTB	#FRM4,CSR(R4),<B,DSB+1(R4)>	
316	020270			CLR	-(SP)	
(9)	020270	005046		BISB	DSB+1(R4),(SP)	
(9)	020272	156416	000005	MOV	CSR(R4),-(SP)	
(8)	020276	016446	000000	MOV	#FRM4,-(SP)	
(7)	020302	012746	021200	MOV	#3,-(SP)	
(6)	020306	012746	000003	MOV	SP,R0	
(3)	020312	010600		TRAP	C\$PNTB	
(4)	020314	104414		ADD	#10,SP	
(4)	020316	062706	000010	JSR	PC,REGDMP	;REGISTER DUMP ROUTINE
317	020322	004737	026120	PRINTB	#FRM5,<SURF>,<CYL>,SECT	
318	020326			MOV	SECT,-(SP)	
(10)	020326	013746	003132	MOV	CYL,-(SP)	
(9)	020332	013746	003122	MOV	SURF,-(SP)	
(8)	020336	013746	003120	MOV	#FRM5,-(SP)	
(7)	020342	012746	021241	MOV	#4,-(SP)	
(6)	020346	012746	000004	MOV	SP,R0	
(3)	020352	010600		TRAP	C\$PNTB	
(4)	020354	104414		ADD	#12,SP	
(4)	020356	062706	000012	PRINTB	#FRM16,CSR(R3),<B,DSB+1(R3)>	
319	020362			CLR	-(SP)	
(9)	020362	005046		BISB	DSB+1(R3),(SP)	
(9)	020364	156316	000005	MOV	CSR(R3),-(SP)	
(8)	020370	016346	000000	MOV	#FRM16,-(SP)	
(7)	020374	012746	022030	MOV	#3,-(SP)	
(6)	020400	012746	000003	MOV	SP,R0	
(3)	020404	010600		TRAP	C\$PNTB	
(4)	020406	104414		ADD	#10,SP	
(4)	020410	062706	000010	ENDMSG		
320				L10002:		
321	020414			TRAP	C\$MESSG	
(3)	020414			BGNMSG	ERR4	
(3)	020414	104423		PRINTB	#FRM4,CSR(R4),<B,DSB+1(R4)>	
322				CLR	-(SP)	
323	020416			BISB	DSB+1(R4),(SP)	
324				MOV	CSR(R4),-(SP)	
325	020416			MOV	#FRM4,-(SP)	
(9)	020416	005046		MOV	#3,-(SP)	
(9)	020420	156416	000005	MOV	SP,R0	
(8)	020424	016446	000000	TRAP	C\$PNTB	
(7)	020430	012746	021200	ADD	#10,SP	
(6)	020434	012746	000003	JSR	PC,REGDMP	;REGISTER DUMP ROUTINE
(3)	020440	010600		PRINTB	#FRM5,<SURF>,<CYL>,SECT	
(4)	020442	104414		MOV	SECT,-(SP)	
(4)	020444	062706	000010			
326	020450	004737	026120			
327	020454					
(10)	020454	013746	003132			

(9)	020460	013746	003122	MOV	CYL,-(SP)
(8)	020464	013746	003120	MOV	SURF,-(SP)
(7)	020470	012746	021241	MOV	#FRM5,-(SP)
(6)	020474	012746	000004	MOV	#4,-(SP)
(3)	020500	010600		MOV	SP,R0
(4)	020502	104414		TRAP	C\$PNTB
(4)	020504	062706	000012	ADD	#12,SP
328	020510			PRINTB	#FRM6,REASON,LSTDRV,LSTCLR,LSTDRV
(11)	020510	013746	003134	MOV	LSTDRV,-(SP)
(10)	020514	013746	003070	MOV	LSTCLR,-(SP)
(9)	020520	013746	003134	MOV	LSTDRV,-(SP)
(8)	020524	013746	003072	MOV	REASON,-(SP)
(7)	020530	012746	021310	MOV	#FRM6,-(SP)
(6)	020534	012746	000005	MOV	#5,-(SP)
(3)	020540	010600		MOV	SP,R0
(4)	020542	104414		TRAP	C\$PNTB
(4)	020544	062706	000014	ADD	#14,SP
329	020550			PRINTB	#FRM7,DIRC
(8)	020550	013746	003116	MOV	DIRC,-(SP)
(7)	020554	012746	021331	MOV	#FRM7,-(SP)
(6)	020560	012746	000002	MOV	#2,-(SP)
(3)	020564	010600		MOV	SP,R0
(4)	020566	104414		TRAP	C\$PNTB
(4)	020570	062706	000006	ADD	#6,SP
330				ENDMSG	
331	020574			L10003:	TRAP C\$MSG
(3)	020574				
(3)	020574	104423			
332				BGNMSG	ERR5
333	020576			PRINTB	#FRM4,CSR(R4),<B,DSB+1(R4)>
334	020576			CLR	-(SP)
(9)	020576	005046		BISB	DSB+1(R4),(SP)
(9)	020600	156416	000005	MOV	CSR(R4),-(SP)
(8)	020604	016446	000000	MOV	#FRM4,-(SP)
(7)	020610	012746	021200	MOV	#3,-(SP)
(6)	020614	012746	000003	MOV	SP,R0
(3)	020620	010600		TRAP	C\$PNTB
(4)	020622	104414		ADD	#10,SP
(4)	020624	062706	000010	JSR	PC,REGDMP
335	020630	004737	026120	ENDMSG	
336	020634			L10004:	TRAP C\$MSG
(3)	020634				
(3)	020634	104423			
337				BGNMSG	ERR6
338	020636			PRINTB	#FRM4,CSR(R4),<B,DSB+1(R4)>
339	020636			CLR	-(SP)
(9)	020636	005046		BISB	DSB+1(R4),(SP)
(9)	020640	156416	000005	MOV	CSR(R4),-(SP)
(8)	020644	016446	000000	MOV	#FRM4,-(SP)
(7)	020650	012746	021200	MOV	#3,-(SP)
(6)	020654	012746	000003	MOV	SP,R0
(3)	020660	010600		TRAP	C\$PNTB
(4)	020662	104414		ADD	#10,SP
(4)	020664	062706	000010	JSR	PC,REGDMP
340	020670	004737	026120	PRINTB	#FRM17,R1,E.MP
341	020674				


```
(9) 020674 013746 003172      MOV      E,MP,-(SP)
(8) 020700 010146              MOV      R1,-(SP)
(7) 020702 012746 022115      MOV      #FRM17,-(SP)
(6) 020706 012746 000003      MOV      #3,-(SP)
(3) 020712 010600              MOV      SP,R0
(4) 020714 104414              TRAP     C$PNTB
(4) 020716 062706 000010      ADD      #10,SP
342 020722                      ENDMMSG
(3) 020722                      L10005:
(3) 020722 104423              TRAP     C$MSG
```

:FORMAT STATMENTS

```
351
352 020724 047045 040445 047125 FRM1:  .ASCIZ  /%N%AUNLOAD DRIVE %01%A ON CONTROLLER %06%A AND REMOVE PACK%N/
353 021021      045 022516 050101 FRM2:  .ASCIZ  /%N%APLACE PACK IN DRIVE %01%A ON CONTROLLER %06%A AND LOAD IT%N/
354 021121      045 022516 053501 FRM3:  .ASCIZ  !%N%A WRONG PACK # IS %05%05%A # S/B %05%05%N%N!
355 021200 040445 047503 052116 FRM4:  .ASCIZ  /%A CONTROLLER: %06%A DRIVE: %01%N/
356 021241      045 044101 040505 FRM5:  .ASCIZ  /%A HEAD: %01%A CYL: %Z3%A SECTOR: %Z2%N/
357 021310 052045 047445 022461 FRM6:  .ASCIZ  /%T%01%A ON %06%N/
358 021331      045 051501 042505 FRM7:  .ASCIZ  /%A SEEK DIRECTION: %T%N%A DATA: %N/
359 021371      045 053501 051117 FRM8:  .ASCIZ  !%A WORD: %Z3%A S/B: %06%A WAS: %06%N!
360 021435      045 031504 040445 FRM9:  .ASCIZ  /%D3%A WORDS BAD OUT OF 128 READ%N/
361 021477      045 041101 052105 FRM10: .ASCIZ  /%A BETWEEN %Z3%A - %Z3%A HEAD: %01%N/
362 021543      045 022516 050101 FRM11: .ASCIZ  /%N%A PWR FAIL NOT SUPPORTED%N/
363 021600 040445 042502 047506 FRM12: .ASCIZ  /%A BEFORE CS: %06%A BA: %06%A DA: %06%A MP: %06/
364 021657      045 022516 040501 FRM13: .ASCIZ  /%N%A AFTER CS: %06%A BA: %06%A DA: %06%A MP: %06%N/
365 021742 047045 040445 042040 FRM14: .ASCIZ  /%N%A DRIVE STATUS: %06/
366 021771      045 022516 041501 FRM15: .ASCIZ  /%N%A CAN'T FIND BAD SECTOR FILE/
367 022030 040445 042101 040512 FRM16: .ASCIZ  /%A ADJACENT WRITTEN BY CONTROLLER: %06%A DRIVE: %01%N/
368 022115      045 042501 050130 FRM17: .ASCIZ  /%A EXP'D: %06%A REC'D: %06%N/
369 022151      045 022516 052501 FRM18: .ASCIZ  /%N%A UNLOAD ALL DRIVES TO BE USED%N/
370 022214 047045 040445 042440 ENDPAS: .ASCIZ  /%N%A END OF PASS%N%N/
371
375      022242                      .EVEN
376
377 022242                      ENDMOD
378
```

```
380
381
382 ;LOAD PROTECTION TABLE
383 022242 BGNPROT
384
385 022242 000000 .WORD 0 ;OFFSET OF CSR IN P-TABLE
386 022244 177777 .WORD -1 ;NOT A MASS-BUS DRIVE
387 022246 000006 .WORD 6 ;OFFSET OF DRIVE IN P-TABLE
388
389 022250 ENDPROT
390
391
392 022250 BGNMOD HPTCODE
393 022250 BGNHW
(3) 022250 000004 .WORD L10007-L$HW/2
394 022252 174400 .WORD 174400 ;CSR
395 022254 000160 .WORD 160 ;VECTOR
396 022256 000001 .WORD 1 ;RL01 OR RL02 (RL01=1)
397 022260 000000 .WORD 0 ;DRIVE NUMBER
398 022262 ENDPHW
(3) 022262 L10007:
399
400 022262 ENDMOD
401
402
403 022262 BGNMOD DSPCODE
404
405 022262 DISPATCH 1
(4) 022262 000001 .WORD 1
(6) 022264 032304 .WORD T1
406
407 022266 ENDMOD
408
409
```



```

411          .SBTTL INITIALIZATION SECTION
412
413 022266    BGNMOD  INITCODE
414
415 022266    BGNINIT
416
417 022266    SETPRI  #340
(3) 022266    012700 000340    MOV      #340,R0
(3) 022272    104441    TRAP     C$SPRI
418
419 022274    023727 002012 000002    CMP      L$UNIT,#2          ;MORE THAN TWO
420 022302    002006    BGE     90$                ;YES, OKAY
421
422 022304    ERRSF   19.,FEW
(4) 022304    104454    TRAP     C$ERSF
(5) 022306    000023    .WORD   19
(5) 022310    017425    .WORD   FEW
(5) 022312    000000    .WORD   0
423 022314    000137 023664    JMP      CMPENA          ;CLEAN CODE WHEN < 2 DRIVES
424
425 022320    023727 002012 000004 90$:    CMP      L$UNIT,#4          ;MORE THAN FOUR
426 022326    003406    BLE     91$                ;NO, OKAY
427
428 022330    ERRSF   20.,MANY
(4) 022330    104454    TRAP     C$ERSF
(5) 022332    000024    .WORD   20
(5) 022334    017464    .WORD   MANY
(5) 022336    000000    .WORD   0
429 022340    000137 023664    JMP      CMPENA          ;CLEAN CODE WHEN > 4 DRIVES
430
431 022344    013737 002012 003130 91$:    MOV      L$UNIT,UUT        ;GET NUMBER OF UNITS
432 022352    005001    CLR     R1                 ;INIT P-TABLE
433 022354    012704 017236    MOV      #DRBUF,R4        ;SET UP DRIVE BUFFER
434 022360    012702 003046    MOV      #PATLST,R2       ;GET LIST OF PATTERNS
435 022364    005737 003130    1$:    TST     UUT                ;ANY P-TABLES LEFT?
436 022370    001423    BEQ     END                ;NO, GO TO END
437 022372    GPHARD  R1,R0                    ;GET A P-TABLE
(3) 022372    010100    MOV     R1,R0
(3) 022374    104442    TRAP     C$GPHRD
438 022376    012064 000000    MOV     (R0)+,CSR(R4)      ;GET CSR
439 022402    012064 000002    MOV     (R0)+,VEC(R4)     ;GET VECTOR
440 022406    012037 003060    MOV     (R0)+,T.DRIVE     ;RL01/2 TYPE ... RL01=1
441 022412    011064 000004    MOV     (R0),DSB(R4)      ;GET DRIVE
442 022416    011264 000006    MOV     (R2),PAT(R4)
443 022422    005722    TST     (R2)+
444 022424    005201    INC     R1                 ;NEXT P TABLE
445 022426    005337 003130    DEC     UUT                ;NEXT DRIVE
446 022432    062704 000010    ADD     #PAT+2,R4
447 022436    000752    BR      1$
448 022440    013737 002012 003130 END:    MOV     L$UNIT,UUT        ;GET BEGINNING OF BUFFER
449 022446    012704 017236    MOV     #DRBUF,R4        ;CLEAR ADJ. TEST FLAG
450 022452    005037 003064    CLR     FADJ              ;CLEAR OVERWRITE FLAG
451 022456    005037 003062    CLR     FOWR
452 022462    READEF  #EF.PWR
(3) 022462    012700 000034    MOV     #EF.PWR,R0
(3) 022466    104447    TRAP     C$REFG

```


453 022470
(2) 022470 103010
454 022472
(7) 022472 012746 021543
(6) 022476 012746 000001
(3) 022502 010600
(4) 022504 104417
(4) 022506 062706 000004

BNCOMPLETE SETUP
BCC SETUP
PRINTF #FRM11
MOV #FRM11, -(SP)
MOV #1, -(SP)
MOV SP, R0
TRAP C\$PNTF
ADD #4, SP

455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470

:INITIALIZE ROUTINE
:WE ATTEMPT TO LOCATE 5 PERFECT ADJACENT TRACKS AT 5 SPOTS
:ACROSS THE PACK.
:THE 5 SPOTS ARE: (EACH SURFACE)
:
:OUTER - TRACK 0 - 16 (BOTH RL01 & RL02)
:INNER - TRACK 238 - 254 (RL01) OR 494 - 510 (RL02)
:MIDDLE - TRACK 120 - 136 (RL01) OR 248 - 264 (RL02)
:ONE QUARTER - TRACK 56 - 72 (RL01) OR 120 - 136 (RL02)
:THREE QUARTER - TRACK 184 - 200 (RL01) OR 376 - 392 (RL02)
:
:IF WE FIND ANY BAD SPOTS, WE WILL REPORT SO.....

471 022512 005237 003076
472 022516 012737 177777 003210
473 022524 012737 177777 003212
474 022532
(7) 022532 012746 022151
(6) 022536 012746 000001
(3) 022542 010600
(4) 022544 104417
(4) 022546 062706 000004
475 022552 004537 031676
476 022556 004537 031124
477 022562 004537 030400
478 022566 012701 002236
479 022572 012700 000062
480 022576 012721 177777
481 022602 005300
482 022604 001374
483
484 022606 004537 030626
485 022612 000001
486 022614 000000
487
488 022616 000000 000020
489 022622 000000 000020
490
491 022626 005737 003160
492 022632 001005
493
494 022634
(4) 022634 104456
(5) 022636 000012
(5) 022640 017675

SETUP: INC STFLG ;INDICATE A START COMMAND
MOV #-1, SERNM1
MOV #-1, SERNM2
PRINTF #FRM18
MOV #FRM18, -(SP)
MOV #1, -(SP)
MOV SP, R0
TRAP C\$PNTF
ADD #4, SP
JSR R5, LOAD ;TELL OPERATOR TO LOAD
JSR R5, SERNUM ;GET SERIAL NUMBER
JSR R5, MERGE ;MERGE BAD SECTOR FILES
MOV #OUT10, R1 ;INITIALIZE ALL TRACKS
1\$: MOV #50, R0
MOV #177777, (R1)+
DEC R0
BNE 1\$
JSR R5, FNDTRK ;TRY TO FIND FIVE TRACKS
1 ;INWARD SEARCH
0 ;TOP SURFACE
;WORD 0,16.
;WORD 0,16.
TST TRKFND ;WAS SEARCH SUCCESSFUL????
BNE 2\$;YES
ERRHRD 10, ERRFND, ERR1 ;NO TRACKS
TRAP C\$ERHRD
;WORD 10
;WORD ERRFND


```

(5) 022642 020172      .WORD  ERR1
495 022644 000404      BR      3$
496
497 022646 012700 002236 2$:  MOV  #OUT10,RO      ;STORE AWAY TRACKS FOUND
498 022652 004537 031070      JSR  R5,FIXCYL
499
500 022656 004537 030626 3$:  JSR  R5,FNDTRK      ;TRY TO FIND FIVE TRACKS
501 022662 000001      1      ;INWARD SEARCH
502 022664 000001      1      ;BOTTOM SURFACE
503 022666 000000 000020      .WORD  0,16.
504 022672 000000 000020      .WORD  0,16.
505
506 022676 005737 003160      TST  TRKFND      ;WAS SEARCH SUCCESSFUL????
507 022702 001005      BNE  4$      ;YES
508
509 022704      ERRHRD 10.,ERRFND,ERR1 ;NO TRACKS
(4) 022704 104456      TRAP  C$ERHRD
(5) 022706 000012      .WORD  10
(5) 022710 017675      .WORD  ERRFND
(5) 022712 020172      .WORD  ERR1
510 022714 000404      BR      5$
511
512 022716 012700 002250 4$:  MOV  #OUT11,RO      ;STORE TRACKS AWAY
513 022722 004537 031070      JSR  R5,FIXCYL
514 022726 004537 030626 5$:  JSR  R5,FNDTRK      ;FIND NEXT 5 TRACK
515 022732 177777      -1     ;OUTWARD SEARCH
516 022734 000000      0      ;TOP SURFACE
517 022736 000376 000356      .WORD  254.,238.      ;TRACK RANGE
518 022742 000776 000756      .WORD  510.,494.
519
520 022746 005737 003160      TST  TRKFND      ;WAS SEARCH SUCCESSFUL?
521 022752 001005      BNE  6$      ;YES
522
523 022754      ERRHRD 10.,ERRFND,ERR1 ;NO TRACKS
(4) 022754 104456      TRAP  C$ERHRD
(5) 022756 000012      .WORD  10
(5) 022760 017675      .WORD  ERRFND
(5) 022762 020172      .WORD  ERR1
524 022764 000404      BR      7$      ;SKIP
525
526 022766 012700 002356 6$:  MOV  #INN10,RO      ;STORE AWAY TRACKS FOUND
527 022772 004537 031070      JSR  R5,FIXCYL
528
529 022776 004537 030626 7$:  JSR  R5,FNDTRK      ;NEXT SET
530 023002 177777      -1     ;OUTWARD SEARCH
531 023004 000001      1      ;BOTTOM SURFACE
532 023006 000376 000356      .WORD  254.,238.
533 023012 000776 000756      .WORD  510.,494.
534
535 023016 005737 003160      TST  TRKFND      ;SEARCH SUCCESSFUL?
536 023022 001005      BNE  8$      ;YES
537
538 023024      ERRHRD 10.,ERRFND,ERR1 ;NO TRACKS
(4) 023024 104456      TRAP  C$ERHRD
(5) 023026 000012      .WORD  10
(5) 023030 017675      .WORD  ERRFND
  
```

```
(5) 023032 020172          .WORD  ERR1
539 023034 000404          BR      9$
540
541 023036 012700 002370    8$:    MOV    #INN11,R0          ;STORE AWAY TRACKS FOUND
542 023042 004537 031070    JSR    R5,FIXCYL
543
544 023046 004537 030626    9$:    JSR    R5,FNDTRK          ;NEXT SET
545 023052 000001          1      ;INWARD SEARCH
546 023054 000000          0      ;TOP SURFACE
547 023056 000176 000210    .WORD  126.,136.          ;TRACK RANGE
548 023062 000376 000410    .WORD  254.,264.
549
550 023066 005737 003160    TST    TRKFND              ;DID WE FIND A SET
551 023072 001020          BNE    10$                 ;YES
552
553 023074 004537 030626    JSR    R5,FNDTRK          ;NEXT SET (OTHER SIDE)
554 023100 177777          -1     ;OUTWARD SEARCH
555 023102 000000          0      ;TOP SURFACE
556 023104 000202 000170    .WORD  130.,120.          ;TRACK RANGE
557 023110 000402 000370    .WORD  258.,248.
558 023114 005737 003160    TST    TRKFND              ;DID WE FIND A SET
559 023120 001005          BNE    10$                 ;YES
560
561 023122          ERRHRD 10.,ERRFND,ERR1 ;NO TRACKS
(4) 023122 104456          TRAP   C$ERHRD
(5) 023124 000012          .WORD  10
(5) 023126 017675          .WORD  ERRFND
(5) 023130 020172          .WORD  ERR1
562 023132 000404          BR      11$
563
564 023134 012700 002306    10$:   MOV    #MID10,R0          ;STORE AWAY
565 023140 004537 031070    JSR    R5,FIXCYL
```


567	023144	004537	030626	11\$:	JSR	R5,FNDTRK		:NEXT SET
568	023150	000001			1			:INWARD SEARCH
569	023152	000001			1			:BOTTOM SURFACE
570	023154	000176	000210		.WORD	126.,136.		:RANGE
571	023160	000376	000410		.WORD	254.,264.		
572								
573	023164	005737	003160		TST	TRKFND		:SUCCESS?
574	023170	001020			BNE	12\$:YES
575								
576	023172	004537	030626		JSR	R5,FNDTRK		:LOOK THE OTHER SIDE
577	023176	177777			-1			:OUTWARD
578	023200	000001			1			:BOTTOM SURFACE
579	023202	000202	000170		.WORD	130.,120.		
580	023206	000402	000370		.WORD	258.,248.		
581								
582	023212	005737	003160		TST	TRKFND		:SUCCESS?
583	023216	001005			BNE	12\$:YES
584								
585	023220				ERRHRD	10.,ERRFND,ERR1	;NO TRACKS	
(4)	023220	104456			TRAP	C\$ERHRD		
(5)	023222	000012			.WORD	10		
(5)	023224	017675			.WORD	ERRFND		
(5)	023226	020172			.WORD	ERR1		
586	023230	000404			BR	13\$		
587								
588	023232	012700	002320	12\$:	MOV	#MID11,R0		:STORE AWAY THE TRACKS FOUND
589	023236	004537	031070		JSR	R5,FXCYL		
590								
591	023242	004537	030626	13\$:	JSR	R5,FNDTRK		:NEXT SET
592	023246	000001			1			:INWARD
593	023250	000000			0			:TOP SURFACE
594	023252	000076	000110		.WORD	62.,72.		:RANGE
595	023256	000176	000210		.WORD	126.,136.		
596								
597	023262	005737	003160		TST	TRKFND		:SUCCESS?
598	023266	001020			BNE	14\$:YES
599								
600	023270	004537	030626		JSR	R5,FNDTRK		:LOOK OTHER SIDE
601	023274	177777			-1			:OUTWARD
602	023276	000000			0			:TOP SURFACE
603	023300	000102	000070		.WORD	66.,56.		:RANGE
604	023304	000202	000170		.WORD	130.,120.		
605								
606	023310	005737	003160		TST	TRKFND		:SUCCESS?
607	023314	001005			BNE	14\$:YES
608								
609	023316				ERRHRD	10.,ERRFND,ERR1	;NO TRACKS	
(4)	023316	104456			TRAP	C\$ERHRD		
(5)	023320	000012			.WORD	10		
(5)	023322	017675			.WORD	ERRFND		
(5)	023324	020172			.WORD	ERR1		
610	023326	000404			BR	15\$		
611								
612	023330	012700	002262	14\$:	MOV	#OQU10,R0		:STORE AWAY NEXT SET
613	023334	004537	031070		JSR	R5,FXCYL		

615	023340	004537	030626	15\$:	JSR	R5,FNDTRK		:LOOK FOR NEXT SET
616	023344	000001			1			:INWARD
617	023346	000001			1			:BOTTOM
618	023350	000076	000110		.WORD	62.,72.		:RANGE
619	023354	000176	000210		.WORD	126.,136.		
620								
621	023360	005737	003160		TST	TRKFND		:SUCCESS?
622	023364	001020			BNE	16\$:YES
623								
624	023366	004537	030626		JSR	R5,FNDTRK		:LOOK FOR ANOTHER SET
625	023372	177777			-1			:OUTWARD
626	023374	000001			1			:BOTTOM
627	023376	000102	000070		.WORD	66.,56.		:RANGE
628	023402	000202	000170		.WORD	130.,120.		
629								
630	023406	005737	003160		TST	TRKFND		:SUCCESS?
631	023412	001005			BNE	16\$:YES
632								
633	023414				ERRHRD	10.,ERRFND,ERR1 ;NO TRACKS		
(4)	023414	104456			TRAP	C\$ERHRD		
(5)	023416	000012			.WORD	10		
(5)	023420	017675			.WORD	ERRFND		
(5)	023422	020172			.WORD	ERR1		
634	023424	000404			BR	17\$		
635								
636	023426	012700	002274	16\$:	MOV	#OQU11,R0		:STORE AWAY TRACKS
637	023432	004537	031070		JSR	R5,FXCYL		
638								
639	023436	004537	030626	17\$:	JSR	R5,FNDTRK		:NEXT SET OF TRACKS
640	023442	000001			1			:INWARD
641	023444	000000			0			:TOP SURFACE
642	023446	000276	000310		.WORD	190.,200.		:RANGE
643	023452	000576	000610		.WORD	382.,392.		
644								
645	023456	005737	003160		TST	TRKFND		:SUCCESS?
646	023462	001020			BNE	18\$:YES
647								
648	023464	004537	030626		JSR	R5,FNDTRK		:LOOK OTHER SIDE
649	023470	177777			-1			:OUTWARD SEARCH
650	023472	000000			0			:TOP
651	023474	000302	000270		.WORD	194.,184.		
652	023500	000602	000570		.WORD	386.,376.		
653								
654	023504	005737	003160		TST	TRKFND		:SUCCESS
655	023510	001005			BNE	18\$:YES
656								
657	023512				ERRHRD	10.,ERRFND,ERR1 ;NO TRACKS		
(4)	023512	104456			TRAP	C\$ERHRD		
(5)	023514	000012			.WORD	10		
(5)	023516	017675			.WORD	ERRFND		
(5)	023520	020172			.WORD	ERR1		
658	023522	000404			BR	19\$		
659								
660	023524	012700	002332	18\$:	MOV	#TQU10,R0		:STORE TRACKS AWAY
661	023530	004537	031070		JSR	R5,FXCYL		


```
663 023534 004537 030626      19$: JSR    R5,FNDTRK      ;NEXT SET
664 023540 000001                1      ;INWARD
665 023542 000001                1      ;BOTTOM SURFACE
666 023544 000276 000310      .WORD 190.,200.      ;RANGE
667 023550 000576 000610      .WORD 382.,392.
668
669 023554 005737 003160      TST    TRKFND      ;SUCCESS?
670 023560 001020      BNE    20$        ;YES
671
672 023562 004537 030626      JSR    R5,FNDTRK      ;OTHER SET
673 023566 177777      -1      ;OUTWARD
674 023570 000001                1      ;BOTTOM SURFACE
675 023572 000302 000270      .WORD 194.,184.      ;RANGE
676 023576 000602 000570      .WORD 386.,376.
677
678 023602 005737 003160      TST    TRKFND      ;SUCCESS
679 023606 001005      BNE    20$        ;YES
680
681 023610      ERRHRD 10.,ERRFND,ERR1 ;NO TRACKS
(4) 023610 104456      TRAP  C$ERHRD
(5) 023612 000012      .WORD 10
(5) 023614 017675      .WORD ERRFND
(5) 023616 020172      .WORD ERR1
682 023620 000404      BR     21$
683
684 023622 012700 002344      20$: MOV    #TQU11,R0      ;STORE SET AWAY
685 023626 004537 031070      JSR    R5,FIXCYL
686
687 023632 012700 002236      21$: MOV    #OUT10,R0      ;DID WE FIND ANY AT ALL
688 023636 012701 000062      MOV    #50.,R1
689 023642 022720 177777      22$: CMP    #-1,(R0)+
690 023646 001017      BNE    EXIT
691 023650 005301      DEC    R1
692 023652 001373      BNE    22$
693 023654      ERRSF 3.,NONE
(4) 023654 104454      TRAP  C$ERSF
(5) 023656 000003      .WORD 3
(5) 023660 017523      .WORD NONE
(5) 023662 000000      .WORD 0
694 023664 005001      CMPENA: CLR   R1
695 023666 013700 002012      MOV    L$UNIT,R0
696 023672      24$: DODU  R1      ;DO DROP UNIT
(3) 023672 010100      MOV    R1,R0
(3) 023674 104451      TRAP  C$DODU
697 023676 005201      INC    R1
698 023700 005300      DEC    R0
699 023702 001373      BNE    24$
700 023704      DOCLN
(3) 023704 104444      TRAP  C$DCLN
701
702 023706      EXIT:
703 023706      L10010: ENDINIT
(3) 023706      TRAP  C$INIT
(3) 023706 104411      ENDMOD
704 023710
705
```

```
707
708 023710          BGNMOD  AUTOCODE          ;AUTO DROP SECTION
709 023710          BGNAUTO
710
711 023710 000240          NOP          ;DO NOTHING
712
713 023712          ENDAUTO
(3) 023712          L10011:
(3) 023712 104461          TRAP    C$AUTO
714 023714          ENDMOD
715
716
717 023714          BGNMOD  CLNCODE
718 023714          BGNCLN
719
720 023714 000240          NOP
721
722 023716          L10012:  ENDCLN
(3) 023716          TRAP    C$CLEAN
(3) 023716 104412          ENDMOD
723 023720
724
725 023720          BGNMOD  DRPCODE
726 023720          BGNDU
727 023720 000240          NOP
728 023722          ENDDU
(3) 023722          L10013:
(3) 023722 104453          TRAP    C$DU
729 023724          ENDMOD
730
731
732
```



```

734          .SBTTL GLOBAL SUBROUTINES SECTION
735
736 023724    BGNMOD GLBSUB
737
738 023724 012737 000160 002116 TIME:  MOV    #160,L$DLY      ;GET OUTER DELAY LOOP
739 023732 005437 017232          NEG    XDELAY        ;GET NEGATIVE OF MULTIPLY FACTOR
740 023736          READBUS          ;Q-BUS?
(3) 023736 104407          TRAP   C$RDBU
741 023740          BCOMPLETE      2$      ;BRANCH - IF YES
(2) 023740 103420          BCS    2$
742 023742          1$: DELAY   #1          ;WAIT
(2) 023742 012727 000001          MOV    ##1,(PC)+
(2) 023746 000000          .WORD  0
(2) 023750 013727 002116          MOV    L$DLY,(PC)+
(2) 023754 000000          .WORD  0
(2) 023756 005367 177772          DEC    -6(PC)
(2) 023762 001375          BNE    -4
(2) 023764 005367 177756          DEC    -22(PC)
(2) 023770 001367          BNE    -20
743 023772 005237 017232          INC    XDELAY        ;WAIT FACTOR EXPIRED?
744 023776 002761          BLT   1$            ;BRANCH - IF NO
745 024000 000422          BR    4$            ;EXIT
746 024002 012737 000150 002116 2$: MOV    #150,L$DLY    ;GET OUTER DELAY LOOP
747 024010          3$: DELAY   #1          ;WAIT WITH RESPECT TO FONZ BUS
(2) 024010 012727 000001          MOV    ##1,(PC)+
(2) 024014 000000          .WORD  0
(2) 024016 013727 002116          MOV    L$DLY,(PC)+
(2) 024022 000000          .WORD  0
(2) 024024 005367 177772          DEC    -6(PC)
(2) 024030 001375          BNE    -4
(2) 024032 005367 177756          DEC    -22(PC)
(2) 024036 001367          BNE    -20
748 024040 005237 017232          INC    XDELAY        ;WAIT FACTOR EXPIRED?
749 024044 002761          BLT   3$            ;BRANCH - IF NO
750 024046 000207          4$: RTS    PC        ;RETURN
751
752 024050 012737 000160 002116 XTIME: MOV    #160,L$DLY    ;GET OUTER DELAY LOOP
753 024056 006337 017234          ASL   YDELAY        ;MULTIPLY FACTOR BY 4
754 024062 006337 017234          ASL   YDELAY
755 024066 005437 017234          NEG   YDELAY        ;GET NEGATIVE OF RESULT
756 024072          READBUS          ;Q-BUS?
(3) 024072 104407          TRAP   C$RDBU
757 024074          BNCOMPLETE  1$      ;BRANCH - IF NO
(2) 024074 103023          BCC   1$
758 024076 012737 000150 002116 2$: MOV    #150,L$DLY    ;GET OUTER DELAY LOOP
759 024104          DELAY   #20        ;WAIT WITH RESPECT TO FONZ BUS
(2) 024104 012727 000020          MOV    ##20,(PC)+
(2) 024110 000000          .WORD  0
(2) 024112 013727 002116          MOV    L$DLY,(PC)+
(2) 024116 000000          .WORD  0
(2) 024120 005367 177772          DEC    -6(PC)
(2) 024124 001375          BNE    -4
(2) 024126 005367 177756          DEC    -22(PC)
(2) 024132 001367          BNE    -20
760 024134 005237 017234          INC    YDELAY        ;WAIT FACTOR EXPIRED?
761 024140 002761          BLT   2$            ;BRANCH - IF NO

```

```
762 024142 000417  
763 024144 1$: BR 3$ :EXIT  
(2) 024144 012727 000050 DELAY #50 :WAIT  
(2) 024150 000000 MOV ##50,(PC)+  
(2) 024152 013727 002116 .WORD 0  
(2) 024156 000000 MOV L$DLY,(PC)+  
(2) 024160 005367 177772 .WORD 0  
(2) 024164 001375 DEC -6(PC)  
(2) 024166 005367 177756 BNE -4  
(2) 024172 001367 DEC -22(PC)  
764 024174 005237 017234 BNE -20  
765 024200 002761 INC YDELAY :WAIT FACTOR EXPIRED?  
766 024202 000207 3$: BLT 1$ :BRANCH - IF NO  
767 RTS PC :RETURN  
768
```



```
770  
771 ;ROUTINE TO PERFORM OVERWRITE  
772 ;CALL: JSR R5,OVWPER  
773 ; SECTORS TO WRITE FORWARD  
774 ; SECTORS TO WRITE REVERSE  
775  
776 024204 010046 OVWPER: MOV R0,-(SP) ;SAVE R0, R1, R2, R3  
777 024206 010146 MOV R1,-(SP)  
778 024210 010246 MOV R2,-(SP)  
779 024212 010346 MOV R3,-(SP)  
780 024214 005000 CLR R0 ;R0 HAS COUNT IF R0<5.  
781 024216 012537 003126 MOV (R5)+,FORSK ;USE TOP SURFACE, IF R0>5.  
782 024222 012537 003124 MOV (R5)+,REVSK ;USE BOTTOM SURFACE, IF R0>1  
783 ;DONE.  
784 024226 012701 003022 1$: MOV #OVWTRK,R1 ;GET START OF LIST OF TRACKS  
785 024232 011102 MOV (R1),R2 ;GET POINTER TO TRACK  
786 024234 021227 177777 CMP (R2),#-1 ;LEGIT TRACK?????  
787 024240 001500 BEQ 3$ ;NO, EXIT  
788  
789 024242 005037 003122 CLR CYL ;CLEAR CYLINDER/HEAD FOR SEEK  
790 024246 005037 003120 CLR SURF  
791 024252 020027 000005 CMP R0,#5 ;TOP/BOTTOM  
792 024256 002402 BLT 2$ ;TOP, BRANCH  
793 024260 005237 003120 INC SURF ;BOTTOM SURFACE  
794 024264 004537 025652 2$: JSR R5,SKCYL ;SEEK TO CYLINDER  
795 024270 005037 003122 CLR CYL  
796 024274 051237 003122 BIS (R2),CYL  
797 024300 004537 025652 JSR R5,SKCYL ;SEEK TO PROPER CYLINDER  
798 024304 013703 003126 MOV FORSK,R3 ;SECTORS TO WRITE  
799 024310 004537 024466 JSR R5,WRSEC ;GO WRITE SECTORS  
800 024314 000034 .WORD 28.  
801 024316 012737 020050 003116 MOV #FWD,DIRC ;SET FORWARD DIRECTION  
802 024324 004537 026576 JSR R5,VEROW ;VERIFY OVERWRITE  
803 024330 004537 027162 JSR R5,VEROD ;VERIFY OTHER DRIVES DATA  
804 024334 005037 003122 CLR CYL  
805 024340 022737 000001 003060 CMP #1,T.DRIVE ;RL01?  
806 024346 001004 BNE 50$ ;NO  
807 024350 052737 000377 003122 BIS #377,CYL ;SET TO GO TO MAX CYL  
808 024356 000403 BR 51$  
809 024360 052737 000777 003122 50$: BIS #777,CYL ;MAX CYL FOR RL02  
810 024366 004537 025652 51$: JSR R5,SKCYL ;SEEK TO MAX CYLINDER ON DRIVE  
811 024372 005037 003122 CLR CYL  
812 024376 005037 003120 CLR SURF  
813 024402 051237 003122 BIS (R2),CYL  
814 024406 004537 025652 JSR R5,SKCYL ;DO ANOTHER SEEK  
815  
816 024412 013703 003124 MOV REVSK,R3 ;SECTORS TO WRITE  
817 024416 004537 024466 JSR R5,WRSEC ;WRITE THEM  
818 024422 000034 .WORD 28.  
819 024424 012737 020060 003116 MOV #REV,DIRC ;SET DIRECTION  
820 024432 004537 026576 JSR R5,VEROW ;VERIFY OVERWRITE  
821 024436 004537 027162 JSR R5,VEROD ;VERIFY OTHER DRIVES DATA  
822  
823 024442 005721 3$: TST (R1)+ ;INCREMENT TO NEXT TRACK  
824 024444 005200 INC R0 ;ACCOUNT FOR IT  
825 024446 020027 000012 CMP R0,#10. ;DONE?
```

CZRLLO RL01/02 DRIVE COMPAT
CZRLLO.MAC 07-DEC-79 10:40

MACY11 30A(1052) 17-DEC-79 11:21 L 4
GLOBAL SUBROUTINES SECTION PAGE 1-24

SEQ 0050

826	024452	001267	BNE	1\$:NO, GO BACK
827					
828	024454	012603	MOV	(SP)+,R3	:RESTORE REG.
829	024456	012602	MOV	(SP)+,R2	
830	024460	012601	MOV	(SP)+,R1	
831	024462	012600	MOV	(SP)+,R0	
832	024464	000205	RTS	R5	:EXIT


```

834 ;ROUTINE TO WRITE SECTORS
835 ;USED IN OVERWRITE TEST;ADJACENT CYLINDER TEST
836 ;CALL JSR R5,WRSEC
837 ;.WRD ;STARTING SECTOR
838 ;R3 HAS BITMAP OF SECTORS TO WRITE
839 ;R4 HAS DRIVE BUFFER POINTER
840
841 024466 010046 WRSEC: MOV R0,-(SP) ;SAVE R0
842 024470 010146 MOV R1,-(SP) ;SAVE R1
843 024472 010246 MOV R2,-(SP) ;SAVE R2
844 024474 012701 003232 MOV #BUF,R1 ;WRITE PATTERN INTO
845 024500 012702 000200 MOV #128.,R2 ;MEMORY THAT WE
846 024504 016421 000006 2$: MOV PAT(R4),(R1)+ ;WILL WRITE ONTO
847 024510 005302 DEC R2 ;PACK FOR THIS
848 024512 001374 BNE 2$ ;DRIVE
849 024514 012701 100000 MOV #100000,R1 ;MASK FOR BIT MAP
850 024520 012737 000007 003056 MOV #7,TEM
851 024526 053702 003122 BIS CYL,R2
852 024532 006302 120$: ASL R2
853 024534 005337 003056 DEC TEM
854 024540 001374 BNE 120$
855 024542 005737 003120 TST SURF
856 024546 001402 BEQ 3$ ;0, SKIP
857 024550 052702 000100 BIS #HEAD,R2 ;SET BOTTOM HEAD
858 024554 052502 3$: BIS (R5)+,R2 ;START AT SECTOR 28.
859 024556 030103 4$: BIT R1,R3 ;WRITE THIS SECTOR?
860 024560 001452 BEQ 5$ ;NO
861
862 024562 005037 003110 CLR HSFLG
863 024566 012737 177600 003206 MOV #-128.,BMP ;LOAD WORD COUNT
864 024574 010237 003204 MOV R2,BDA ;LOAD DISK ADDRESS
865 024600 010237 003066 MOV R2,TEMP ;SAVE DISK ADDRESS
866 024604 042702 177700 BIC #177700,R2
867 024610 020227 000047 CMP R2,#39.
868 024614 003403 BLE 6$
869 024616 162737 000050 003204 SUB #40.,BDA
870 024624 012737 003232 003202 6$: MOV #BUF,BBA ;LOAD BUS ADDRESS
871 024632 013702 003066 MOV TEMP,R2 ;RESTORE DISK ADDRESS
872 024636 004537 032002 11$: JSR R5,LDFUNC ;GO WRITE
873 024642 000012 WRITE
874 024644 005737 003074 TST ERFLG ;ERROR IN WRITING
875 024650 001416 BEQ 5$ ;NO,OKAY
876 024652 005737 003110 TST HSFLG
877 024656 001007 BNE 10$
878 024660 ERRSOFT 100.,WRIT1,ERR2
(4) 024660 104457 TRAP C$ERSOFT
(5) 024662 000144 .WORD 100
(5) 024664 020070 .WORD WRIT1
(5) 024666 020230 .WORD ERR2
879 024670 005237 003110 INC HSFLG
880 024674 000760 BR 11$
881 024676 10$: ERRHRD 110.,WRIT1,ERR2
(4) 024676 104456 TRAP C$ERHRD
(5) 024700 000156 .WORD 110
(5) 024702 020070 .WORD WRIT1
(5) 024704 020230 .WORD ERR2

```

```
882  
883 024706 005202          5$:  INC    R2          :NEXT SECTOR  
884 024710 000241          CLC          :CLEAR CARRY BIT  
885 024712 006001          ROR    R1          :DONE?  
886 024714 103320          BCC    4$          :NO GO BACK  
887 024716 012602          MOV    (SP)+,R2    :REGISTERS AND EXIT  
888 024720 012601          MOV    (SP)+,R1  
889 024722 012600          MOV    (SP)+,R0  
890 024724 000205          RTS    R5
```


892	024726	005037	003214		ADJCYL: CLR	ADJTRK	:INSIDE/OUTSIDE TRACK FLAG
893	024732	005037	003114		CLR	HEAD01	:INIT TO TOP SURFACE
894	024736	012737	000001	003216	MOV	#1,ADJUUT	:START OF TRACK LIST
895	024744	012701	002236		21\$: MOV	#OUT10,R1	:
896	024750	012537	003100		20\$: MOV	(R5)+,ADJLOC	:PICK UP TRACK OFFSET
897	024754	001003			BNE	1\$:IS THERE ONE?
898	024756	005037	003104		CLR	ADJDIR	
899	024762	000205			RTS	R5	:NO EXIT
900	024764	012537	003220		1\$: MOV	(R5)+,ADJLC2	:YES, GET REST OF INFO
901	024770	012537	003222		MOV	(R5)+,ADJLC3	
902	024774	012537	003224		MOV	(R5)+,ADJLC4	
903	025000	113700	003100		2\$: MOV	ADJLOC,R0	:GET OFFSET
904	025004	012737	000020	003230	MOV	#16.,STSEC	:STARTING SECTOR IS 16
905							
906	025012	010102			MOV	R1,R2	:GET START INTO R2
907							
908	025014	005300			3\$: DEC	R0	:DOWN COUNT OFFSET
909	025016	001414			BEQ	4\$:FOUND IT?
910							
911	025020	005722			TST	(R2)+	:INDEX (R2)
912	025022	062737	000042	003230	ADD	#34.,STSEC	:NO, NEXT SECTOR
913	025030	022737	000050	003230	CMP	#40.,STSEC	
914	025036	003366			BGT	3\$	
915	025040	162737	000050	003230	SUB	#40.,STSEC	
916	025046	000762			BR	3\$:BACK FOR NEXT
917							
918	025050	021227	177777		4\$: CMP	(R2),#-1	:LEGAL TRACK?
919	025054	001002			BNE	5\$:YES, CONTINUE
920							
921	025056	000137	025524		JMP	13\$:NO PICK UP NEXT SET
922							
923	025062	005037	003120		5\$: CLR	SURF	:SET UP FOR OUTER TRACK
924	025066	005037	003122		CLR	CYL	
925							
926	025072	005737	003114		TST	HEAD01	:WHICH HEAD?
927	025076	001403			BEQ	6\$:TOP, SKIP
928							
929	025100	052737	000001	003120	BIS	#1,SURF	:LOWER HEAD, SET IT!
930							
931	025106	004537	025652		6\$: JSR	R5,SKCYL	:SEEK TO OUTER TRACK
932							
933	025112	011237	003122		MOV	(R2),CYL	:GET DESIRED TRACK
934							
935	025116	004537	025652		JSR	R5,SKCYL	:SEEK TO IT
936	025122	012737	020050	003116	MOV	#FWD,DIRC	:SEEK DIRECTION
937	025130	113703	003101		MOV	ADJLOC+1,R3	:GET SECTORS TO WRITE
938	025134	000303			SWAB	R3	:ALIGN IT
939	025136	042703	000377		BIC	#377,R3	:CLEAR OUT HIGH BYTE
940							
941	025142	022737	000047	003230	CMP	#39.,STSEC	:OVER FORTY?
942	025150	002003			BGE	7\$:NO, CONTINUE
943							
944	025152	162737	000050	003230	SUB	#40.,STSEC	:YES BACK IT UP
945	025160	013737	003230	025172	7\$: MOV	STSEC,8\$:STARTING SECTOR
946							
947	025166	004537	024466		JSR	R5,WRSEC	:WRITE SECTORS

948	025172	000000			8\$:	.WORD	0	
949	025174	013737	025172	025206		MOV	8\$,108\$	
950	025202	004537	027510			JSR	R5,VAJWR	;VERIFY THIS WRITE
951	025206	000000			108\$:	.WORD	0	
952	025210	013737	025206	025222		MOV	108\$,208\$	
953	025216	004537	027754			JSR	R5,BSVWR	
954	025222	000000			208\$:	.WORD	0	
955	025224	013737	003230	003226		MOV	STSEC,STSEC1	;GET OTHER SECTORS TO WRITE
956	025232	062737	000010	003226		ADD	#8.,STSEC1	;8 SECTORS GONE BY
957	025240	022737	000047	003226		CMP	#39.,STSEC1	;GONE PAST 40?
958	025246	002003				BGE	9\$;NO, OKAY
959								
960	025250	162737	000050	003226		SUB	#40.,STSEC1	;YES BACK IT UP
961								
962	025256	013703	003220		9\$:	MOV	ADJLC2,R3	;GET SECTORS TO WRITE
963								
964	025262	013737	003226	025274		MOV	STSEC1,10\$;STARTING SECTORS
965								
966	025270	004537	024466			JSR	R5,WRSEC	;WRITE SECTORS
967	025274	000000			10\$:	.WORD	0	
968	025276	013737	025274	025310		MOV	10\$,110\$	
969	025304	004537	027510			JSR	R5,VAJWR	;VERIFY THIS WRITE
970	025310	000000			110\$:	.WORD	0	
971	025312	013737	025310	025324		MOV	110\$,210\$	
972	025320	004537	027754			JSR	R5,BSVWR	;VERIFY ADJ CYL + 1
973	025324	000000			210\$:	.WORD	0	
974	025326	022737	000001	003060		CMP	#1,T.DRIVE	
975	025334	001004				BNE	77\$	
976	025336	012737	000377	003122		MOV	#377,CYL	
977	025344	000403				BR	88\$	
978								
979	025346	012737	000777	003122	77\$:	MOV	#777,CYL	
980								
981	025354	004537	025652		88\$:	JSR	R5,SKCYL	
982								
983	025360	011237	003122			MOV	(R2),CYL	;SEEK BACK TO PROPER TRACK
984								
985	025364	004537	025652			JSR	R5,SKCYL	;SEEK TO PROPER CYLINDER
986	025370	012737	020060	003116		MOV	#REV,DIRC	;SEEK DIRECTION
987	025376	113703	003223			MOVB	ADJLC3+1,R3	;GET SECTORS TO WRITE
988								
989	025402	000303				SWAB	R3	;ALIGN IT
990	025404	042703	000377			BIC	#377,R3	;CLEAR OUT HIGH BYTE
991	025410	013737	003230	025422		MOV	STSEC,11\$	
992								
993	025416	004537	024466			JSR	R5,WRSEC	;WRITE PROPER SECTOR
994	025422	000000			11\$:	.WORD	0	
995								
996	025424	013737	025422	025436		MOV	11\$,111\$	
997	025432	004537	027510			JSR	R5,VAJWR	;VERIFY THIS WRITE
998	025436	000000			111\$:	.WORD	0	
999	025440	013737	025436	025452		MOV	111\$,211\$	
1000	025446	004537	027754			JSR	R5,BSVWR	
1001	025452	000000			211\$:	.WORD	0	
1002	025454	013703	003224			MOV	ADJLC4,R3	;GET SECTORS
1003	025460	013737	003226	025472		MOV	STSEC1,12\$;GET SECTORS TO WRITE


```

1004
1005 025466 004537 024466          12$: JSR    R5,WRSEC      ;WRITE PROPER SECTORS
1006 025472 000000
1007
1008
1009 025474 013737 025472 025506    MOV    12$,112$
1010 025502 004537 027510          JSR    R5,VAJWR      ;VERIFY THIS WRITE
1011 025506 000000          112$: .WORD    0
1012
1013
1014 025510 013737 025506 025522    MOV    112$,212$
1015 025516 004537 027754          JSR    R5,BSVWR      ;VERIFY ADJ CYLINDERS + 1
1016 025522 000000          212$: .WORD    0
1017
1018
1019 025524 005737 003114          13$: TST    HEAD01    ;WHICH HEAD WERE WE DOING?
1020 025530 001003          BNE    14$
1021 025532 005237 003114          INC    HEAD01
1022 025536 000402          BR     99$
1023 025540 005037 003114          14$: CLR    HEAD01    ;NEXT SET OF TRACKS
1024 025544 062701 000012          99$: ADD    #10.,R1    ;NEXT SET OF TRACKS
1025 025550 020127 002400          CMP    R1,#INN51    ;END OF LIST
1026 025554 002002          BGE    18$          ;END OF TRACK LIST
1027 025556 000137 025000          JMP    2$          ;NO GO BACK
1028
1029          ;AT END OF TRACK LIST NEXT GROUP OF WRITES
1030
1031 025562 005737 003064          18$: TST    FADJ
1032 025566 001403          BEQ    15$          ;FIRST SET?
1033 025570 005037 003064          CLR    FADJ          ;NO, CONTINUE
1034 025574 000421          BR     17$          ;YES, CLEAR FIRST
1035 025576 005737 003214          15$: TST    ADJTRK    ;EXIT
1036 025602 001004          BNE    16$          ;DONE BOTH INSIDE OUTSIDE
1037 025604 005237 003214          INC    ADJTRK        ;TRACKS, YES 16$
1038 025610 000137 024744          JMP    21$          ;NO, SET INSIDE FLAG
1039 025614 005037 003214          16$: CLR    ADJTRK    ;GO DO INSIDE TRACK
1040 025620 005237 003216          INC    ADJUUT        ;BACK TO OUTSIDE TRACK
1041 025624 023737 003216 003130  CMP    ADJUUT,UUT    ;DONE WITH ANOTHER
1042 025632 001402          BEQ    17$          ;DONE TABLE FOR ALL UUT?
1043 025634 000137 024744          JMP    21$          ;YES, FOR EXIT
1044 025640 005725          17$: TST    (R5)+    ;NO, GO BACK FOR NEXT
1045 025642 001376          BNE    17$          ;BUMP EXIT TO END OF
1046 025644 005037 003104          CLR    ADJDIR        ;TABLE FOR PROPER RETURN
1047 025650 000205          RTS    R5          ;EXIT
  
```

```
1049 ;ROUTINE TO SEEK TO A DESIRED CYLINDER
1050 ;CALL: JSR R5,SKCYL
1051 ;ROUTINE HAS DESIRED CYLINDER IN LOC 'CYL'
1052 :
1053 :
1054 025652 010146 SKCYL: MOV R1,-(SP) ;SAVE R1
1055 025654 004537 032002 90$: JSR R5,LDFUNC ;GET PRESENT POSITION
1056 025660 000010 RDHDR
1057 :
1058 025662 005737 003074 TST ERFLG ;ERROR FLAG SET
1059 025666 001104 BNE 5$ ;YES, SKIP
1060 :
1061 025670 005001 CLR R1
1062 025672 012737 000007 003056 MOV #7,TEM
1063 025700 053701 003122 BIS CYL,R1 ;GET THE SELECTED CYLINDER NUMBER
1064 :
1065 025704 006301 120$: ASL R1
1066 025706 005337 003056 DEC TEM
1067 025712 001374 BNE 120$
1068 025714 042737 000177 003172 BIC #177,E.MP
1069 025722 163701 003172 SUB E.MP,R1 ;CALCULATE DIFFERENCE WORD
1070 025726 103002 BCC 1$ ;IF POSITIVE SET DIRECTION
```


1072	025730	005401			NEG	R1		:NEGATE
1073	025732	000402			BR	2\$:SKIP SETTING DIRECTION
1074	025734	052701	000004		1\$: BIS	#SIGN,R1		:SET FOR FORWARD SEEK
1075	025740	052701	000001		2\$: BIS	#MK,R1		:SET MARKER BIT
1076	025744	005737	003120		TST	SURF		
1077	025750	001402			BEQ	3\$:TOP
1078	025752	052701	000020		BIS	#SKHS,R1		:BOTTOM
1079	025756	010137	003204		3\$: MOV	R1,BDA		:LOAD DIFFERENCE WORD
1080	025762	004537	032002		JSR	R5,LDFUNC		:EXECUTE SEEK
1081	025766	000006			SEEK			
1082								
1083	025770	005737	003074		TST	ERFLG		:ERROR?
1084	025774	001041			BNE	5\$:YES, SKIP
1085								
1086	025776	004537	032002		JSR	R5,LDFUNC		:VERIFY POSITION?
1087	026002	000010			RDHDR			
1088	026004	005737	003074		TST	ERFLG		
1089	026010	001033			BNE	5\$		
1090	026012	042737	000077	003172	BIC	#77,E.MP		:VERIFY POSITION
1091	026020	005001			CLR	R1		
1092	026022	012737	000007	003056	MOV	#7,TEM		
1093	026030	053701	003122		BIS	CYL,R1		
1094	026034	006301			220\$: ASL	R1		
1095	026036	005337	003056		DEC	TEM		
1096	026042	001374			BNE	220\$		
1097	026044	005737	003120		TST	SURF		
1098	026050	001402			BEQ	4\$		
1099	026052	052701	000100		BIS	#HEAD,R1		
1100	026056	020137	003172		4\$: CMP	R1,E.MP		
1101	026062	001414			BEQ	6\$		
1102								
1103	026064				ERRDF	12.,SKER,ERR6		:SEEK ERROR
(4)	026064	104455			TRAP	C\$ERDF		
(5)	026066	000014			.WORD	12		
(5)	026070	020031			.WORD	SKER		
(5)	026072	020636			.WORD	ERR6		
1104	026074	000137	025654		JMP	90\$		
1105								
1106	026100				5\$: ERRDF	13.,FUNERR,ERR5		:FUNCTION ERROR IN SEEK
(4)	026100	104455			TRAP	C\$ERDF		
(5)	026102	000015			.WORD	13		
(5)	026104	020001			.WORD	FUNERR		
(5)	026106	020576			.WORD	ERR5		
1107	026110	000137	025654		JMP	90\$		
1108	026114	012601			6\$: MOV	(SP)+,R1		:CANT GET THERE
1109	026116	000205			RTS	R5		:EXIT

```
1111 ;ROUTINE TO PERFORM REGISTER PRINTOUT DUMP
1112 ;CALL: JSR PC,REGDMP
1113 026120 REGDMP: PRINTB #FRM12,BCS,BBA,BDA,BMP
(11) 026120 013746 003206 MOV BMP,-(SP)
(10) 026124 013746 003204 MOV BDA,-(SP)
(9) 026130 013746 003202 MOV BBA,-(SP)
(8) 026134 013746 003200 MOV BCS,-(SP)
(7) 026140 012746 021600 MOV #FRM12,-(SP)
(6) 026144 012746 000005 MOV #5,-(SP)
(3) 026150 010600 MOV SP,R0
(4) 026152 104414 TRAP C$PNTB
(4) 026154 062706 000014 ADD #14,SP
1114 026160 PRINTB #FRM13,E.CS,E.BA,E.DA,E.MP
(11) 026160 013746 003172 MOV E.MP,-(SP)
(10) 026164 013746 003170 MOV E.DA,-(SP)
(9) 026170 013746 003166 MOV E.BA,-(SP)
(8) 026174 013746 003164 MOV E.CS,-(SP)
(7) 026200 012746 021657 MOV #FRM13,-(SP)
(6) 026204 012746 000005 MOV #5,-(SP)
(3) 026210 010600 MOV SP,R0
(4) 026212 104414 TRAP C$PNTB
(4) 026214 062706 000014 ADD #14,SP
1115 026220 032737 040000 003164 BIT #BIT14,E.CS
1116 026226 001437 BEQ 1$
1117 026230 016403 000000 MOV CSR(R4),R3
1118 026234 012763 000013 000004 MOV #13,DA(R3)
1119 026242 012737 000004 003200 MOV #4,BCS
1120 026250 056437 000004 003200 BIS DSB(R4),BCS
1121 026256 013763 003200 000000 MOV BCS,CS(R3)
1122 026264 032763 000200 000000 2$: BIT #200,CS(R3)
1123 026272 001774 BEQ 2$
1124 026274 016337 000006 003106 MOV MP(R3),DRSTAT
1125 026302 PRINTB #FRM14,DRSTAT
(8) 026302 013746 003106 MOV DRSTAT,-(SP)
(7) 026306 012746 021742 MOV #FRM14,-(SP)
(6) 026312 012746 000002 MOV #2,-(SP)
(3) 026316 010600 MOV SP,R0
(4) 026320 104414 TRAP C$PNTB
(4) 026322 062706 000006 ADD #6,SP
1126 026326 000207 1$: RTS PC
```



```

1128 ;ROUTINE TO STORE OR RETRIEVE ADJACENT CYLINDER SECTOR DRIVE
1129 ;INFORMATION FROM THE 24X5 'SECLST' BUFFER.
1130 ;ENTER WITH R0 = SECTOR REQUEST
1131 ;EXIT WITH R0 = ADJACENT CYLINDER DRIVE INFORMATION FOR SECTOR
1132 ;EXIT WITH R0 = 0 IF SECTOR REQUESTED IS NOT IN BUFFER MAP
1133 ;CALL 1: JSR R5,RSADJS
1134 ; .WORD 0 ;RETRIEVE SECTOR INFO.
1135 ;CALL 2: JSR R5,RSADJS
1136 ; .WORD 1 ;STORE SECTOR INFO.
1137 026330 010146 RSADJS: MOV R1,-(SP)
1138 026332 010246 MOV R2,-(SP)
1139 026334 010346 MOV R3,-(SP)
1140 026336 042700 177700 BIC #177700,R0 ;SAVE SECTOR BITS
1141 026342 012537 003102 MOV (R5)+,ADJFLG ;SAVE RETRIEVE/STORE FLAG
1142 026346 012701 000001 MOV #1,R1 ;START WITH TRACK (N-2)
1143 026352 012702 002442 MOV #SECBUF,R2 ;START OF 24X5 BUFFER
1144 026356 012703 000020 MOV #16.,R3 ;SECTOR 16 START FOR (N-2) TRACK
1145 026362 123701 003100 1$: CMPB ADJLOC,R1 ;CHECK TRACK INDEX
1146 026366 001413 BEQ 2$ ;
1147 026370 005201 INC R1 ;INDEX TRACK REFERENCE
1148 026372 062702 000060 ADD #48.,R2 ;UPDATE BUFFER TO NEXT TRACK REF.
1149 026376 062703 000042 ADD #34.,R3 ;UPDATE SECTOR START FOR NEXT TRACK
1150 026402 020327 000050 CMP R3,#40.
1151 026406 002765 BLT 1$
1152 026410 162703 000050 SUB #40.,R3
1153 026414 000762 BR 1$
1154 026416 012701 000030 2$: MOV #24.,R1 ;SET COUNTER FOR 24 SECTORS
1155 026422 020003 3$: CMP R0,R3 ;COMPARE SECTOR TO SECTOR TABLE
1156 026424 001413 BEQ 5$ ;YES,STORE OR RETRIEVE SECTOR INFO.
1157 026426 005722 TST (R2)+ ;INDEX SECLST BUFFER IN WORD FORMAT
1158 026430 005203 INC R3 ;INDEX SECTOR COUNT
1159 026432 020327 000047 CMP R3,#39. ;COMPARE SECTOR COUNT FOR <40
1160 026436 003402 BLE 4$
1161 026440 162703 000050 SUB #40.,R3 ;KEEP SECTOR COUNT<40
1162 026444 005301 4$: DEC R1 ;PASSED 24 SECTORS?
1163 026446 001365 BNE 3$ ;COMPARE NEXT SECTOR
1164 026450 005000 CLR R0 ;SETUP R0 FOR EXIT
1165 026452 000405 BR 7$ ;EXIT ROUTINE,SECTOR NOT FOUND
1166 026454 005737 003102 5$: TST ADJFLG ;FLAG=0 FOR RETRIEVE
1167 026460 001401 BEQ 6$
1168 026462 010412 MOV R4,(R2) ;STORE DRIVE INFO. INTO BUFFER
1169 026464 011200 6$: MOV (R2),R0 ;SAVE DRIVE INFO. INTO R0 FOR EXIT
1170 026466 012603 7$: MOV (SP)+,R3
1171 026470 012602 MOV (SP)+,R2
1172 026472 012601 MOV (SP)+,R1
1173 026474 000205 RTS R5 ;EXIT

```

```

1175 ;ROUTINE TO SET DRIVE IN SECTOR LIST
1176 ;CALL: JSR R5,SETLST ;R0 HAS SECTOR
1177 ;DRIVE GOTTEN FROM R4
1178
1179 026476 010146 SETLST: MOV R1,-(SP) ;SAVE R1
1180
1181 026500 162700 000034 SUB #28.,R0 ;START LIST AT 0
1182 026504 100002 BPL 3$
1183 026506 062700 000050 ADD #40.,R0
1184 026512 012701 002402 3$: MOV #SECLST,R1 ;BEGINNING OF SECTOR LIST
1185 026516 005700 1$: TST R0 ;FOUND SECTOR?
1186 026520 001403 BEQ 2$ ;BRANCH IF YES
1187 026522 005300 DEC R0 ;DECREMENT SECTOR
1188 026524 005721 TST (R1)+ ;NEXT ENTRY IN LIST
1189 026526 000773 BR 1$ ;GO BACK
1190 026530 010411 2$: MOV R4,(R1) ;STORE DRIVE BITS IN LIST
1191 026532 012601 MOV (SP)+,R1 ;RESTORE R1
1192 026534 000205 RTS R5
1193
1194 ;ROUTINE TO LOCATE DRIVE THAT WROTE SECTOR LAST
1195 ;CALL: JSR R5,FNDDRV ;R0-CONTAINS SECTOR
1196 ;ON EXIT R0-DRIVE
1197
1198 026536 010146 FNDDRV: MOV R1,-(SP) ;SAVE R1
1199 026540 162700 000034 SUB #28.,R0 ;START LIST AT 0
1200 026544 100002 BPL 3$
1201 026546 062700 000050 ADD #40.,R0
1202 026552 012701 002402 3$: MOV #SECLST,R1 ;START OF LIST
1203 026556 005700 1$: TST R0 ;FOUND SECTOR?
1204 026560 001403 BEQ 2$ ;YES, GET DRIVE #, EXIT
1205 026562 005300 DEC R0 ;NO, DOWN COUNT SECTOR
1206 026564 005721 TST (R1)+ ;NEXT ENTRY IN LIST
1207 026566 000773 BR 1$ ;GO BACK
1208 026570 011100 2$: MOV (R1),R0 ;GET DRIVE BUFFER POINTER
1209 026572 012601 MOV (SP)+,R1 ;RESTORE R1
1210 026574 000205 RTS R5 ;EXIT

```



```

1212
1213
1214
1215
1216
1217
1218
1219
1220 026576 010046
1221 026600 010146
1222 026602 010246
1223 026604 012737 000034 003132
1224 026612 012701 100000
1225 026616 016437 000006 003136
1226
1227 026624 012737 177600 003206 1$:
1228 026632 012737 003232 003202
1229 026640 042737 000077 003204 2$:
1230 026646 053737 003132 003204
1231 026654 030103
1232 026656 001521
1233 026660 004537 032002
1234 026664 000014
1235
1236 026666 005737 003164
1237 026672 100107
1238
1239 026674 005737 003062
1240 026700 001412
1241 026702 012737 017325 003072
1242 026710 016437 000000 003070
1243 026716 016437 000005 003134
1244 026724 000415
1245 026726 012737 017574 003072 21$:
1246 026734 013700 003132
1247 026740 004537 026536
1248 026744 016037 000000 003070
1249 026752 116037 000005 003134
1250 026760
(4) 026760 104455
(5) 026762 000015
(5) 026764 017735
(5) 026766 020416
1251 026770 005037 003142
1252 026774 005037 003144
1253 027000 012702 003232
1254 027004 023712 003136 3$:
1255 027010 001417
1256 027012 005237 003142
1257 027016
(10) 027016 011246
(9) 027020 013746 003136
(8) 027024 013746 003144
(7) 027030 012746 021371
(6) 027034 012746 000004
(3) 027040 010600

```

```

:
:ROUTINE TO VERIFY THAT THE OVERWRITE DID ACTUALLY OVERWRITE THE
:PREVIOUS DATA ON THE PACK.
:
:CALL: JSR R5,VEROW USES R3 AS BIT MAP OF SECTORS TO
: CHECK. R3 IS LOADED PRIOR TO
: WRITING SECTORS.
:
VEROW: MOV R0,-(SP) ;SAVE REGISTER CONTENTS
MOV R1,-(SP)
MOV R2,-(SP)
MOV #28,SECT ;START VERIFY AT SECTOR 28
MOV #100000,R1 ;BIT MASK FOR VERIFICATION
MOV PAT(R4),GDATA ;GET PATTERN FOR THIS DRIVE
1$: MOV #-128.,BMP ;SET UP READ-ONE SECTOR
MOV #BUF,BBA ;BUS ADDRESS
2$: BIC #77,BDA ;CLEAR OUT SECTOR BITS
BIS SECT,BDA ;SET SECTOR
BIT R1,R3 ;DO WE READ THIS ONE?
BEQ 5$ ;NO, BRANCH
JSR R5,LDFUNC ;READ
READ
TST E.CS ;ERROR
BPL 4$ ;NO CONTINUE
TST FOWR ;INITIAL WRITE
BEQ 21$ ;NO
MOV #INITWR,REASON ;SETUP INITIAL WRITE OF SECTOR
MOV CSR(R4),LSTCLR
MOV DSB+1(R4),LSTDRV
BR 22$
21$: MOV #OVMS,REASON ;SET MESSAGE FOR OVERWRITE
MOV SECT,R0 ;FIND DRIVE THAT LAST WROTE
JSR R5, FNDDRV ;SECTOR
MOV CSR(R0),LSTCLR ;GET IT'S CSR
MOVB DSB+1(R0),LSTDRV ;GET THE DRIVE
22$: ERRDF 13.,OVWER,ERR4 ;PRINT ERROR
TRAP C$ERDF
.WORD 13
.WORD OVWER
.WORD ERR4
CLR WCOUNT ;CLEAR BAD WORD COUNT W/IN SECTOR
CLR SECWRD ;CLEAR WORD IN SECTOR
MOV #BUF,R2 ;GET BUFFER START
3$: CMP GDATA,(R2) ;IS DATA CORRECT?
BEQ 31$ ;YES CHECK NEXT
INC WCOUNT ;NO ACCOUNT FOR IT
PRINTF #FRM8,SECWRD,GDATA,(R2)
MOV (R2),-(SP)
MOV GDATA,-(SP)
MOV SECWRD,-(SP)
MOV #FRM8,-(SP)
MOV #4,-(SP)
MOV SP,R0

```

```

(4) 027042 104417          TRAP  C$PNTF
(4) 027044 062706 000012  ADD   #12,SP
1258
1259 027050 005722          31$:  TST   (R2)+      ;NEXT
1260 027052 005237 003144  INC   SECWRD      ;NEXT
1261 027056 023727 003144 000200  CMP   SECWRD,#128. ;DONE WITH SECTOR?
1262 027064 001347          BNE   3$          ;NO GO BACK
1263
1264 027066          PRINTF #FRM9,WCOUNT ;PRINT SUMMARY
(8) 027066 013746 003142  MOV   WCOUNT,-(SP)
(7) 027072 012746 021435  MOV   #FRM9,-(SP)
(6) 027076 012746 000002  MOV   #2,-(SP)
(3) 027102 010600          MOV   SP,R0
(4) 027104 104417          TRAP  C$PNTF
(4) 027106 062706 000006  ADD   #6,SP
1265
1266 027112 013700 003132  4$:  MOV   SECT,R0      ;SET SECTOR IN LIST TO THE
1267 027116 004537 026476  JSR   R5,SETLST      ;CREDIT OF THIS DRIVE
1268
1269 027122 005237 003132  5$:  INC   SECT          ;NEXT SECTOR
1270 027126 023727 003132 000050  CMP   SECT,#40.
1271 027134 001003          BNE   6$
1272 027136 162737 000050 003132  SUB   #40.,SECT
1273 027144 000241          6$:  CLC           ;CLEAR CARRY
1274 027146 006001          ROR   R1           ;NEXT BIT
1275 027150 103225          BCC   1$          ;IF CLEAR NEXT
1276
1277 027152 012602          MOV   (SP)+,R2      ;RESTORE R2-R0, EXIT
1278 027154 012601          MOV   (SP)+,R1
1279 027156 012600          MOV   (SP)+,R0
1280 027160 000205          RTS   R5
  
```



```

1282 ;ROUTINE TO VERIFY THAT A DRIVE CAN RECOVER ANOTHER DRIVE'S DATA.
1283 ;
1284 ;CALL: JSR R5,VEROD USES R3 AS BIT MAP OF SECTORS TO
1285 ; CHECK. R3 IS LOAD BY WRSEC (WE
1286 ; USE R3 COMPLIMENTED.
1287 ;
1288 ;
1289 027162 010046 VEROD: MOV R0,-(SP) ;SAVE R0-R2
1290 027164 010146 MOV R1,-(SP)
1291 027166 010246 MOV R2,-(SP)
1292 027170 012701 100000 MOV #100000,R1 ;BIT MASK FOR SECTORS.
1293 027174 012737 000034 003132 MOV #28.,SECT ;START WITH SECTOR 28
1294 027202 005737 003062 TST FOWR ;CHECK FOR FIRST OVERWRITE
1295 027206 001134 BNE 6$
1296 ;
1297 027210 012737 177600 003206 1$: MOV #-128.,BMP ;SET UP READ (ONE SECTOR)
1298 027216 012737 003232 003202 MOV #BUF,BBA ;BUS ADDRESS
1299 027224 042737 000077 003204 2$: BIC #77,BDA ;CLEAR SECTOR BITS
1300 027232 053737 003132 003204 BIS SECT,BDA ;SET IN SECTOR BITS
1301 027240 030103 BIT R1,R3 ;CHECK THIS SECTOR?
1302 027242 001103 BNE 5$ ;NO BRANCH
1303 ;
1304 027244 013700 003132 MOV SECT,R0 ;FIND DRIVE THAT WROTE
1305 027250 004537 026536 JSR R5,FNDDRV ;SECTOR LAST
1306 027254 016037 000000 003070 MOV CSR(R0),LSTCLR ;GET CSR OF DRIVE
1307 027262 116037 000005 003134 MOVSB DSB+1(R0),LSTDV ;GET DRIVE
1308 027270 016037 000006 003136 MOV PAT(R0),GDATA ;GET PATTERN
1309 ;
1310 027276 004537 032002 JSR R5,LDFUNC ;READ
1311 027302 000014 READ
1312 ;
1313 027304 005737 003164 TST E.CS ;ERROR?
1314 027310 100060 BPL 5$ ;NO, NEXT SECTOR
1315 027312 012737 017627 003072 MOV #RECMS,REASON ;SET READ RECOVERY MESSAGE
1316 027320 ERRDF 14.,RECER,ERR4 ;REPORT ERROR
(4) 027320 104455 TRAP C$ERDF
(5) 027322 000016 .WORD 14
(5) 027324 017755 .WORD RECER
(5) 027326 020416 .WORD ERR4
1317 ;
1318 027330 005037 003142 CLR WCOUNT ;CLEAR BAD WORD COUNT
1319 027334 005037 003144 CLR SECWRD ;CLEAR WORD W/I SECTOR
1320 027340 012702 003232 MOV #BUF,R2 ;START OF BUFFER
1321 027344 023712 003136 3$: CMP GDATA,(R2) ;DATA COMPARE
1322 027350 001417 BEQ 4$ ;YES, CHECK NEXT
1323 ;
1324 027352 005237 003142 INC WCOUNT ;ACCOUNT FOR ERROR
1325 027356 PRINTF #FRM8,SECWRD,GDATA,(R2) ;PRINT ERROR
(10) 027356 011246 MOV (R2),-(SP)
(9) 027360 013746 003136 MOV GDATA,-(SP)
(8) 027364 013746 003144 MOV SECWRD,-(SP)
(7) 027370 012746 021371 MOV #FRM8,-(SP)
(6) 027374 012746 000004 MOV #4,-(SP)
(3) 027400 010600 MOV SP,R0
(4) 027402 104417 TRAP C$PNTF
(4) 027404 062706 000012 ADD #12,SP

```

```
1326  
1327 027410 005722          4$:  TST      (R2)+      ;NEXT  
1328 027412 005237 003144  INC      SECWRD     ;NEXT WORD IN SECTOR  
1329 027416 023727 003144 000200 CMP      SECWRD,#128. ;DONE?  
1330 027424 001347          BNE      3$         ;NO  
1331 027426          PRINTF  #FRM9,WCOUNT ;PRINT SUMMARY  
   (8) 027426 013746 003142  MOV      WCOUNT,-(SP)  
   (7) 027432 012746 021435  MOV      #FRM9,-(SP)  
   (6) 027436 012746 000002  MOV      #2,-(SP)  
   (3) 027442 010600          MOV      SP,R0  
   (4) 027444 104417          TRAP    C$PNTF  
   (4) 027446 062706 000006  ADD      #6,SP  
1332  
1333 027452 005237 003132 5$:  INC      SECT      ;NEXT SECTOR  
1334 027456 023727 003132 000050 CMP      SECT,#40.  
1335 027464 001002          BNE      7$  
1336 027466 005037 003132  CLR      SECT  
1337 027472 000241          7$:  CLC  
1338 027474 006001          ROR      R1      ;NEXT BIT MAP  
1339 027476 103244          BCC     1$  
1340  
1341 027500 012602          6$:  MOV      (SP)+,R2  ;RESTORE R2-R0, EXIT  
1342 027502 012601          MOV      (SP)+,R1  
1343 027504 012600          MOV      (SP)+,R0  
1344 027506 000205          RTS      R5
```



```

1346 ;ROUTINE TO VERIFY THE ADJ. CYL. WRITE IS GOOD
1347 ;USES R3 AND WORD FOLLOWING CALL
1348 ;IF WRITE WAS GOOD,SECTOR WILL BE STORED IN MAP
1349 ;USING RSADJS/.WORD 1
1350
1351 027510 010046          VAJWR:  MOV    R0,-(SP)          ;SAVE REGISTERS
1352 027512 010146          MOV    R1,-(SP)
1353 027514 010246          MOV    R2,-(SP)
1354 027516 012701 100000  MOV    #100000,R1      ;BIT MASK FOR CYLINDER
1355 027522 012502          MOV    (R5)+,R2      ;STARTING SECTOR
1356 027524 005000          CLR    R0
1357 027526 053700 003122  BIS    CYL,R0
1358 027532 012737 000007 003056  MOV    #7,TEM
1359
1360 027540 006300          2$:   ASL    R0
1361 027542 005337 003056  DEC    TEM
1362 027546 001374          BNE   2$
1363 027550 005737 003120  TST   SURF
1364 027554 001402          BEQ   3$
1365 027556 052700 000100  BIS   #HEAD,R0
1366 027562 050200          3$:   BIS   R2,R0
1367 027564 030103          4$:   BIT   R1,R3
1368 027566 001462          BEQ   5$
1369 027570 012737 177600 003206  MOV   #-128.,BMP
1370 027576 010037 003204  MOV   R0,BDA
1371 027602 010037 003066  MOV   R0,TEMP
1372 027606 042700 177700  BIC   #177700,R0
1373 027612 020027 000047  CMP   R0,#39.
1374 027616 003406          BLE   6$
1375 027620 162737 000050 003204  SUB   #40.,BDA
1376 027626 162737 000050 003066  SUB   #40.,TEMP
1377 027634 012737 003232 003202  6$:   MOV   #BUF,BBA
1378 027642 005037 003110  CLR   HSFLG
1379 027646 013700 003066  MOV   TEMP,R0
1380 027652 004537 032002  10$:  JSR   R5,LDFUNC      ;READ FUNCTION
1381 027656 000014          READ
1382 027660 005737 003074  TST   ERFLG
1383 027664 001416          BEQ   7$
1384 027666 005737 003110  TST   HSFLG
1385 027672 001007          BNE   11$
1386 027674          ERRSOFT 120.,READ1,ERR2
1387 (4) 027674 104457          TRAP  C$ERSOFT
1388 (5) 027676 000170          .WORD 120
1389 (5) 027700 020115          .WORD READ1
1390 (5) 027702 020230          .WORD ERR2
1391 1387 027704 005237 003110  INC   HSFLG
1392 1388 027710 000760          BR    10$
1393 1389 027712          ERRHRD 130.,READ1,ERR2
1394 (4) 027712 104456          TRAP  C$ERHRD
1395 (5) 027714 000202          .WORD 130
1396 (5) 027716 020115          .WORD READ1
1397 (5) 027720 020230          .WORD ERR2
1398 1390 027722 010046          7$:   MOV   R0,-(SP)
1399 1391 027724 004537 026330  JSR   R5,RSADJS      ;STORE ADJ. CYL. SECTOR INFO.
1400 1392 027730 000001          .WORD 1
1401 1393 027732 012600          MOV   (SP)+,R0      ;RESTORE R0

```

1394	027734	005200	5\$:	INC	R0
1395	027736	000241		CLC	
1396	027740	006001		ROR	R1
1397	027742	103310		BCC	4\$
1398	027744	012602		MOV	(SP)+,R2
1399	027746	012601		MOV	(SP)+,R1
1400	027750	012600		MOV	(SP)+,R0
1401	027752	000205		RTS	R5


```

1403 :ROUTINE TO VERIFY THAT WRITE DID NOT DISTURB ADJACENT TRACKS
1404 :WRITTEN BY OTHER DRIVES.
1405 :CALL JSR R5,BSVWR
1406 :
1407 :
1408 :
1409 :USES 'ADJLOC' TO GET +/-1 CYLINDER OFFSET
1410 :USES R3 FOR SECTOR MAP, USES MAP AT 'SECBUF' FOR INFO
1411 027754 010046 BSVWR: MOV R0,-(SP) ;SAVE REGISTERS
1412 027756 010146 MOV R1,-(SP)
1413 027760 010246 MOV R2,-(SP)
1414 027762 013746 003122 MOV CYL,-(SP)
1415 027766 013746 003120 MOV SURF,-(SP)
1416 027772 012546 MOV (R5)+,-(SP) ;GET STARTING SECTOR
1417 027774 123727 003100 000003 CMPB ADJLOC,#3 ;ON MIDDLE TRACK???
1418 030002 001455 BEQ BSEXIT ;YES, THEN NO CHECK
1419 030004 162716 000042 SUB #34.,(SP) ;SETUP SECTOR START FOR OUTSIDE
1420 030010 100002 BPL 1$ ;IF POSITIVE OKAY ELSE FIX
1421 030012 062716 000050 ADD #40.,(SP) ;FIX IT
1422 030016 123727 003100 000001 1$: CMPB ADJLOC,#1 ;ON OUTER LIMIT???
1423 030024 001412 BEQ INAWR ;YES,SKIP CHECK
1424 030026 105337 003100 DECB ADJLOC ;OUTER ADJ TRACK
1425 030032 005337 003122 DEC CYL
1426 030036 004537 030164 JSR R5,CHECK ;GO CHECK ADJ SECTORS
1427 030042 005237 003122 INC CYL ;FIX BACK
1428 030046 105237 003100 INCB ADJLOC
1429 030052 062716 000104 INAWR: ADD #68.,(SP) ;INNER SECTOR START
1430 030056 021627 000050 CMP (SP),#40. ;WITHIN LIMITS???
1431 030062 002407 BLT 1$ ;YES, OKAY
1432 030064 162716 000050 SUB #40.,(SP) ;FIX SECTOR
1433 030070 021627 000050 CMP (SP),#40.
1434 030074 002402 BLT 1$
1435 030076 162716 000050 SUB #40.,(SP)
1436 030102 123727 003100 000005 1$: CMPB ADJLOC,#5 ;INNER LIMIT??
1437 030110 001412 BEQ BSEXIT ;YES,SKIP CHECK
1438 030112 105237 003100 INCB ADJLOC ;FIX FOR INNER
1439 030116 005237 003122 INC CYL
1440 030122 004537 030164 JSR R5,CHECK ;GO CHECK ADJ SECTORS
1441 030126 105337 003100 DECB ADJLOC ;FIX BACK
1442 030132 005337 003122 DEC CYL
1443 030136 005726 BSEXIT: TST (SP)+ ;THROW OFF SECTOR
1444 030140 012637 003120 MOV (SP)+,SURF
1445 030144 012637 003122 MOV (SP)+,CYL
1446 030150 012602 NCHECK: MOV (SP)+,R2
1447 030152 012601 MOV (SP)+,R1
1448 030154 012600 MOV (SP)+,R0
1449 030156 004537 025652 JSR R5,SKCYL ;SEEK BACK
1450 030162 000205 RTS R5 ;RETURN
  
```

```
1452 ;ROUTINE TO VERIFY AN ADJACENT SECTOR
1453 ;CALLED FROM BSVWR
1454 ;
1455 ;
1456 030164 012701 100000 CHECK: MOV #100000,R1 ;SECTOR MASK
1457 030170 004537 025652 JSR R5,SKCYL ;GET TO DESIRED CYLINDER
1458 030174 005002 CLR R2 ;CREATE ADDRESS
1459 030176 053702 003122 BIS CYL,R2
1460 030202 012737 000007 003056 MOV #7,TEM
1461 030210 006302 2$: ASL R2
1462 030212 005337 003056 DEC TEM
1463 030216 001374 BNE 2$
1464 030220 005737 003120 TST SURF
1465 030224 001402 BEQ 3$ ;NO
1466 030226 052702 000100 BIS #HEAD,R2
1467 030232 056602 000002 3$: BIS 2(SP),R2 ;SET IN SECTOR
1468 030236 030103 4$: BIT R1,R3 ;THIS SECTOR IN LIST???
1469 030240 001452 BEQ 5$ ;NO, NEXT
1470 030242 010200 MOV R2,R0 ;COPY SECTOR
1471 030244 042700 177700 BIC #177700,R0 ;ONLY SECTOR LEFT
1472 030250 020027 000050 CMP R0,#40. ;SECTOR OKAY???
1473 030254 002404 BLT 6$ ;YES
1474 030256 162700 000050 SUB #40.,R0
1475 030262 162702 000050 SUB #40.,R2 ;FIX SECTOR
1476 030266 004537 026330 6$: JSR R5,RSADJS ;FIND IF SECTOR PREVIOUSLY WRITTEN
1477 030272 000000 .WORD 0
1478 030274 005700 TST R0 ;WAS IT??
1479 030276 001433 BEQ 5$ ;NO
1480 030300 010237 003204 MOV R2,BDA ;LOAD DISK ADDRESS
1481 030304 012737 177600 003206 MOV #-128.,BMP ;LOAD WC
1482 030312 004537 032002 JSR R5,LDFUNC ;LOAD
1483 030316 000014 READ
1484 030320 005737 003074 TST ERFLG ;WAS READ GOOD
1485 030324 001420 BEQ 5$
1486 030326 010346 MOV R3,-(SP)
1487 030330 010237 003132 MOV R2,SECT
1488 030334 010003 MOV R0,R3
1489 030336 042737 177700 003132 BIC #177700,SECT
1490 030344 ERRHRD 140.,ADJTXT,ERR3
(4) 030344 104456 TRAP C$ERHRD
(5) 030346 000214 .WORD 140
(5) 030350 020142 .WORD ADJTXT
(5) 030352 020270 .WORD ERR3
1491 030354 012603 MOV (SP)+,R3
1492 030356 ERRHRD 110.,READ1,ERR2
(4) 030356 104456 TRAP C$ERHRD
(5) 030360 000156 .WORD 110
(5) 030362 020115 .WORD READ1
(5) 030364 020230 .WORD ERR2
1493 030366 005202 5$: INC R2 ;NEXT SECTOR
1494 030370 000241 CLC
1495 030372 006001 ROR R1 ;SHIFT MASK
1496 030374 103320 BCC 4$
1497 030376 000205 RTS R5
```



```

1499 ;ROUTINE TO MERGE BAD SECTOR FILES
1500 ;ENTRY INTO THIS ROUTINE WILL OCCUR AFTER THE 'SERNUM' ROUTINE
1501 ;IS PERFORMED. THE FACTORY BAD SECTOR FILE WILL BE LOCATED IN
1502 ;FIRST 400(8) LOCATIONS.
1503 ;THIS ROUTINE WILL STORE THE FIELD BAD SECTORS INTO THE NEXT
1504 ;400 LOCATIONS AND THEN MERGE THE FACTORY BAD FILE
1505 ;WITH THE FIELD BAD FILE.
1506
1507 ;FACTORY BAD AT BUF
1508 ;FIELD BAD AT BUF + 512.
1509
1510 030400 010146 MERGE: MOV R1,-(SP) ;SAVE R1, R2, R3
1511 030402 010246 MOV R2,-(SP)
1512 030404 010346 MOV R3,-(SP)
1513 030406 012737 003632 003202 MOV #BUF+400,BBA ;BUFFER START FOR FIELD BAD
1514 030414 022737 000001 003060 CMP #1,T.DRIVE
1515 030422 001004 BNE 55$
1516 030424 012737 077724 003204 MOV #77724,BDA
1517 030432 000403 BR 66$
1518 030434 012737 177724 003204 55$: MOV #177724,BDA
1519
1520 030442 012737 177400 003206 66$: MOV #-256.,BMP
1521 030450 004537 032002 97$: JSR R5,LDFUNC ;LOAD READ FUNCTION
1522 030454 000014 READ
1523 030456 005737 003074 TST ERFLG ;TEST ERROR FLAG
1524 030462 001431 BEQ 98$ ;YES;MERGE BAD SECTOR FILES
1525 030464 062737 000004 003204 ADD #4,BDA ;TRY NEXT FIELD BAD SECTOR FILE
1526 030472 022737 000001 003060 CMP #1,T.DRIVE
1527 030500 001004 BNE 400$
1528 030502 022737 077750 003204 CMP #77750,BDA
1529 030510 001357 BNE 97$
1530
1531 030512 022737 177750 003204 400$: CMP #177750,BDA
1532 030520 001353 BNE 97$ ;NO,DO NEXT FIELD BAD SECTOR
1533 030522 PRINTF #FRM15
(7) 030522 012746 021771 MOV #FRM15,-(SP)
(6) 030526 012746 000001 MOV #1,-(SP)
(3) 030532 010600 MOV SP,R0
(4) 030534 104417 TRAP C$PNTF
(4) 030536 062706 000004 ADD #4,SP
1534 030542 999$: BREAK
(3) 030542 104422 TRAP C$BRK
1535 030544 000776 BR 999$
1536 030546 012701 003242 98$: MOV #BUF+10,R1 ;GET PAST ID ETC.
1537 030552 012702 000176 MOV #126.,R2 ;MAX = 126
1538 030556 005721 1$: TST (R1)+ ;SECTOR OR END
1539 030560 100404 BMI 2$ ;END, GO GET FIELD
1540 030562 005721 TST (R1)+ ;REST OF SECTOR
1541 030564 005302 DEC R2 ;MAX REACHED
1542 030566 001373 BNE 1$ ;NO, KEEP GOINC
1543 030570 000401 BR 3$ ;YES, SKIP BACK UP
1544 030572 005741 2$: TST -(R1) ;BACK UP PAST TERMINATOR
1545 030574 012703 000176 3$: MOV #126.,R3 ;SET 126 MAX
1546 030600 012702 003642 MOV #BUF+410,R2 ;GET FIELD SECTORS
1547 030604 012221 4$: MOV (R2)+,(R1)+ ;MERGE AT END OF FACTORY
1548 030606 100403 BMI 5$ ;DONE?

```

```
1549 030610 012221      MOV      (R2)+,(R1)+      ;NO, MERGE REST OF SECTOR
1550 030612 005303      DEC      R3              ;DONE
1551 030614 001373      BNE      4$              ;NO, GO BACK
1552 030616 012603      5$:    MOV      (SP)+,R3      ;RESTORE R3, R2, R1
1553 030620 012602      MOV      (SP)+,R2
1554 030622 012601      MOV      (SP)+,R1
1555 030624 000205      RTS      R5              ;EXIT
```


1557	030626	012537	003146		FNDTRK:	MOV	(R5)+,OFFSET	:GET INCREMENT/DECREMENT
1558	030632	012537	003156			MOV	(R5)+,SURFACE	:GET HEAD (SURFACE)
1559	030636	022737	000001	003060		CMP	#1,T.DRIVE	
1560	030644	001001				BNE	80\$	
1561	030646	000401				BR	90\$	
1562	030650	022525			80\$:	CMP	(R5)+,(R5)+	
1563	030652	012537	003152		90\$:	MOV	(R5)+,FRTRK	
1564	030656	012537	003150			MOV	(R5)+,LSTTRK	
1565	030662	005037	003160			CLR	TRKFND	:CLEAR OUT FLAG FOUND
1566	030666	005037	003162			CLR	TRKCNT	:CLEAR OUT TRACK COUNT
1567	030672	013737	003152	003154		MOV	FRTRK,PRSTRK	:GET FIRST TRACK
1568	030700				1\$:			
1569	030700	004537	031000			JSR	R5,FNDBSC	:IS TRACK IN BAD SECTOR FILE
1570	030704	005737	002234			TST	HDRFND	:WAS IT?
1571	030710	001003				BNE	2\$:YES, CLEAR TRKCNT
1572	030712	005237	003162			INC	TRKCNT	:NO, INDICATE GOOD TRACK
1573	030716	000402				BR	3\$:CONTINUE
1574	030720	005037	003162		2\$:	CLR	TRKCNT	:START COUNT OVER
1575	030724	023727	003162	000005	3\$:	CMP	TRKCNT,#5	:FIND 5 TRACKS YET?
1576	030732	001011				BNE	4\$:NO, CONTINUE
1577	030734	005237	003160			INC	TRKFND	:YES, EXIT WITH GOOD FLAG
1578	030740	022737	000001	003060		CMP	#1,T.DRIVE	
1579	030746	001002				BNE	81\$	
1580	030750	062705	000004			ADD	#4,R5	
1581								
1582	030754	000205			81\$:	RTS	R5	
1583	030756	023737	003154	003150	4\$:	CMP	PRSTRK,LSTTRK	:ARE WE DONE?
1584	030764	001001				BNE	5\$:NO, KEEP LOOKING
1585	030766	000205				RTS	R5	:EXIT WITH NOT FOUND
1586	030770	063737	003146	003154	5\$:	ADD	OFFSET,PRSTRK	:NEXT TRACK
1587	030776	000740				BR	1\$	

```

1589 ;ROUTINE TO FIND BAD TRACK IN FILE
1590
1591 031000 005037 002234 FNDBSC: CLR HDRFND ;INITIALIZE FLAG
1592 031004 010146 MOV R1,-(SP) ;SAVE R1, R2
1593 031006 010246 MOV R2,-(SP)
1594 031010 012701 003242 MOV #BUF+10,R1 ;SETUP FOR BEGINNING OF FILE
1595 031014 005711 1$: TST (R1) ;END?
1596 031016 100421 BMI 2$ ;IF MINUS AT END, EXIT
1597 031020 023721 003154 CMP PRSTRK,(R1)+ ;CYLINDER CORRECT?
1598 031024 001011 BNE 3$ ;NO, NEXT
1599 031026 105724 TSTB (R4)+ ;UPPER HALF OF WORD
1600 031030 123711 003156 CMPB SURFACE,(R1) ;CORRECT SURFACT
1601 031034 001402 BEQ 4$
1602 031036 105744 TSTB -(R4)
1603 031040 000403 BR 3$
1604 031042 005237 002234 4$: INC HDRFND ;SET FOUND
1605 031046 000405 BR 2$
1606
1607 031050 005721 3$: TST (R1)+ ;NEXT WORD
1608 031052 005202 INC R2 ;ACCOUNT FOR IT
1609 031054 020227 000374 CMP R2,#252. ;DONE?
1610 031060 001355 BNE 1$ ;NO, KEEP CHECKING
1611 031062 012601 2$: MOV (SP)+,R1 ;RESTORE R2, R1, EXIT
1612 031064 012602 MOV (SP)+,R2
1613 031066 000205 RTS R5
1614
1615 031070 013701 003154 FIXCYL: MOV PRSTRK,R1 ;GET TRACK WHICH IS GOOD
1616 031074 005737 003146 TST OFFSET ;WHICH WAY WERE WE LOOKING
1617 031100 100402 BMI 1$ ;IN WORD, BRANCH
1618 031102 162701 000004 SUB #4,R1 ;BACK IT UP BY FOUR
1619 031106 012702 000005 1$: MOV #5,R2 ;GOING STORE AWAY 5 TRACKS
1620 031112 010120 2$: MOV R1,(R0)+ ;STORE THEM 1 WD/PER
1621 031114 005201 INC R1
1622 031116 005302 DEC R2
1623 031120 001374 BNE 2$
1624 031122 000205 RTS R5
  
```



```

1626 ;ROUTINE TO GET SERIAL NUMBER
1627
1628 ;CALL JSR R5,SERNUM
1629
1630 SERNUM: MOV #13,BDA
1631 JSR R5,LDFUNC ;GET STATUS
1632 GSTAT
1633 JSR R5,LDFUNC ;READ HEADER
1634 RDHDR
1635 MOV E.MP,R0 ;GET THE HEADER
1636 1$: BIC #77,R0 ;CLEAR SECTOR BITS
1637 CMP #1,T.DRIVE
1638 BNE 23$
1639 CMP R0,#77700
1640 BEQ 2$
1641 23$: CMP R0,#177700
1642 BEQ 2$
1643 BIC #100,R0 ;CLEAR HEAD
1644 CMP #1,T.DRIVE
1645 BNE 32$
1646 MOV #77600,R1
1647 BR 33$
1648 32$: MOV #177600,R1
1649
1650 33$: SUB R0,R1
1651 MOV R1,BDA ;SET UP DIF WORD
1652 003204 BIS #25,BDA ;SEEK IN, HEAD 1
1653 JSR R5,LDFUNC ;SEEK
1654 SEEK
1655 JSR R5,LDFUNC ;VERIFY POSITION
1656 RDHDR
1657 MOV E.MP,R0 ;GET HEADER
1658 CMP #1,T.DRIVE
1659 BNE 42$
1660 CMP #77700,R0
1661 BR 43$
1662 42$: CMP #177700,R0
1663
1664 43$: BHIS 1$
1665 2$: CMP #1,T.DRIVE
1666 BNE 52$
1667 MOV #77700,BDA
1668 BR 97$
1669
1670 52$: MOV #177700,BDA
1671 97$: MOV #BUF,BBA
1672 MOV #-256.,BMP
1673 JSR R5,LDFUNC ;READ
1674 READ
1675 TST ERFLG ;TEST ERROR FLAG
1676 BEQ 98$ ;YES,COMPARE SERIAL NUMBERS
1677 ADD #4,BDA ;NO,SETUP FOR NEXT FACTORY BAD SECTOR
1678 CMP #1,T.DRIVE
1679 BNE 62$
1680 CMP #77724,BDA
1681 BNE 97$

```

```

1682 031414 000453          BR      99$
1683 031416 022737 177724 003204 62$:  CMP    #177724,BDA
1684 031424 001344          BNE    97$          ;GET NEXT FACTORY BAD SECTOR
1685 031426 000446          BR      99$          ;REPORT ERROR
1686 031430 012701 003232          MOV    #BUF,R1      ;COMPARE SERIAL NUMBERS
1687 031434 005737 003210          TST    SERNM1       ;HAVE WE GOT ONE TO COMPARE
1688 031440 100005          BPL    3$           ;YES, BRANCH
1689 031442 011137 003210          MOV    (R1),SERNM1  ;NO, CALL THIS ONE IT
1690 031446 016137 000002 003212          MOV    2(R1),SERNM2
1691 031454 021137 003210          CMP    (R1),SERNM1  ;
1692 031460 001004          BNE    4$           ;SERNUM OKAY
1693 031462 026137 000002 003212          CMP    2(R1),SERNM2 ;NO, PRINT ERROR
1694 031470 001437          BEQ    5$           ;OTHER HALF OKAY
1695 031472          4$:  PRINTF #FRM3,2(R1),(R1),SERNM2,SERNM1 ;YES, EXIT
      (11) 031472 013746 003210          MOV    SERNM1,-(SP)
      (10) 031476 013746 003212          MOV    SERNM2,-(SP)
      (9)  031502 011146          MOV    R1,-(SP)
      (8)  031504 016146 000002          MOV    2(R1),-(SP)
      (7)  031510 012746 021121          MOV    #FRM3,-(SP)
      (6)  031514 012746 000005          MOV    #5,-(SP)
      (3)  031520 010600          MOV    SP,R0
      (4)  031522 104417          TRAP   C$PNTF
      (4)  031524 062706 000014          ADD    #14,SP
1696 031530 004537 031572          JSR    R5,UNLOAD    ;LET OPERATOR CHANGE
1697 031534 004537 031676          JSR    R5,LOAD      ;PACK
1698 031540 000137 031124          JMP    SERNUM       ;GO CHECK IT AGAIN.
1699 031544          99$: PRINTF #FRM15          ;MESSAGE
      (7)  031544 012746 021771          MOV    #FRM15,-(SP)
      (6)  031550 012746 000001          MOV    #1,-(SP)
      (3)  031554 010600          MOV    SP,R0
      (4)  031556 104417          TRAP   C$PNTF
      (4)  031560 062706 000004          ADD    #4,SP
1700 031564          999$: BREAK
      (3)  031564 104422          TRAP   C$BRK
1701 031566 000776          BR     999$
1702 031570 000205          5$:  RTS    R5
  
```



```
1704 ;ROUTINE UNLOAD
1705
1706 ;CALL JSR R5,UNLOAD
1707
1708 UNLOAD: PRINTF #FRM1,<B,DSB+1(R4)>,CSR(R4)
(9) 031572 016446 000000 MOV CSR(R4),-(SP)
(8) 031576 005046 CLR -(SP)
(8) 031600 156416 000005 BISB DSB+1(R4),(SP)
(7) 031604 012746 020724 MOV #FRM1,-(SP)
(6) 031610 012746 000003 MOV #3,-(SP)
(3) 031614 010600 MOV SP,R0
(4) 031616 104417 TRAP C$PNTF
(4) 031620 062706 000010 ADD #10,SP
1709 031624 012701 000074 MOV #60.,R1 ;SETUP 60 SECOND TIMER
1710 031630 012700 000200 MOV #200,R0
1711 031634 056400 000004 BIS DSB(R4),R0
1712 031640 010074 000000 MOV R0,@CSR(R4)
1713 031644 032774 000001 000000 2$: BIT #DRDY,@CSR(R4) ;CHECK DRDY FOR ZERO
1714 031652 001410 BEQ 3$ ;PACK UNLOADED
1715 031654 WAITMS #10. ;WAIT 1 SECOND
1716 031666 005301 DEC R1 ;HAS 60 SEC PASSED?
1717 031670 001365 BNE 2$ ;NO, RETEST DRDY, CONTINUE WAIT
1718 031672 000737 BR UNLOAD ;YES, REPEAT MESSAGE CONTINUE WAIT
1719 031674 000205 3$: RTS R5 ;RETURN WITH PACK UNLOADED
1720
1721 ;ROUTINE LOAD
1722
1723 ;CALL JSR R5,LOAD
1724
1725 LOAD: PRINTF #FRM2,<B,DSB+1(R4)>,CSR(R4)
(9) 031676 016446 000000 MOV CSR(R4),-(SP)
(8) 031702 005046 CLR -(SP)
(8) 031704 156416 000005 BISB DSB+1(R4),(SP)
(7) 031710 012746 021021 MOV #FRM2,-(SP)
(6) 031714 012746 000003 MOV #3,-(SP)
(3) 031720 010600 MOV SP,R0
(4) 031722 104417 TRAP C$PNTF
(4) 031724 062706 000010 ADD #10,SP
1726 031730 012701 000170 MOV #120.,R1 ;SETUP 120 SEC TIMER
1727 031734 012700 000200 MOV #200,R0 ;SETUP CONTROLLER READY BIT
1728 031740 056400 000004 BIS DSB(R4),R0 ;SELECT DRIVE
1729 031744 010074 000000 MOV R0,@CSR(R4)
1730 031750 032774 000001 000000 2$: BIT #DRDY,@CSR(R4)
1731 031756 001010 BNE 3$
1732 031760 WAITMS #10.
1733 031772 005301 DEC R1
1734 031774 001365 BNE 2$
1735 031776 000737 BR LOAD
1736
1737 032000 000205 3$: RTS R5
```

```

1739          :ROUTINE LDFUNC
1740          :CALL   JSR     R5,LDFUNC
1741
1742 032002 010046          LDFUNC: MOV     R0,-(SP)
1743 032004 010346          MOV     R3,-(SP)
1744 032006 010146          MOV     R1,-(SP)
1745 032010 005037 003074  CLR     ERFLG          ;CLEAR ERROR FLAG
1746 032014 016403 000000  MOV     CSR(R4),R3      ;GET CSR
1747 032020 013763 003206 000006  MOV     BMP,MP(R3)     ;LOAD MULTIPURPOSE
1748 032026 013763 003204 000004  MOV     BDA,DA(R3)    ;LOAD DISK ADDRESS
1749 032034 013763 003202 000002  MOV     BBA,BA(R3)    ;LOAD BUS ADDRESS
1750 032042 011537 003200          MOV     (R5),BCS      ;GET FUNCTION TO LOAD
1751 032046 056437 000004 003200  BIS     DSB(R4),BCS   ;SELECT BITS
1752 032054 012701 000031          MOV     #25.,R1      ;SET WATCHDOG TO 250MS
1753 032060 052737 000200 003200  BIS     #200,BCS
1754 032066 013763 003200 000000  MOV     BCS,CS(R3)    ;LOAD FUNCTION
1755 032074 016337 000000 003200  MOV     CS(R3),BCS
1756 032102 042763 000200 000000  BIC     #200,CS(R3)
1757 032110 032763 000200 000000 1$: BIT     #200,CS(R3)  ;CNTLR READY?
1758 032116 001036          BNE     2$           ;YES, GO
1759 032120          WAITUS #100.      ;WAIT 100 MILLISECONDS
1760 032132 005301          DEC     R1
1761 032134 001365          BNE     1$
1762
1763 032136 016337 000000 003164  MOV     CS(R3),E.CS   ;READ ALL REGISTERS
1764 032144 016337 000002 003166  MOV     BA(R3),E.BA
1765 032152 016337 000004 003170  MOV     DA(R3),E.DA
1766 032160 016337 000006 003172  MOV     MP(R3),E.MP
1767 032166 016337 000006 003174  MOV     MP(R3),E.MP1
1768 032174 016337 000006 003176  MOV     MP(R3),E.MP2
1769 032202          ERRDF 210.,CNTTOT,ERR5;CNTLR TIMEOUT
1770 032212 000425          TRAP  C$ERDF
1771          (4) 032202 104455          .WORD 210
1772          (5) 032204 000322          .WORD CNTTOT
1773          (5) 032206 017300          .WORD ERR5
1774          (5) 032210 020576          BR     4$
1775
1776 032214 016337 000000 003164 2$: MOV     CS(R3),E.CS   ;READ ALL REGISTERS
1777 032222 016337 000002 003166  MOV     BA(R3),E.BA
1778 032230 016337 000004 003170  MOV     DA(R3),E.DA
1779 032236 016337 000006 003172  MOV     MP(R3),E.MP
1780 032244 016337 000006 003174  MOV     MP(R3),E.MP1
1781 032252 016337 000006 003176  MOV     MP(R3),E.MP2
1782
1783 032260 005737 003164          TST     E.CS          ;ANY ERRORS?
1784 032264 100002          BPL     3$           ;YES, GO SERVICE
1785 032266 005237 003074          4$: INC     ERFLG
1786 032272 005725          3$: TST     (R5)+
1787 032274 012601          MOV     (SP)+,R1
1788 032276 012603          MOV     (SP)+,R3
1789 032300 012600          MOV     (SP)+,R0
1790 032302 000205          RTS     R5
1791
1792          ENDMOD

```



```

1790 032304
1791
1792
1793
1794
1795
1796
1797 032304 012701 002442
1798 032310 012700 000170
1799 032314 005021
1800 032316 005300
1801 032320 001375
1802 032322 005237 003062
1803 032326 004537 024204
1804 032332 177400
1805 032334 000377
1806 032336 005037 003062
1807 032342 005237 003064
1808 032346 005237 003104
1809 032352 004537 024726
1810 032356 003 377
1811 032360 170000
1812 032362 003 000
1813 032364 007777
1814 032366 000000
1815 032370 004537 031572
1816 032374 062704 000010
1817 032400 004537 031676
1818 032404 004537 031124
1819 032410 004537 024204
1820 032414 000360
1821 032416 000017
1822 032420 005237 003104
1823 032424 004537 024726
1824 032430 002 360
1825 032432 000000
1826 032434 002 017
1827 032436 000000
1828 032440 004 360
1829 032442 000000
1830 032444 004 017
1831 032446 000000
1832 032450 000000
1833 032452 004537 031572
1834 032456 023727 003130 000002
1835 032464 001002
1836 032466 000137 033102
1837 032472 062704 000010
1838 032476 004537 031676
1839 032502 004537 031124
1840 032506 004537 024204
1841 032512 006014
1842 032514 001403
1843 032516 005237 003104
1844 032522 004537 024726
1845 032526 002 000
  
```

BGNTST

```

:CONTROL SECTION COMPATIBILITY PROGRAM
:PRINT UNLOAD AND LOAD DRIVE MESSAGES
:PERFORM SERIAL CHECK ROUTINE
:PERFORM READ/WRITE CHECKS ON DRIVES
  
```

```

COMPAT: MOV #SECBUF,R1 ;ADJ. CYLINDER BUFFER
MOV #120.,R0 ;ADJ. CYLINDER BUFFER COUNT
4$: CLR (R1)+ ;CLEAR ADJ. CYL. BUFFER AT STARTUP
DEC R0 ;BUFFER CLEARED?
BNE 4$ ;CLEAR NEXT BUFFER WORD
INC F0WR ;SET FIRST OVERWRITE FLAG
JSR R5,OVWPER ;PERFORM OVERWRITE ON FIRST DRIVE
177400
377
CLR F0WR ;CLEAR FIRST OVERWRITE
INC FADJ ;SET FIRST ADJ. FLAG
INC ADJDIR ;UP = 1
JSR R5,ADJCYL
.BYTE 3,377 ;TRACK AND SECTORS FOR
.WORD 170000 ;INWARD SEEK
.BYTE 3,0 ;TRACK AND SECTORS FOR
.WORD 7777 ;OUTWARD SEEK
.WORD 0 ;TERMINATOR
JSR R5,UNLOAD ;UNLOAD PACK FROM DRIVE UNIT
ADD #PAT+2,R4 ;UPDATE POINTER FOR NEXT DRIVE
JSR R5,LOAD ;LOAD INTO SECOND DRIVE UNIT
JSR R5,SERNUM ;CHECK PACK SERIAL NUMBER
JSR R5,OVWPER ;PERFORM R/W OVERWRITE
360
17
INC ADJDIR
JSR R5,ADJCYL
.BYTE 2,360 ;IN 1/0 OUTSIDE
.WORD 0
.BYTE 2,17 ;OUT 1/0 OUTSIDE
.WORD 0
.BYTE 4,360 ;IN 1/0 INSIDE
.WORD 0
.BYTE 4,17 ;OUT 1/0 INSIDE
.WORD 0
JSR R5,UNLOAD ;UNLOAD PACK FROM DRIVE UNIT
CMP UUT,#2 ;CHECK FOR > 2 DRIVES
BNE 10$ ;YES,GO TO NEXT DRIVE
JMP 2$ ;GO TO FIRST DRIVE
10$: ADD #PAT+2,R4 ;UPDATE DRIVE BUFFER FOR THIRD DRIVE
JSR R5,LOAD ;LOAD PACK FOR THIRD DRIVE
JSR R5,SERNUM ;CHECK SERIAL NUMBERS
JSR R5,OVWPER ;PERFORM R/W OVERWRITE ON THIRD DRIVE
6014
1403
INC ADJDIR
JSR R5,ADJCYL
.BYTE 2,0 ;IN 2/0 OUTSIDE
  
```


1846	032530	170000			.WORD	170000	
1847	032532	002	000		.BYTE	2,0	;OUT 2/0 OUTSIDE
1848	032534	007400			.WORD	7400	
1849	032536	004	000		.BYTE	4,0	;IN 2/0 INSIDE
1850	032540	170000			.WORD	170000	
1851	032542	004	000		.BYTE	4,0	;OUT 2/0 INSIDE
1852	032544	007400			.WORD	7400	
1853	032546	001	200		.BYTE	1,200	;IN 2/1 OUTSIDE
1854	032550	000000			.WORD	0	
1855	032552	001	100		.BYTE	1,100	;OUT 2/1 OUTSIDE
1856	032554	000000			.WORD	0	
1857	032556	005	200		.BYTE	5,200	;IN 2/1 INSIDE
1858	032560	000000			.WORD	0	
1859	032562	005	100		.BYTE	5,100	;OUT 2/1 INSIDE
1860	032564	000000			.WORD	0	
1861	032566	000000			.WORD	0	;TERMINATOR
1862	032570	004537	031572		JSR	R5,UNLOAD	;UNLOAD PACK ON THIRD DRIVE
1863	032574	023727	003130	000003	CMP	UUT,#3	;CHECK FOR > 3 DRIVES
1864	032602	001500			BEQ	1\$;NO, GO TO 2ND DRIVE
1865	032604	062704	000010		ADD	#PAT+2,R4	;UPDATE DRIVE BUFFER FOR 4TH DRIVE
1866	032610	004537	031676		JSR	R5,LOAD	;LOAD PACK ON 4TH DRIVE
1867	032614	004537	031124		JSR	R5,SERNUM	;CHECK PACK ON FOURTH DRIVE
1868	032620	004537	024204		JSR	R5,OVWPER	;PERFORM R/W OVERWRITE
1869	032624	001042				1042	
1870	032626	000421				421	
1871	032630	005237	003104		INC	ADJDIR	
1872	032634	004537	024726		JSR	R5,ADJCYL	
1873	032640	002	000		.BYTE	2,0	;IN 3/0 OUTSIDE
1874	032642	000360			.WORD	360	
1875	032644	002	000		.BYTE	2,0	;OUT 3/0 OUTSIDE
1876	032646	000017			.WORD	17	
1877	032650	004	000		.BYTE	4,0	;IN 3/0 INSIDE
1878	032652	000360			.WORD	360	
1879	032654	004	000		.BYTE	4,0	;OUT 3/0 INSIDE
1880	032656	000017			.WORD	17	
1881	032660	001	040		.BYTE	1,40	;IN 3/1 OUTSIDE
1882	032662	000000			.WORD	0	
1883	032664	001	020		.BYTE	1,20	;OUT 3/1 OUTSIDE
1884	032666	000000			.WORD	0	
1885	032670	005	040		.BYTE	5,40	;IN 3/1 INSIDE
1886	032672	000000			.WORD	0	
1887	032674	005	020		.BYTE	5,20	;OUT 3/1 INSIDE
1888	032676	000000			.WORD	0	
1889	032700	001	000		.BYTE	1,0	;IN 3/2 OUTSIDE
1890	032702	100000			.WORD	100000	
1891	032704	001	000		.BYTE	1,0	;OUT 3/2 OUTSIDE
1892	032706	040000			.WORD	40000	
1893	032710	005	000		.BYTE	5,0	;IN 3/2 INSIDE
1894	032712	100000			.WORD	100000	
1895	032714	005	000		.BYTE	5,0	;OUT 3/2 INSIDE
1896	032716	040000			.WORD	40000	
1897	032720	000000			.WORD	0	;TERMINATOR
1898	032722	004537	031572		JSR	R5,UNLOAD	;UNLOAD PACK FROM 4TH DRIVE
1899	032726	162704	000010		SUB	#PAT+2,R4	;SET DRIVE BUFFER FOR 3RD DRIVE
1900	032732	004537	031676		JSR	R5,LOAD	;LOAD PACK ON 3RD DRIVE
1901	032736	004537	031124		JSR	R5,SERNUM	;CHECK FOR PACK SERIAL NUMBER

1902	032742	004537	024204	JSR	R5,OVWPER	:PERFORM R/W OVERWRITE ON 3RD DRIVE
1903	032746	020000		20000		
1904	032750	010000		10000		
1905	032752	004537	024726	JSR	R5,ADJCYL	
1906	032756	001	000	.BYTE	1,0	:IN 2/3 OUTSIDE
1907	032760	000200		.WORD	200	
1908	032762	001	000	.BYTE	1,0	:OUT 2/3 OUTSIDE
1909	032764	000100		.WORD	100	
1910	032766	005	000	.BYTE	5,0	:IN 2/3 INSIDE
1911	032770	000200		.WORD	200	
1912	032772	005	000	.BYTE	5,0	:OUT 2/3 INSIDE
1913	032774	000100		.WORD	100	
1914	032776	000000		.WORD	0	:TERMINATOR
1915	033000	004537	031572	JSR	R5,UNLOAD	:UNLOAD PACK FROM 3RD DRIVE
1916	033004	162704	000010	1\$: SUB	#PAT+2,R4	:SET DRIVE BUFFER FOR 2ND DRIVE
1917	033010	004537	031676	JSR	R5,LOAD	:LOAD PACK ON THIRD DRIVE
1918	033014	004537	031124	JSR	R5,SERNUM	:CHECK PACK SERIAL NUMBER
1919	033020	004537	024204	JSR	R5,OVWPER	:PERFORM R/W OVERWRITE ON 2ND DRIVE
1920	033024	004040		4040		
1921	033026	002020		2020		
1922	033030	004537	024726	JSR	R5,ADJCYL	
1923	033034	001	000	.BYTE	1,0	:IN 1/2 OUTSIDE
1924	033036	020000		.WORD	20000	
1925	033040	001	000	.BYTE	1,0	:OUT 1/2 OUTSIDE
1926	033042	010000		.WORD	10000	
1927	033044	005	000	.BYTE	5,0	:IN 1/2 INSIDE
1928	033046	020000		.WORD	20000	
1929	033050	005	000	.BYTE	5,0	:OUT 1/2 INSIDE
1930	033052	010000		.WORD	10000	
1931	033054	001	000	.BYTE	1,0	:IN 1/3 OUTSIDE
1932	033056	000040		.WORD	40	
1933	033060	001	000	.BYTE	1,0	:OUT 1/3 OUTSIDE
1934	033062	000020		.WORD	20	
1935	033064	005	000	.BYTE	5,0	:IN 1/3 INSIDE
1936	033066	000040		.WORD	40	
1937	033070	005	000	.BYTE	5,0	:OUT 1/3 INSIDE
1938	033072	000020		.WORD	20	
1939	033074	000000		.WORD	0	:TERMINATOR
1940	033076	004537	031572	JSR	R5,UNLOAD	:UNLOAD PACK FROM 2ND DRIVE
1941	033102	162704	000010	2\$: SUB	#PAT+2,R4	:SET DRIVE BUFFER FOR 1ST DRIVE
1942	033106	004537	031676	JSR	R5,LOAD	:LOAD PACK INTO FIRST DRIVE UNIT
1943	033112	004537	031124	JSR	R5,SERNUM	:CHECK SERIAL NUMBER
1944	033116	004537	024204	JSR	R5,OVWPER	:PERFORM R/W OVERWRITE
1945	033122	001042		1042		
1946	033124	000421		421		
1947	033126	004537	024726	JSR	R5,ADJCYL	
1948	033132	001	010	.BYTE	1,10	:IN 0/1 OUTSIDE
1949	033134	000000		.WORD	0	
1950	033136	001	004	.BYTE	1,4	:OUT 0/1 OUTSIDE
1951	033140	000000		.WORD	0	
1952	033142	005	010	.BYTE	5,10	:IN 0/1 INSIDE
1953	033144	000000		.WORD	0	
1954	033146	005	004	.BYTE	5,4	:OUT 0/1 INSIDE
1955	033150	000000		.WORD	0	
1956	033152	001	000	.BYTE	1,0	:IN 0/2 OUTSIDE
1957	033154	004000		.WORD	4000	

1958	033156	001	000	.BYTE	1,0	;OUT 0/2 OUTSIDE
1959	033160	002000		.WORD	2000	
1960	033162	005	000	.BYTE	5,0	;IN 0/2 INSIDE
1961	033164	004000		.WORD	4000	
1962	033166	005	000	.BYTE	5,0	;OUT 0/2 INSIDE
1963	033170	002000		.WORD	2000	
1964	033172	001	000	.BYTE	1,0	;IN 0/3 OUTSIDE
1965	033174	000010		.WORD	10	
1966	033176	001	000	.BYTE	1,0	;OUT 0/3 OUTSIDE
1967	033200	000004		.WORD	4	
1968	033202	005	000	.BYTE	5,0	;IN 0/3 INSIDE
1969	033204	000010		.WORD	10	
1970	033206	005	000	.BYTE	5,0	;OUT 0/3 INSIDE
1971	033210	000004		.WORD	4	
1972	033212	000000		.WORD	0	; TERMINATOR
1973	033214	004537	031572	JSR	R5,UNLOAD	;UNLOAD PACK
1974	033220			PRINTF	#ENDPAS	;END OF PASS
(7)	033220	012746	022214	MOV	#ENDPAS,-(SP)	
(6)	033224	012746	000001	MOV	#1,-(SP)	
(3)	033230	010600		MOV	SP,R0	
(4)	033232	104417		TRAP	C\$PNTF	
(4)	033234	062706	000004	ADD	#4,SP	
1975	033240			3\$: BREAK		
(3)	033240	104422		TRAP	C\$BRK	
1976	033242	000776		BR	3\$	
1977						
1978						
1979						
1980	033244			ENDTST		
(3)	033244			L10014:		
(3)	033244	104401		TRAP	C\$SETST	
1981						
1982	033246			BGNMOD	HRDPRM	
1983	033246			BGNHRD		
(3)	033246	000020		.WORD	L10015-L\$HARD/2	
1984						
1985	033250			GPRMA	CSRMSG,CSR,0,160000,177776,YES	
(4)	033250	000031		.WORD	T\$CODE	
(4)	033252	033310		.WORD	CSRMSG	
(4)	033254	160000		.WORD	T\$LOLIM	
(4)	033256	177776		.WORD	T\$HILIM	
1986	033260			GPRMA	VECMSG,VECT,0,0,776,YES	
(4)	033260	001031		.WORD	T\$CODE	
(4)	033262	033346		.WORD	VECMSG	
(4)	033264	000000		.WORD	T\$LOLIM	
(4)	033266	000776		.WORD	T\$HILIM	
1987	033270			GPRMD	DRMSG,DRBT,0,03400,0,7,YES	
(4)	033270	003032		.WORD	T\$CODE	
(4)	033272	033355		.WORD	DRMSG	
(4)	033274	003400		.WORD	03400	
(4)	033276	000000		.WORD	T\$LOLIM	
(4)	033300	000007		.WORD	T\$HILIM	
1988	033302			GPRML	DRTYPE,TYPDR,1,YES	
(4)	033302	002130		.WORD	T\$CODE	
(4)	033304	033324		.WORD	DRTYPE	
(4)	033306	000001		.WORD	1	


```
1989 033310          ENDHRD
      (2)           .EVEN
      (3) 033310    L10015:
1990
1991 033310 052502 020123 042101 CSRMSG: .ASCIZ /BUS ADDRESS/
      033316 051104 051505 000123
1992 033324 051104 053111 020105 DRTYPE: .ASCIZ /DRIVE TYPE = RL01/
      033332 054524 042520 036440
      033340 051040 030114 000061
1993 033346 042526 052103 051117 VECMSG: .ASCIZ /VECTOR/
      033354      000
1994 033355      104 044522 042526 DRMSG: .ASCIZ /DRIVE/
      033362      000
1995
1996          033364          .EVEN
1997
1998 033364          ENDMOD
1999
2000 033364          LASTAD
      (2)           .EVEN
      (4) 033364 000000          .WORD 0
      (4) 033366 000000          .WORD 0
      (3) 033370
2001          L$LAST::
2002          000001          .END
```


FSSW = 000014	22#							
F\$TEST= 000001	22#	1790	1980					
GDATA 003136	212#	1225*	1254	1257	1308*	1321	1325	
GLBDAT 002234 G	98#							
GLBEQA 002234 G	40#							
GLBERR 020172 G	302#							
GLBSUB 023724 G	736#							
GLBTXT 017300 G	269#							
G\$BIT = 000003	77#							
G\$STAT = 000004	71#	1632						
G\$CNT0= 000200	22#							
G\$DELM= 000372	22#	742	747	759	763			
G\$DISP= 000003	22#							
G\$EXCP= 000400	22#							
G\$HILI= 000002	22#							
G\$LOLI= 000001	22#							
G\$NO = 000000	22#							
G\$OFFS= 000400	22#	1985	1986	1987	1988			
G\$OF SI= 000376	22#	1985	1986	1987	1988			
G\$PRMA= 000001	22#	1985	1986					
G\$PRMD= 000002	22#	1987						
G\$PRML= 000000	22#	1988						
G\$RADA= 000140	22#							
G\$RADB= 000000	22#							
G\$RADD= 000040	22#							
G\$RADL= 000100	22#	1988						
G\$RADO= 000000	22#	1985	1986	1987				
G\$XFER= 000004	22#							
G\$YES = 000010	22#	1985	1986	1987	1988			
HCRC = 004000	64#							
HDRFND 002234	100#	1570	1591*	1604*				
HEAD = 000100	81#	857	1099	1365	1466			
HEAD01 003114	203#	893*	926	1019	1021*	1023*		
HNF = 010000	65#							
HOE = 100000 G	42#							
HPTCOD 022250 G	392#							
HRDPRM 033246 G	1982#							
HSFLG 003110	201#	862*	876	879*	1378*	1384	1387*	
IBE = 010000 G	42#							
IDU = 000040 G	42#							
IER = 020000 G	42#							
INAWR 030052	1423	1429#						
INITCO 022266 G	413#							
INITWR 017325	279#	1241						
INN10 002356	145#	526						
INN11 002370	150#	541						
INN20 002360	146#							
INN21 002372	151#							
INN30 002362	147#	174						
INN31 002374	152#	179						
INN40 002364	148#							
INN41 002376	153#							
INN50 002366	149#							
INN51 002400	154#	1025						
INTEN = 000100	58#							
ISR = 000100 G	42#							

CZRLLO RL01/02 DRIVE COMPAT
CZRLLO.MAC 07-DEC-79 10:40

MACY11 30A(1052) 17-DEC-79 11:21 PAGE 8-9
CROSS REFERENCE TABLE -- USER SYMBOLS

	454#	474#	1113#	1114#	1125#	1257#	1264#	1325#	1331#	1533#	1695#	1699#	1708#
T\$CODE= 002130	1725#	1974#											
T\$ERRN= 000322	1985#	1986#	1987#	1988#									
	22#	422#	428#	494#	509#	523#	538#	561#	585#	609#	633#	657#	681#
T\$EXCP= 000000	693#	878#	881#	1103#	1106#	1250#	1316#	1386#	1389#	1490#	1492#	1769#	
T\$GMAN= 000000	1985#	1986#	1987#										
T\$HILI= 000007	22#												
T\$LAST= 000001	1985#	1986#	1987#										
T\$LOLI= 000000	22#	2000#											
T\$LSYM= 010000	1985#	1986#	1987#										
	22#	308	313	321	331	336	342	398	703	713	722	728	1980
T\$LTNO= 000001	1989												
T\$NEST= 177777	2000#												
	22#	28#	30#	40#	90#	98#	266#	269#	298#	302#	304#	308#	310#
	313#	315#	321#	323#	331#	333#	336#	338#	342#	377#	383#	389#	392#
	393#	398#	400#	403#	407#	413#	415#	703#	704#	708#	709#	713#	714#
	717#	718#	722#	723#	725#	726#	728#	729#	736#	1788#	1790#	1980#	1982#
T\$NSO = 000000	1983#	1989#	1998#										
	28#	30	40#	90	98#	266	269#	298	302#	377	383#	389	392#
	400	403#	407	413#	704	708#	714	717#	723	725#	729	736#	1788
T\$NS1 = 000004	1790#	1980	1982#	1998									
	304#	308	310#	313	315#	321	323#	331	333#	336	338#	342	393#
	398	415#	703	709#	713	718#	722	726#	728	1983#	1989		
T\$PTNU= 000000	22#												
T\$SAVL= 177777	22#												
T\$SEGL= 177777	22#												
T\$SUBN= 000000	22#	1790#											
T\$TAGL= 177777	22#												
T\$TAGN= 010016	22#	304#	310#	315#	323#	333#	338#	383#	393#	415#	709#	718#	726#
T\$TEMP= 000000	1790#	1983#											
	30#	90#	266#	298#	308#	313#	321#	331#	336#	342#	377#	389#	398#
	400#	405#	407#	703#	704#	713#	714#	722#	723#	728#	729#	1788#	1980#
T\$TEST= 000001	1985#	1986#	1987#	1988#	1989#	1998#							
T\$TSTM= 177777	22#	1790#	2000										
	22#	306	308	311	313	316	318	319	321	325	327	328	329
	331	334	336	339	341	342	417	422	428	437	452	454	474
	494	509	523	538	561	585	609	633	657	681	693	696	700
	703	713	722	728	740	756	878	881	1103	1106	1113	1114	1125
	1250	1257	1264	1316	1325	1331	1386	1389	1490	1492	1533	1534	1695
	1699	1700	1708	1725	1769	1974	1975	1980					
T\$TSTS= 000001	22#	1790#											
T\$\$AUT= 010011	709#	713											
T\$\$CLE= 010012	718#	722											
T\$\$DU = 010013	726#	728											
T\$\$HAR= 010015	1983#	1989											
T\$\$HW = 010007	393#	398											
T\$\$INI= 010010	415#	703											
T\$\$MSG= 010005	304#	308	310#	313	315#	321	323#	331	333#	336	338#	342	
T\$\$PRO= 010006	383#												
T\$\$TES= 010014	1790#	1980											
T.DRIV 003060	189#	440*	805	974	1514	1526	1559	1578	1637	1644	1658	1665	1678
T1 032304 G	405	1790#											
UAM = 000200 G	42#												
UNLOAD 031572	1696	1708#	1718	1815	1833	1862	1898	1915	1940	1973			
UUT 003130	209#	431*	435	445*	448*	1041	1834	1863					
VAJWR 027510	950	969	997	1010	1351#								

BCOMPL	741														
BGNAUT	709														
BGNCLN	718														
BGNDU	726														
BGNHRD	1983														
BGNHW	393														
BGNINI	415														
BGNMOD	28	40	98	269	302	392	403	413	708	717	725	736	1982		
BGNMSG	304	310	315	323	333	338									
BGNPRO	383														
BGNTST	1790														
BNCOMP	453	757													
BREAK	1534	1700	1975												
DELAY	742	747	759	763											
DESCRI	32														
DEVTYP	34														
DISPAT	405														
DOCLN	700														
DODU	696														
ENDAUT	713														
ENDCLN	722														
ENDDU	728														
ENDHRD	1989														
ENDHW	398														
ENDINI	703														
ENDMOD	30	90	266	298	377	400	407	704	714	723	729	1788	1998		
ENDMSG	308	313	321	331	336	342									
ENDPRO	389														
ENDTST	1980														
EQUALS	42														
ERRDF	1103	1106	1250	1316	1769										
ERRHRD	494	509	523	538	561	585	609	633	657	681	881	1389	1490	1492	
ERRSF	422	428	693												
ERRSOF	878	1386													
GPHARD	437														
GPRMA	1985	1986													
GPRMD	1987														
GPRML	1988														
HEADER	29														
LASTAD	2000														
MSBYTE	29#														
MSCNTO	1985#	1986#	1987#	1988#											
MSCOUN	306#	311#	316#	318#	319#	325#	327#	328#	329#	334#	339#	341#	454#	474#	1113#
	1114#	1125#	1257#	1264#	1325#	1331#	1533#	1695#	1699#	1708#	1725#	1974#			
MSDATA	29#	32#	34#												
MSDECR	30#	90#	266#	298#	308#	313#	321#	331#	336#	342#	377#	389#	398#	400#	407#
	703#	704#	713#	714#	722#	723#	728#	729#	1788#	1980#	1989#	1998#			
MSDEFA	1985#	1986#	1987#	1988#											
MSENDE	30#	90#	266#	298#	308#	313#	321#	331#	336#	342#	377#	398#	400#	407#	703#
	704#	713#	714#	722#	723#	728#	729#	1788#	1980#	1989#	1998#				
MSERRI	422#	428#	494#	509#	523#	538#	561#	585#	609#	633#	657#	681#	693#	878#	881#
	1103#	1106#	1250#	1316#	1386#	1389#	1490#	1492#	1769#						
MSEXCP	1985#	1986#	1987#												
MSGEN	28#	29#	32#	34#	40#	98#	269#	302#	304#	308#	310#	313#	315#	321#	323#
	331#	333#	336#	338#	342#	383#	392#	393#	398#	403#	405#	413#	415#	703#	708#
	709#	713#	717#	718#	722#	725#	726#	728#	736#	1790#	1980#	1982#	1983#	1989#	2000#

MSGETS	30#	90#	266#	298#	308#	313#	321#	331#	336#	342#	377#	389#	398#	400#	407#
	703#	704#	713#	714#	722#	723#	728#	729#	1788#	1980#	1989#	1998#			
MSGNGB	28#	29#	32#	34#	40#	98#	269#	302#	304#	310#	315#	323#	333#	338#	383#
	392#	393#	403#	405#	413#	415#	708#	709#	717#	718#	725#	726#	736#	1982#	1983#
	2000#														
MSGNIN	29#	32#	34#	306#	308#	311#	313#	316#	318#	319#	321#	325#	327#	328#	329#
	331#	334#	336#	339#	341#	342#	393#	405#	417#	422#	428#	437#	452#	453#	454#
	474#	494#	509#	523#	538#	561#	585#	609#	633#	657#	681#	693#	696#	700#	703#
	713#	722#	728#	740#	741#	742#	747#	756#	757#	759#	763#	878#	881#	1103#	1106#
	1113#	1114#	1125#	1250#	1257#	1264#	1316#	1325#	1331#	1386#	1389#	1490#	1492#	1533#	1534#
	1695#	1699#	1700#	1708#	1725#	1769#	1974#	1975#	1980#	1983#	1985#	1986#	1987#	1988#	1989#
	2000#														
MSGNTA	308#	313#	321#	331#	336#	342#	398#	703#	713#	722#	728#	1980#	1989#		
MSGNTE	1790#														
MSHAPT	29#														
MSHNAP	29#														
MSINCR	28#	40#	98#	269#	302#	304#	306#	308#	310#	311#	313#	315#	316#	318#	319#
	321#	323#	325#	327#	328#	329#	331#	333#	334#	336#	338#	339#	341#	342#	383#
	392#	393#	403#	413#	415#	417#	422#	428#	437#	452#	454#	474#	494#	509#	523#
	538#	561#	585#	609#	633#	657#	681#	693#	696#	700#	703#	708#	709#	713#	717#
	718#	722#	725#	726#	728#	736#	740#	756#	878#	881#	1103#	1106#	1113#	1114#	1125#
	1250#	1257#	1264#	1316#	1325#	1331#	1386#	1389#	1490#	1492#	1533#	1534#	1695#	1699#	1700#
	1708#	1725#	1769#	1790#	1974#	1975#	1980#	1982#	1983#						
MSLDRO	417#	437#	452#	696#											
MSMCHI	22#														
MSMCLO	22#														
MSPOP	30#	90#	266#	298#	308#	313#	321#	331#	336#	342#	377#	389#	398#	400#	407#
	703#	704#	713#	714#	722#	723#	728#	729#	1788#	1980#	1989#	1998#			
MSPRIN	306#	311#	316#	318#	319#	325#	327#	328#	329#	334#	339#	341#	454#	474#	1113#
	1114#	1125#	1257#	1264#	1325#	1331#	1533#	1695#	1699#	1708#	1725#	1974#			
MSPUSH	28#	40#	98#	269#	302#	304#	310#	315#	323#	333#	338#	383#	392#	393#	403#
	413#	415#	708#	709#	717#	718#	725#	726#	736#	1790#	1982#	1983#			
MSPUT	306#	311#	316#	318#	319#	325#	327#	328#	329#	334#	339#	341#	454#	474#	1113#
	1114#	1125#	1257#	1264#	1325#	1331#	1533#	1695#	1699#	1708#	1725#	1974#			
MSPUT1	306#	311#	316#	318#	319#	325#	327#	328#	329#	334#	339#	341#	454#	474#	1113#
	1114#	1125#	1257#	1264#	1325#	1331#	1533#	1695#	1699#	1708#	1725#	1974#			
MSRADI	1985#	1986#	1987#	1988#											
MSRNRO	437#														
MSSETS	28#	40#	98#	269#	302#	304#	310#	315#	323#	333#	338#	383#	392#	393#	403#
	413#	415#	708#	709#	717#	718#	725#	726#	736#	1790#	1982#	1983#			
MS SVC	306#	308#	311#	313#	316#	318#	319#	321#	325#	327#	328#	329#	331#	334#	336#
	339#	341#	342#	417#	422	428	437#	452#	454#	474#	494	509	523	538	561
	585	609	633	657	681	693	696#	700#	703#	713#	722#	728#	740#	756#	878
	881	1103	1106	1113#	1114#	1125#	1250	1257#	1264#	1316	1325#	1331#	1386	1389	1490
	1492	1533#	1534#	1695#	1699#	1700#	1708#	1725#	1769	1974#	1975#	1980#			
MS TLAB	306#	308#	311#	313#	316#	318#	319#	321#	325#	327#	328#	329#	331#	334#	336#
	339#	341#	342#	417#	422#	428#	437#	452#	454#	474#	494#	509#	523#	538#	561#
	585#	609#	633#	657#	681#	693#	696#	700#	703#	713#	722#	728#	740#	756#	878#
	881#	1103#	1106#	1113#	1114#	1125#	1250#	1257#	1264#	1316#	1325#	1331#	1386#	1389#	1490#
	1492#	1533#	1534#	1695#	1699#	1700#	1708#	1725#	1769#	1974#	1975#	1980#			
MS STL	306#	308#	311#	313#	316#	318#	319#	321#	325#	327#	328#	329#	331#	334#	336#
	339#	341#	342#	417#	422#	428#	437#	452#	454#	474#	494#	509#	523#	538#	561#
	585#	609#	633#	657#	681#	693#	696#	700#	703#	713#	722#	728#	740#	756#	878#
	881#	1103#	1106#	1113#	1114#	1125#	1250#	1257#	1264#	1316#	1325#	1331#	1386#	1389#	1490#
	1492#	1533#	1534#	1695#	1699#	1700#	1708#	1725#	1769#	1974#	1975#	1980#			
MSWORD	29#	405#	422#	428#	494#	509#	523#	538#	561#	585#	609#	633#	657#	681#	693#

	878#	881#	1103#	1106#	1250#	1316#	1386#	1389#	1490#	1492#	1769#	1985#	1986#	1987#	1988#
POINTE	2000														
PRINTB	26														
PRINTF	306	311	316	318	319	325	327	328	329	334	339	341	1113	1114	1125
READBU	454	474	1257	1264	1325	1331	1533	1695	1699	1708	1725	1974			
REDEF	740	756													
SETPRI	452														
SVC	417														
WAITMS	4#	22													
WAITUS	15#	1715	1732												
	10#	1759													

. ABS. 033370 000

ERRORS DETECTED: 0

.CZRLB.LST/CRF=SVC33/ML.CZRLB.MAC
RUN-TIME: 60 49 5 SECONDS
RUN-TIME RATIO: 276/115=2.3
CORE USED: 15K (29 PAGES)