

RL11,RLV11

DRIVE COMPATABILITY
CZRLFBO

AH-E251B-MC

COPYRIGHT © 77-78

FICHE 1 OF 1

JAN 1979

digital

MADE IN USA

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30
31	32	33	34	35	36
37	38	39	40	41	42
43	44	45	46	47	48
49	50	51	52	53	54
55	56	57	58	59	60
61	62	63	64	65	66
67	68	69	70	71	72
73	74	75	76	77	78
79	80	81	82	83	84
85	86	87	88	89	90
91	92	93	94	95	96
97	98	99	100	101	102

IDENTIFICATION

B 1

SEQ 0001

PRODUCT CODE: AC-E250B-MC
PRODUCT NAME: CZRLFBO RL01 DRIVE COMPATABILITY TEST
DATE CREATED: 11-OCT-78
MAINTAINER: DIAGNOSTIC ENGINEERING
AUTHOR: D. DEKNIS

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) 1977, 1978, DIGITAL EQUIPMENT CORPORATION

TH
CL
GI
*E
PA
DE
**
RE
**
TH
SP
CC
TH
NE
EN
TH

**
CC
**
CC
CC
TE
SC
MA

TH

TABLE OF CONTENTS

- 1.0 GENERAL INFORMATION
- 1.1 PROGRAM ABSTRACT
- 1.2 SYSTEM 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 SIX STEPS OF EXECUTION
 - 2.1.2 SAMPLE RUN-THROUGH
- 2.2 HOW TO CREATE A CHAINABLE FILE
- 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

- 4.0 PERFORMANCE AND PROGRESS REPORTS

- 5.0 DEVICE INFORMATION TABLES

- 6.0 TEST SUMMARIES

*1
PF
*1
CO
OF
EP
TH
*1
CO
*1
TH
HA
ME
TH
RU
EJ
TH
WJ
WJ
IF
FF
BE
AN
FJ
WJ
TO
LA
UT
*1
DF
*1
TH
OF
PF
TH
HC

1.0 GENERAL INFORMATION
1.1 PROGRAM ABSTRACT
1.1.1 STRUCTURE OF PROGRAM

THIS DIAGNOSTIC OCCUPIES 14.5K WORDS OF MEMORY AND 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 "CREATE CORE IMAGE" COMMAND BELOW FOR DETAILS OF CHAINING PROCEDURE). IT IS A SINGLE PROGRAM FROM THE STANDPOINT OF THE DIAGNOSTIC USER, BUT WE HAVE INCORPORATED INTO IT A CONTROL MODULE WHICH WILL LATER BE RELEASED INDEPENDENTLY AS A DIAGNOSTIC SUPERVISOR.

WHEN THIS DIAGNOSTIC IS STARTED AT ADDRESS 200, 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 (DS B>). AT COMMAND MODE THE OPERATOR MAY ENTER ANY OF SEVERAL COMMANDS AS DESCRIBED BELOW.

THE SUPERVISOR CODING FOLLOWS IMMEDIATELY THE DIAGNOSTIC TEST CODING, BUT THE SUPERVISOR LISTING HAS BEEN SUPPRESSED FOR GENERAL DISTRIBUTION. A LIMITED DISTRIBUTION HAS BEEN MADE TO FIELD SERVICE OF THE SUPERVISOR ASSEMBLY LISTING, AND IT MAY BE CONSULTED IN EVENT OF A SOFTWARE PROBLEM.

1.1.2 DIAGNOSTIC INFORMATION

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

1.2 SYSTEM REQUIREMENTS

1.2.1 HARDWARE REQUIREMENTS

PDP-11/LSI-11 PROCESSOR WITH 16K OR MORE OF MEMORY
CONSOLE DEVICE (LA30,LA36,VT50,ETC.)
RL11/RLV11 CONTROLLER(S)
1 - 8 RLO1 DRIVES
1 - 8 RLO1K CARTRIDGES WITH BAD SECTOR FILE
KW11P, KW11L (OPTIONAL)
LINEPRINTER(OPTIONAL)

1.2.2 SOFTWARE REQUIREMENTS

CZRLFBO RLO1 DRIVE COMPATABILITY
(FORMERLY MD-11-DZRLF-A)

**
AD
**
TH
DR
FC
**
PR
**
AL
IS
**
DI
**
TH
FC
OF
**
FL
**
TH
**
ZF
**
AL
2.
TH
FC
AS
SA
AR
BE
FC

1.3 RELATED DOCUMENTS AND STANDARDS

RL01 USERS MANUAL (EK-RL01-UG-PRE)
XXDP USERS MANUAL

1.4 DIAGNOSTIC HIERARCY PREREQUISITES

THE RL01 SUBSYSTEM SHOULD HAVE SUCCESSFULLY RUN THE FOLLOWING PROGRAMS:

CZRLABO	RL11/RLV11	RL01 CONTROLLER TEST (PART 1)
CZRLBBO	RL11/RLV11	RL01 CONTROLLER TEST (PART 2)
CVRLAAO	RLV11	RL01 DISKLESS TEST (RLV11 ONLY)
CZRLCBO		RL01 DRIVE TEST (PART 1)
CZRLDBO		RL01 DRIVE TEST (PART 2)
CZRLEBO	RL11/RLV11	RL01 PERFORMANCE EXERCISER

1.5 ASSUMPTIONS

THE HARDWARE OTHER THAN THE RL01 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 DIAGNOSTIC

2.1.1 THE SIX STEPS OF EXECUTION

THIS DIAGNOSTIC SHOULD BE LOADED AND STARTED USING NORMAL XXDP PROCEDURES. THE START COMMAND SHOULD NOT SPECIFY AN ADDRESS, BECAUSE THE DIAGNOSTIC HAS THE PROPER TRANSFER ADDRESS CODED INTO IT.

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

* STEP 1 *

A SHORT SERIES OF 'HARDCORE QUESTIONS' WILL BE ASKED:

QUESTION	MEANING	-----	-----
L-CLK (L) N ?	IS THERE AN L-CLOCK?		
P-CLK (L) N ?	IS THERE A P-CLOCK?		
50HZ (L) N ?	IS THE POWER 50 CYCLES (AS IN EUROPE)?		
LSI (L) N ?	IS MACHINE AN LSI?		
LPT (L) N ?	IS THERE A LINE PRINTER?		
MEM (K) (D) 16 ?	HOW MANY K OF MEMORY ARE THERE?		

THE DEFAULTS (SHOWN AFTER EACH QUESTION) CAN BE SELECTED BY HITTING CARRIAGE RETURN. IT IS POSSIBLE THAT NOT ALL OF THE QUESTIONS WILL BE ASKED: FOR EXAMPLE, IF YOU SAY 'YES' TO THE L-CLOCK QUESTION, THE

PP
VA
IN
BE
EX
ON
OL
VA
PL
TH
RE
IN
US
A
VA
TH
AC
IN
NC
OF
HA
HA
FI
VA
(1
LE
FI
TH

UN
CC
CC
CC
UN
CC
CC
CC

P-CLOCK QUESTION WILL NOT BE ASKED.

IF NEITHER P OR L CLOCK ARE ANSWERED YES THE OPERATOR WILL BE ASKED TO TYPE TWO CHARACTERS 4 SECONDS APART.

* STEP 2 *

WHEN YOU HAVE ANSWERED ALL THE HARDCORE QUESTIONS, THE DIAGNOSTIC WILL ISSUE THE PROMPT 'DS-B>'. 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 'DS-B>' 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:

LOE	LOOP ONE ERROR
HOE	HALT ON ERROR
IER	INHIBIT ERROR PRINTOUT

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

* STEP 3 *

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.

A
TH
(1
CC
TE

TH

3.

ER
AL
IN

3.

AL
AF

ER
WI

EX

RE

ER

ER
WA

* STEP 4 *

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 5 *

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 6 *

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 DS-B>).

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 ENLESSLY 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 OCCURED.

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 REISSUED.

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 2, 3, 4, 5, AND 6 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 OCCURED. 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:IFR=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:

BY
WHOM
ENTERED:
O
D
D.O
D
D
D
D
D.O
D.O
D
D.O
D.O
D.O
D.O
D.O
D
D.O
D.O
D.O
D.O
D.O
D
D
D.O

.R DZRKXX
DZRKXX
L-CLK (L) N ? Y
SOHZ (L) N ?
LSI (L) N ?
LPT (L) N ?
MEM (K) (D) 16 ?
DS-B>STA/PASS:1/FLAGS:HOE
UNITS (D) ? 2
UNIT 1
CSR (O) ?
VECTOR (O) ?
BR LEVEL (O) ?
DRIVE (O) ? 0
UNIT 2
CSR (O) ?
VECTOR (O) ?
BR LEVEL (O) ?
DRIVE (O) ? 1
CHANGE SW (L) ? N
DZRKXX HARD ERR 00004 TST 003 SUB 002 PC:004130
ERR HLT
DS-B>PRO/FLAGS:IER:LOE:HOE=0

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
DS-B>CON/FLAGS:HOE:IER:LOE=0
CHANGE SW (L) ? N
DZRKXX EOP 1
DS-B>RESTART/PASS:1
CHANGE SW (L) ? N

6.
TI
TI
SU
TI

TI
Q

W
SI
B
SI
TI
O
A

2.2 HOW TO CREATE A CHAINABLE FILE

THE DIAGNOSTIC AS RECEIVED FROM RELEASE ENGINEERING CANNOT BE RUN IN CHAIN MODE. THAT IS WHY IT BEARS THE EXTENSION 'BIN' INSTEAD OF 'BIC'. THERE IS A WAY, HOWEVER, TO CREATE A CHAINABLE PROGRAM FROM WHAT YOU'VE GOT.

IT CONSISTS OF RUNNING THE PROGRAM WITH THE SPECIAL COMMAND 'CCI' ISSUED WHERE YOU WOULD NORMALLY ISSUE A START COMMAND (TO THE PROMPT DS-B>). THIS COMMAND CAUSES THE DIAGNOSTIC TO GO THRU ALL THE QUESTIONS AND ANSWERS AND THEN TO HALT, JUST WHERE IT WOULD ORDINARILY BEGIN EXECUTION OF THE HARDWARE TEST CODE. AT THIS POINT YOU CAN DUMP THE PROGRAM AS IT SITS IN CORE TO THE LOAD MEDIUM, WITH THE NEW EXTENSION 'BIC'.

HERE IS A SAMPLE DIALOGUE TO ACCOMPLISH THIS:

```
.R UPD2
RESTART: XXXXXX
*CLR
*LOAD DIAG.BIN
XFER:200 CORE:0,60602
*START 200
L-CLK (L) N ?
-----
-----
```

```
DS-B>CCI
# UNITS (D) ? 4
-----
-----
```

```
CHANGE SW (L) ? N
PTAB END: 60632
```

```
*****
*AT THIS POINT THE MACHINE HALTS AND*
*YOU MUST RESTART AT ADDRESS XXXXXX*
*****
```

```
*HICORE 60632
CORE: 0,60632
*DUMP DK0: DIAG.BIC
```

THE RESULT OF DOING THIS IS THAT YOU CAN NOW BUILD AN XXDP CHAIN FILE CONTAINING THE XXDP COMMAND

```
.R DIAG.BIC
```

AND THE DIAGNOSTIC WILL EXECUTE WITHOUT MANUAL INTERVENTION, USING THE ANSWERS THAT YOU GAVE IT WHEN YOU DID THE CCI COMMAND.

2.3 DETAILS OF COMMANDS AND SYNTAX

2.3.1 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:

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

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 EXECUTION..B '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 TES 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

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 IF 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 REEXECUTED. HARDWARE PARAMETERS MAY NOT BE CHANGED.

THE SWITCH ARGUMENTS ARE AS IN THE START COMMAND EXCEPT:

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

PRO(CEED)/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

CCI/TEST:TEST-LIST/PASS:PASS-CNT/FLAGS:FLAG-LIST/EOP:EOP-INCR

THE DIAGNOSTIC EXECUTES THRU ALL OPERATOR DIALOGUE AND HALTS AT THE HARDWARE TEST CODE. NOW THE OPERATOR CAN DUMP THE CORE IMAGE TO THE MEDIUM WITH A BIC EXTENSION.

THE BIC FILE MUST BE HANDLED DIFFERENTLY DEPENDING ON WHETHER IT IS RUN MANUALLY OR IN CHAIN MODE. IF RUN MANUALLY IT CAN BE INVOKED EITHER WITH A 'START' (IN WHICH CASE IT WILL BEHAVE LIKE THE BIN FILE: THE PRE-GENERATED ANSWERS TO OPERATOR QUESTIONS WILL BE IGNORED) OR WITH A 'RESTART' (IN WHICH CASE THE PRE-GENERATED OPERATOR ANSWERS WILL BE USED).

IF RUN IN CHAIN MODE, AUTOMATIC EXECUTION WILL COMMENCE IMMEDIATELY FROM THE XXDP COMMAND '.R DIAG'. THE COMMAND PROMPT 'DS-B>' WILL NOT BE ISSUED.

ANY SWITCHES SPECIFIED ON THE CCI COMMAND WILL CARRY OVER WHEN THE BIC FILE IS RUN IN CHAIN MODE (EXCEPT THAT UAM IS ALWAYS SET THERE) BUT WILL NOT CARRY OVER WHEN IT IS RUN MANUALLY.

TO DO A CCI ON A FULL SIZED DIAGNOSTIC (14.5K WORDS), A MACHINE SIZE LARGER THAN 16K IS REQUIRED. THE EXACT SIZE NEEDED DEPENDS ON WHICH UTILITY IS USED TO EXECUTE THE DIAGNOSTIC AT CCI TIME.

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, SAY) 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.

ON THE FIRST TRIP THRU THE QUESTIONS, ALL OF THE SLOTS IN ALL OF THE P-TABLES ARE FILLED. IF THE OPERATOR TYPES IN LESS THAN N EXPLICIT VALUES IN RESPONSE TO A PARTICULAR QUESTION, THESE VALUES ARE PLACED IN THE P-TABLES (ONE VALUE GOING INTO THE PROPER SLOT OF EACH P-TABLE BEGINNING WITH THE FIRST P-TABLE) UNTIL THE STRING OF VALUES IS EXHAUSTED. THE LAST VALUE THAT SLOT IN THE REMAINING P-TABLES.

ON SUBSEQUENT TRIPS THRU THE QUESTIONS, THE SAME PROCESS IS CARRIED OUT, EXCEPT THAT THE EARLIEST P-TABLE NOT TO HAVE RECEIVED AN EXPLICIT VALUE IN ANY OF ITS SLOTS NOW ASSUMES THE ROLE THAT TABLE NUMBER ONE PLAYED IN THE FIRST TRIP.

THE SERIES OF QUESTIONS IS REISSUED UNTIL AT LEAST ONE QUESTION HAS RECEIVED N EXPLICIT VALUES FROM THE OPERATOR.

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 TRANLATES 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 64 UNITS, AND THAT THERE ARE THREE HARDWARE PARAMETERS FOR EACH (THREE SLOTS IN THE P-TABLE, THREE HARDWARE QUESTIONS IN THE DIALOGUE). LET THE DESIRED VALUE FOR THE FIRST PARAMETER BE THE NUMBER 75 FOR ALL 64 TABLES. LET THE DESIRED VALUE FOR THE SECOND PARAMETER BE EQUAL TO THE UNIT NUMBER (1,2,3,...,64) EXCEPT FOR UNIT 50, WHICH SOULD RECEIVE THE VALUE 49. LET THE DESIRED VALUE FOR THE THIRD PARAMETER BE THE NUMBER 76 FOR THE FIRST 20 UNITS AND THE NUMBER 77 FOR THE LAST 44 UNITS.

THE FOLLOWING DIALOGUE WOULD ACCOMPLISH THIS GOAL:

UNITS (D) ? 64

UNIT 1
<QUESTION 1> ? 75
<QUESTION 2> ? 1-20
<QUESTION 3> ? 76

UNIT 21
<QUESTION 1> ?
<QUESTION 2> ? 21-49,,51-64
<QUESTION 3> ? 77

THE FIRST TIME THE SERIES IS ASKED, SLOT ONE RECEIVES A 75 IN ALL 64 TABLES. SLOT TWO RECEIVES THE VALUES 1,2,3,...,20 IN TABLES 1 THRU 20 AND A CONSTANT 20 IN TABLES 21 THRU 64. SLOT THREE RECEIVES A CONSTANT 76 IN ALL 64 TABLES.

THE SECOND TIME THRU THE SERIES, TABLES 21 THRU THE END ARE GOING TO BE AFFECTED (NOTE THAT THIS PIECE OF INFORMATION IS PRINTED OUT FOR THE OPERATOR IN THE FORM 'UNIT XX' AT THE BEGINNING OF EACH SERIES). QUESTION 1 IS RESPONDED TO BY A <CR>, SO SLOT ONE STAYS A CONSTANT 75 IN TABLES 21 THRU 64, SINCE NO NEW EXPLICIT VALUES ARE TYPED IN. SLOT TWO GETS THE VALUES 21,22,23,...,49 IN TABLES 21 THRU 49, AND GETS A 49 IN SLOT 50, AND GETS THE VALUES 51,52,53,...,64 IN TABLES 51 THRU 64. SLOT THREE GETS THE VALUE 77 IN TABLES 21 THRU 64.

THE DIALOGUE IS TERMINATED WHEN THE SOFTWARE RECOGNIZES THAT 64 EXPLICIT VALUES HAVE BEEN GIVEN FOR AT LEAST ON QUESTION (NAMELY QUESTION 2).

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.

RL11 (L) Y?

ANSWER YES(Y) IF YOU HAVE AN RL11 CONTROLLER, NO(N) IF YOU HAVE AN RLV11 CONTROLLER.

BUS ADDRESS (O) 174400?

ANSWER WITH THE BUS ADDRESS OF THE CONTROLLER.

VECTOR (O) 330?

ANSWER WITH THE INTERRUPT VECTOR OF THE CONTROLLER.

BR LEVEL (O) 5?

ANSWER WITH THE INTERRUPT PRIORITY OF THE CONTROLLER.

DRIVE (O) 0?

ANSWER WITH THE DRIVE(S) CONNECTED TO THE CONTROLLER.

2.6 SOFTWARE PARAMETERS

THE FOLLOWING QUESTIONS ARE ASKED IF REQUESTED ON A START, RESTART, OR CONTINUE. THEY ALLOW FLEXIBILITY 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 AS 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:

DZRL? 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 - MUST BE ZERO(0)
- BIT 14-7 - CYLINDER ADDRESS FOR TRANSFER
- BIT 6 - SURFACE FOR TRANSFER
- BIT 5-0 - SECTOR FOR TRANSFER (0-47)

FOR SEEK FUNCTION

BIT 15 - MUST BE ZERO(0)

BIT 14-7 - DIFFERENCE TO NEW CYLINDER
BIT 6-5 - MUST BE ZERO(0)
BIT 4 - SURFACE
BIT 3 - MUST BE ZERO
BIT 2 - SEEK DIRECTION(1 - IN / 0 - OUT)
BIT 1 - MUST BE ZERO
BIT 0 - MUST BE ONE(1)

FOR GET STATUS FUNCTION

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

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 - RESERVED(0)
BIT 6 - SURFACE
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 RLO1K 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.

.MAIN. MACY11 30A(1052) 22-NOV-78 15:54
CZRLFB.P11 22-NOV-78 15:47 TABLE OF CONTENTS

M 2

SEQ 0025

26	GLOBAL EQUATES SECTION
84	GLOBAL DATA SECTION
253	GLOBAL TEXT SECTION
293	GLOBAL ERROR REPORT SECTION
400	INITIALIZATION SECTION
699	GLOBAL SUBROUTINES SECTION
1902	DIAGNOSTIC SUPERVISOR -- LOW CORE SET UP

1			.ENABLE AMA
2			.ENABLE ABS
3			.NLIST ME,CND,MD
4			
5			
6	002000		.=2000
7			
8	002000		SVC
9		000000	SVCINS=0
10		000000	SVCTAG=0
11			
12			
13			
14	002000		POINTER NONE
15			
16			
17	002000		BGNMOD MDHEDR
18	002000		HEADER CZRLF,B,0,0,0,0,RL01,1
(4)	002000	103	.ASCII /C/
(4)	002001	132	.ASCII /Z/
(4)	002002	122	.ASCII /R/
(4)	002003	114	.ASCII /L/
(4)	002004	106	.ASCII /F/
(6)	002005	000	.BYTE 0
(6)	002006	000	.BYTE 0
(5)	002007	000	.BYTE 0
(4)	002010	102	.ASCII /B/
(4)	002011	060	.ASCII /O/
(4)	002012	000000	.WORD 0
(4)	002014	000000	.WORD 0
(4)	002016	032022	.WORD L\$HARD
(4)	002020	000000	.WORD 0
(4)	002022	022034	.WORD L\$HW
(4)	002024	000000	.WORD 0
(4)	002026	032244	.WORD L\$LAST
(4)	002030	000000	.WORD 0
(4)	002032	000000	.WORD 0
(4)	002034	000001	.WORD 1
(4)	002036	000000	.WORD 0
(4)	002040	022050	.WORD L\$DISPATCH
(4)	002042	000000	.WORD 0
(4)	002044	000000	.WORD 0
(4)	002046	000000	.WORD 0
(4)	002050	002	.BYTE C\$REVISION
(3)	002051	002	.BYTE C\$EDIT
(4)	002052	000000	.WORD 0
(4)	002054	000000	.WORD 0
(4)	002056	000000	.WORD 0
(5)	002060	000000	.WORD 0
(4)	002062	000000	.WORD 0
(4)	002064	002114	.WORD L\$DVTYP
(4)	002066	000000	.WORD 0
(4)	002070	002112	.WORD L\$DR
(4)	002072	002112	.WORD L\$DRST
(4)	002074	000000	.WORD 0
(4)	002076	000000	.WORD 0
(5)	002100	000014	.WORD 14

```

(4) 002102 000000      .WORD 0
(4) 002104 022052      .WORD L$INIT
(4) 002106 023324      .WORD L$CLEAN
19
20 002110
21 002110      ENDMOD
                DEVREG
(5) 002110 000000      .WORD 0
(2) 002112 000001      .BLKW
22
23 002114
(3) 002114 046122 030460 000      DEVTYP <RL01>
                .ASCIZ /RL01/
(2) 002114 002122      .EVEN
24
25
26      .SBTTL GLOBAL EQUATES SECTION
27
28      ;DEFINITIONS
29
30
31 002122      BGNMOD GLBEQAT
32
33 002122      EQUALS
34
35      000000      CS=0      ;CONTROL AND STATUS OFFSET
36      000002      BA=2      ;BUSADDRESS OFFSET
37      000004      DA=4      ;DISK ADDRESS OFFSET
38      000006      MP=6      ;MULTI PURPOSE OFFSET
39      ;CONSTANT OFFSETS FOR INDIVIDUAL DRIVE BUFFERS
40
41      000000      CSR=0      ;CONTROLLER ADDRESS
42      000002      VEC=2      ;VECTOR OF CONTROLLER
43      000004      DSB=4      ;DRIVE SELECT
44      000006      PAT=6      ;PATTERN UNIQUE TO DRIVE
45
46
47      000001      DRDY=BIT0      ;DRIVE READY
48      000100      INTEN=BIT6      ;INTERRUPT ENABLE
49      100000      ERR=BIT15      ;COMPOSITE ERROR
50      040000      DERR=BIT14      ;DRIVE ERROR
51      020000      NXM=BIT13      ;NON-EXISTANT MEMORY ERROR
52      010000      DLT=BIT12      ;DATA LATE
53      004000      DCRC=BIT11      ;DATA CRC ERROR
54      004000      HCRC=BIT11      ;HEADER CRC ERROR
55      010000      HNF=BIT12      ;HEADER NOT FOUND ERROR
56      002000      OPI=BIT10      ;OPERATION INCOMPLETE ERROR
57      000200      CRDY=BIT7      ;CONTROLLER READY
58      000040      BA17=BIT5      ;EXTENDED BUS ADDRESS BIT 17
59      000020      BA16=BIT4      ;EXTENDED BUS ADDRESS BIT 16
60      000002      CRSET=BIT1      ;CONTROLLER RESET FUNCTION CODE
61      000004      GSTAT=BIT2      ;GET DRIVE STATUS FUNCTION CODE
62      000006      SEEK=BIT1!BIT2      ;SEEK FUNCTION CODE
63      000010      RDHDR=BIT3      ;READ HEADER FUNCTION CODE
64      000012      WRITE=BIT3!BIT1      ;WRITE FUNCTION CODE
65      000014      READ=BIT3!BIT2      ;READ FUNCTION CODE
66      000013      DRST=BIT3!BIT1!BIT0      ;DRIVE RESET COMMAND CODE FOR DRIVE COMMAD WORD
67      000003      GSBIT=BIT1!BIT0      ;GET STATUS COMMAND CODE FOR DRIVE COMMAND WORD
    
```

```

68      000001      MK=BIT0      ;MARKER BIT FOR DRIVE COMMAND WORD(SEEK,GET STATUS)
69      000004      SIGN=BIT2     ;DIRECTION FOR SEEK(0=AWAY FROM SPINDLE)
70      000020      SKHS=BIT4     ;HEAD SELECT FOR SEEK
71      000100      HEAD=BIT6     ;HEAD SELECT FOR READ,WRITE,GET STATUS
72
73      ;OFFSET FOR HARDWARE P-TABLE
74
75      000000      CSR=0
76      000002      VECT=2
77      000004      PRIOR=4
78      000006      DRBT=6
79      000010      RLCNT=10
    
```

```

83 002122      .SBTTL GLOBAL DATA SECTION
84
85
86
87
88
    
```

```

89 002122      BGNMOD  GLBDAT
90
91 002122 000000      HDRFND: .WORD 0      ;1=HEADER IN BAD SECTOR LIST
92 002124      000      OUT10: .BYTE 0
93 002125      000      OUT20: .BYTE 0
94 002126      000      OUT30: .BYTE 0
95 002127      000      OUT40: .BYTE 0
96 002130      000      OUT50: .BYTE 0
97 002131      000      OUT11: .BYTE 0
98 002132      000      OUT21: .BYTE 0
99 002133      000      OUT31: .BYTE 0
100 002134      000      OUT41: .BYTE 0
101 002135      000      OUT51: .BYTE 0
102 002136      000      OQU10: .BYTE 0
103 002137      000      OQU20: .BYTE 0
104 002140      000      OQU30: .BYTE 0
105 002141      000      OQU40: .BYTE 0
106 002142      000      OQU50: .BYTE 0
107 002143      000      OQU11: .BYTE 0
108 002144      000      OQU21: .BYTE 0
109 002145      000      OQU31: .BYTE 0
110 002146      000      OQU41: .BYTE 0
111 002147      000      OQU51: .BYTE 0
112 002150      000      MID10: .BYTE 0
113 002151      000      MID20: .BYTE 0
114 002152      000      MID30: .BYTE 0
115 002153      000      MID40: .BYTE 0
116 002154      000      MID50: .BYTE 0
117 002155      000      MID11: .BYTE 0
118 002156      000      MID21: .BYTE 0
119 002157      000      MID31: .BYTE 0
120 002160      000      MID41: .BYTE 0
121 002161      000      MID51: .BYTE 0
122 002162      000      TQU10: .BYTE 0
123 002163      000      TQU20: .BYTE 0
    
```

124 002164 000
125 002165 000
126 002166 000
127 002167 000
128 002170 000
129 002171 000
130 002172 000
131 002173 000
132 002174 000
133 002175 000
134 002176 000
135 002177 000
136 002200 000
137 002201 000
138 002202 000
139 002203 000
140 002204 000
141 002205 000
142
143
144
145
146
147 002206 000020
148
149
150
151 002246 000170
152
153
154
155
156
157
158 002626 002126
159 002630 002140
160 002632 002152
161 002634 002164
162 002636 002176
163 002640 002133
164 002642 002145
165 002644 002157
166 002646 002171
167 002650 002203
168
169 002652 152525
170 002654 133333
171 002656 066666
172 002660 155555
173
174 002662 000000
175 002664 000000
176 002666 000000
177 002670 000000
178 002672 000000
179 002674 000000

TQU30: .BYTE 0
TQU40: .BYTE 0
TQU50: .BYTE 0
TQU11: .BYTE 0
TQU21: .BYTE 0
TQU31: .BYTE 0
TQU41: .BYTE 0
TQU51: .BYTE 0
INN10: .BYTE 0
INN20: .BYTE 0
INN30: .BYTE 0
INN40: .BYTE 0
INN50: .BYTE 0
INN11: .BYTE 0
INN21: .BYTE 0
INN31: .BYTE 0
INN41: .BYTE 0
INN51: .BYTE 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 BYTE ADDRESSES OF TOP SURFACE.
;LAST FIVE ARE BYTE ADDRESSES OF BOTTOM SURFACE.

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

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

FOWR: .WORD 0
FADJ: .WORD 0
TEMP: .WORD 0
LSTCLR: .WORD 0
REASON: .WORD 0
ERFLG: .WORD 0

;LAST CONTROLLER
;DRIVE ERROR REASON
;ERROR FLAG

180	002676	000000	STFLG: .WORD	0	:PROGRAM START UP FLAG
181	002700	000000	ADJLOC: .WORD	0	:TRACK INDEX FOR ADJ. CYL TEST
182	002702	000000	ADJFLG: .WORD	0	:FLAG FOR ADJ. STORE OR RETRIEVE
183	002704	000000	ADJDIR: .WORD	0	:ADJACENT SEEK DIRECTION
184	002706	000000	DRSTAT: .WORD	0	
185	002710	000000	HSFLG: .WORD	0	
186	002712	000000	OSECT: .WORD	0	
187	002714	000000	HEAD01: .WORD	0	:SURFACE FLAG
188	002716	000000	DIRC: .WORD	0	:DIRECTION OF SEEK
189	002720	000000	DESCYL: .WORD	0	:PACK SURFACE
190	002722	000000	REVSK: .WORD	0	:REVERSE SEEK
191	002724	000000	FORSK: .WORD	0	:FORWARD SEEK
192	002726	000000	UUT: .WORD	0	:UNIT UNDER TEST
193	002730	000000	SECT: .WORD	0	:SECTOR
194	002732	000000	LSTDRV: .WORD	0	:LAST DRIVE
195	002734	000000	GDATA: .WORD	0	:GOOD DATA
196	002736	000000	BDATA: .WORD	0	:BAD DATA
197	002740	000000	WCOUNT: .WORD	0	:WORD COUNT
198	002742	000000	SECWRD: .WORD	0	:SECTOR WORD
199	002744	000000	OFFSET: .WORD	0	:INCREMENT
200	002746	000000	LSTTRK: .WORD	0	:LAST TRACK OF SEARCH
201	002750	000000	FRTTRK: .WORD	0	:FIRST TRACK OF SEARCH
202	002752	000000	PRSTRK: .WORD	0	:PRESENT TRACK
203	002754	000000	SURFACE: .WORD	0	:SURFACE
204	002756	000000	TRKFND: .WORD	0	:TRACK FOUND
205	002760	000000	TRKCNT: .WORD	0	:TRACK COUNT
206	002762	000000	E.CS: .WORD	0	:IMAGE OF CSR
207	002764	000000	E.BA: .WORD	0	:IMAGE OF BUS ADDRESS
208	002766	000000	E.DA: .WORD	0	:IMAGE OF DISK ADDRESS
209	002770	000000	E.MP: .WORD	0	:IMAGE OF MULTI-PURPOSE WORD 1
210	002772	000000	E.MP1: .WORD	0 2
211	002774	000000	E.MP2: .WORD	0 3
212	002776	000000	BCS: .WORD	0	:COMMAND LOADED
213	003000	000000	BBA: .WORD	0	:BUS ADDRESS LOADED
214	003002	000000	BDA: .WORD	0	:DISK ADDRESS LOADED
215	003004	000000	BMP: .WORD	0	:WORD COUNT LOADED
216	003006	000000	SERNM1: .WORD	0	:SERIAL NUMBER OF CATRIDGE
217	003010	000000	SERNM2: .WORD	0	
218	003012	000000	ADJTRK: .WORD	0	:INSIDE/OUTSIDE FLAG
219	003014	000000	ADJUUT: .WORD	0	:UUT FOR 'ADJCYL'
220	003016	000000	ADJLC2: .WORD	0	:TEMP LOC FOR 'ADJCYL' ..
221	003020	000000	ADJLC3: .WORD	0
222	003022	000000	ADJLC4: .WORD	0
223	003024	000000	STSEC1: .WORD	0	:SECTORS TO WRITE 'ADJCYL'
224	003026	000000	STSEC: .WORD	0	
225	003030	006000	BUF: .BLKW	3072.	:BUFFER FOR 24 SECTOR READS
226					
227	017030		DRBUF:		:DRIVE INFORMATION BUFFERS
228					
232					
240					
(1)	017030	000000	CSR		:CONTROLLER ADDRESS
(1)	017032	000002	VFC		:VECTOR
(1)	017034	000004	DSB		:DRIVE SELECT BITS
(1)	017036	000006	PAT		:PATTERN UNIQUE TO DRIVE
(1)					

```
(1)
(1) 017040 000000 CSR ;CONTROLLER ADDRESS
(1) 017042 000002 VEC ;VECTOR
(1) 017044 000004 DSB ;DRIVE SELECT BITS
(1) 017046 000006 PAT ;PATTERN UNIQUE TO DRIVE
(1)
(1)
(1) 017050 000000 CSR ;CONTROLLER ADDRESS
(1) 017052 000002 VEC ;VECTOR
(1) 017054 000004 DSB ;DRIVE SELECT BITS
(1) 017056 000006 PAT ;PATTERN UNIQUE TO DRIVE
(1)
(1)
(1) 017060 000000 CSR ;CONTROLLER ADDRESS
(1) 017062 000002 VEC ;VECTOR
(1) 017064 000004 DSB ;DRIVE SELECT BITS
(1) 017066 000006 PAT ;PATTERN UNIQUE TO DRIVE
```

```
241
245 017070 000000 ENDBUF: .WORD 0 ;END OF DRIVE BUFFERS
246
247
```

```
248 017072 ENDMOD
249
```

```
250
251
252
253 .SBTTL GLOBAL TEXT SECTION
254 017072 BGNMOD GLBTXT
255
256 ;GLOBAL TEXT
257
```

```
262
263 017072 047103 046124 020122 CNTTOT: .ASCIZ /CNTLR TIMED OUT/
264 017112 051105 047522 020122 INITWR: .ASCIZ /ERROR ON RECOVERING INITIAL WRITE BY FIRST DRIVE /
265 017174 051105 047522 020122 DCKER: .ASCIZ /ERROR ON READ/
266 017212 044515 044516 052515 FEW: .ASCIZ /MINIMUM OF TWO DRIVES REQUIRED/
267 017251 115 054101 046511 MANY: .ASCIZ /MAXIMUM OF FOUR DRIVES ALLOWED/
268 017310 042524 052123 040440 NONE: .ASCIZ /TEST ABORTED - CAN'T FIND ANY GOOD SPOTS/
269 017361 124 054522 047111 OVMES: .ASCIZ /TRYING TO OVERWRITE DRIVE /
270 017414 051124 044531 043516 RECMS: .ASCIZ /TRYING TO READ DATA WRITTEN BY DRIVE /
271 017462 040503 023516 020124 ERRFND: .ASCIZ /CAN'T FIND FIVE ADJACENT TRACKS/
272 017522 053117 051105 051127 OVWER: .ASCIZ /OVERWRITE ERROR/
273 017542 042522 042101 051040 RECER: .ASCIZ /READ RECOVERY ERROR/
274 017566 051105 047522 020122 FUNERR: .ASCIZ /ERROR IN SEEK OPERATION/
275 017616 044515 020123 042523 SKER: .ASCIZ /MIS SEEK ERROR/
276 017635 106 051117 040527 FWD: .ASCIZ /FORWARD/
277 017645 122 053105 051105 REV: .ASCIZ /REVERSE/
278 017655 105 051122 051117 WRIT1: .ASCIZ /ERROR WRITING SECTOR/
279 017702 051105 047522 020122 READ1: .ASCIZ /ERROR READING SECTOR/
280 017727 101 045104 041501 ADJTXT: .ASCIZ /ADJACENT CYLINDER TEST/
```

```
281
282
283
284
```



```

288
289
290
291 017756
292
293
294 017756
295
296 017756
297
298 017756
(10) 017756 013746 002754
(9) 017762 013746 002746
(8) 017766 013746 002750
(7) 017772 012746 021270
(6) 017776 012746 000004
(3) 020002 010600
(4) 020004 104014
(4) 020006 062706 000012
299
300 020012
(3) 020012
(3) 020012 104023
301
302 020014
303 020014
(9) 020014 005046
(9) 020016 156416 000005
(8) 020022 016446 000000
(7) 020026 012746 020771
(6) 020032 012746 000003
(3) 020036 010600
(4) 020040 104014
(4) 020042 062706 000010
304 020046 004737 025144
305 020052
(3) 020052
(3) 020052 104023
306
307 020054
308 020054
(9) 020054 005046
(9) 020056 156416 000005
(8) 020062 016446 000000
(7) 020066 012746 020771
(6) 020072 012746 000003
(3) 020076 010600
(4) 020100 104014
(4) 020102 062706 000010
309 020106 004737 025144
310 020112
(10) 020112 013746 002730
(9) 020116 005046
(9) 020120 153716 002720
(8) 020124 005046
(8) 020126 153716 002721

.EVEN
ENDMOD

.SBTTL GLOBAL ERROR REPORT SECTION
BGNMOD GLBERR
BGNMSG ERR1
PRINTB #FRM10,FRTTRK,LSTTRK,SURFACE
MOV SURFACE,-(SP)
MOV LSTTRK,-(SP)
MOV FRTTRK,-(SP)
MOV #FRM10,-(SP)
MOV #4,-(SP)
MOV SP,R0
EMT C$PNTB
ADD #12,SP
ENDMSG
L10000:
EMT C$MSG
BGNMSG ERR2
PRINTB #FRM4,CSR(R4),<B,DSB+1(R4)>
CLR -(SP)
BISB DSB+1(R4),(SP)
MOV CSR(R4),-(SP)
MOV #FRM4,-(SP)
MOV #3,-(SP)
MOV SP,R0
EMT C$PNTB
ADD #10,SP
JSR PC,REGDMP ;REGISTER DUMP ROUTINE
ENDMSG
L10001:
EMT C$MSG
BGNMSG ERR3
PRINTB #FRM4,CSR(R4),<B,DSB+1(R4)>
CLR -(SP)
BISB DSB+1(R4),(SP)
MOV CSR(R4),-(SP)
MOV #FRM4,-(SP)
MOV #3,-(SP)
MOV SP,R0
EMT C$PNTB
ADD #10,SP
JSR PC,REGDMP ;REGISTER DUMP ROUTINE
PRINTB #FRM5,<B,DESCYL+1>,<B,DESCYL>,SECT
MOV SECT,-(SP)
CLR -(SP)
BISB DESCYL,(SP)
CLR -(SP)
BISB DESCYL+1,(SP)
    
```

(7)	020132	012746	021032	MOV	#FRM5,-(SP)	
(6)	020136	012746	000004	MOV	#4,-(SP)	
(3)	020142	010600		MOV	SP,R0	
(4)	020144	104014		EMT	C\$PNTB	
(4)	020146	062706	000012	ADD	#12,SP	
311	020152			PRINTB	#FRM16,CSR(R3),<B,DSB+1(R3)>	
(9)	020152	005046		CLR	-(SP)	
(9)	020154	156316	000005	BISB	DSB+1(R3),(SP)	
(8)	020160	016346	000000	MOV	CSR(R3),-(SP)	
(7)	020164	012746	021621	MOV	#FRM16,-(SP)	
(6)	020170	012746	000003	MOV	#3,-(SP)	
(3)	020174	010600		MOV	SP,R0	
(4)	020176	104014		EMT	C\$PNTB	
(4)	020200	062706	000010	ADD	#10,SP	
312						
313	020204			ENDMSG		
(3)	020204			L10002:		
(3)	020204	104023		EMT	C\$MSG	
314						
315	020206			BGNMSG	ERR4	
316						
317	020206			PRINTB	#FRM4,CSR(R4),<B,DSB+1(R4)>	
(9)	020206	005046		CLR	-(SP)	
(9)	020210	156416	000005	BISB	DSB+1(R4),(SP)	
(8)	020214	016446	000000	MOV	CSR(R4),-(SP)	
(7)	020220	012746	020771	MOV	#FRM4,-(SP)	
(6)	020224	012746	000003	MOV	#3,-(SP)	
(3)	020230	010600		MOV	SP,R0	
(4)	020232	104014		EMT	C\$PNTB	
(4)	020234	062706	000010	ADD	#10,SP	
318	020240	004737	025144	JSR	PC,REGDMP ;REGISTER DUMP ROUTINE	
319	020244			PRINTB	#FRM5,<B,DESCYL+1>,<B,DESCYL>,SECT	
(10)	020244	013746	002730	MOV	SECT,-(SP)	
(9)	020250	005046		CLR	-(SP)	
(9)	020252	153716	002720	BISB	DESCYL,(SP)	
(8)	020256	005046		CLR	-(SP)	
(8)	020260	153716	002721	BISB	DESCYL+1,(SP)	
(7)	020264	012746	021032	MOV	#FRM5,-(SP)	
(6)	020270	012746	000004	MOV	#4,-(SP)	
(3)	020274	010600		MOV	SP,R0	
(4)	020276	104014		EMT	C\$PNTB	
(4)	020300	062706	000012	ADD	#12,SP	
320	020304			PRINTB	#FRM6,REASON,LSTDRV,LSTCLR,LSTDRV	
(11)	020304	013746	002732	MOV	LSTDRV,-(SP)	
(10)	020310	013746	002670	MOV	LSTCLR,-(SP)	
(9)	020314	013746	002732	MOV	LSTDRV,-(SP)	
(8)	020320	013746	002672	MOV	REASON,-(SP)	
(7)	020324	012746	021101	MOV	#FRM6,-(SP)	
(6)	020330	012746	000005	MOV	#5,-(SP)	
(3)	020334	010600		MOV	SP,R0	
(4)	020336	104014		EMT	C\$PNTB	
(4)	020340	062706	000014	ADD	#14,SP	
321	020344			PRINTB	#FRM7,DIRC	
(8)	020344	013746	002716	MOV	DIRC,-(SP)	
(7)	020350	012746	021122	MOV	#FRM7,-(SP)	
(6)	020354	012746	000002	MOV	#2,-(SP)	

```

(3) 020360 010600      MOV      SP,R0
(4) 020362 104014      EMT      C$PNTB
(4) 020364 062706 000006  ADD      #6,SP
322
323 020370      ENDMSG
(3) 020370      L10003:
(3) 020370 104023      EMT      C$MSG
324
325 020372      BGNMSG
326 020372      ERR5
PRINTB #FRM4,CSR(R4),<B,DSB+1(R4)>
(9) 020372 005046      CLR      -(SP)
(9) 020374 156416 000005  BISB    DSB+1(R4),(SP)
(8) 020400 016446 000000  MOV     CSR(R4),-(SP)
(7) 020404 012746 020771  MOV     #FRM4, -(SP)
(6) 020410 012746 000003  MOV     #3, -(SP)
(3) 020414 010600      MOV     SP,R0
(4) 020416 104014      EMT     C$PNTB
(4) 020420 062706 000010  ADD     #10,SP
327 020424 004737 025144  JSR    PC,REGDMP
328 020430      ENDMSG
(3) 020430      L10004:
(3) 020430 104023      EMT     C$MSG
329
330 020432      BGNMSG
331 020432      ERR6
PRINTB #FRM4,CSR(R4),<B,DSB+1(R4)>
(9) 020432 005046      CLR      -(SP)
(9) 020434 156416 000005  BISB    DSB+1(R4),(SP)
(8) 020440 016446 000000  MOV     CSR(R4),-(SP)
(7) 020444 012746 020771  MOV     #FRM4, -(SP)
(6) 020450 012746 000003  MOV     #3, -(SP)
(3) 020454 010600      MOV     SP,R0
(4) 020456 104014      EMT     C$PNTB
(4) 020460 062706 000010  ADD     #10,SP
332 020464 004737 025144  JSR    PC,REGDMP
333 020470      PRINTB #FRM17,R1,E.MP
(9) 020470 013746 002770  MOV     E.MP, -(SP)
(8) 020474 010146      MOV     R1, -(SP)
(7) 020476 012746 021706  MOV     #FRM17, -(SP)
(6) 020502 012746 000003  MOV     #3, -(SP)
(3) 020506 010600      MOV     SP,R0
(4) 020510 104014      EMT     C$PNTB
(4) 020512 062706 000010  ADD     #10,SP
334 020516      ENDMSG
(3) 020516      L10005:
(3) 020516 104023      EMT     C$MSG
335
336
337
338
339

```

;FORMAT STATEMENTS

```

343
344 020520 047045 040445 047125 FRM1: .ASCIZ /%N%UNLOAD DRIVE %01%A ON CONTROLLER %06%A AND REMOVE PACK%N/
345 020615 045 022516 050101 FRM2: .ASCIZ /%N%APLACE PACK IN DRIVE %01%A ON CONTROLLER %06%A AND LOAD%N/
346 020712 047045 040445 051127 FRM3: .ASCIZ !%N%AWRONG PACK # IS %05%05%A # S/B %05%05%N%N!
347 020771 045 041501 047117 FRM4: .ASCIZ /%A^CONTROLLER: %06%A DRIVE: %01%N/
348 021032 040445 042510 042101 FRM5: .ASCIZ /%AHEAD: %01%A CYL: %23%A SECTOR: %22%N/

```

```

349 021101      045 022524 030517 FRM6:  .ASCIZ  /%T%01% ON %06%/
350 021122 040445 042523 045505 FRM7:  .ASCIZ  /%A%SEEK DIRECTION: %T%N%ADATA:%N/
351 021162 040445 047527 042122 FRM8:  .ASCIZ  !%A%WORD: %Z3%A S/B: %06%A WAS: %06%N!
352 021226 042045 022463 020101 FRM9:  .ASCIZ  /%D3%A WORDS BAD OUT OF 128 READ%N/
353 021270 040445 042502 053524 FRM10: .ASCIZ  /%A%BETWEEN %Z3%A - %Z3%A HEAD: %01%N/
354 021334 047045 040445 053520 FRM11: .ASCIZ  /%N%APWR FAIL NOT SUPPORTED%N/
355 021371      045 041101 043105 FRM12: .ASCIZ  /%A%BEFORE CS: %06%A BA: %06%A DA: %06%A MP: %06%/
356 021450 047045 040445 043101 FRM13: .ASCIZ  /%N%AAFTER CS: %06%A BA: %06%A DA: %06%A MP: %06%N/
357 021533      045 022516 020101 FRM14: .ASCIZ  /%N%A DRIVE STATUS: %06/
358 021562 047045 040445 040503 FRM15: .ASCIZ  /%N%ACAN'T FIND BAD SECTOR FILE/
359 021621      045 040501 045104 FRM16: .ASCIZ  /%A%ADJACENT WRITTEN BY CONTROLLER: %06%A DRIVE: %01%N/
360 021706 040445 054105 023520 FRM17: .ASCIZ  /%A%EXP'D: %06%A REC'D: %06%N/
361 021742 047045 040445 047125 FRM18: .ASCIZ  /%N%AUNLOAD ALL DRIVES TO BE USED%N/
362 022005      045 022516 020101 ENDPAS: .ASCIZ  /%N%A END OF PASS%N%N/
363
364
368
369
370
371
372          .EVEN
373
374 022032          ENDMOD
375
376 022032          BGNMOD  HPTCODE
377
378 022032          BGNHW
(3) 022032 000005          .WORD  L10006-L$HW/2
379
380 022034 174400          .WORD  174400
381 022036 000160          .WORD  160
382 022040 000240          .WORD  240
383 022042 000000          .WORD  0
384 022044 000001          .WORD  1
385
386 022046          ENDPHW
(3) 022046          L10006:
387
388 022046          ENDMOD
389
390
391 022046          BGNMOD  DSPCODE
392
393 022046          DISPATCH  1
(4) 022046 000001          .WORD  1
(6) 022050 031056          .WORD  T1
394
395 022052          ENDMOD
396
397
398
399
400          .SBTTL  INITIALIZATION SECTION
401 022052          BGNMOD  INITCODE
402
403 022052          BGNINIT

```

```

404
405 022052          SETPRI #340
(3) 022052 012700 000340  MOV    #340,R0
(3) 022056 104041      EMT    C$SPRI
406
407 022060 023727 002012 000002  CMP    L$UNIT,#2          ;MORE THAN TWO
408 022066 002005      BGE    90$                ;YES, OKAY
409
410 022070          ERRSF  19.,FEW
(3) 022070 104421      TRAP  T$ERCODE
(5) 022072 000023      .WORD 19
(5) 022074 017212      .WORD FEW
411 022076 000137 023300      JMP    CMPENA              ;CLEAN CODE WHEN < 2 DRIVES
412
413 022102 023727 002012 000004 90$:  CMP    L$UNIT,#4          ;MORE THAN FOUR
414 022110 003405      BLE    91$                ;NO, OKAY
415
416 022112          ERRSF  20.,MANY
(3) 022112 104421      TRAP  T$ERCODE
(5) 022114 000024      .WORD 20
(5) 022116 017251      .WORD MANY
417 022120 000137 023300      JMP    CMPENA              ;CLEAN CODE WHEN > 4 DRIVES
418
419 022124 013737 002012 002726 91$:  MOV    L$UNIT,UUT        ;GET NUMBER OF UNITS
420 022132 005001      CLR    R1                 ;INIT P-TABLE
421 022134 012704 017030      MOV    #DRBUF,R4         ;SET UP DRIVE BUFFER
422 022140 012702 002652      MOV    #PATLST,R2        ;GET LIST OF PATTERNS
423 022144 005737 002726      TST    UUT                ;ANY P-TABLES LEFT?
424 022150 001422      BEQ    END                ;NO,GO TO END
425 022152          GPHARD R1,R0             ;GET A P-TABLE
(3) 022152 010100      MOV    R1,R0
(3) 022154 104042      EMT    C$GPHRD
426 022156 012064 000000      MOV    (R0)+,CSR(R4)     ;GET CSR
427 022162 012064 000002      MOV    (R0)+,VEC(R4)     ;GET VECTOR
428 022166 005720      TST    (R0)+             ;SKIP PAST BR
429 022170 011064 000004      MOV    (R0),DSB(R4)      ;GET DRIVE
430 022174 011264 000006      MOV    (R2),PAT(R4)
431 022200 005722      TST    (R2)+
432 022202 005201      INC    R1                 ;NEXT P TABLE
433 022204 005337 002726      DEC    UUT                ;NEXT DRIVE
434 022210 062704 000010      ADD    #PAT+2,R4
435 022214 000753      BR     1$
436 022216 013737 002012 002726  END:  MOV    L$UNIT,UUT        ;GET BEGINNING OF BUFFER
437 022224 012704 017030      MOV    #DRBUF,R4         ;CLEAR ADJ. TEST FLAG
438 022230 005037 002664      CLR    FADJ              ;CLEAR OVERWRITE FLAG
439 022234 005037 002662      CLR    FOWR
440 022240          READEF #EF.PWR
(3) 022240 012700 000034      MOV    #EF.PWR,R0
(3) 022244 104050      EMT    C$REFG
441 022246          BNCOMPLETE  SETUP
(2) 022246 103010      BCC   SETUP
442 022250          PRINTF #FRM11
(7) 022250 012746 021334      MOV    #FRM11,-(SP)
(6) 022254 012746 000001      MOV    #1,-(SP)
(3) 022260 010600      MOV    SP,R0
(4) 022262 104017      EMT    C$PNTF

```

```

(4) 022264 062706 000004          ADD    #4,SP
443
444          :INITIALIZE ROUTINE
445          :WE ATTEMPT TO LOCATE 5 PERFECT ADJACENT TRACKS AT 5 SPOTS
446          :ACROSS THE PACK.
447          :THE 5 SPOTS ARE: (EACH SURFACE)
448          :
449          :OUTER - TRACK 0 - 16
450          :INNER - TRACK 238 - 254
451          :MIDDLE - TRACK 120 - 136
452          :ONE QUARTER - TRACK 56 - 72
453          :THREE QUARTER - TRACK 184 - 200
454          :
455          :IF WE FIND ANY BAD SPOTS, WE WILL REPORT SO.....
456
457
458 022270 005237 002676          SETUP:  INC    STFLG          ;INDICATE A START COMMAND
459 022274 012737 177777 003006  MOV    #-1,SERNM1
460 022302 012737 177777 003010  MOV    #-1,SERNM2
461 022310          PRINTF    #FRM18
(7) 022310 012746 021742          MOV    #FRM18,-(SP)
(6) 022314 012746 000001          MOV    #1,-(SP)
(3) 022320 010600          MOV    SP,R0
(4) 022322 104017          EMT    C$PNTF
(4) 022324 062706 000004          ADD    #4,SP
462 022330 004537 030460          JSR    R5,LOAD          ;TELL OPERATOR TO LOAD
463 022334 004537 030030          JSR    R5,SERNUM        ;GET SERIAL NUMBER
464 022340 004537 027374          JSR    R5,MERGE        ;MERGE BAD SECTOR FILES
465 022344 012701 002124          MOV    #OUT10,R1      ;INITIALIZE ALL TRACKS
466 022350 012700 000031          MOV    #25,R0
467 022354 012721 177777 1$:    MOV    #177777,(R1)+
468 022360 005300          DEC    R0
469 022362 001374          BNE    1$
470
471 022364 004537 027562          JSR    R5,FNDTRK      ;TRY TO FIND FIVE TRACKS
472 022370 000001          1          ;INWARD SEARCH
473 022372 000000          0          ;TOP SURFACE
474 022374          000          020          .BYTE 0,16.          ;TRACK RANGE
475
476 022376 005737 002756          TST    TRKFND          ;WAS SEARCH SUCCESSFUL???
477 022402 001005          BNE    2$          ;YES
478
479 022404          ERRHRD 10,ERRFND,ERR1 ;NO TRACKS
(3) 022404 104463          TRAP  T$ERCODE
(5) 022406 000012          .WORD 10
(5) 022410 017462          .WORD ERRFND
(5) 022412 017756          .WORD ERR1
480 022414 000404          BR     3$
481
482 022416 012700 002124          2$:    MOV    #OUT10,R0          ;STORE AWAY TRACKS FOUND
483 022422 004537 027774          JSR    R5,FIXCYL
484
485 022426 004537 027562          3$:    JSR    R5,FNDTRK      ;TRY TO FIND FIVE TRACKS
486 022432 000001          1          ;INWARD SEARCH
487 022434 000001          1          ;BOTTOM SURFACE
488 022436          000          020          .BYTE 0,16.          ;TRACK RANGE
    
```

```

489
490 022440 005737 002756      TST      TRKFND      ;WAS      SEARCH  SUCCESSFUL????
491 022444 001005                BNE      4$          ;YES
492
493 022446                ERRHRD  10.,ERRFND,ERR1 ;NO TRACKS
(3) 022446 104463      TRAP    T$ERCODE
(5) 022450 000012      .WORD  10
(5) 022452 017462      .WORD  ERRFND
(5) 022454 017756      .WORD  ERR1
494 022456 000404      BR      5$
495
496 022460 012700 002131      4$:  MOV    #OUT11,RO      ;STORE TRACKS AWAY
497 022464 004537 027774      JSR    R5,FIXCYL
498 022470 004537 027562      5$:  JSR    R5,FNDTRK      ;FIND NEXT 5 TRACK
499 022474 177777      -1     ;OUTWARD SEARCH
500 022476 000000      0      ;TOP SURFACE
501 022500      376      356      .BYTE  254.,238.      ;TRACK RANGE
502
503 022502 005737 002756      TST      TRKFND      ;WAS SEARCH SUCCESSFUL?
504 022506 001005                BNE      6$          ;YES
505
506 022510                ERRHRD  10.,ERRFND,ERR1 ;NO TRACKS
(3) 022510 104463      TRAP    T$ERCODE
(5) 022512 000012      .WORD  10
(5) 022514 017462      .WORD  ERRFND
(5) 022516 017756      .WORD  ERR1
507 022520 000404      BR      7$          ;SKIP
508
509 022522 012700 002174      6$:  MOV    #INN10,RO     ;STORE AWAY TRACKS FOUND
510 022526 004537 027774      JSR    R5,FIXCYL
511
512 022532 004537 027562      7$:  JSR    R5,FNDTRK      ;NEXT SET
513 022536 177777      -1     ;OUTWARD SEARCH
514 022540 000001      1      ;BOTTOM SURFACE
515 022542      376      356      .BYTE  254.,238.      ;TRACK RANGE
516
517 022544 005737 002756      TST      TRKFND      ;SEARCH SUCCESSFUL?
518 022550 001005                BNE      8$          ;YES
519
520 022552                ERRHRD  10.,ERRFND,ERR1 ;NO TRACKS
(3) 022552 104463      TRAP    T$ERCODE
(5) 022554 000012      .WORD  10
(5) 022556 017462      .WORD  ERRFND
(5) 022560 017756      .WORD  ERR1
521 022562 000404      BR      9$
522
523 022564 012700 002201      8$:  MOV    #INN11,RO     ;STORE AWAY TRACKS FOUND
524 022570 004537 027774      JSR    R5,FIXCYL
525
526 022574 004537 027562      9$:  JSR    R5,FNDTRK      ;NEXT SET
527 022600 000001      1      ;INWARD SEARCH
528 022602 000000      0      ;TOP SURFACE
529 022604      176      210      .BYTE  126.,136.      ;TRACK RANGE
530
531 022606 005737 002756      TST      TRKFND      ;DID WE FIND A SET
532 022612 001015                BNE      10$         ;YES

```

```
533
534 022614 004537 027562      JSR    R5,FNDTRK      ;NEXT SET (OTHER SIDE)
535 022620 177777      -1      ;OUTWARD SEARCH
536 022622 000000      0      ;TOP SURFACE
537 022624      202      170    .BYTE   130.,120.    ;TRACK RANGE
538
539 022626 005737 002756      TST    TRKFND      ;DID WE FIND A SET
540 022632 001005      BNE    10$        ;YES
541
542 022634      ERRHRD  10.,ERRFND,ERR1 ;NO TRACKS
(3) 022634 104463      TRAP   T$ERCODE
(5) 022636 000012      .WORD  10
(5) 022640 017462      .WORD  ERRFND
(5) 022642 017756      .WORD  ERR1
543 022644 000404      BR     11$
544
545 022646 012700 002150      10$:  MOV    #MID10,R0      ;STORE AWAY
546 022652 004537 027774      JSR    R5,FIXCYL
547
548 022656 004537 027562      11$:  JSR    R5,FNDTRK      ;NEXT SET
549 022662 000001      1      ;INWARD SEARCH
550 022664 000001      1      ;BOTTOM SURFACE
551 022666      176      210    .BYTE   126.,136.    ;RANGE
552
553 022670 005737 002756      TST    TRKFND      ;SUCCESS?
554 022674 001015      BNE    12$        ;YES
555
556 022676 004537 027562      JSR    R5,FNDTRK      ;LOOK THE OTHER SIDE
557 022702 177777      -1     ;OUTWARD
558 022704 000001      1      ;BOTTOM SURFACE
559 022706      202      170    .BYTE   130.,120.    ;RANGE
560
561 022710 005737 002756      TST    TRKFND      ;SUCCESS?
562 022714 001005      BNE    12$        ;YES
563
564 022716      ERRHRD  10.,ERRFND,ERR1 ;NO TRACKS
(3) 022716 104463      TRAP   T$ERCODE
(5) 022720 000012      .WORD  10
(5) 022722 017462      .WORD  ERRFND
(5) 022724 017756      .WORD  ERR1
565 022726 000404      BR     13$
566
567 022730 012700 002155      12$:  MOV    #MID11,R0      ;STORE AWAY THE TRACKS FOUND
568 022734 004537 027774      JSR    R5,FIXCYL
569
570 022740 004537 027562      13$:  JSR    R5,FNDTRK      ;NEXT SET
571 022744 000001      1      ;INWARD
572 022746 000000      0      ;TOP SURFACE
573 022750      076      110    .BYTE   62.,72.      ;RANGE
574
575 022752 005737 002756      TST    TRKFND      ;SUCCESS?
576 022756 001015      BNE    14$        ;YES
577
578 022760 004537 027562      JSR    R5,FNDTRK      ;LOOK OTHER SIDE
579 022764 177777      -1     ;OUTWARD
580 022766 000000      0      ;TOP SURFACE
```


581	022770	102	070		.BYTE	66.,56.		:RANGE
582								
583	022772	005737	002756		TST	TRKFND		:SUCCESS?
584	022776	001005			BNE	14\$:YES
585								
586	023000				ERRHRD	10.,ERRFND,ERR1		:NO TRACKS
(3)	023000	104463			TRAP	T\$ERCODE		
(5)	023002	000012			.WORD	10		
(5)	023004	017462			.WORD	ERRFND		
(5)	023006	017756			.WORD	ERR1		
587	023010	000404			BR	15\$		
588								
589	023012	012700	002136	14\$:	MOV	#OQU10,RO		:STORE AWAY NEXT SET
590	023016	004537	027774		JSR	R5,FIXCYL		
591								
592	023022	004537	027562	15\$:	JSR	R5,FNDTRK		:LOOK FOR NEXT SET
593	023026	000001			1			:INWARD
594	023030	000001			1			:BOTTOM
595	023032	076	110		.BYTE	62.,72.		:RANGE
596								
597	023034	005737	002756		TST	TRKFND		:SUCCESS?
598	023040	001015			BNE	16\$:YES
599								
600	023042	004537	027562		JSR	R5,FNDTRK		:LOOK FOR ANOTHER SET
601	023046	177777			-1			:OUTWARD
602	023050	000001			1			:BOTTOM
603	023052	102	070		.BYTE	66.,56.		:RANGE
604								
605	023054	005737	002756		TST	TRKFND		:SUCCESS?
606	023060	001005			BNE	16\$:YES
607								
608	023062				ERRHRD	10.,ERRFND,ERR1		:NO TRACKS
(3)	023062	104463			TRAP	T\$ERCODE		
(5)	023064	000012			.WORD	10		
(5)	023066	017462			.WORD	ERRFND		
(5)	023070	017756			.WORD	ERR1		
609	023072	000404			BR	17\$		
610								
611	023074	012700	002143	16\$:	MOV	#OQU11,RO		:STORE AWAY TRACKS
612	023100	004537	027774		JSR	R5,FIXCYL		
613								
614	023104	004537	027562	17\$:	JSR	R5,FNDTRK		:NEXT SET OF TRACKS
615	023110	000001			1			:INWARD
616	023112	000000			0			:TOP SURFACE
617	023114	276	310		.BYTE	190.,200.		:RANGE
618								
619	023116	005737	002756		TST	TRKFND		:SUCCESS?
620	023122	001015			BNE	18\$:YES
621								
622	023124	004537	027562		JSR	R5,FNDTRK		:LOOK OTHER SIDE
623	023130	177777			-1			:OUTWARD SEARCH
624	023132	000000			0			:TOP
625	023134	302	270		.BYTE	194.,184.		:RANGE
626								
627	023136	005737	002756		TST	TRKFND		:SUCCESS
628	023142	001005			BNE	18\$:YES

```

629
630 023144          ERRHRD 10.,ERRFND,ERR1 ;NO TRACKS
(3) 023144 104463   TRAP    T$ERCODE
(5) 023146 000012   .WORD  10
(5) 023150 017462   .WORD  ERRFND
(5) 023152 017756   .WORD  ERR1
631 023154 000404   BR      19$
632
633 023156 012700 002162   18$:  MOV    #TQU10,R0          ;STORE TRACKS AWAY
634 023162 004537 027774   JSR    R5,FIXCYL
635
636 023166 004537 027562   19$:  JSR    R5,FNDTRK          ;NEXT SET
637 023172 000001           1          ;INWARD
638 023174 000001           1          ;BOTTOM SURFACE
639 023176      276      310   .BYTE  190.,200.          ;RANGE
640
641 023200 005737 002756   TST    TRKFND          ;SUCCESS?
642 023204 001015           BNE    20$             ;YES
643
644 023206 004537 027562   JSR    R5,FNDTRK          ;OTHER SET
645 023212 177777           -1         ;OUTWARD
646 023214 000001           1          ;BCTTOM SURFACE
647 023216      302      270   .BYTE  194.,184.          ;RANGE
648
649 023220 005737 002756   TST    TRKFND          ;SUCCESS
650 023224 001005           BNE    20$             ;YES
651
652 023226          ERRHRD 10.,ERRFND,ERR1 ;NO TRACKS
(3) 023226 104463   TRAP    T$ERCODE
(5) 023230 000012   .WORD  10
(5) 023232 017462   .WORD  ERRFND
(5) 023234 017756   .WORD  ERR1
653 023236 000404   BR      21$
654
655 023240 012700 002167   20$:  MOV    #TQU11,R0          ;STORE SET AWAY
656 023244 004537 027774   JSR    R5,FIXCYL
657
658 023250 012700 002124   21$:  MOV    #OUT10,R0          ;DID WE FIND ANY AT ALL
659 023254 012701 000062   MOV    #50.,R1
660 023260 122720 000377   22$:  CMPB   #377,(R0)+
661 023264 001016           BNE    EXIT
662 023266 005301           DEC    R1
663 023270 001373           BNE    22$
664 023272          ERRSF 3.,NONE
(3) 023272 104421   TRAP    T$ERCODE
(5) 023274 000003   .WORD  3
(5) 023276 017310   .WORD  NONE
665 023300 005001           CLR    R1
666 023302 013700 002012   CMPENA: MOV   L$UNIT,R0
667 023306          24$:  DODU   R1          ;DO DROP UNIT
(3) 023306 010100   MOV    R1,R0
(3) 023310 104053   EMT    C$DODU
668 023312 005201           INC    R1
669 023314 005300           DEC    R0
670 023316 001373           BNE    24$
671 023320          DOCLN
    
```

```

(3) 023320 104044          EMT      C$DCLN
672
673
674 023322          EXIT:
675
676 023322          ENDINIT
(3) 023322          L10007:
(3) 023322 104011      EMT      C$INIT
677
678 023324          ENDMOD
679
680 023324          BGNMOD  CLNCODE
681
682
683 023324          BGNCLN
684
685
686
687 023324 000240      NOP
688
689
690 023326          ENDCLN
(3) 023326          L10010:
(3) 023326 104012      EMT      C$CLEAN
691
692 023330          ENDMOD
693
694 023330          BGNMOD  DRPCODE
695 023330          BGNMOD  BGNDU
696 023330 000240      NOP
697 023332          ENDDU
(3) 023332          L10011:
(3) 023332 104055      EMT      C$DU
698 023334          ENDMOD
699          .SBTTL GLOBAL SUBROUTINES SECTION
700 023334          BGNMOD  GLBSUB
701
702          ;ALL COMMON OR GLOBAL SUBROUTINES GO HERE
703
704          ;ROUTINE TO PERFORM OVERWRITE
705          ;CALL: JSR      R5,OVWPER
706          ;      SECTORS TO WRITE FORWARD
707          ;      SECTORS TO WRITE REVERSE
708
709 023334 010046      OVWPER: MOV      R0,-(SP)      ;SAVE R0, R1, R2, R3
710 023336 010146      MOV      R1,-(SP)
711 023340 010246      MOV      R2,-(SP)
712 023342 010346      MOV      R3,-(SP)
713 023344 005000      CLR      R0
714 023346 012537 002724  MOV      (R5)+,FORSK  ;R0 HAS COUNT IF R0<5.
715 023352 012537 002722  MOV      (R5)+,REVS  ;USE TOP SURFACE, IF R0>5.
716          ;USE BOTTOM SURFACE, IF R0>1
717 023356 012701 002626  ;DONE.
718 023362 011102          1$:  MOV      #OVWTRK,R1  ;GET START OF LIST OF TRACKS
719 023364 121227 177777  MOV      (R1),R2      ;GET POINTER TO TRACK
720 023370 001464          CMPB   (R2),#-1      ;LEGIT TRACK??????
          BEQ      3$      ;NO, EXIT
  
```

```

721
722 023372 005037 002720 CLR DESCYL ;CLEAR CYLINDER/HEAD FOR SEEK
723 023376 020027 000005 CMP R0,#5 ;TOP/BOTTOM
724 023402 002402 BLT 2$ ;TOP, BRANCH
725 023404 105237 002721 INCB DESCYL+1 ;BOTTOM SURFACE
726 023410 004537 024716 2$: JSR R5,SKCYL ;SEEK TO CYLINDER
727 023414 105037 002720 CLRB DESCYL
728 023420 151237 002720 BISB (R2),DESCYL
729 023424 004537 024716 JSR R5,SKCYL ;SEEK TO PROPER CYLINDER
730 023430 013703 002724 MOV FORSK,R3 ;SECTORS TO WRITE
731 023434 004537 023566 JSR R5,WRSEC ;GO WRITE SECTORS
732 023440 000034 .WORD 28.
733 023442 012737 017635 002716 MOV #FWD,DIRC ;SET FORWARD DIRECTION
734 023450 004537 025622 JSR R5,VEROW ;VERIFY OVERWRITE
735 023454 004537 026206 JSR R5,VEROD ;VERIFY OTHER DRIVES DATA
736 023460 105037 002720 CLRB DESCYL ;SET UP FOR SEEK TO
737 023464 052737 000377 002720 BIS #377,DESCYL ;INNER GUARD BAND
738 023472 004537 024716 JSR R5,SKCYL ;DO THE SEEK
739
740 023476 105037 002720 CLRB DESCYL ;SET UP FOR SEEK TO
741 023502 151237 002720 BISB (R2),DESCYL ;DESIRED TRACK
742 023506 004537 024716 JSR R5,SKCYL ;DO ANOTHER SEEK
743
744 023512 013703 002722 MOV REVSK,R3 ;SECTORS TO WRITE
745 023516 004537 023566 JSR R5,WRSEC ;WRITE THEM
746 023522 000034 .WORD 28.
747 023524 012737 017645 002716 MOV #REV,DIRC ;SET DIRECTION
748 023532 004537 025622 JSR R5,VEROW ;VERIFY OVERWRITE
749 023536 004537 026206 JSR R5,VEROD ;VERIFY OTHER DRIVES DATA
750
751 023542 005721 3$: TST (R1)+ ;INCREMENT TO NEXT TRACK
752 023544 005200 INC R0 ;ACCOUNT FOR IT
753 023546 020027 000012 CMP R0,#10. ;DONE?
754 023552 001303 BNE 1$ ;NO, GO BACK
755
756 023554 012603 MOV (SP)+,R3 ;RESTORE REG.
757 023556 012602 MOV (SP)+,R2
758 023560 012601 MOV (SP)+,R1
759 023562 012600 MOV (SP)+,R0
760 023564 000205 RTS R5 ;EXIT
761
762 ;ROUTINE TO WRITE SECTORS
763 ;USED IN OVERWRITE TEST;ADJACENT CYLINDER TEST
764 ;CALL JSR R5,WRSEC
765 ; .WRD ;STARTING SECTOR
766 ;R3 HAS BITMAP OF SECTORS TO WRITE
767 ;R4 HAS DRIVE BUFFER POINTER
768
769 023566 010046 WRSEC: MOV R0,-(SP) ;SAVE R0
770 023570 010146 MOV R1,-(SP) ;SAVE R1
771 023572 010246 MOV R2,-(SP) ;SAVE R2
772 023574 012701 003030 MOV #BUF,R1 ;WRITE PATTERN INTO
773 023600 012702 000200 MOV #128,R2 ;MEMORY THAT WE
774 023604 016421 000006 2$: MOV PAT(R4),(R1)+ ;WILL WRITE ONTO
775 023610 005302 DEC R2 ;PACK FOR THIS
776 023612 001374 BNE 2$ ;DRIVE
    
```

```

777 023614 012701 100000      MOV      #100000,R1      ;MASK FOR BIT MAP
778 023620 153702 002720      BISB     DESCYL,R2      ;GET CYLINDER
779 023624 000302              SWAB     R2              ;PUT IN HIGH BYTE
780 023626 006002              ROR     R2              ;ALIGN FOR DISK ADDRESS
781 023630 032737 000400 002720  BIT      #400,DESCYL    ;WHICH SURFACE
782 023636 001402              BEQ     3$              ;0, SKIP
783 023640 052702 000100      BIS      #HEAD,R2      ;SET BOTTOM HEAD
784 023644 052502          3$:  BIS      (R5)+,R2      ;START AT SECTOR 36
785 023646 030103          4$:  BIT      R1,R3      ;WRITE THIS SECTOR?
786 023650 001452              BEQ     5$              ;NO
787
788 023652 005037 002710      CLR     HSFLG          ;
789 023656 012737 177600 003004  MOV     #-128.,BMP     ;LOAD WORD COUNT
790 023664 010237 003002      MOV     R2,BDA        ;LOAD DISK ADDRESS
791 023670 010237 002666      MOV     R2,TEMP       ;SAVE DISK ADDRESS
792 023674 042702 177700      BIC     #177700,R2
793 023700 020227 000047      CMP     R2,#39.
794 023704 003403              BLE     6$
795 023706 162737 000050 003002  SUB     #40.,BDA
796 023714 012737 003030 003000  6$:  MOV     #BUF,BBA      ;LOAD BUS ADDRESS
797 023722 013702 002666      MOV     TEMP,R2       ;RESTORE DISK ADDRESS
798 023726 004537 030560 11$:  JSR     R5,LDFUNC     ;GO WRITE
799 023732 000012              WRITE
800 023734 005737 002674      TST     ERFLG         ;ERROR IN WRITING
801 023740 001416              BEQ     5$             ;NO,OKAY
802 023742 005737 002710      TST     HSFLG
803 023746 001007              BNE     10$
804 023750              ERRSOFT 100.,WRIT1,ERR2
(3) 023750 104464              TRAP   T$ERCODE
(5) 023752 000144              .WORD 100
(5) 023754 017655              .WORD WRIT1
(5) 023756 020014              .WORD ERR2
805 023760 005237 002710      INC     HSFLG
806 023764 000760              BR     11$
807 023766          10$:  ERRHRD 110.,WRIT1,ERR2
(3) 023766 104463              TRAP   T$ERCODE
(5) 023770 000156              .WORD 110
(5) 023772 017655              .WORD WRIT1
(5) 023774 020014              .WORD ERR2
808
809 023776 005202          5$:  INC     R2              ;NEXT SECTOR
810 024000 000241              CLC                    ;CLEAR CARRY BIT
811 024002 006001              ROR     R1              ;DONE?
812 024004 103320              BCC     4$             ;NO GO BACK
813 024006 012602              MOV     (SP)+,R2       ;REGISTERS AND EXIT
814 024010 012601              MOV     (SP)+,R1
815 024012 012600              MOV     (SP)+,R0
816 024014 000205              RTS     R5
817
818 024016 005037 003012  ADJCYL: CLR     ADJTRK   ;INSIDE/OUTSIDE TRACK FLAG
819 024022 005037 002714      CLR     HEAD01        ;INIT TO TOP SURFACE
820 024026 012737 000001 003014  MOV     #1,ADJUJ      ;START OF TRACK LIST
821 024034 012701 002124          21$:  MOV     #OUT10,R1
822 024040 012537 002700          20$:  MOV     (R5)+,ADJLOC  ;PICK UP TRACK OFFSET
823 024044 001003              BNE     1$             ;IS THERE ONE?
824 024046 005037 002704      CLR     ADJDIR
    
```

825	024052	000205				RTS	R5		:NO EXIT
826	024054	012537	003016		1\$:	MOV	(R5)+,ADJLC2		:YES, GET REST OF INFO
827	024060	012537	003020			MOV	(R5)+,ADJLC3		
828	024064	012537	003022			MOV	(R5)+,ADJLC4		
829	024070	113700	002700		2\$:	MOVB	ADJLOC,R0		:GET OFFSET
830	024074	012737	000020	003026		MOV	#16.,STSEC		:STARTING SECTOR IS 16
831									
832	024102	010102				MOV	R1,R2		:GET START INTO R2
833									
834	024104	005300			3\$:	DEC	R0		:DOWN COUNT OFFSET
835	024106	001414				BEQ	4\$:FOUND IT?
836									
837	024110	105722				TSTB	(R2)+		:INDEX (R2)
838	024112	062737	000042	003026		ADD	#34.,STSEC		:NO, NEXT SECTOR
839	024120	022737	000050	003026		CMP	#40.,STSEC		
840	024126	003366				BGT	3\$		
841	024130	162737	000050	003026		SUB	#40.,STSEC		
842	024136	000762				BR	3\$:BACK FOR NEXT
843									
844	024140	121227	000377		4\$:	CMPB	(R2),#377		:LEGAL TRACK?
845	024144	001002				BNE	5\$:YES, CONTINUE
846									
847	024146	000137	024570			JMP	13\$:NO PICK UP NEXT SET
848									
849	024152	005037	002720		5\$:	CLR	DESCYL		:SET UP FOR OUTER TRACK
850									
851	024156	005737	002714			TST	HEAD01		:WHICH HEAD?
852	024162	001403				BEQ	6\$:TOP, SKIP
853									
854	024164	052737	000400	002720		BIS	#400,DESCYL		:LOWER HEAD, SET IT!
855									
856	024172	004537	024716		6\$:	JSR	R5,SKCYL		:SEEK TO OUTER TRACK
857									
858	024176	111237	002720			MOVB	(R2),DESCYL		:GET DESIRED TRACK
859									
860	024202	004537	024716			JSR	R5,SKCYL		:SEEK TO IT
861	024206	012737	017635	002716		MOV	#FWD,DIRC		:SEEK DIRECTION
862	024214	113703	002701			MOVB	ADJLOC+1,R3		:GET SECTORS TO WRITE
863	024220	000303				SWAB	R3		:ALIGN IT
864	024222	042703	000377			BIC	#377,R3		:CLEAR OUT HIGH BYTE
865									
866	024226	022737	000047	003026		CMP	#39.,STSEC		:OVER FORTY?
867	024234	002003				BGE	7\$:NO, CONTINUE
868									
869	024236	162737	000050	003026		SUB	#40.,STSEC		:YES BACK IT UP
870	024244	013737	003026	024256	7\$:	MOV	STSEC,8\$:STARTING SECTOR
871									
872	024252	004537	023566			JSR	R5,WRSEC		:WRITE SECTORS
873	024256	000000			8\$:	.WORD	0		
874	024260	013737	024256	024272		MOV	8\$,108\$		
875	024266	004537	026534			JSR	R5,VAJWR		:VERIFY THIS WRITE
876	024272	000000			108\$:	.WORD	0		
877	024274	013737	024272	024306		MOV	108\$,208\$		
878	024302	004537	026770			JSR	R5,BSVWR		
879	024306	000000			208\$:	.WORD	0		
880	024310	013737	003026	003024		MOV	STSEC,STSEC1		:GET OTHER SECTORS TO WRITE


```

937
938 024570 005737 002714      13$:  TST      HEAD01      :WHICH HEAD WERE WE DOING?
939 024574 001003
940 024576 005237 002714      INC      HEAD01
941 024602 000402      BR       99$
942 024604 005037 002714      14$:  CLR      HEAD01      :NEXT SET OF TRACKS
943 024610 0062701 000005      99$:  ADD      #5,R1      :NEXT SET OF TRACKS
944 024614 020127 002205      CMP      R1,#INN51      :END OF LIST
945 024620 002002      BGE     18$      :END OF TRACK LIST
946 024622 000137 024070      JMP     2$      :NO GO BACK
947
948      :AT END OF TRACK LIST NEXT GROUP OF WRITES
949
950 024626 005737 002664      18$:  TST      FADJ      :FIRST SET?
951 024632 001403      BEQ     15$      :NO, CONTINUE
952 024634 005037 002664      CLR     FADJ      :YES, CLEAR FIRST
953 024640 000421      BR      17$      :EXIT
954 024642 005737 003012      15$:  TST      ADJTRK     :DONE BOTH INSIDE OUTSIDE
955 024646 001004      BNE     16$      :TRACKS, YES 16$
956 024650 005237 003012      INC     ADJTRK     :NO, SET INSIDE FLAG
957 024654 000137 024034      JMP     21$      :GO DO INSIDE TRACK
958 024660 005037 003012      16$:  CLR     ADJTRK     :BACK TO OUTSIDE TRACK
959 024664 005237 003014      INC     ADJUUT     :DONE WITH ANOTHER
960 024670 023737 003014 002726      CMP     ADJUUT,UUT :DONE TABLE FOR ALL UUT?
961 024676 001402      BEQ     17$      :YES, FOR EXIT
962 024700 000137 024034      JMP     21$      :NO, GO BACK FOR NEXT
963 024704 005725      17$:  TST      (R5)+      :BUMP EXIT TO END OF
964 024706 001376      BNE     17$      :TABLE FOR PROPER RETURN
965 024710 005037 002704      CLR     ADJDIR     :EXIT
966 024714 000205      RTS     R5
967
968
969      :ROUTINE TO SEEK TO A DESIRED CYLINDER
970      :CALL: JSR      R5,SKCYL
971      :ROUTINE HAS DESIRED CYLINDER IN LOC 'DESCYL'
972      :
973      :
974 024716 010146      SKCYL: MOV     R1,-(SP)      :SAVE R1
975 024720 004537 030560      90$:  JSR     R5,LDFUNC      :GET PRESENT POSITION
976 024724 000010      RDHDR
977
978 024726 005737 002674      TST     ERFLG      :ERROR FLAG SET
979 024732 001074      BNE     5$      :YES, SKIP
980
981 024734 005001      CLR     R1
982 024736 153701 002720      BISB   DESCYL,R1      :GET DESIRED CYLINDER
983 024742 000301      SWAB   R1      :GET IN HIGH BYTE
984 024744 006001      ROR    R1      :ALIGN IT
985 024746 042737 000177 002770      BIC    #177,E,MP      :CLEAR PRESENT HD:SEC
986 024754 163701 002770      SUB    E,MP,R1      :CALCULATE DIFFERENCE WORD
987 024760 100002      BPL    1$      :IF POSITIVE SET DIRECTION
988 024762 005401      NEG    R1      :NEGATE
989 024764 000402      BR     2$      :SKIP SETTING DIRECTION
990 024766 052701 000004      1$:  BIS    #SIGN,R1      :SET FOR FORWARD SEEK
991 024772 052701 000001      2$:  BIS    #1,R1      :SET MARKER BIT
992 024776 032737 000400 002720      BIT    #400,DESCYL      :WHICH HEAD

```



```

993 025004 001402          BEQ      3$          ;TOP
994 025006 052701 000020   BIS      #SKHS,R1    ;BOTTOM
995 025012 010137 003002   3$:    MOV      R1,BDA  ;LOAD DIFFERENCE WORD
996 025016 004537 030560   JSR      R5,LDFUNC  ;EXECUTE SEEK
997 025022 000006          SEEK
998
999 025024 005737 002674   TST      ERFLG      ;ERROR?
1000 025030 001035          BNE      5$          ;YES, SKIP
1001
1002 025032 004537 030560   JSR      R5,LDFUNC  ;VERIFY POSITION?
1003 025036 000010          RDHDR
1004 025040 005737 002674   TST      ERFLG
1005 025044 001027          BNE      5$
1006 025046 042737 000077 002770   BIC      #77,E.MP   ;VERIFY POSITION
1007 025054 005001          CLR      R1
1008 025056 153701 002720   BISB    DESCYL,R1  ;IS CORRECT AND IF
1009 025062 000301          SWAB    R1         ;NOT CORRECT THEN
1010 025064 006001          ROR     R1         ;RESEEK
1011 025066 032737 000400 002720   BIT     #400,DESCYL
1012 025074 001402          BEQ     4$
1013 025076 052701 000100          BIS     #HEAD,R1
1014 025102 020137 002770   4$:    CMP     R1,E.MP
1015 025106 001414          BEQ     6$
1016
1017 025110          ERRDF  12.,SKER,ERR6 ;SEEK ERROR
   (3) 025110 104462          TRAP   T$ERCODE
   (5) 025112 000014          .WORD 12
   (5) 025114 017616          .WORD SKER
   (5) 025116 020432          .WORD ERR6
1018 025120 000137 024720   JMP     90$
1019
1020 025124          5$:    ERRDF  13.,FUNERR,ERR5 ;FUNCTION ERROR IN SEEK
   (3) 025124 104462          TRAP   T$ERCODE
   (5) 025126 000015          .WORD 13
   (5) 025130 017566          .WORD FUNERR
   (5) 025132 020372          .WORD ERR5
1021 025134 000137 024720   JMP     90$
1022 025140 012601          6$:    MOV     (SP)+,R1   ;CANT GET THERE
1023 025142 000205          RTS     R5         ;EXIT
1024
1025
1026          ;ROUTINE TO PERFORM REGISTER PRINTOUT DUMP
1027          ;CALL: JSR PC,REGDMP
REGDMP: PRINTB #FRM12,BCS,BBA,BDA,BMP
   (11) 025144 013746 003004   MOV     BMP,-(SP)
   (10) 025150 013746 003002   MOV     BDA,-(SP)
   (9)  025154 013746 003000   MOV     BBA,-(SP)
   (8)  025160 013746 002776   MOV     BCS,-(SP)
   (7)  025164 012746 021371   MOV     #FRM12,-(SP)
   (6)  025170 012746 000005   MOV     #5,-(SP)
   (3)  025174 010600          MOV     SP,R0
   (4)  025176 104014          EMT    C$PNTB
   (4)  025200 062706 000014   ADD     #14,SP
1029 025204          PRINTB #FRM13,E.CS,E.BA,E.DA,E.MP
   (11) 025204 013746 002770   MOV     E.MP,-(SP)
   (10) 025210 013746 002766   MOV     E.DA,-(SP)

```

```

(9) 025214 013746 002764      MOV      E.BA,-(SP)
(8) 025220 013746 002762      MOV      E.CS,-(SP)
(7) 025224 012746 021450      MOV      #FRM13,-(SP)
(6) 025230 012746 000005      MOV      #5,-(SP)
(3) 025234 010600      MOV      SP,R0
(4) 025236 104014      EMT      C$PNTB
(4) 025240 062706 000014      ADD      #14,SP
1030 025244 032737 040000 002762      BIT      #BIT14,E.CS
1031 025252 001437      BEQ      1$
1032 025254 016403 000000      MOV      CSR(R4),R3
1033 025260 012763 000013 000004      MOV      #13,DA(R3)
1034 025266 012737 000004 002776      MOV      #4,BCS
1035 025274 056437 000004 002776      BIS      DSB(R4),BCS
1036 025302 013763 002776 000000      MOV      BCS,CS(R3)
1037 025310 032763 000200 000000 2$:      BIT      #200,CS(R3)
1038 025316 001774      BEQ      2$
1039 025320 016337 000006 002706      MOV      MP(R3),DRSTAT
1040 025326      PRINTB #FRM14,DRSTAT
(8) 025326 013746 002706      MOV      DRSTAT,-(SP)
(7) 025332 012746 021533      MOV      #FRM14,-(SP)
(6) 025336 012746 000002      MOV      #2,-(SP)
(3) 025342 010600      MOV      SP,R0
(4) 025344 104014      EMT      C$PNTB
(4) 025346 062706 000006      ADD      #6,SP
1041 025352 000207      1$:      RTS      PC
1042
1043      ;ROUTINE TO SET DRIVE IN SECTOR LIST
1044      ;CALL: JSR R5,SETLST      ;R0 HAS SECTOR
1045      ;DRIVE GOTTEN FROM R4
1046
1047 025354 010146      SETLST: MOV      R1,-(SP)      ;SAVE R1
1048
1049 025356 162700 000034      SUB      #28.,R0      ;START LIST AT 0
1050 025362 100002      BPL      3$
1051 025364 062700 000050      ADD      #40.,R0
1052 025370 012701 002206 3$:      MOV      #SECLST,R1      ;BEGINNING OF SECTOR LIST
1053 025374 005700 1$:      TST      R0      ;FOUND SECTOR?
1054 025376 001403      BEQ      2$      ;BRANCH IF YES
1055 025400 005300      DEC      R0      ;DECREMENT SECTOR
1056 025402 005721      TST      (R1)+      ;NEXT ENTRY IN LIST
1057 025404 000773      BR      1$      ;GO BACK
1058 025406 010411 2$:      MOV      R4,(R1)      ;STORE DRIVE BITS IN LIST
1059 025410 012601      MOV      (SP)+,R1      ;RESTORE R1
1060 025412 000205      RTS      R5
1061
1062      ;ROUTINE TO STORE OR RETRIEVE ADJACENT CYLINDER SECTOR DRIVE
1063      ;INFORMATION FROM THE 24X5 'SECLST' BUFFER.
1064      ;ENTER WITH R0 = SECTOR REQUEST
1065      ;EXIT WITH R0 = ADJACENT CYLINDER DRIVE INFORMATION FOR SECTOR
1066      ;EXIT WITH R0 = 0 IF SECTOR REQUESTED IS NOT IN BUFFER MAP
1067      ;CALL 1: JSR R5,RSADJS
1068      ;      .WORD 0      ;RETRIEVE SECTOR INFO.
1069      ;CALL 2: JSR R5,RSADJS
1070      ;      .WORD 1      ;STORE SECTOR INFO.
1071 025414 010146      RSADJS: MOV      R1,-(SP)
1072 025416 010246      MOV      R2,-(SP)
  
```

1073	025420	010346		MOV	R3, -(SP)	
1074	025422	042700	177700	BIC	#177700, R0	:SAVE SECTOR BITS
1075	025426	012537	002702	MOV	(R5)+, ADJFLG	:SAVE RETRIEVE/STORE FLAG
1076	025432	012701	000001	MOV	#1, R1	:START WITH TRACK (N-2)
1077	025436	012702	002246	MOV	#SECBUF, R2	:START OF 24X5 BUFFER
1078	025442	012703	000020	MOV	#16, R3	:SECTOR 16 START FOR (N-2) TRACK
1079	025446	123701	002700	1\$: CMPB	ADJLOC, R1	:CHECK TRACK INDEX
1080	025452	001413			2\$: BEQ	
1081	025454	005201			INC	R1
1082	025456	062702	000060		ADD	#48, R2
1083	025462	062703	000042		ADD	#34, R3
1084	025466	020327	000050		CMP	R3, #40.
1085	025472	002765			BLT	1\$
1086	025474	162703	000050		SUB	#40, R3
1087	025500	000762			BR	1\$
1088	025502	012701	000030	2\$:	MOV	#24, R1
1089	025506	020003		3\$:	CMP	R0, R3
1090	025510	001413			BEQ	5\$
1091	025512	005722			TST	(R2)+
1092	025514	005203			INC	R3
1093	025516	020327	000047		CMP	R3, #39.
1094	025522	003402			BLE	4\$
1095	025524	162703	000050		SUB	#40, R3
1096	025530	005301		4\$:	DEC	R1
1097	025532	001365			BNE	3\$
1098	025534	005000			CLR	R0
1099	025536	000405			BR	7\$
1100	025540	005737	002702	5\$:	TST	ADJFLG
1101	025544	001401			BEQ	6\$
1102	025546	010412			MOV	R4, (R2)
1103	025550	011200		6\$:	MOV	(R2), R0
1104	025552	012603		7\$:	MOV	(SP)+, R3

```

:SAVE SECTOR BITS
:SAVE RETRIEVE/STORE FLAG
:START WITH TRACK (N-2)
:START OF 24X5 BUFFER
:SECTOR 16 START FOR (N-2) TRACK
:CHECK TRACK INDEX
:INDEX TRACK REFERENCE
:UPDATE BUFFER TO NEXT TRACK REF.
:UPDATE SECTOR START FOR NEXT TRACK

:SET COUNTER FOR 24 SECTORS
:COMPARE SECTOR TO SECTOR TABLE
:YES, STORE OR RETRIEVE SECTOR INFO.
:INDEX SECLST BUFFER IN WORD FORMAT
:INDEX SECTOR COUNT
:COMPARE SECTOR COUNT FOR <40

:KEEP SECTOR COUNT<40
:PASSED 24 SECTORS?
:COMPARE NEXT SECTOR
:SETUP R0 FOR EXIT
:EXIT ROUTINE, SECTOR NOT FOUND
:FLAG=0 FOR RETRIEVE

:STORE DRIVE INFO. INTO BUFFER
:SAVE DRIVE INFO. INTO R0 FOR EXIT

```

```

1106 025554 012602          MOV      (SP)+,R2
1107 025556 012601          MOV      (SP)+,R1
1108 025560 000205          RTS      R5              ;EXIT
1109
1110          ;ROUTINE TO LOCATE DRIVE THAT WROTE SECTOR LAST
1111          ;CALL: JSR      R5,FNDDRV          ;R0-CONTAINS SECTOR
1112          ;ON EXIT R0-DRIVE
1113
1114 025562 010146          FNDDRV: MOV      R1,-(SP)          ;SAVE R1
1115 025564 162700 000034    SUB      #28.,R0          ;START LIST AT 0
1116 025570 100002          BPL     3$
1117 025572 062700 000050    ADD     #40.,R0
1118 025576 012701 002206    3$:     MOV     #SECLST,R1      ;START OF LIST
1119 025602 005700          1$:     TST     R0              ;FOUND SECTOR?
1120 025604 001403          BEQ     2$                  ;YES, GET DRIVE #, EXIT
1121 025606 005300          DEC     R0                  ;NO, DOWN COUNT SECTOR
1122 025610 005721          TST     (R1)+              ;NEXT ENTRY IN LIST
1123 025612 000773          BR      1$                  ;GO BACK
1124 025614 011100          2$:     MOV     (R1),R0        ;GET DRIVE BUFFER POINTER
1125 025616 012601          MOV     (SP)+,R1          ;RESTORE R1
1126 025620 000205          RTS      R5              ;EXIT
1127
1128          ;
1129          ;ROUTINE TO VERIFY THAT THE OVERWRITE DID ACTUALLY OVERWRITE THE
1130          ;PREVIOUS DATA ON THE PACK.
1131          ;
1132          ;CALL: JSR      R5,VEROW          USES R3 AS BIT MAP OF SECTORS TO
1133          ;CHECK. R3 IS LOADED PRIOR TO
1134          ;WRITING SECTORS.
1135          ;
1136 025622 010046          VEROW: MOV     R0,-(SP)          ;SAVE REGISTER CONTENTS
1137 025624 010146          MOV     R1,-(SP)
1138 025626 010246          MOV     R2,-(SP)
1139 025630 012737 000034 002730    MOV     #28.,SECT          ;START VERIFY AT SECTOR 28
1140 025636 012701 100000    MOV     #100000,R1         ;BIT MASK FOR VERIFICATION
1141 025642 016437 000006 002734    MOV     PAT(R4),GDATA     ;GET PATTERN FOR THIS DRIVE
1142
1143 025650 012737 177600 003004 1$:     MOV     #-128.,BMP        ;SET UP READ-ONE SECTOR
1144 025656 012737 003030 003000    MOV     #BUF,BBA          ;BUS ADDRESS
1145 025664 042737 000077 003002 2$:     BIC     #77,BDA          ;CLEAR OUT SECTOR BITS
1146 025672 053737 002730 003002    BIS     SECT,BDA          ;SET SECTOR
1147 025700 030103          BIT     R1,R3              ;DO WE READ THIS ONE?
1148 025702 001521          BEQ     5$                  ;NO, BRANCH
1149 025704 004537 030560    JSR     R5,LDFUNC          ;READ
1150 025710 000014          READ
1151
1152 025712 005737 002762          TST     E.CS              ;ERROR
1153 025716 100107          BPL     4$                  ;NO CONTINUE
1154
1155 025720 005737 002662          TST     FOWR              ;INITIAL WRITE
1156 025724 001412          BEQ     21$                ;NO
1157 025726 012737 017112 002672    MOV     #INITWR,REASON    ;SETUP INITIAL WRITE OF SECTOR
1158 025734 016437 000000 002670    MOV     CSR(R4),LSTCLR
1159 025742 016437 000005 002732    MOV     DSB+1(R4),LSTDRV
1160 025750 000415          BR      22$
1161 025752 012737 017361 002672 21$:    MOV     #OVMES,REASON    ;SET MESSAGE FOR OVERWRITE

```

```

1162 025760 013700 002730      MOV      SECT,R0      ;FIND DRIVE THAT LAST WROTE
1163 025764 004537 025562      JSR      R5,FNDDRV   ;SECTOR
1164 025770 016037 000000 002670  MOV      CSR(R0),LSTCLR ;GET IT'S CSR
1165 025776 116037 000005 002732  MOVVB   DSB+1(R0),LSTDRV ;GET THE DRIVE
1166 026004      ERRDF 13.,OVWER,ERR4 ;PRINT ERROR
(3) 026004 104462      TRAP   T$ERCODE
(5) 026006 000015      .WORD 13
(5) 026010 017522      .WORD OVWER
(5) 026012 020206      .WORD ERR4
1167 026014 005037 002740      CLR      WCOUNT     ;CLEAR BAD WORD COUNT W/IN SECTOR
1168 026020 005037 002742      CLR      SECWRD      ;CLEAR WORD IN SECTOR
1169 026024 012702 003030      MOV      #BUF,R2     ;GET BUFFER START
1170 026030 023712 002734      CMP      GDATA,(R2)  ;IS DATA CORRECT?
1171 026034 001417      BEQ     31$          ;YES CHECK NEXT
1172 026036 005237 002740      INC      WCOUNT     ;NO ACCOUNT FOR IT
1173 026042      PRINTF #FRM8,SECWRD,GDATA,(R2)
(10) 026042 011246      MOV      (R2),-(SP)
(9) 026044 013746 002734      MOV      GDATA,-(SP)
(8) 026050 013746 002742      MOV      SECWRD,-(SP)
(7) 026054 012746 021162      MOV      #FRM8,-(SP)
(6) 026060 012746 000004      MOV      #4,-(SP)
(3) 026064 010600      MOV      SP,R0
(4) 026066 104017      EMT     C$PNTF
(4) 026070 062706 000012      ADD     #12,SP
1174
1175 026074 005722      TST     (R2)+        ;NEXT
1176 026076 005237 002742      INC     SECWRD       ;NEXT
1177 026102 023727 002742 000200  CMP     SECWRD,#128. ;DONE WITH SECTOR?
1178 026110 001347      BNE    3$           ;NO GO BACK
1179
1180 026112      PRINTF #FRM9,WCOUNT ;PRINT SUMMARY
(8) 026112 013746 002740      MOV     WCOUNT,-(SP)
(7) 026116 012746 021226      MOV     #FRM9,-(SP)
(6) 026122 012746 000002      MOV     #2,-(SP)
(3) 026126 010600      MOV     SP,R0
(4) 026130 104017      EMT     C$PNTF
(4) 026132 062706 000006      ADD     #6,SP
1181
1182 026136 013700 002730      MOV     SECT,R0     ;SET SECTOR IN LIST TO THE
1183 026142 004537 025354      JSR     R5,SETLST  ;CREDIT OF THIS DRIVE
1184
1185 026146 005237 002730      INC     SECT        ;NEXT SECTOR
1186 026152 023727 002730 000050  CMP     SECT,#40.
1187 026160 001003      BNE    6$
1188 026162 162737 000050 002730  SUB     #40.,SECT
1189 026170 000241      CLC
1190 026172 006001      ROR    R1           ;CLEAR CARRY
1191 026174 103225      BCC   1$           ;NEXT BIT
1192
1193 026176 012602      MOV     (SP)+,R2    ;RESTORE R2-R0, EXIT
1194 026200 012601      MOV     (SP)+,R1
1195 026202 012600      MOV     (SP)+,R0
1196 026204 000205      RTS     R5
1197
1198
1199
;ROUTINE TO VERIFY THAT A DRIVE CAN RECOVER ANOTHER DRIVE'S DATA.

```

```

1200 ;CALL: JSR R5,VEROD USES R3 AS BIT MAP OF SECTORS TO
1201 ;: CHECK. R3 IS LOAD BY WRSEC (WE
1202 ;: USE R3 COMPLIMENTED.
1203 ;:
1204 ;:
1205 026206 010046 VEROD: MOV R0,-(SP) ;SAVE R0-R2
1206 026210 010146 MOV R1,-(SP)
1207 026212 010246 MOV R2,-(SP)
1208 026214 012701 100000 MOV #100000,R1 ;BIT MASK FOR SECTORS
1209 026220 012737 000034 002730 MOV #28.,SECT ;START WITH SECTOR 28
1210 026226 005737 002662 TST F0WR ;CHECK FOR FIRST OVERWRITE
1211 026232 001134 BNE 6$
1212
1213 026234 012737 177600 003004 1$: MOV #-128.,BMP ;SET UP READ (ONE SECTOR)
1214 026242 012737 003030 003000 MOV #BUF,BBA ;BUS ADDRESS
1215 026250 042737 000077 003002 2$: BIC #77,BDA ;CLEAR SECTOR BITS
1216 026256 053737 002730 003002 BIS SECT,BDA ;SET IN SECTOR BITS
1217 026264 030103 BIT R1,R3 ;CHECK THIS SECTOR?
1218 026266 001103 BNE 5$ ;NO BRANCH
1219
1220 026270 013700 002730 MOV SECT,R0 ;FIND DRIVE THAT WROTE
1221 026274 004537 025562 JSR R5,FNDDRV ;SECTOR LAST
1222 026300 016037 000000 002670 MOV CSR(R0),LSTCLR ;GET CSR OF DRIVE
1223 026306 116037 000005 002732 MOVSB DSB+1(R0),LSTDRV ;GET DRIVE
1224 026314 016037 000006 002734 MOV PAT(R0),GDATA ;GET PATTERN
1225
1226 026322 004537 030560 JSR R5,LDFUNC ;READ
1227 026326 000014 READ
1228
1229 026330 005737 002762 TST E.CS ;ERROR?
1230 026334 100060 BPL 5$ ;NO, NEXT SECTOR
1231 026336 012737 017414 002672 MOV #RECMS,REASON ;SET READ RECOVERY MESSAGE
1232 026344 ERRDF 14.,RECER,ERR4 ;REPORT ERROR
1233 (3) 026344 104462 TRAP T$ERCODE
1234 (5) 026346 000016 .WORD 14
1235 (5) 026350 017542 .WORD RECER
1236 (5) 026352 020206 .WORD ERR4
1237
1238 026354 005037 002740 CLR WCOUNT ;CLEAR BAD WORD COUNT
1239 026360 005037 002742 CLR SECWRD ;CLEAR WORD W/I SECTOR
1240 026364 012702 003030 MOV #BUF,R2 ;START OF BUFFER
1241 026370 023712 002734 3$: CMP GDATA,(R2) ;DATA COMPARE
1242 026374 001417 BEQ 4$ ;YES, CHECK NEXT
1243
1244 026376 005237 002740 INC WCOUNT ;ACCOUNT FOR ERROR
1245 026402 PRINTF #FRM8,SECWRD,GDATA,(R2) ;PRINT ERROR
1246 (10) 026402 011246 MOV (R2),-(SP)
1247 (9) 026404 013746 002734 MOV GDATA,-(SP)
1248 (8) 026410 013746 002742 MOV SECWRD,-(SP)
1249 (7) 026414 012746 021162 MOV #FRM8,-(SP)
1250 (6) 026420 012746 000004 MOV #4,-(SP)
1251 (3) 026424 010600 MOV SP,R0
1252 (4) 026426 104017 EMT C$PNTF
1253 (4) 026430 062706 000012 ADD #12,SP
1254
1255 026434 005722 4$: TST (R2)+ ;NEXT
    
```

1244	026436	005237	002742		INC	SECWRD	:NEXT WORD IN SECTOR
1245	026442	023727	002742	000200	CMP	SECWRD,#128.	:DONE?
1246	026450	001347			BNE	3\$:NO

```

1248
1249 026452          PRINTF #FRM9,WCOUNT ;PRINT SUMMARY
      (8) 026452 013746 002740  MOV      WCOUNT,-(SP)
      (7) 026456 012746 021226  MOV      #FRM9,-(SP)
      (6) 026462 012746 000002  MOV      #2,-(SP)
      (3) 026466 010600          MOV      SP,R0
      (4) 026470 104017          EMT      C$PNTF
      (4) 026472 062706 000006  ADD      #6,SP
1250
1251 026476 005237 002730      5$: INC      SECT ;NEXT SECTOR
1252 026502 023727 002730 000050  CMP      SECT,#40.
1253 026510 001002          BNE      7$
1254 026512 005037 002730      CLR      SECT
1255 026516 000241          7$: CLC
1256 026520 006001          ROR      R1 ;NEXT BIT MAP
1257 026522 103244          BCC      1$
1258
1259 026524 012602          6$: MOV      (SP)+,R2 ;RESTORE R2-R0, EXIT
1260 026526 012601          MOV      (SP)+,R1
1261 026530 012600          MOV      (SP)+,R0
1262 026532 000205          RTS      R5
1263
1264
1265 ;ROUTINE TO VERIFY THE ADJ. CYL. WRITE IS GOOD
1266 ;USES R3 AND WORD FOLLOWING CALL
1267 ;IF WRITE WAS GOOD,SECTOR WILL BE STORED IN MAP
1268 ;USING RSADJS/.WORD 1
1269
1270 026534 010046          VAJWR: MOV      R0,-(SP) ;SAVE REGISTERS
1271 026536 010146          MOV      R1,-(SP)
1272 026540 010246          MOV      R2,-(SP)
1273 026542 012701 100000      MOV      #100000,R1 ;BIT MASK FOR CYLINDER
1274 026546 012502          MOV      (R5)+,R2 ;STARTING SECTOR
1275 026550 105000          CLRB    R0
1276 026552 153700 002720      BLSB    DESCYL,R0
1277 026556 000300          SWAB   R0
1278 026560 006000          ROR    R0
1279 026562 032737 000400 002720      BIT     #400,DESCYL
1280 026570 001402          BEQ     3$
1281 026572 052700 000100      BIS     #HEAD,R0
1282 026576 050200          3$: BIS     R2,R0
1283 026600 030103          4$: BIT     R1,R3
1284 026602 001462          BEQ     5$
1285 026604 012737 177600 003004      MOV     #-128.,BMP
1286 026612 010037 003002      MOV     R0,BDA
1287 026616 010037 002666      MOV     R0,TEMP
1288 026622 042700 177700      BIC     #177700,R0
1289 026626 020027 000047      CMP     R0,#39.
1290 026632 003406          BLE     6$
1291 026634 162737 000050 003002      SUB     #40.,BDA
1292 026642 162737 000050 002666      SUB     #40.,TEMP
1293 026650 012737 003030 003000  6$: MOV     #BUF,BBA
1294 026656 005037 002710      CLR     HSFLG
1295 026662 013700 002666      MOV     TEMP,R0
1296 026666 004537 030560      JSR     R7,LDFUNC ;READ FUNCTION
1297 026672 000014
    
```



```

1298 026674 005737 002674      TST      ERFLG
1299 026700 001416      BEQ      7$
1300 026702 005737 002710      TST      HSFLG
1301 026706 001007      BNE     11$
1302 026710      ERRSOFT 120.,READ1,ERR2
(3) 026710 104464      TRAP    T$ERCODE
(5) 026712 000170      .WORD  120
(5) 026714 017702      .WORD  READ1
(5) 026716 020014      .WORD  ERR2
1303 026720 005237 002710      INC     HSFLG
1304 026724 000760      BR      10$
1305 026726      11$:    ERRHRD 130.,READ1,ERR2
(3) 026726 104463      TRAP    T$ERCODE
(5) 026730 000202      .WORD  130
(5) 026732 017702      .WORD  READ1
(5) 026734 020014      .WORD  ERR2
1306 026736 010046      7$:    MOV     R0,-(SP)
1307 026740 004537 025414      JSR     R5,RSADJS      ;STORE ADJ. CYL. SECTOR INFO.
1308 026744 000001      .WORD  1
1309 026746 012600      MOV     (SP)+,R0      ;RESTORE R0
1310 026750 005200      5$:    INC     R0
1311 026752 000241      CLC
1312 026754 006001      ROR     R1
1313 026756 103310      BCC    4$
1314 026760 012602      MOV     (SP)+,R2
1315 026762 012601      MOV     (SP)+,R1
1316 026764 012600      MOV     (SP)+,R0
1317 026766 000205      RTS     R5
1318
1319      ;ROUTINE TO VERIFY THAT WRITE DID NOT DISTURB ADJACENT TRACKS
1320      ;WRITTEN BY OTHER DRIVES.
1321      ;CALL JSR R5,BSVWR
1322      ;
1323      ;      .WORD      ;STARTING SECTOR
1324      ;
1325      ;USES "ADJLOC" TO GET +1/-1 CYLINDER OFFSET
1326      ;USES R3 FOR SECTOR MAP, USES MAP AT "SECBUF" FOR INFO
1327 026770 010046      BSVWR: MOV     R0,-(SP)      ;SAVE REGISTERS
1328 026772 010146      MOV     R1,-(SP)
1329 026774 010246      MOV     R2,-(SP)
1330 026776 013746 002720      MOV     DESCYL,-(SP) ;SAVE PRESENT POSITION
1331 027002 012546      MOV     (R5)+,-(SP) ;GET STARTING SECTOR
1332 027004 123727 002700 000003      CMPB   ADJLOC,#3      ;ON MIDDLE TRACK???
1333 027012 001455      BEQ     B$EXIT        ;YES, THEN NO CHECK
1334 027014 162716 000042      SUB     #34.,(SP)     ;SETUP SECTOR START FOR OUTSIDE
1335 027020 100002      BPL     1$           ;IF POSITIVE OKAY ELSE FIX
1336 027022 062716 000050      ADD     #40.,(SP)    ;FIX IT
1337 027026 123727 002700 000001 1$:    CMPB   ADJLOC,#1      ;ON OUTER LIMIT???
1338 027034 001412      BEQ     INAWR        ;YES, SKIP CHECK
1339 027036 105337 002700      DECB   ADJLOC        ;OUTER ADJ TRACK
1340 027042 105337 002720      DECB   DESCYL        ;CREATE CYLINDER
1341 027046 004537 027170      JSR     R5,CHECK     ;GO CHECK ADJ SECTORS
1342 027052 105237 002720      INCB   DESCYL        ;FIX BACK
1343 027056 105237 002700      INCB   ADJLOC
1344 027062 062716 000104      INAWR: ADD     #8.,(SP)    ;INNER SECTOR START
1345 027066 021627 000050      CMP     (SP),#40.    ;WITHIN LIMITS???

```

```

1346 027072 002407          BLT      1$          ;YES, OKAY
1347 027074 162716 000050  SUB      #40.,(SP)   ;FIX SECTOR
1348 027100 021627 000050  CMP      (SP),#40.
1349 027104 002402          BLT      1$
1350 027106 162716 000050  SUB      #40.,(SP)
1351 027112 123727 002700 000005 1$:  CMPB    ADJLOC,#5   ;INNER LIMIT??
1352 027120 001412          BEQ      BSEXIT     ;YES,SKIP CHECK
1353 027122 105237 002700          INCB    ADJLOC     ;FIX FOR INNER
1354 027126 105237 002720          INCB    DESCYL
1355 027132 004537 027170          JSR     R5,CHECK   ;GO CHECK ADJ SECTORS
1356 027136 105337 002700          DECB    ADJLOC     ;FIX BACK
1357 027142 105337 002720          DECB    DESCYL
1358 027146 005726          BSEXIT: TST      (SP)+ ;THROW OFF SECTOR
1359 027150 012637 002720          MOV     (SP)+,DESCYL ;GET OLD CYLINDER
1360 027154 012602          MOV     (SP)+,R2
1361 027156 012601          MOV     (SP)+,R1
1362 027160 012600          MOV     (SP)+,R0
1363 027162 004537 024716          JSR     R5,SKCYL   ;SEEK BACK
1364 027166 000205          RTS      R5        ;RETURN
1365
1366
1367          ;ROUTINE TO VERIFY AN ADJACENT SECTOR
1368          ;CALLED FROM BSVWR
1369          ;
1370
1371 027170 012701 100000  CHECK:  MOV     #100000,R1 ;SECTOR MASK
1372 027174 004537 024716          JSR     R5,SKCYL   ;GET TO DESIRED CYLINDER
1373 027200 005002          CLR     R2         ;CREATE ADDRESS
1374 027202 153702 002720          BISB   DESCYL,R2
1375 027206 000302          SWAB   R2
1376 027210 006002          ROR     R2
1377 027212 032737 000400 002720  BIT     #400,DESCYL ;HEAD SET???
1378 027220 001402          BEQ     3$         ;NO
1379 027222 052702 000100          BIS     #HEAD,R2
1380 027226 056602 000002  3$:  BIS     2(SP),R2   ;SET IN SECTOR
1381 027232 030103  4$:  BIT     R1,R3     ;THIS SECTOR IN LIST???
1382 027234 001452          BEQ     5$         ;NO, NEXT
1383 027236 010200          MOV     R2,R0     ;COPY SECTOR
1384 027240 042700 177700          BIC     #177700,R0 ;ONLY SECTOR LEFT
1385 027244 020027 000050          CMP     R0,#40. ;SECTOR OKAY???
1386 027250 002404          BLT     6$         ;YES
1387 027252 162700 000050          SUB     #40.,R0
1388 027256 162702 000050          SUB     #40.,R2   ;FIX SECTOR
1389 027262 004537 025414  6$:  JSR     R5,RSADJS  ;FIND IF SECTOR PREVIOUSLY WRITTEN
1390 027266 000000          .WORD  0
1391 027270 005700          TST     R0         ;WAS IT??
1392 027272 001433          BEQ     5$         ;NO
1393 027274 010237 003002          MOV     R2,BDA    ;LOAD DISK ADDRESS
1394 027300 012737 177600 003004  MOV     #-128.,BMP ;LOAD WC
1395 027306 004537 030560          JSR     R5,LDFUNC ;LOAD
1396 027312 000014          READ
1397 027314 005737 002674          TST     ERFLG     ;WAS READ GOOD
1398 027320 001420          BFQ     5$
1399
1400 027322 010346          MOV     R3,-(SP)
1401 027324 010237 002730          MOV     R2,SECT
    
```

```

1402 027330 010003          MOV     R0,R3
1403
1404 027332 042737 177700 002730      BIC     #177700,SECT
1405 027340          ERRHRD  140.,ADJTXT,ERR3
      (3) 027340 104463          TRAP   T$ERCODE
      (5) 027342 000214          .WORD  140
      (5) 027344 017727          .WORD  ADJTXT
      (5) 027346 020054          .WORD  ERR3
1406 027350 012603          MOV     (SP)+,R3
1407 027352          ERRHRD  110.,READ1,ERR2
      (3) 027352 104463          TRAP   T$ERCODE
      (5) 027354 000156          .WORD  110
      (5) 027356 017702          .WORD  READ1
      (5) 027360 020014          .WORD  ERR2
1408
1409 027362 005202          5$:    INC     R2          ;NEXT SECTOR
1410 027364 000241          CLC
1411 027366 006001          ROR     R1          ;SHIFT MASK
1412 027370 103320          BCC    4$
1413 027372 000205          RTS     R5
1414
1415
1416          ;ROUTINE TO MERGE BAD SECTOR FILES
1417          ;ENTRY INTO THIS ROUTINE WILL OCCUR AFTER THE 'SERNUM' ROUTINE
1418          ;IS PERFORMED. THE FACTORY BAD SECTOR FILE WILL BE LOCATED IN
1419          ;FIRST 400(8) LOCATIONS.
1420          ;THIS ROUTINE WILL STORE THE FIELD BAD SECTORS INTO THE NEXT
1421          ;400 LOCATIONS AND THEN MERGE THE FACTORY BAD FILE
1422          ;WITH THE FIELD BAD FILE.
1423
1424          ;FACTORY BAD AT BUF
1425          ;FIELD BAD AT BUF + 512.
1426
1427 027374 010146          MERGE: MOV     R1,-(SP)          ;SAVE R1, R2, R3
1428 027376 010246          MOV     R2,-(SP)
1429 027400 010346          MOV     R3,-(SP)
1430 027402 012737 003430 003000          MOV     #BUF+400,BBA          ;BUFFER START FOR FIELD BAD
1431 027410 012737 077724 003002          MOV     #77724,BDA          ;DA OF FIELD BAD SEC. START
1432 027416 012737 177400 003004          MOV     #-256.,BMP          ;SETUP TO READ TWO SECTORS
1433 027424 004537 030560          97$:   JSR     R5,LDFUNC          ;LOAD READ FUNCTION
1434 027430 000014          READ
1435 027432 005737 002674          TST     ERFLG          ;TEST ERROR FLAG
1436 027436 001421          BEQ    98$          ;YES;MERGE BAD SECTOR FILES
1437 027440 062737 000004 003002          ADD     #4,BDA          ;TRY NEXT FIELD BAD SECTOR FILE
1438 027446 022737 077750 003002          CMP     #77750,BDA          ;COMPLETED FIELD BAD SECTORS?
1439 027454 001363          BNE    97$          ;NG,DO NEXT FIELD BAD SECTOR
1440 027456          PRINTF #FRM15
      (7) 027456 012746 021562          MOV     #FRM15,-(SP)
      (6) 027462 012746 000001          MOV     #1,-(SP)
      (3) 027466 010600          MOV     SP,R0
      (4) 027470 104017          EMT    C$PNTF
      (4) 027472 062706 000004          ADD     #4,SP
1441 027476          999$: BREAK
      (3) 027476 104022          EMT    C$BRK
1442 027500 000776          BR     999$
1443 027502 012701 003040          98$:   MOV     #BUF+10,R1          ;GET PAST ID ETC.
    
```

```

1444 027506 012702 000176          MOV    #126.,R2          ;MAX = 126
1445 027512 005721          1$:   TST    (R1)+          ;SECTOR OR END
1446 027514 100404          BMI    2$              ;END, GO GET FIELD
1447 027516 005721          TST    (R1)+          ;REST OF SECTOR
1448 027520 005302          DEC    R2              ;MAX REACHED
1449 027522 001373          BNE    1$              ;NO, KEEP GOING
1450 027524 000401          BR     3$              ;YES, SKIP BACK UP
1451 027526 005741          2$:   TST    -(R1)        ;BACK UP PAST TERMINATOR
1452 027530 012703 000176          3$:   MOV    #126.,R3    ;SET 126 MAX
1453 027534 012702 003440          MOV    #BUF+410,R2    ;GET FIELD SECTORS
1454 027540 012221          4$:   MOV    (R2)+,(R1)+ ;MERGE AT END OF FACTORY
1455 027542 100403          BMI    5$              ;DONE?
1456 027544 012221          MOV    (R2)+,(R1)+    ;NO, MERGE REST OF SECTOR
1457 027546 005303          DEC    R3              ;DONE
1458 027550 001373          BNE    4$              ;NO, GO BACK
1459 027552 012603          5$:   MOV    (SP)+,R3    ;RESTORE R3, R2, R1
1460 027554 012602          MOV    (SP)+,R2
1461 027556 012601          MOV    (SP)+,R1
1462 027560 000205          RTS    R5              ;EXIT
1463
1464
1465 027562 012537 002744          FNDTRK: MOV (R5)+,OFFSET ;GET INCREMENT/DECREMENT
1466 027566 012537 002754          MOV    (R5)+,SURFACE ;GET HEAD (SURFACE)
    
```

```

1468 027572 112537 002750      MOVB      (R5)+,FRTRK      ;BEGINNING TRACK
1469 027576 112537 002746      MOVB      (R5)+,LSTTRK     ;ENDING TRACK
1470 027602 005037 002756      CLR       TRKFND          ;CLEAR OUT FLAG FOUND
1471 027606 005037 002760      CLR       TRKCNT          ;CLEAR OUT TRACK COUNT
1472 027612 013737 002750 002752  MOV       FRTRK,PRSTRK     ;GET FIRST TRACK
1473 027620                                     1$:
1474 027620 004537 027704      JSR       R5,FNDBSC        ;IS TRACK IN BAD SECTOR FILE
1475 027624 005737 002122      TST      HDRFND          ;WAS IT?
1476 027630 001003                                     BNE      2$              ;YES, CLEAR TRKCNT
1477 027632 005237 002760      INC      TRKCNT          ;NO, INDICATE GOOD TRACK
1478 027636 000402                                     BR       3$              ;CONTINUE
1479 027640 005037 002760                                     CLR      TRKCNT          ;START COUNT OVER
1480 027644 023727 002760 000005 3$:      CMP      TRKCNT,#5        ;FIND 5 TRACKS YET?
1481 027652 001003                                     BNE      4$              ;NO, CONTINUE
1482 027654 005237 002756      INC      TRKFND          ;YES, EXIT WITH GOOD FLAG
1483 027660 000205                                     RTS      R5              ;EXIT
1484 027662 023737 002752 002746 4$:      CMP      PRSTRK,LSTTRK    ;ARE WE DONE?
1485 027670 001001                                     BNE      5$              ;NO, KEEP LOOKING
1486 027672 000205                                     RTS      R5              ;EXIT WITH NOT FOUND
1487 027674 063737 002744 002752 5$:      ADD      OFFSET,PRSTRK    ;NEXT TRACK
1488 027702 000746      BR       1$
1489
1490                                     ;ROUTINE TO FIND BAD TRACK IN FILE
1491
1492 027704 005037 002122      FNDBSC: CLR      HDRFND          ;INITIALIZE FLAG
1493 027710 010146      MOV      R1,-(SP)         ;SAVE R1, R2
1494 027712 010246      MOV      R2,-(SP)
1495 027714 012701 003040      MOV      #BUF+10,R1       ;SETUP FOR BEGINNING OF FILE
1496 027720 005711      1$:      TST      (R1)             ;END?
1497 027722 100421      BMI      2$              ;IF MINUS AT END, EXIT
1498 027724 023721 002752      CMP      PRSTRK,(R1)+     ;CYLINDER CORRECT?
1499 027730 001011      BNE      3$              ;NO, NEXT
1500 027732 105724      TSTB    (R4)+            ;UPPER HALF OF WORD
1501 027734 123711 002754      CMPB    SURFACE,(R1)     ;CORRECT SURFACT
1502 027740 001402      BEQ     4$
1503 027742 105744      TSTB    -(R4)
1504 027744 000403      BR      3$
1505 027746 005237 002122      4$:      INC      HDRFND          ;SET FOUND
1506 027752 000405      BR      2$
1507
1508 027754 005721      3$:      TST      (R1)+            ;NEXT WORD
1509 027756 005202      INC     R2               ;ACCOUNT FOR IT
1510 027760 020227 000374      CMP     R2,#252.         ;DONE?
1511 027764 001355      BNE     1$              ;NO, KEEP CHECKING
1512 027766 012601      2$:      MOV     (SP)+,R1         ;RESTORE R2, R1, EXIT
1513 027770 012602      MOV     (SP)+,R2
1514 027772 000205      RTS     R5
1515
1516 027774 013701 002752      FIXCYL: MOV     PRSTRK,R1   ;GET TRACK WHICH IS GOOD
1517 030000 005737 002744      TST     OFFSET          ;WHICH WAY WERE WE LOOKING
1518 030004 100402      BMI     1$              ;IN WORD, BRANCH
1519 030006 162701 000004      SUB     #4,R1           ;BACK IT UP BY FOUR
1520 030012 012702 000005      1$:      MOV     #5,R2           ;GOING STORE AWAY 5 TRACKS
1521 030016 110120      2$:      MOVB   R1,(R0)+         ;STORE THEM
1522 030020 005201      INC     R1
1523 030022 005302      DEC     R2

```

```

1524 030024 001374          BNE 2$
1525 030026 000205          RTS  R5
1526
1527
1528          ;ROUTINE TO GET SERIAL NUMBER
1529
1530          ;CALL  JSR  R5,SERNUM
1531
1532 030030 012737 000013 003002  SERNUM: MOV  #13,BDA
1533 030036 004537 030560          JSR  R5,LDFUNC          ;GET STATUS
1534 030042 000004          GSTAT
1535 030044 004537 030560          JSR  R5,LDFUNC          ;READ HEADER
1536 030050 000010          RDHDR
1537 030052 013700 002770          MOV  E,MP,R0          ;GET THE HEADER
1538 030056 042700 000077 1$:  BIC  #77,R0          ;CLEAR SECTOR BITS
1539 030062 020027 077700          CMP  R0,#77700        ;ON LAST TRACK
1540 030066 001425          BEQ  2$              ;IF SO, DON'T SEEK
1541 030070 042700 000100          BIC  #100,R0         ;CLEAR HEAD
1542 030074 012701 077600          MOV  #77600,R1       ;LAST CYLINDER
1543 030100 160001          SUB  R0,R1           ;CALCULATE DIF OF SEEK
1544 030102 010137 003002          MOV  R1,BDA          ;SET UP DIF WORD
1545 030106 052737 000025 003002  BIS  #25,BDA          ;SEEK IN, HEAD 1
1546 030114 004537 030560          JSR  R5,LDFUNC          ;SEEK
1547 030120 000006          SEEK
1548 030122 004537 030560          JSR  R5,LDFUNC          ;VERIFY POSITION
1549 030126 000010          RDHDR
1550 030130 013700 002770          MOV  E,MP,R0          ;GET HEADER
1551 030134 022700 077700          CMP  #77700,R0       ;COMPARE AGAINST LAST
1552 030140 003346          BGT  1$              ;IF WRONG, RE-SEEK
1553 030142 012737 077700 003002 2$:  MOV  #77700,BDA       ;BAD SECTOR DA START
1554 030150 012737 003030 003000 97$:  MOV  #BUF,BBA         ;READ IN BAD SECTOR FILE
1555 030156 012737 177400 003004          MOV  #-256.,BMP
1556 030164 004537 030560          JSR  R5,LDFUNC          ;READ
1557 030170 000014          READ
1558 030172 005737 002674          TST  ERFLG           ;TEST ERROR FLAG
1559 030176 001410          BEQ  98$             ;YES,COMPARE SERIAL NUMBERS
1560 030200 062737 000004 003002  ADD  #4,BDA           ;NO,SETUP FOR NEXT FACTORY BAD SECTOR
1561 030206 022737 077724 003002  CMP  #77724,BDA       ;END OF FACTORY BAD FILE?
1562 030214 001355          BNE  97$             ;GET NEXT FACTORY BAD SECTOR
1563 030216 000445          BR   99$             ;REPORT ERROR
1564 030220 012701 003030 98$:  MOV  #BUF,R1          ;COMPARE SERIAL NUMBERS
1565 030224 005737 003006          TST  SERNM1           ;HAVE WE GOT ONE TO COMPARE
1566 030230 100005          BPL  3$              ;YES, BRANCH
1567 030232 011137 003006          MOV  (R1),SERNM1      ;NO, CALL THIS ONE IT
1568 030236 016137 000002 003010  MOV  2(R1),SERNM2
1569 030244 021137 003006 3$:  CMP  (R1),SERNM1
1570 030250 001004          BNE  4$              ;SERNUM OKAY
1571 030252 026137 000002 003010  CMP  2(R1),SERNM2     ;NO, PRINT ERROR
1572 030260 001436          BEQ  5$              ;OTHER HALF OKAY
1573 030262 4$:  PRINTF #FRM3,2(R1),(R1),SERNM2,SERNM1 ;YES, EXIT
(11) 030262 013746 003006          MOV  SERNM1,-(SP)
(10) 030266 013746 003010          MOV  SERNM2,-(SP)
(9)  030272 011146          MOV  (R1),-(SP)
(8)  030274 016146 000002          MOV  2(R1),-(SP)
(7)  030300 012746 020712          MOV  #i?M3,-(SP)
(6)  030304 012746 000005          MOV  #5,-(SP)
    
```

(3)	030310	010600		MOV	SP,R0	
(4)	030312	104017		EMT	C\$PNTF	
(4)	030314	062706	000014	ADD	#14,SP	
1574	030320	004537	030360	JSR	R5,UNLOAD	:LET OPERATOR CHANGE
1575	030324	004537	030460	JSR	R5,LOAD	:PACK
1576	030330	000637		BR	SERNUM	:GO CHECK IT AGAIN.
1577	030332			99\$: PRINTF	#FRM15	:MESSAGE
(7)	030332	012746	021562	MOV	#FRM15,-(SP)	
(6)	030336	012746	000001	MOV	#1,-(SP)	
(3)	030342	010600		MOV	SP,R0	
(4)	030344	104017		EMT	C\$PNTF	
(4)	030346	062706	000004	ADD	#4,SP	
1578	030352			999\$: BREAK		
(3)	030352	104022		EMT	C\$BRK	
1579	030354	000776		BR	999\$	
1580	030356	000205		5\$: RTS	R5	
1581						
1582						

```

1584      ;ROUTINE UNLOAD
1585
1586      ;CALL   JSR   R5,UNLOAD
1587
1588      UNLOAD: PRINTF #FRM1,<B,DSB+1(R4)>,CSR(R4)
(9) 030360 016446 000000      MOV   CSR(R4),-(SP)
(8) 030364 005046      CLR   -(SP)
(8) 030366 156416 000005      BISB  DSB+1(R4),(SP)
(7) 030372 012746 020520      MOV   #FRM1,-(SP)
(6) 030376 012746 000003      MOV   #3,-(SP)
(3) 030402 010600      MOV   SP,R0
(4) 030404 104017      EMT   C$PNTF
(4) 030406 062706 000010      ADD   #10,SP
1589 030412 012701 000074      MOV   #60,R1      ;SETUP 60 SECOND TIMER
1590 030416 012700 000200      MOV   #200,R0
1591 030422 056400 000004      BIS   DSB(R4),R0
1592 030426 010074 000000      MOV   R0,@CSR(R4)
1593 030432 032774 000001 000000 2$: BIT   #DRDY,@CSR(R4) ;CHECK DRDY FOR ZERO
1594 030440 001406      BEQ   3$          ;PACK UNLOADED
1595 030442      WAITMS #10.      ;WAIT 1 SECOND
(3) 030442 012700 000012      MOV   #10,R0
(3) 030446 104026      EMT   C$WTM
1596 030450 005301      DEC   R1
1597 030452 001367      BNE   2$          ;HAS 60 SEC PASSED?
1598 030454 000741      BR    UNLOAD     ;NO, RETEST DRDY, CONTINUE WAIT
1599 030456 000205      3$:  BR    UNLOAD     ;YES, REPEAT MESSAGE CONTINUE WAIT
1600      RTS   R5          ;RETURN WITH PACK UNLOADED
1601      ;ROUTINE LOAD
1602
1603      ;CALL   JSR   R5,LOAD
1604
1605      LOAD:  PRINTF #FRM2,<B,DSB+1(R4)>,CSR(R4)
(9) 030460 016446 000000      MOV   CSR(R4),-(SP)
(8) 030464 005046      CLR   -(SP)
(8) 030466 156416 000005      BISB  DSB+1(R4),(SP)
(7) 030472 012746 020615      MOV   #FRM2,-(SP)
(6) 030476 012746 000003      MOV   #3,-(SP)
(3) 030502 010600      MOV   SP,R0
(4) 030504 104017      EMT   C$PNTF
(4) 030506 062706 000010      ADD   #10,SP
1606 030512 012701 000170      MOV   #120,R1     ;SETUP 120 SEC TIMER
1607 030516 012700 000200      MOV   #200,R0     ;SETUP CONTROLLER READY BIT
1608 030522 056400 000004      BIS   DSB(R4),R0 ;SELECT DRIVE
1609 030526 010074 000000      MOV   R0,@CSR(R4)
1610 030532 032774 000001 000000 2$: BIT   #DRDY,@CSR(R4)
1611 030540 001006      BNE   3$
1612 030542      WAITMS #10.
(3) 030542 012700 000012      MOV   #10,R0
(3) 030546 104026      EMT   C$WTM
1613 030550 005301      DEC   R1
1614 030552 001367      BNE   2$
1615 030554 000741      BR    LOAD

```



```

1617 030556 000205 3$: RTS R5
1618
1619 ;ROUTINE LDFUNC
1620 ;CALL JSR R5,LDFUNC
1621
1622 030560 010046 LDFUNC: MOV R0,-(SP)
1623 030562 010346 MOV R3,-(SP)
1624 030564 010146 MOV R1,-(SP)
1625 030566 005037 002674 CLR ERFLG ;CLEAR ERROR FLAG
1626 030572 016403 000000 MOV CSR(R4),R3 ;GET CSR
1627 030576 013763 003004 000006 MOV BMP,MP(R3) ;LOAD MULTIPURPOSE
1628 030604 013763 003002 000004 MOV BDA,DA(R3) ;LOAD DISK ADDRESS
1629 030612 013763 003000 000002 MOV BBA,BA(R3) ;LOAD BUS ADDRESS
1630 030620 011537 002776 MOV (R5),BCS ;GET FUNCTION TO LOAD
1631 030624 056437 000004 002776 BIS DSB(R4),BCS ;SELECT BITS
1632 030632 012701 000031 MOV #25.,R1 ;SET WATCHDOG TO 250MS
1633 030636 052737 000200 002776 BIS #200,BCS
1634 030644 013763 002776 000000 MOV BCS,CS(R3) ;LOAD FUNCTION
1635 030652 016337 000000 002776 MOV CS(R3),BCS
1636 030660 042763 000200 000000 BIC #200,CS(R3)
1637 030666 032763 000200 000000 1$: BIT #200,CS(R3) ;CNTLR READY?
1638 030674 001034 BNE 2$ ;YES, GO
1639 030676 WAITUS #100. ;WAIT 10 MILLISECONDS
(3) 030676 012700 000144 MOV #100.,R0
(3) 030702 104027 EMT C$WTU
1640 030704 005301 DEC R1
1641 030706 001367 BNE 1$
1642
1643 030710 016337 000000 002762 MOV CS(R3),E.CS ;READ ALL REGISTERS
1644 030716 016337 000002 002764 MOV BA(R3),E.BA
1645 030724 016337 000004 002766 MOV DA(R3),E.DA
1646 030732 016337 000006 002770 MOV MP(R3),E.MP
1647 030740 016337 000006 002772 MOV MP(R3),E.MP1
1648 030746 016337 000006 002774 MOV MP(R3),E.MP2
1649 030754 ERRDF 210.,CNTTOT,ERR5;CNTLR TIMEOUT
(3) 030754 104462 TRAP T$ERCODE
(5) 030756 000322 .WORD 210
(5) 030760 017072 .WORD CNTTOT
(5) 030762 020372 .WORD ERR5
1650 030764 000425 BR 4$
1651
1652 030766 016337 000000 002762 2$: MOV CS(R3),E.CS ;READ ALL REGISTERS
1653 030774 016337 000002 002764 MOV BA(R3),E.BA
1654 031002 016337 000004 002766 MOV DA(R3),E.DA
1655 031010 016337 000006 002770 MOV MP(R3),E.MP
1656 031016 016337 000006 002772 MOV MP(R3),E.MP1
1657 031024 016337 000006 002774 MOV MP(R3),E.MP2
1658
1659 031032 005737 002762 TST E.CS ;ANY ERRORS?
1660 031036 100002 BPL 3$ ;YES, GO SERVICE
1661 031040 005237 002674 4$: INC ERFLG
1662 031044 005725 3$: TST (R5)+
1663 031046 012601 MOV (SP)+,R1
1664 031050 012603 MOV (SP)+,R3
1665 031052 012600 MOV (SP)+,R0
1666 031054 000205 RTS R5
    
```

```

1667
1668
1669
1670 031056          ENDMOD
1671
1672
1673
1674
1675
1676 031056          BGNTST
1677
1678
1679                ;CONTROL SECTION COMPATABILITY PROGRAM
1680                ;PRINT UNLOAD AND LOAD DRIVE MESSAGES
1681                ;PERFORM SERIAL CHECK ROUTINE
1682                ;PERFORM READ/WRITE CHECKS ON DRIVES
1683
1684 031056 012701 002246  COMPAT: MOV      #SECBUF,R1      ;ADJ. CYLINDER BUFFER
1685 031062 012700 000170      MOV      #120.,R0      ;ADJ. CYLINDER BUFFER COUNT
1686 031066 005021      4$: CLR      (R1)+      ;CLEAR ADJ. CYL. BUFFER AT STARTUP
1687 031070 005300      DEC      R0          ;BUFFER CLEARED?
1688 031072 001375      BNE     4$          ;CLEAR NEXT BUFFER WORD
1689 031074 005237 002662      INC     F0WR        ;SET FIRST OVERWRITE FLAG
1690 031100 004537 023334      JSR    R5,OVWPER    ;PERFORM OVERWRITE ON FIRST DRIVE
1691 031104 177400
1692 031106 000377      177400
1693 031110 005037 002662      377
1694 031114 005237 002664      CLR     F0WR        ;CLEAR FIRST OVERWRITE
1695 031120 005237 002704      INC     FADJ        ;SET FIRST ADJ. FLAG
1696 031124 004537 024016      INC     ADJDIR      ;UP = 1
1697 031130      003      377
1698 031132 170000      JSR    R5,ADJCYL    ;TRACK AND SECTORS FOR
1699 031134      003      000      .BYTE 3,377      ;INWARD SEEK
1700 031136 007777      .WORD 170000      ;TRACK AND SECTORS FOR
1701 031140 000000      .BYTE 3,0         ;OUTWARD SEEK
1702 031142 004537 030360      .WORD 7777       ;TERMINATOR
1703 031146 062704 000010      JSR    R5,UNLOAD    ;UNLOAD PACK FROM DRIVE UNIT
1704 031152 004537 030460      ADD     #PAT+2,R4   ;UPDATE POINTER FOR NEXT DRIVE
1705 031156 004537 030030      JSR    R5,LOAD      ;LOAD INTO SECOND DRIVE UNIT
1706 031162 004537 023334      JSR    R5,SERNUM    ;CHECK PACK SERIAL NUMBER
1707 031166 000360      JSR    R5,OVWPER    ;PERFORM R/W OVERWRITE
1708 031170 000017      360
1709 031172 005237 002704      17
1710 031176 004537 024016      INC     ADJDIR
1711 031202      002      360      JSR    R5,ADJCYL
1712 031204 000000      .BYTE 2,360      ;IN 1/0 OUTSIDE
1713 031206      002      017      .WORD 0
1714 031210 000000      .BYTE 2,17       ;OUT 1/0 OUTSIDE
1715 031212      004      360      .WORD 0
1716 031214 000000      .BYTE 4,360      ;IN 1/0 INSIDE
1717 031216      004      017      .WORD 0
1718 031220 000000      .BYTE 4,17       ;OUT 1/0 INSIDE
1719 031222 000000      .WORD 0
1720 031224 004537 030360      .WORD 0
1721 031230 023727 002726 000002      JSR    R5,UNLOAD    ;UNLOAD PACK FROM DRIVE UNIT
1722 031236 001002      CMP     UC*,#2     ;CHECK FOR > 2 DRIVES
                                BNE     10$          ;YES,GO TO NEXT DRIVE

```

1723	031240	000137	031654		JMP	2\$:GO TO FIRST DRIVE
1724	031244	062704	000010	10\$:	ADD	#PAT+2,R4		:UPDATE DRIVE BUFFER FOR THIRD DRIVE
1725	031250	004537	030460		JSR	R5,LOAD		:LOAD PACK FOR THIRD DRIVE
1726	031254	004537	030030		JSR	R5,SERNUM		:CHECK SERIAL NUMBERS
1727	031260	004537	023334		JSR	R5,OVWPER		:PERFORM R/W OVERWRITE ON THIRD DRIVE
1728	031264	006014				6014		
1729	031266	001403				1403		
1730	031270	005237	002704		INC	ADJDIR		
1731	031274	004537	024016		JSR	R5,ADJCYL		
1732	031300	002	000		.BYTE	2,0		:IN 2/0 OUTSIDE
1733	031302	170000			.WORD	170000		
1734	031304	002	000		.BYTE	2,0		:OUT 2/0 OUTSIDE
1735	031306	007400			.WORD	7400		
1736	031310	004	000		.BYTE	4,0		:IN 2/0 INSIDE
1737	031312	170000			.WORD	170000		
1738	031314	004	000		.BYTE	4,0		:OUT 2/0 INSIDE
1739	031316	007400			.WORD	7400		
1740	031320	001	200		.BYTE	1,200		:IN 2/1 OUTSIDE
1741	031322	000000			.WORD	0		
1742	031324	001	100		.BYTE	1,100		:OUT 2/1 OUTSIDE
1743	031326	000000			.WORD	0		
1744	031330	005	200		.BYTE	5,200		:IN 2/1 INSIDE
1745	031332	000000			.WORD	0		
1746	031334	005	100		.BYTE	5,100		:OUT 2/1 INSIDE
1747	031336	000000			.WORD	0		
1748	031340	000000			.WORD	0		:TERMINATOR
1749	031342	004537	030360		JSR	R5,UNLOAD		:UNLOAD PACK ON THIRD DRIVE
1750	031346	023727	002726	000003	CMP	UUT,#3		:CHECK FOR > 3 DRIVES
1751	031354	001500			BEQ	1\$:NO, GO TO 2ND DRIVE
1752	031356	062704	000010		ADD	#PAT+2,R4		:UPDATE DRIVE BUFFER FOR 4TH DRIVE
1753	031362	004537	030460		JSR	R5,LOAD		:LOAD PACK ON 4TH DRIVE
1754	031366	004537	030030		JSR	R5,SERNUM		:CHECK PACK ON FOURTH DRIVE
1755	031372	004537	023334		JSR	R5,OVWPER		:PERFORM R/W OVERWRITE
1756	031376	021040				21040		
1757	031400	010420				10420		
1758	031402	005237	002704		INC	ADJDIR		
1759	031406	004537	024016		JSR	R5,ADJCYL		
1760	031412	002	000		.BYTE	2,0		:IN 3/0 OUTSIDE
1761	031414	000360			.WORD	360		
1762	031416	002	000		.BYTE	2,0		:OUT 3/0 OUTSIDE
1763	031420	000017			.WORD	17		
1764	031422	004	000		.BYTE	4,0		:IN 3/0 INSIDE
1765	031424	000360			.WORD	360		
1766	031426	004	000		.BYTE	4,0		:OUT 3/0 INSIDE
1767	031430	000017			.WORD	17		
1768	031432	001	040		.BYTE	1,40		:IN 3/1 OUTSIDE
1769	031434	000000			.WORD	0		
1770	031436	001	020		.BYTE	1,20		:OUT 3/1 OUTSIDE
1771	031440	000000			.WORD	0		
1772	031442	005	040		.BYTE	5,40		:IN 3/1 INSIDE
1773	031444	000000			.WORD	0		
1774	031446	005	020		.BYTE	5,20		:OUT 3/1 INSIDE
1775	031450	000000			.WORD	0		
1776	031452	001	000		.BYTE	1,0		:IN 3/2 OUTSIDE
1777	031454	100000			.WORD	100000		
1778	031456	001	000		.BYTE	1,0		:OUT 3/2 OUTSIDE

1779	031460	040000		.WORD	40000	
1780	031462	005	000	.BYTE	5,0	:IN 3/2 INSIDE
1781	031464	100000		.WORD	100000	
1782	031466	005	000	.BYTE	5,0	:OUT 3/2 INSIDE
1783	031470	040000		.WORD	40000	
1784	031472	000000		.WORD	0	:TERMINATOR
1785	031474	004537	030360	JSR	R5,UNLOAD	:UNLOAD PACK FROM 4TH DRIVE
1786	031500	162704	000010	SUB	#PAT+2,R4	:SET DRIVE BUFFER FOR 3RD DRIVE
1787	031504	004537	030460	JSR	R5,LOAD	:LOAD PACK ON 3RD DRIVE
1788	031510	004537	030030	JSR	R5,SERNUM	:CHECK FOR PACK SERIAL NUMBER
1789	031514	004537	023334	JSR	R5,OVWPER	:PERFORM R/W OVERWRITE ON 3RD DRIVE
1790	031520	020000			20000	
1791	031522	010000			10000	
1792	031524	004537	024016	JSR	R5,ADJCYL	
1793	031530	001	000	.BYTE	1,0	:IN 2/3 OUTSIDE
1794	031532	000200		.WORD	200	
1795	031534	001	000	.BYTE	1,0	:OUT 2/3 OUTSIDE
1796	031536	000100		.WORD	100	
1797	031540	005	000	.BYTE	5,0	:IN 2/3 INSIDE
1798	031542	000200		.WORD	200	
1799	031544	005	000	.BYTE	5,0	:OUT 2/3 INSIDE
1800	031546	000100		.WORD	100	
1801	031550	000000		.WORD	0	:TERMINATOR
1802	031552	004537	030360	JSR	R5,UNLOAD	:UNLOAD PACK FROM 3RD DRIVE
1803	031556	162704	000010	SUB	#PAT+2,R4	:SET DRIVE BUFFER FOR 2ND DRIVE
1804	031562	004537	030460	JSR	R5,LOAD	:LOAD PACK ON THIRD DRIVE
1805	031566	004537	030030	JSR	R5,SERNUM	:CHECK PACK SERIAL NUMBER
1806	031572	004537	023334	JSR	R5,OVWPER	:PERFORM R/W OVERWRITE ON 2ND DRIVE
1807	031576	004040			4040	
1808	031600	002020			2020	
1809	031602	004537	024016	JSR	R5,ADJCYL	
1810	031606	001	000	.BYTE	1,0	:IN 1/2 OUTSIDE
1811	031610	020000		.WORD	20000	
1812	031612	001	000	.BYTE	1,0	:OUT 1/2 OUTSIDE
1813	031614	010000		.WORD	10000	
1814	031616	005	000	.BYTE	5,0	:IN 1/2 INSIDE
1815	031620	020000		.WORD	20000	
1816	031622	005	000	.BYTE	5,0	:OUT 1/2 INSIDE
1817	031624	010000		.WORD	10000	
1818	031626	001	000	.BYTE	1,0	:IN 1/3 OUTSIDE
1819	031630	000040		.WORD	40	
1820	031632	001	000	.BYTE	1,0	:OUT 1/3 OUTSIDE
1821	031634	000020		.WORD	20	
1822	031636	005	000	.BYTE	5,0	:IN 1/3 INSIDE
1823	031640	000040		.WORD	40	
1824	031642	005	000	.BYTE	5,0	:OUT 1/3 INSIDE
1825	031644	000020		.WORD	20	
1826	031646	000000		.WORD	0	:TERMINATOR
1827	031650	004537	030360	JSR	R5,UNLOAD	:UNLOAD PACK FROM 2ND DRIVE
1828	031654	162704	000010	SUB	#PAT+2,R4	:SET DRIVE BUFFER FOR 1ST DRIVE
1829	031660	004537	030460	JSR	R5,LOAD	:LOAD PACK INTO FIRST DRIVE UNIT
1830	031664	004537	030030	JSR	R5,SERNUM	:CHECK SERIAL NUMBER
1831	031670	004537	023334	JSR	R5,OVWPER	:PERFORM R/W OVERWRITE
1832	031674	001042			1042	
1833	031676	000421			421	
1834	031700	004537	024016	JSR	R5,ADJCYL	

1\$:

2\$:

1835	031704	001	010	.BYTE	1,10	;IN 0/1 OUTSIDE
1836	031706	000000		.WORD	0	
1837	031710	001	004	.BYTE	1,4	;OUT 0/1 OUTSIDE
1838	031712	000000		.WORD	0	
1839	031714	005	010	.BYTE	5,10	;IN 0/1 INSIDE
1840	031716	000000		.WORD	0	
1841	031720	005	004	.BYTE	5,4	;OUT 0/1 INSIDE
1842	031722	000000		.WORD	0	
1843	031724	001	000	.BYTE	1,0	;IN 0/2 OUTSIDE
1844	031726	004000		.WORD	4000	
1845	031730	001	000	.BYTE	1,0	;OUT 0/2 OUTSIDE
1846	031732	002000		.WORD	2000	
1847	031734	005	000	.BYTE	5,0	;IN 0/2 INSIDE
1848	031736	004000		.WORD	4000	
1849	031740	005	000	.BYTE	5,0	;OUT 0/2 INSIDE
1850	031742	002000		.WORD	2000	
1851	031744	001	000	.BYTE	1,0	;IN 0/3 OUTSIDE
1852	031746	000010		.WORD	10	
1853	031750	001	000	.BYTE	1,0	;OUT 0/3 OUTSIDE
1854	031752	000004		.WORD	4	
1855	031754	005	000	.BYTE	5,0	;IN 0/3 INSIDE
1856	031756	000010		.WORD	10	
1857	031760	005	000	.BYTE	5,0	;OUT 0/3 INSIDE
1858	031762	000004		.WORD	4	
1859	031764	000000		.WORD	0	; TERMINATOR
1860	031766	004537	030360	JSR	R5,UNLOAD	;UNLOAD PACK
1861	031772			PRINTF	#ENDPAS	;END OF PASS
(7)	031772	012746	022005	MOV	#ENDPAS,-(SP)	
(6)	031776	012746	000001	MOV	#1,-(SP)	
(3)	032002	010600		MOV	SP,R0	
(4)	032004	104017		EMT	C\$PNTF	
(4)	032006	062706	000004	ADD	#4,SP	
1862	032012			3\$: BREAK		
(3)	032012	104022		EMT	C\$BRK	
1863	032014	000776		BR	3\$	
1864						
1865						
1866						
1867	032016			ENDTST		
(3)	032016			L10012:		
(3)	032016	104001		EMT	C\$ETST	
1868						
1869	032020			BGNMOD	HRDPRM	
1870						
1871	032020			BGNHRD		
(3)	032020	000025		.WORD	L10013-L\$HARD/2	
1872						
1873	032022			GPRML	RLMSG,RLCNT,1,YES	
(4)	032022	004130		.WORD	T\$CODE	
(4)	032024	032074		.WORD	RLMSG	
(4)	032026	000001		.WORD	1	
1874	032030			GPRMA	CSRMSG,CSR,0,160000,177776,YES	
(4)	032030	000031		.WORD	T\$CODE	
(4)	032032	032101		.WORD	CSRMSG	
(4)	032034	160000		.WORD	T\$LOLIM	
(4)	032036	177776		.WORD	T\$HILIM	

1875	032040				GPRMA	VECMSG,VECT,0,0,776,YES
(4)	032040	001031			.WORD	T\$CODE
(4)	032042	032126			.WORD	VECMSG
(4)	032044	000000			.WORD	T\$LLOLIM
(4)	032046	000776			.WORD	T\$HILIM
1876	032050				GPRMD	BRMSG,PRIOR,0,340,0,7,YES
(4)	032050	002032			.WORD	T\$CODE
(4)	032052	032115			.WORD	BRMSG
(4)	032054	000340			.WORD	340
(4)	032056	000000			.WORD	T\$LLOLIM
(4)	032060	000007			.WORD	T\$HILIM
1877	032062				GPRMD	DRMSG,DRBT,0,03400,0,7,YES
(4)	032062	003032			.WORD	T\$CODE
(4)	032064	032135			.WORD	DRMSG
(4)	032066	003400			.WORD	03400
(4)	032070	000000			.WORD	T\$LLOLIM
(4)	032072	000007			.WORD	T\$HILIM
1878						
1879	032074				ENDHRD	
(2)					.EVEN	
(3)	032074			L10013:		
1880						
1881	032074	046122	030461	000	RLMSG:	.ASCIZ /RL11/
1882	032101	102	051525	040440	CSRMSG:	.ASCIZ /BUS ADDRESS/
	032106	042104	042522	051523		
	032114	000				
1883	032115	102	020122	042514	BRMSG:	.ASCIZ /BR LEVEL/
	032122	042526	000114			
1884	032126	042526	052103	051117	VECMSG:	.ASCIZ /VECTOR/
	032134	000				

```
1886 032135 104 044522 042526 DRMSG: .ASCIZ /DRIVE/
      032142 000
1887 032144 .EVEN
1888
1889 032144 ENDMOD
1890
1891
1892
1893 032244 .=32244
1894
1895 ;AREA RESERVED AS PATCH AREA FOR DIAGNOSTICS.
1896 ;THIS DIAGNOSTIC DOES NOT RUN IN APT MODE.
1897
1898 032244 LASTAD
      (2) .EVEN
      (3) 032244 L$LAST::
1899
1900
```

1902
12773 063040 000000
12774 063042 000000
12775 063044 000000
12776 063046 000000
12777 063052
12778 000200

.SBTTL DIAGNOSTIC SUPERVISOR -- LOW CORE SET UP
.WORD 0 ;SPACE FOR USER POOL POINTER
.WORD 0 ;SIZE
.WORD 0 ;CHECKSUM (NOT CURRENTLY USED)
.WORD 0 ;SIZE OF H.W. PTAB. ALLOCATION
END.SUPV=+.2
.END 200

ABOFLA	032570	G	BIT5	=	000040	G	CSRMSG	032101		C\$REVI	=	000002	EF09	=	000011	G
ABOPAS	032506	G	BIT6	=	000100	G	CURR.S	032252	G	C\$RPT	=	000025	EF10	=	000012	G
ABO.FM	035050		BIT7	=	000200	G	CURR.T	032254	G	C\$SEFG	=	000047	EF11	=	000013	G
ADJCYL	024016		BIT8	=	000400	G	C\$AAD	044612		C\$SPRI	=	000041	EF12	=	000014	G
ADJDIR	002704		BIT9	=	001000	G	C\$AAE	044624		C\$SVEC	=	000037	EF13	=	000015	G
ADJFLG	002702		BLD.HW	037732		C\$AAK	045622		C\$TPRI	=	000013	EF14	=	000016	G	
ADJLC2	003016		BLOCK	055344		C\$AAL	045766		C\$UNBU	=	000031	EF15	=	000017	G	
ADJLC3	003020		BMP	003004		C\$ABRT	=	000021	C\$WTM	=	000026	EF16	=	000020	G	
ADJLC4	003022		BRMSG	032115		C\$ADR	=	000020	C\$WTU	=	000027	EMT.TR	032574	G		
ADJLOC	002700		BSEXIT	027146		C\$AU	=	000054	DA	=	000004	END	022216			
ADJTRK	003012		BSVWR	026770		C\$BRK	=	000022	DCKER	017174		ENDBUF	017070			
ADJTXT	017727		BUF	003030		C\$BSEG	=	000004	DCRC	=	004000	ENDPAS	022005			
ADJUUT	003014		B\$AAB	041334		C\$BSUB	=	000002	DECMG	051534		END.OF	040620			
AFSI	032276	G	B\$AAF	041246		C\$BUFF	=	000030	DERR	=	040000	END.SU	=	063052		
ALLOC	053210		CALLPC	=	000022	C\$CEFG	=	000046	DESCYL	002720		ENVIRO	032316	G		
APT.ER	034200		CALLPS	=	000024	C\$CLEA	=	000012	DEV.CO	032256	G	EOP.CH	061520	G		
ASSEMB	=	000010	CALLSP	=	000026	C\$CLP1	=	000006	DIAGMC	=	000000	EOP.FM	035064			
ASAAV	037046		CALLTC	=	000030	C\$CVEC	=	000036	DIAG.T	032576	G	EOP.IN	037230			
ASAAW	037062		CAL.CL	057732		C\$DCLN	=	000044	DIRC	002716		ERFLG	002674			
ASAAZ	037074		CAL.TI	057770	G	C\$DODU	=	000053	DLT	=	010000	ERR	=	100000		
ASAAZ	037102		CHECK	027170		C\$DRPT	=	000024	DPDVD	062206	G	ERRFND	017462			
ASABA	037116		CHKLUP	041350		C\$DU	=	000055	DPMUL	062074	G	ERRFOR	046044			
ASABA	037126		CHKSTR	053552		C\$EDIT	=	000002	DRBT	=	000006	ERRHAN	044644			
BA	=	000002	CHKTTY	051640		C\$ERDF	=	000002	DRBUF	017030		ERR.HR	045602			
BA16	=	000020	CHK.MA	037510		C\$ERHR	=	000003	DRDY	=	000001	ERR.NU	032246	G		
BA17	=	000040	CHK.PC	044640		C\$ERSF	=	000001	DRMSG	032135		ERR.SF	045606			
BBA	003000		CHK.SW	033700		C\$ERSO	=	000004	DRPCOD	023330	G	ERR1	017756	G		
BCS	002776		CHRCNT	053072		C\$ESCA	=	000010	DRST	=	000013	ERR1FO	046130			
BDA	003002		CH.FLA	037216		C\$ESEG	=	000005	DRSTAT	002706		ERR2	020014	G		
BDATA	002736		CH.PAS	037234		C\$ESUB	=	000003	DSB	=	000004	ERR3	020054	G		
BGN.SU	=	032244	CLEAR.	040632		C\$ETST	=	000001	DSPCOD	022046	G	ERR4	020206	G		
BINMSG	051520		CLKACC	032504	G	C\$EXIT	=	000032	DUNIT.	032512	G	ERR5	020372	G		
BIT0	=	000001	CLKBFR	057734		C\$GMAN	=	000043	DVC.FT	045572		ERR6	020432	G		
BIT00	=	000001	CLKCNT	032502	G	C\$GPHR	=	000042	DSAAG	046476		ESC.PC	044636			
BIT01	=	000002	CLKJUM	060340	G	C\$GPRI	=	000040	DSAAH	046514		EV.COU	032250	G		
BIT02	=	000004	CLKRES	061342	G	C\$GTIM	=	000052	DSAAI	051262		EXIT	023322			
BIT03	=	000010	CLKSER	061476	G	C\$INIT	=	000011	DSAAJ	051266		E.BA	002764			
BIT04	=	000020	CLKSON	032542	G	C\$INLP	=	000020	DSAAK	051304		E.CS	002762			
BIT05	=	000040	CLK.SE	037312		C\$KWOF	=	000035	DSAAL	051322		E.DA	002766			
BIT06	=	000100	CLNCOD	023324	G	C\$KWON	=	000034	DSAAM	051332		E.MP	002770			
BIT07	=	000200	CLR.MA	037566		C\$LOOP	=	000100	EF.CON	=	000036	G	E.MP1	002772		
BIT08	=	000400	COMPEN	023300		C\$MANI	=	000051	EF.NEW	=	000035	G	E.MP2	002774		
BIT09	=	001000	CNTTOT	017072		C\$MSG	=	000023	EF.PWR	=	000034	G	FADJ	002664		
BIT1	=	000002	CNVT	056010		C\$PNTB	=	000014	EF.RES	=	000037	G	FEW	017212		
BIT10	=	002000	COMMAN	032314	G	C\$PNTF	=	000017	EF.STA	=	000040	G	FILL	052370		
BIT11	=	004000	COMMTA	055624		C\$PNTS	=	000016	EF01	=	000001	G	FILL.C	000204	G	
BIT12	=	010000	COMPAT	031056		C\$PNTX	=	000015	EF02	=	000002	G	FIXCYL	027774		
BIT13	=	020000	CONTCL	061422	G	C\$POIN	=	000040	EF03	=	000003	G	FLAGS	032310	G	
BIT14	=	040000	CRDY	=	000200	C\$QIO	=	000377	EF04	=	000004	G	FLAGS1	032312	G	
BIT15	=	100000	CRLF	051722		C\$RDBU	=	000007	EF05	=	000005	G	FLAGTA	055542		
BIT2	=	000004	CRSET	=	000002	C\$REFG	=	000050	EF06	=	000006	G	FLAG.I	037276		
BIT3	=	000010	CS	=	000000	C\$REQT	=	000045	EF07	=	000007	G	FLA.SE	055510		
BIT4	=	000020	CSR	=	000000	C\$RESE	=	000033	EF08	=	000010	G	FLG.MA	037236		

FNDBSC	027704	GET.TW	053730	INN20	002175	L\$DEVP	002064	G	MAN.TI	001244	
FNDDRV	025562	GLBDAT	002122	INN21	002202	L\$DISP	022050	G	MAP16	062444	G
FNDTRK	027562	GLBEQA	002122	INN30	002176	L\$DR	002112	G	MASK.B	041346	
FORM.T	046140	GLBERR	017756	INN31	002203	L\$DRCT	002070	G	MASK.W	041344	
FORSK	002724	GLBSUB	023334	INN40	002177	L\$DRS	002072	G	MDHEDR	002000	G
FOWR	002662	GLBTXT	017072	INN41	002204	L\$DRST	002112	G	MEM.SI	036704	
FREE	053446	G\$BIT =	000003	INN50	002200	L\$DTP	002040	G	MERGE	027374	
FRM1	020520	G\$STAT =	000004	INN51	002205	L\$DU	023330	G	MID10	002150	
FRM10	021270	G\$EXCP =	000400	INPUTA	052476	L\$DUT	002076	G	MID11	002155	
FRM11	021334	G\$HILI =	000002	INTEN =	000100	L\$DVTY	002114	G	MID20	002151	
FRM12	021371	G\$LOLI =	000001	INTFOR	045774	L\$EF	002056	G	MID21	002156	
FRM13	021450	G\$NO =	000000	INVAL.	036602	L\$EFLG	002034	G	MID30	002152	
FRM14	021533	G\$OFFS =	000400	INVINT	045632	L\$EXP1	002042	G	MID31	002157	
FRM15	021562	G\$OF SI =	000376	INV.SW	033634	L\$EXP2	002044	G	MID40	002153	
FRM16	021621	G\$PRMA =	000001	IN.SUF	040604	L\$EXP3	002046	G	MID41	002160	
FRM17	021706	G\$PRMD =	000002	I\$AU =	000041	L\$HARD	032022	G	MID50	002154	
FRM18	021742	G\$PRML =	000000	I\$CLN =	000041	L\$HPCP	002016	G	MID51	002161	
FRM2	020615	G\$RADA =	000140	I\$DU =	000041	L\$HPTP	002022	G	MIN.IN	032322	G
FRM3	020712	G\$RADB =	000000	I\$HRD =	000041	L\$HW	022034	G	MIN.US	032324	G
FRM4	020771	G\$RADD =	000040	I\$INIT =	000041	L\$I CP	002104	G	MK =	000001	
FRM5	021032	G\$RADF =	000200	I\$MOD =	000041	L\$INIT	022052	G	MODR	062006	G
FRM6	021101	G\$RADL =	000120	I\$MSG =	000041	L\$LADP	002026	G	MP =	000006	
FRM7	021122	G\$RADO =	000020	I\$PWR =	000041	L\$LAST	032244	G	MSG.AD	032270	G
FRM8	021162	G\$RADT =	000100	I\$RPT =	000041	L\$MREV	002050	G	MSG.TY	032244	G
FRM9	021226	G\$XFER =	000004	I\$SEG =	000041	L\$NAME	002000	G	MUL	061742	G
FRTTRK	002750	G\$YES =	000010	I\$SRV =	000041	L\$REPP	002066	G	NEWPRI	061466	G
FUNERR	017566	HCORED	037006	I\$SUB =	000041	L\$REV	002010	G	NEXTAR	055726	
FWD	017635	HCOREQ	036716	I\$TST =	000041	L\$SPC	002062	G	NONE	017310	
F\$AU =	000015	HCORET	032532	J\$JMP =	000167	L\$SPCP	002020	G	NO.CLK	036632	
F\$BGN =	000040	HCRC =	004000	KBPTR	032354	L\$SPTP	002024	G	NO.FLA	055522	
F\$CLEA =	000007	HC.ADR	032302	KBUF	032356	L\$STA	002030	G	NO.LPT	053040	
F\$DU =	000016	HC.DEF	032274	LDUNC	030560	L\$TIML	002014	G	NO.PTA	037036	
F\$END =	000041	HC.DIA	032272	LINE.F	032572	L\$TIMU	002054	G	NR =	000000	
F\$HARD =	000004	HDRFND	002122	LOAD	030460	L\$TIM1	002052	G	NUMBIN	046164	
F\$HW =	000013	HEAD =	000100	LOAD.F	037232	L\$TSTI	002100	G	NUM.LA	046332	
F\$INIT =	000006	HEAD01	002714	LOGMSG	051542	L\$UNIT	002012	G	NUM.NO	032306	G
F\$JMP =	000050	HERTZ.	036656	LPBFR	032352	L.CLK.	036642		NUM.UN	032714	
F\$MOD =	000000	HNF =	010000	LPCNTR	032350	L10000	020012		NUNITS	041322	
F\$MSG =	000011	HOLDSP =	000020	LPT.AD	036674	L10001	020052		NXM =	020000	
F\$PWR =	000017	HPTCOD	022032	LPT.RE	036670	L10002	020204		NXTFOR	056002	
F\$RPT =	000012	HRDPRM	032020	LSI.RE	036664	L10003	020370		OCTMSG	051526	
F\$SEG =	000003	HSFLC	002710	LSTCLR	002670	L10004	020430		OFFSET	002744	
F\$SOFT =	000005	HW.ADR	032300	LSTDRV	002732	L10005	020516		OPI =	002000	
F\$SRV =	000010	H\$AAB	056336	LSTTRK	002746	L10006	022046		OQU10	002136	
F\$SUB =	000002	INAWR	027062	LUP	057636	L10007	023322		OQU11	002143	
F\$SW =	000014	ININIT	032522	LUP.AD	044642	L10010	023326		OQU20	002137	
F\$TEST =	000001	INITCO	022052	L\$APT	002036	L10011	023332		OQU21	002144	
GARBAG	053074	INITIA	051550	L\$AUT	002074	L10012	032016		OQU30	002140	
GDATA	002734	INITWR	017112	L\$CCP	002106	L10013	032074		OQU31	002145	
GETCHR	051600	INIT.M	037634	L\$CLEA	023324	MAJ.IN	032326	G	OQU40	002141	
GETCMN	055164	INIT.R	032336	L\$CO	002032	MAJ.LO	057736		OQU41	002146	
GETPAR	046656	INN10	002174	L\$DEPO	002011	MAJ.US	032330	G	OQU50	002142	
GETSWI	054160	INN11	002201	L\$DESC	002102	MANY	017251		OQU51	002147	

OSECT	002712	PWR.FA	062700	G	SPEC.U	037136	TQU50	002166	UNLOAD	030360	
OUT10	002124	PWR.FL	032334	G	SPV.SE	000400	TQU51	002173	USER.P	032524	G
OUT11	002131	PWR.MS	063026		STARTC	061416	TRKCN	002760	USER.T	032526	G
OUT20	002125	PWR.SA	063022		STFLG	002676	TRKFND	002756	UUT	002726	
OUT21	002132	PWR.UP	063024		STRCHR	052430	TST.AB	041460	VAJWR	026534	
OUT30	002126	P.CLK.	036650		STRT.T	037214	TST.TO	033662	VALID.	032764	
OUT31	002133	RDHDR =	000010		STSEC	003026	TYPEC	052066	VAL.LA	033604	
OUT40	002127	READ =	000014		STSEC1	003024	TYPEPC	045762	VAL.SW	037250	
OUT41	002134	READ.P	057740	G	ST.SET	034046	TYPFLA	055404	VEC =	000002	
OUT50	002130	READ1	017702		SUNIT.	037220	TYPLIN	051764	VECMG	032126	
OUT51	002135	REASON	002672		SUPERV	035102	TYPNUM	051346	VECT =	000002	
OVMS	017361	RECER	017542		SUPFLA	032510	TYPSTR	052004	VEROD	026206	
OVWER	017522	RECMS	017414		SUPV.T	032662	TYP.ER	045612	VEROW	025622	
OVWPER	023334	REGBAC	062430	G	SUP.PR	033620	TY.UNI	040624	WCOUNT	002740	
OVWTRK	002626	REGDMP	025144		SURFAC	002754	TSARGC=	000001	WIDTH	046532	
OSAPTS=	000000	REGSAV	062414	G	SVCGBL=	000000	TSCODE=	003032	WRITE =	000012	
OSAU =	000000	REQN.P	032320	G	SVCHAN	041536	TSERCO=	000062	WRIT1	017655	
OSBGNR=	000000	REQN.T	037212		SVCINS=	000000	TSERRN=	000322	WRSEC	023566	
OSBGNS=	000000	REV	017645		SVCSUB=	177777	TSEXCP=	000000	XEQDIA	061554	G
OSDU =	000000	REVSK	002722		SVCTAG=	000000	TSHILI=	000007	XEQSUB	061542	G
OSGNSW=	000000	RE.SET	034002		SVCTST=	177777	TSLOLI=	000000	XEQ.CL	041264	
OSPOIN=	000001	RLCNT =	000010		SWCHAN	037030	TSLSYM=	010000	XEQ.CM	036574	
OSPWR =	000000	RLMSG	032074		SWITCH	055702	TSNEST=	177777	XEQ.IN	040746	
PARSES	055236	RSADJS	025414		SW.ADR	032304	TSNSKO=	000000	XEQ.LA	035036	
PAR.LA	051224	RSTACK	061670	G	SW.PTA	037014	TSNSK1=	000004	XEQ.OP	041040	
PASS.C	032260	SAVEDO=	034200		SYS.FT	045562	TSSAVL=	177777	XEQ.PR	034240	
PAT =	000006	SEARCH	053676		S\$LSYM=	010000	TSSEGL=	177777	XEQ.TE	041104	
PATLST	002652	SECBUF	002246		TEMP	002666	TSSUBN=	000000	XTIME	060426	G
PRINTC	053050	SECLST	002206		TERMI	057726	TSTAGL=	177777	XTIMEN	061252	
PRINTF	056356	SECT	002730		TERMLI	055530	TSTAGN=	010014	XTIMST	060450	
PRIOR =	000004	SECWRD	002742		TERMTA	051512	TSTEMP=	000000	XXDP.D	036614	
PRI00 =	000000	SEEK =	000006		TEST.M	037150	TSTEST=	000001	XSALWA=	000000	
PRI01 =	000040	SEGSTA	032544	G	TIMFLG	032500	TSTSTM=	177777	XSALS=	000040	
PRI02 =	000100	SERNM1	003006		TIM.CO	032332	TSTSTS=	000001	X\$OFFS=	000400	
PRI03 =	000140	SERNM2	003010		TIM.OP	046136	TSSCLE=	010010	X\$TRUE=	000020	
PRI04 =	000200	SERNUM	030030		TOO.MA	051472	TSSDU =	010011	\$BREG	037310	
PRI05 =	000240	SETLST	025354		TQU10	002162	TSSHAR=	010013	\$ENDAD	061526	G
PRI06 =	000300	SETUP	0222.J		TQU11	002167	TSSHW =	010006	\$SAV2	062572	G
PRI07 =	000340	SET.MA	037422		TQU20	002163	TSSINI=	010007	\$SAV3	062606	G
PRNTST	052740	SHIFT	062526	G	TQU21	002170	TSSMSG=	010005	\$SAV4	062624	G
PRO.CM	037210	SIGN =	000004		TQU30	002164	T\$STES=	010012	\$SAV5	062644	G
PRSTRK	002752	SKCYL	024716		TQU31	002171	T1	031056	.	=	063050
PTAB.S	032530	SKER	017616		TQU40	002165	UNIT.D	032262			
PUTCHR	051554	SKHS =	000020		TQU41	002172	UNI.MA	037140			

. ABS. 063050 000

ERRORS DETECTED: 0

DSKZ:CZRLFBSUP, DSKZ:CZRLFBSUP=CZRLFBSUP/ML, CZRLFBSUP.P11, CZRLFBSUP.SUP
 RUN-TIME: 28 25 .9 SECONDS
 RUN-TIME RATIO: 99/54=1.8
 CORE USED: 15K (29 PAGES)