

ML11

ML11 PERF EXERCISER  
CZMLBAO

AH-S430A-MC  
FICHE 1 OF 2

FEB 1981  
COPYRIGHT © 1981  
MADE IN USA



The main body of the document is a large grid of data, likely a performance exerciser. It consists of approximately 15 columns and 25 rows of small, faint text blocks. Each block appears to contain a small table or set of data points, possibly representing test results or performance metrics. The text is too small to read clearly, but the overall structure is a dense, organized array of information.

ML11

ML11 PERF EXERCISER  
CZMLBAO

AH-S430A-MC  
FICHE 2 OF 2

FEB 1981  
COPYRIGHT © 1981  
MADE IN USA



1

.SBTTL USER DOCUMENTATION  
.REM 8

IDENTIFICATION

PRODUCT CODE: AC-S428A-MC  
PRODUCT NAME: CZMLBA0 ML11 PERFORMANCE EXERCISER  
PRODUCT DATE: 02-FEB-81  
MAINTAINER: STORAGE SYSTEMS DIAGNOSTICS  
AUTHOR: MICHELE D. ROSEN

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 DOCUMENT.

NO RESPONSIBILITY IS ASSUMED FOR THE USE OR RELIABILITY OF SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL OR ITS AFFILIATED COMPANIES.

COPYRIGHT (C) 1981 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL  
DEC

PDP  
DECUS

UNIBUS  
DECTAPE

MASSBUS

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

.TITLE ML11 PERFORMANCE EXERCISER

:DURING INITIAL PROGRAM DEVELOPMENT, IT IS DESIRABLE  
:TO TREAT EACH OF THE 4 SKELETON FILES INDIVIDUALLY,  
:AND TO ASSEMBLE THEM INTO 4 SEPARATE OBJECT FILES.  
:THIS IS DONE TO HASTEN THE PROCESS OF DETECTING AND  
:CORRECTING SYNTAX ERRORS.

:WHEN THE TIME COMES, HOWEVER, TO PRODUCE A DIAGNOSTIC  
:PACKAGE WHICH WILL BE TAKEN TO A TEST STATION AND RUN,  
:THE SKELETONS MUST BE HANDLED DIFFERENTLY. THEY MUST  
:BE ASSEMBLED TOGETHER TO PRODUCE A SINGLE OBJECT FILE.  
:A FURTHER RESTRICTION IS THAT THE FLAG 'ONEFILE' MUST  
:BE DEFINED (TO INDICATE THAT ALL 4 OF THE SKELETONS  
:ARE NOW COMBINED INTO THE REQUIRED SINGLE OBJECT FILE).

:NOTE THAT 'ONEFILE' IS NOT DEFINED WITHIN THE SOURCE  
:CODE OF ANY OF THE SKELETONS, BUT THAT IT CAN EASILY  
:BE ACCOMPLISHED BY INCLUDING THIS FILE (ONEFILEX.MAC)  
:AS PART OF THE INPUT SPECIFICATIONS OF THE ASSEMBLY  
:COMMAND LINE.

.NLIST TOC

000001

ONEFILE = 1

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	COMMANDS
2.2	SWITCHES
2.3	FLAGS
2.4	HARDWARE QUESTIONS
2.5	SOFTWARE QUESTIONS
2.6	EXTENDED P-TABLE DIALOGUE
2.7	QUICK STARTUP PROCEDURE
3.0	ERROR INFORMATION
4.0	PERFORMANCE AND PROGRESS REPORTS
5.0	DEVICE INFORMATION TABLES
6.0	TEST SUMMARIES

## 1.0 GENERAL INFORMATION

### 1.1 PROGRAM ABSTRACT

THE ML11 PERFORMANCE EXERCISER IS A BLISS PROGRAM WHICH RUNS UNDER THE PDP-11 DIAGNOSTIC SUPERVISOR AND WHICH EXERCISES UP TO 8 ML11 UNITS ON A SINGLE RH CONTROLLER. AN ML11 UNIT IS A FAST, RANDOM ACCESS, BLOCK MODE MOS MEMORY SYSTEM WITH ECC CAPABILITY. IT IS MADE UP OF 3 CONTROL MODULES AND UP TO 16 ARRAY MODULES. IT IS A MASSBUS DEVICE, AND AS SUCH, CONFORMS TO MASSBUS STANDARDS.

THIS DIAGNOSTIC HAS BEEN WRITTEN FOR USE WITH THE DIAGNOSTIC RUNTIME SERVICES SOFTWARE (SUPERVISOR). THESE SERVICES PROVIDE THE INTERFACE TO THE OPERATOR AND TO THE SOFTWARE ENVIRONMENT. THIS PROGRAM CAN BE USED WITH XXDP+, ACT, APT, SLIDE AND PAPER TAPE. FOR A COMPLETE DESCRIPTION OF THE RUNTIME SERVICES, REFER TO THE XXDP+ USER'S MANUAL. THERE IS A BRIEF DESCRIPTION OF THE RUNTIME SERVICES IN SECTION 2 OF THIS DOCUMENT.

### 1.2 SYSTEM REQUIREMENTS

1. PDP-11 WITH 28K WORDS OF MEMORY
2. CONSOLE TERMINAL
3. RH11 OR RH70 CONTROLLER
4. 1 TO 8 ML11 DRIVES
5. RK05 OR OTHER XXDP+ LOAD MEDIA

### 1.3 RELATED DOCUMENTS AND STANDARDS

THE HARDWARE DESIGN IS EXPECTED TO CONFORM TO THE STANDARDS SET FORTH IN THE MASSBUS SPECIFICATION (DEC STANDARD 159).

THE FOLLOWING DOCUMENTATION MAY PROVE USEFUL IN LEARNING MORE ABOUT THE ML11:

1. ML11 ENGINEERING SPECIFICATION
2. RWS04 FIXED-HEAD DISK SUBSYSTEM USER'S MANUAL
3. THE ML11 PERFORMANCE EXERCISER'S PROGRAM LISTING
4. THE ML11 LOGIC TEST'S SPECIFICATIONS AND LISTINGS
5. ML11 PROJECT PLAN
6. XXDP+ USER'S MANUAL (CHQUS)
7. PDP-11 DIAGNOSTIC SUPERVISOR DOCUMENTATION (SUPINT, SUPFUN, SUPPRGC)

#### 1.4 DIAGNOSTIC HIERARCY PREREQUISITES

ML11 SUBSYSTEM FAULTS WILL BE DETECTED BUT NOT ISOLATED, SINCE OTHER DIAGNOSTICS WILL BE AVAILABLE FOR TROUBLESHOOTING THE EXACT CAUSE OF FAILURE. A LIST OF AVAILABLE DIAGNOSTIC TOOLS IS INCLUDED IN APPENDIX A OF THE ML11 FUNCTIONAL SPECIFICATION.

#### 1.5 ASSUMPTIONS

#### 2.0 OPERATING INSTRUCTIONS

THIS SECTION CONTAINS A VERY BRIEF DESCRIPTION OF THOSE PARTS OF THE DIAGNOSTIC RUNTIME SERVICES OF THE PDP-11 DIAGNOSTIC SUPERVISOR WHICH ARE APPLICABLE FOR THIS EXERCISER. CONSULT THE XXDP+ USER'S MANUAL (CHQUS) FOR MORE DETAILS.

#### 2.1 COMMANDS

THE FOLLOWING IS A LIST OF THE 11 COMMANDS AVAILABLE TO THE DRS. ANY COMMAND IS RECOGNIZED BY ITS FIRST 3 CHARACTERS, AND MANY COMMANDS MAY BE MODIFIED BY OPTIONAL SWITCHES WHICH ARE DESCRIBED IN THE NEXT SECTION.

DRS COMMAND	EFFECT
START	START THE DIAGNOSTIC FROM AN INITIAL STATE
RESTART	START THE DIAGNOSTIC WITHOUT INITIALIZING
CONTINUE	CONTINUE AT TEST THAT WAS INTERRUPTED (AFTER ^C)
PROCEED	CONTINUE FROM AN ERROR HALT
EXIT	RETURN TO THE XXDP+ MONITOR
ADD	ACTIVATE A UNIT FOR TESTING (ALL UNITS ARE CONSIDERED TO BE ACTIVE AT START TIME)
DROP	DEACTIVATE A UNIT
PRINT	PRINT STATISTICAL INFORMATION
DISPLAY	TYPE A LIST OF ALL DEVICE INFORMATION
FLAGS	TYPE THE STATE OF ALL FLAGS (SEE SECTION 2.3)
ZFLAGS	CLEAR ALL FLAGS (SEE SECTION 2.3)

2.2 SWITCHES

SWITCHES ARE USED TO MODIFY PROGRAM OPERATION, AND ARE APPENDED TO COMMANDS. THE SWITCHES WHICH HAVE MEANING FOR THIS EXERCISER AND INFORMATION ABOUT THEIR USE ARE GIVEN IN THE FOLLOWING TWO TABLES.

SWITCH	EFFECT
/PASS:DDDDD	EXECUTE DDDDD PASSES (DDDDD = 1 TO 64000)
/FLAGS:FLGS	SET SPECIFIED FLAGS (SEE SECTION 2.3)
/EOP:DDDDD	REPORT END OF PASS MESSAGE AFTER EVERY DDDDD PASSES ONLY (DDDDD = 1 TO 64000).
/UNITS:LIST	TEST/ADD/DROP ONLY THOSE UNITS SPECIFIED IN THE LIST. LIST EXAMPLE: /UNITS:0:3:5-7 USE UNITS 0,3,5,6,7 (UNIT NUMBERS = 0-7)

EXAMPLE OF SWITCH USAGE:

START/PASS:1000/EOP:100

THE EFFECT OF THIS COMMAND WILL BE:

1. ALL UNITS WILL BE TESTED 1000 TIMES
2. THE END OF PASS MESSAGES WILL BE PRINTED AFTER EACH 100 PASSES ONLY.

BELOW IS A TABLE THAT SPECIFIES WHICH SWITCHES CAN BE USED BY EACH COMMAND.

	PASS	FLAGS	EOP	UNITS
START	X	X	X	X
RESTART	X	X	X	X
CONTINUE	X	X	X	
PROCEED		X		
DROP				X
ADD				X
PRINT				
DISPLAY				X
FLAGS				
ZFLAGS				
EXIT				



## 2.3 FLAGS

FLAGS ARE USED TO SET UP CERTAIN OPERATION PARAMETERS SUCH AS LOOPING ON ERROR. ALL FLAGS ARE CLEARED AT STARTUP AND REMAIN CLEARED UNTIL EXPLICITLY SET USING THE FLAGS SWITCH. FLAGS ARE ALSO CLEARED AFTER A START COMMAND UNLESS SET USING THE FLAG SWITCH. THE ZFLAGS COMMAND MAY ALSO BE USED TO CLEAR ALL FLAGS. WITH THE EXCEPTION OF THE START AND ZFLAGS COMMANDS, NO COMMANDS AFFECT THE STATE OF THE FLAGS. THEY REMAIN SET OR CLEARED AS SPECIFIED BY THE LAST FLAG SWITCH.

FLAG	EFFECT
HOE	HALT ON ERROR - CONTROL IS RETURNED TO DRS COMMAND MODE
LOE	LOOP ON ERROR
IER*	INHIBIT ALL ERROR REPORTS
IBR*	INHIBIT ALL ERROR REPORTS EXCEPT FIRST LEVEL (FIRST LEVEL CONTAINS ERROR TYPE, NUMBER, PC, TEST AND UNIT)
IXR*	INHIBIT EXTENDED ERROR REPORTS (THOSE CALLED BY PRINTX MACRO'S)
PRI	DIRECT MESSAGES TO LINE PRINTER
BOE	'BELL' ON ERROR
ISR	INHIBIT STATISTICAL REPORTS
IDR	INHIBIT PROGRAM DROPPING OF UNITS
ADR	EXECUTE AUTODROP CODE

\* ERROR MESSAGES ARE DESCRIBED IN SECTION 4.7

SEE THE XXDP+ USER'S MANUAL FOR MORE DETAILS ON FLAGS. MORE THAN ONE FLAG MAY BE SPECIFIED WITH THE /FLAGS SWITCH. FOR EXAMPLE, TO CAUSE THE PROGRAM TO LOOP ON ERROR, INHIBIT ERROR REPORTS AND TYPE A 'BELL' ON ERROR, YOU MAY USE THE FOLLOWING STRING:

/FLAGS:LOE:IER:BOE

## 2.4 HARDWARE QUESTIONS

WHEN A DIAGNOSTIC IS STARTED, THE DIAGNOSTIC RUNTIME SERVICES PROMPTS THE USER FOR HARDWARE INFORMATION BY TYPING "CHANGE HW (L) ?". YOU MUST ANSWER 'Y' AFTER A START COMMAND UNLESS THE HARDWARE INFORMATION HAS BEEN PRELOADED USING THE SETUP UTILITY (SEE CHAP. 6 OF THE XXDP+ USER'S MANUAL). WHEN YOU ANSWER THIS QUESTION WITH A 'Y' THE RUNTIME SERVICES WILL ASK FOR THE NUMBER OF UNITS (IN DECIMAL). YOU WILL THEN BE ASKED THE FOLLOWING QUESTIONS FOR EACH UNIT:

### 2.4.1 QUESTIONS AND ANSWERS

#### 1. CSR ADDRESS (172000) 0?

ANSWER: THE CSR ADDRESS OF THE RH CONTROLLER  
(OCTAL, DEFAULT 172000).

#### 2. INTERRUPT VECTOR (204) 0?

ANSWER: THE INTERRUPT VECTOR ADDRESS  
(OCTAL, DEFAULT 204).

#### 3. BR LEVEL FOR INTERRUPT (5) 0?

ANSWER: THE BUS REQUEST LEVEL FOR INTERRUPT  
(OCTAL, DEFAULT 5).

#### 4. DRIVE NUMBER (0) 0?

ANSWER: THE PHYSICAL DRIVE NUMBER(S) OF THE DRIVE(S) TO BE TESTED  
(OCTAL, {0-7}, DEFAULT 0).

2.4.2 SUMMARY OF HARDWARE QUESTION SEQUENCE

```
*****
*                                     *
* # UNITS?                           *
*                                     *
*****
!
*****
*                                     *
* CSR ADDRESS?                       *
*                                     *
*****
!
*****
*                                     *
* VECTOR?                            *
*                                     *
*****
!
*****
*                                     *
* BR LEVEL?                          *
*                                     *
*****
!
*****
*                                     *
* DRIVE NUMBER?                      *
*                                     *
*****
```

## 2.5 SOFTWARE QUESTIONS

AFTER YOU HAVE ANSWERED THE HARDWARE QUESTIONS, OR AFTER A RESTART OR OR CONTINUE COMMAND, THE RUNTIME SERVICES WILL ASK FOR SOFTWARE PARAMETERS. THESE PARAMETERS WILL GOVERN SOME DIAGNOSTIC SPECIFIC OPERATION MODES. YOU WILL BE PROMPTED BY "CHANGE SW (L) ?" IF YOU WISH TO CHANGE ANY PARAMETERS, ANSWER BY TYPING "Y". THE SOFTWARE QUESTIONS AND THE DEFAULT VALUES ARE DESCRIBED IN THE NEXT PARAGRAPH(S).

INCLUDED IN THIS SECTION ARE QUESTIONS ABOUT PARAMETERS WHICH AFFECT PROGRAM OPERATION. REQUESTS FOR SOFTWARE OPTIONS SHOULD BE DISCUSSED AND INCORPORATED INTO THIS SPECIFICATION AS SOON AS POSSIBLE, AND CERTAINLY BEFORE SIGNOFF TIME.

IF A ^Z (CONTROL-Z) IS TYPED AS THE ANSWER TO ANY QUESTION, THE PROGRAM WILL START TO EXECUTE.

IF A ^C (CONTROL-C) IS TYPED AT ANY TIME, CONTROL WILL RETURN TO THE DIAGNOSTIC SUPERVISOR.

### 2.5.1 QUESTIONS AND ANSWERS

#### 1. LIMIT RANGE OF SECTORS TO BE TESTED (N) L?

ANSWER: TO TEST ONLY A CERTAIN RANGE OF SECTORS INSTEAD OF AN ENTIRE UNIT. ALTHOUGH THIS QUESTION IS NOT OPTIONAL, IT SHOULD BE ANSWERED 'NO' UNLESS THE ANSWER TO THE HARDWARE QUESTION '# UNITS ?' WAS 1. (LOGICAL, {Y TO TEST PARTIAL UNIT, N FOR ENTIRE UNIT}, DEFAULT N).

THE QUESTIONS WHICH FOLLOW ARE OPTIONAL, AND DEPEND ON AN AFFIRMATIVE ANSWER TO THE PREVIOUS QUESTION:

#### 2. DO YOU KNOW THE EXACT RANGE OF SECTORS TO BE TESTED (N) L?

ANSWER: THIS DECIDES WHETHER TO ASK FOR SECTOR NUMBERS OR BOARD NUMBER.

THE NEXT TWO QUESTIONS WILL BE ASKED ONLY IF QUESTION 2 IS ANSWERED YES:

#### 3. FIRST SECTOR TO TEST (O) O?

ANSWER: THE SECTOR NUMBER WHERE TESTING BEGINS (OCTAL, {0-7777 FOR ML11A, 0-37777 FOR ML11B}, DEFAULT 0).

4. LAST SECTOR TO TEST (LAST) 0?

ANSWER: THE LAST SECTOR NUMBER TO BE TESTED  
(OCTAL, {0-7777 FOR ML11A, 0-37777 FOR ML11B},  
DEFAULT 37777).

THE NEXT QUESTION WILL BE ASKED ONLY IF QUESTION 2 IS ANSWERED  
NO:

5. WHICH BOARD {1-16} SHOULD BE TESTED (1) 0?

ANSWER: THE NUMBER OF THE ONLY BOARD TO BE TESTED  
(DECIMAL, {1-16}, DEFAULT 1).

6. THE PROGRAM OPTIONS INCLUDE:

1. ADDRESS CHECK
2. PATTERN TEST
3. UNIQUE DATA CHECK
4. MARCH TEST
5. RANDOMNESS TESTS

DO YOU WANT TO DROP ANY OF THESE FROM THE EXERCISER (N) L?

ANSWER: THIS DECIDES WHETHER TO ASK FOR OPTION NUMBERS.  
(LOGICAL, {Y FOR PARTIAL EXERCISER, N FOR COMPLETE EX-  
ERCISER}, DEFAULT N).

THE FOLLOWING SET OF QUESTIONS WILL BE ASKED ONLY IF QUESTION  
6 IS ANSWERED YES:

7. DROP OPTION 1 (N) L?

8. DROP OPTION 2 (N) L?

9. DROP OPTION 3 (N) L?

10. DROP OPTION 4 (N) L?

11. DROP OPTION 5 (N) L?

12. ENABLE REFRESH MARGINING (N) L?

ANSWER: NORMAL OPERATION IS WITHOUT REFRESH MARGINING.  
(LOGICAL, {Y TO ENABLE, N TO DISABLE}, DEFAULT N).

13. DISABLE ERROR CORRECTION (N) L?

ANSWER: NORMAL OPERATION IS WITH ECC ENABLED.  
(LOGICAL, {Y TO DISABLE, N TO ENABLE}, DEFAULT N).

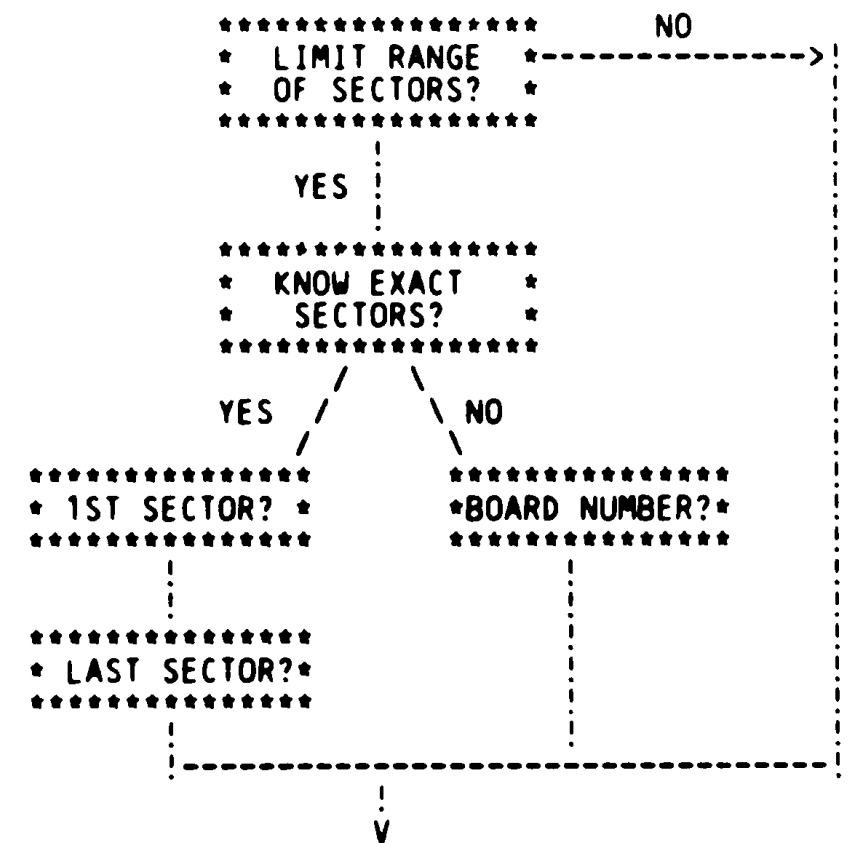
14. ENABLE END OF PASS SUMMARY PRINTOUT (N) L?

ANSWER: NORMAL OPERATION IS WITHOUT END OF PASS SUMMARY PRINTOUT.  
(LOGICAL, {Y TO ENABLE, N TO DISABLE PRINTOUT}, DEFAULT N).

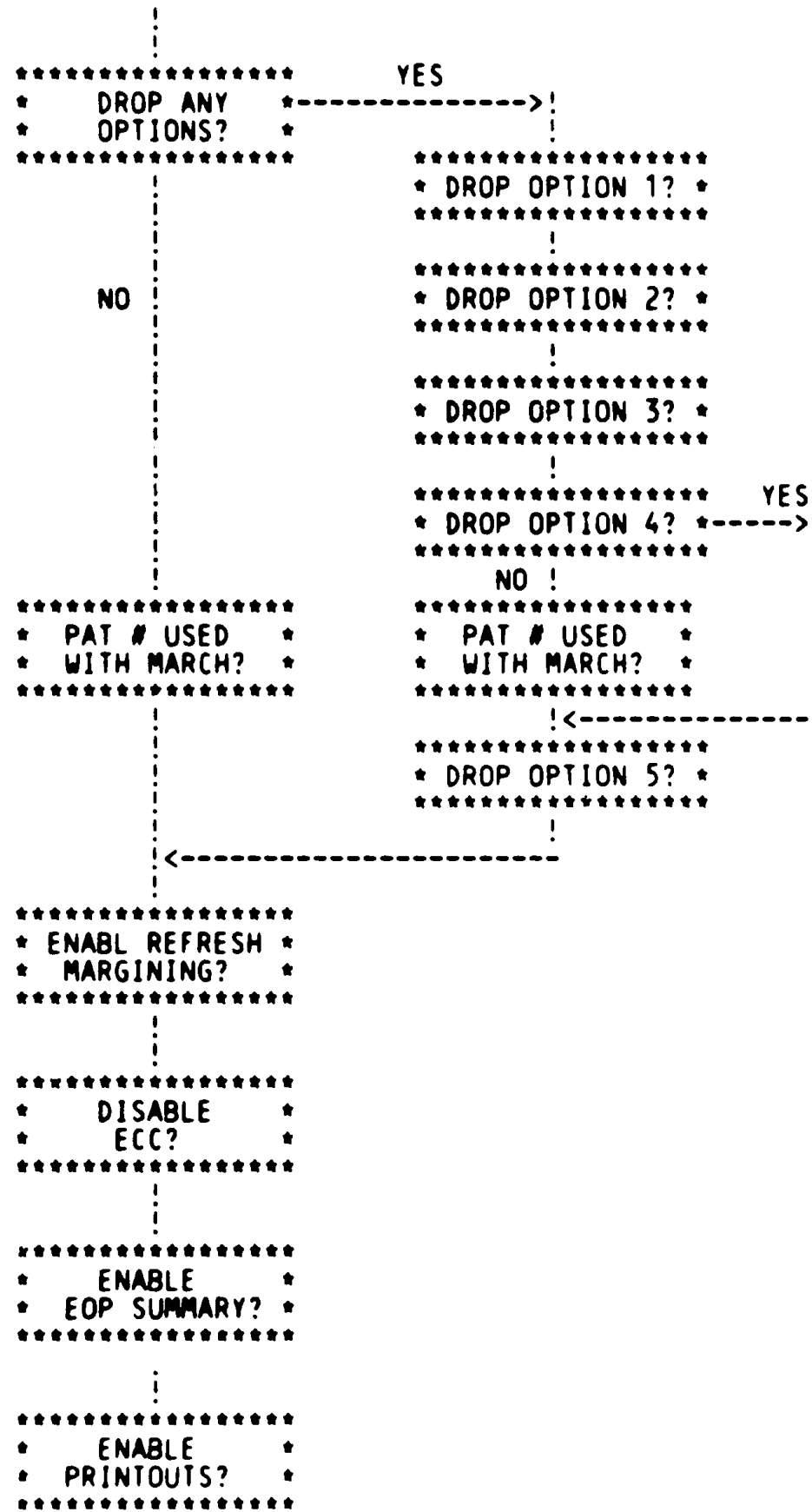
15. ENABLE ERROR PRINTOUTS (N) L?

ANSWER: NORMAL OPERATION IS WITHOUT ERROR PRINTOUTS.  
(LOGICAL, {Y TO ENABLE, N TO DISABLE PRINTOUT}, DEFAULT N).

2.5.2 SUMMARY OF SOFTWARE QUESTION SEQUENCE



(CONTINUED ON NEXT PAGE)



## 2.6 EXTENDED P-TABLE DIALOGUE

WHEN YOU ANSWER THE HARDWARE QUESTIONS, YOU ARE BUILDING ENTRIES IN A TABLE THAT DESCRIBES THE DEVICES UNDER TEST. THE SIMPLEST WAY TO BUILD THIS TABLE IS TO ANSWER ALL QUESTIONS FOR EACH UNIT TO BE TESTED. IF YOU HAVE A MULTIPLEXED DEVICE, SUCH AS A MASS STORAGE CONTROLLER WITH SEVERAL DRIVES OR A COMMUNICATION DEVICE WITH SEVERAL LINES, THE DIALOGUE BECOMES TEDIOUS BECAUSE MOST OF THE ANSWERS ARE REPETITIOUS.

SUPPOSE YOU ARE TESTING A HYPOTHETICAL DEVICE, THE XY11 WHICH IS MADE UP OF ONE CONTROL MODULE WITH 8 UNITS (SUB-DEVICES). THESE 8 UNITS, NUMBERED 0 THROUGH 7, HAVE JUST ONE HARDWARE PARAMETER THAT CAN VARY AMONG UNITS, CALLED THE Q-FACTOR.

THE 3 EXAMPLES WHICH FOLLOW SHOW DIFFERENT WAYS OF ANSWERING THE HARDWARE QUESTIONS TO ACHIEVE IDENTICAL RESULTS. IN EACH EXAMPLE, THERE ARE TO BE 8 XY11'S TESTED WHICH HAVE THE FOLLOWING Q-FACTORS:

UNIT #	Q-FACTOR
0	0
1	1
2	0
3	0
4	0
5	0
6	1
7	1

### EXAMPLE 1: ANSWER SEPARATELY FOR EACH UNIT

# UNITS (D) ? 8<CR>

UNIT 0  
 CSR ADDRESS (O) ? 160000<CR>  
 SUB-DEVICE # (O) ? 0<CR>  
 Q-FACTOR (C) 0 ? 0<CR>

UNIT 1  
 CSR ADDRESS (O) ? 160000<CR>  
 SUB-DEVICE # (O) ? 1<CR>  
 Q-FACTOR (O) 0 ? 1<CR>

UNIT 2  
 CSR ADDRESS (O) ? 160000<CR>  
 SUB-DEVICE # (O) ? 2<CR>  
 Q-FACTOR (O) 1 ? 0<CR>

UNIT 3  
 CSR ADDRESS (O) ? 160000<CR>  
 SUB-DEVICE # (O) ? 3<CR>  
 Q-FACTOR (O) 0 ? <CR>



UNIT 4  
CSR ADDRESS (0) ? 16000<CR>  
SUB-DEVICE # (0) ? 4<CR>  
Q-FACTOR (0) 0 ? <CR>

UNIT 5  
CSR ADDRESS (0) ? 160000<CR>  
SUB-DEVICE # (0) ? 5<CR>  
Q-FACTOR (0) 0 ? 1<CR>

UNIT 6  
CSR ADDRESS (0) ? 160000<CR>  
SUB-DEVICE # (0) ? 6<CR>  
Q-FACTOR (0) 1 ? 1<CR>

UNIT 7  
CSR ADDRESS (0) ? 160000<CR>  
SUB-DEVICE # (0) ? 7<CR>  
Q-FACTOR (0) 1 ? <CR>

NOTICE THAT THE DEFAULT VALUE FOR THE Q-FACTOR CHANGES  
WHEN A NON-DEFAULT RESPONSE IS GIVEN. BE CAREFUL WHEN  
SPECIFYING MULTIPLE UNITS!

AS YOU CAN SEE, THE HARDWARE PARAMETERS DO NOT DIFFER  
SIGNIFICANTLY FROM UNIT TO UNIT. THE PROCEDURE SHOWN  
IS NOT VERY EFFICIENT.

EXAMPLE 2: USE THE MULTIPLE SPECIFICATION FEATURE OF THE RUNTIME  
SERVICES TO MAKE FEWER PASSES THROUGH THE QUESTIONS.

# UNITS (0) ? 8<CR>

UNIT 0  
CSR ADDRESS (0) ? 160000<CR>  
SUB-DEVICE # (0) ? 0,1<CR>  
Q-FACTOR (0) 0 ? 0,1<CR>

UNIT 2  
CSR ADDRESS (0) ? 160000<CR>  
SUB-DEVICE # (0) ? 2-5<CR>  
Q-FACTOR (0) 1 ? 0<CR>

UNIT 6  
CSR ADDRESS (0) ? 160000<CR>  
SUB-DEVICE # (0) ? 6,7<CR>  
Q-FACTOR (0) 0 ? 1,1<CR>

AS YOU CAN SEE IN THE ABOVE DIALOGUE, RUNTIME SERVICES

WILL BUILD AS MANY ENTRIES AS IT CAN WITH THE INFORMATION GIVEN IN ANY ONE PASS THROUGH THE QUESTIONS. THE FIRST PASS BUILDS 2 ENTRIES, BECAUSE 2 SUB-DEVICES AND Q-FACTORS WERE SPECIFIED. THE DRS ASSUMES THAT THE CSR ADDRESS IS 160000 FOR BOTH UNITS. IN THE SECOND PASS, 4 ENTRIES WERE BUILT. THE "-" CONSTRUCT TELLS THE DRS TO INCREMENT THE DATA FROM THE FIRST VALUE TO THE SECOND. IN THIS CASE, "2-5" SPECIFIES SUB-DEVICES 2,3,4, AND 5. THE CSR ADDRESS AND Q-FACTOR FOR THE 4 ENTRIES ARE ASSUMED TO BE 160000 AND 0 RESPECTIVELY SINCE THEY WERE ONLY SPECIFIED ONCE. THE LAST 2 UNITS ARE SPECIFIED IN THE THIRD PASS.

EXAMPLE 3: ACCOMPLISH THE WHOLE PROCESS IN JUST ONE PASS THROUGH THE QUESTIONS.

```
# UNITS (D) ? 8<CR>
UNIT 1
CSR ADDRESS (0) ? 160000<CR>
SUB-DEVICE # (0) ? 0-7<CR>
Q-FACTOR (0) 0 ? 0,1,0,,,,1,1<CR>
```

AS YOU CAN SEE FROM THIS EXAMPLE, NULL REPLIES (COMMAS ENCLOSING A BLANK FIELD) DIRECT THE DRS TO REPEAT THE LAST REPLY.

## 2.7 QUICK START-UP PROCEDURE

### 2.7.1 TO START-UP THIS PROGRAM WHEN RUNNING UNDER XXDP+

1. BOOT XXDP+
2. GIVE THE DATE AND ANSWER THE LSI AND 50HZ QUESTIONS
3. TYPE 'R NAME', WHERE 'NAME' IS THE FILENAME OF THE .BIN OR .BIC FILE FOR THIS PROGRAM
4. TYPE "START"
5. ANSWER THE "CHANGE HW" QUESTION WITH 'Y'
6. ANSWER ALL THE HARDWARE QUESTIONS
7. ANSWER THE "CHANGE SW" QUESTION WITH 'N'

WHEN YOU FOLLOW THIS PROCEDURE YOU WILL BE USING ONLY THE DEFAULTS FOR FLAGS AND SOFTWARE PARAMETERS. THE DEFAULTS ARE DESCRIBED IN SECTIONS 2.3 AND 2.5.

### 2.7.2 TO START-UP THIS PROGRAM WHEN RUNNING UNDER ACT

THE FILE WHICH WILL BE STARTED MUST, AT SOME TIME, HAVE BEEN CREATED WITH THE 'SETUP' UTILITY PROGRAM. ASSUMING THAT THIS HAS BEEN DONE, THE START-UP PROCEDURE IS THE SAME AS THE ONE IN SECTION 2.7.1.

TO CREATE A FILE USING THE SETUP UTILITY:

1. TYPE 'R SETUP'
2. THE TARGET ENVIRONMENT IS ACT, SO TYPE 'AC'
3. TYPE THE SETUP COMMAND:  
\*SETUP OUTFILE.EXT=INFILE.EXT  
WHERE OUTFILE.EXT = THE NEWLY CREATED FILE  
AND INFILE.EXT = THE RELEASED .BIN FILE
4. YOU WILL HAVE AN OPPORTUNITY TO SET UP A PERMANENT DEFAULT TEST CONFIGURATION BY ANSWERING THE HARDWARE AND SOFTWARE QUESTIONS. ONCE THIS IS DONE, YOU WILL NO LONGER BE FORCED TO ANSWER 'Y' TO 'CHANGE HW ?' EVERY TIME YOU START THE PROGRAM. YOU WILL ALSO BE ASKED THE LSI AND 50HZ QUESTIONS HERE.
5. TYPE 'EXIT'

## 3.0 ERROR INFORMATION

### 3.1 TYPES OF ERROR MESSAGES

THERE ARE THREE LEVELS OF ERROR MESSAGES THAT MAY BE ISSUED BY A DIAGNOSTIC: GENERAL, BASIC AND EXTENDED. GENERAL ERROR MESSAGES ARE ALWAYS PRINTED UNLESS THE 'IER' FLAG IS SET (SECTION 2.3). THE GENERAL ERROR MESSAGE IS OF THE FORM:

NAME TYPE NUMBER ON UNIT NUMBER TST NUMBER PC:XXXXXX  
ERROR MESSAGE

WHERE:

NAME = DIAGNOSTIC NAME  
TYPE = ERROR TYPE (SYS FATAL, DEV FATAL, HARD OR SOFT)  
NUMBER = ERROR NUMBER  
UNIT NUMBER = 0 - N (N IS LAST UNIT IN PTABLE)  
TST NUMBER = TEST AND SUBTEST WHERE ERROR OCCURRED  
PC:XXXXXX = ADDRESS OF ERROR MESSAGE CALL

BASIC ERROR MESSAGES ARE MESSAGES THAT CONTAIN SOME ADDITIONAL

INFORMATION ABOUT THE ERROR. THESE ARE ALWAYS PRINTED UNLESS THE 'IER' OR 'IBR' FLAGS ARE SET (SECTION 2.3). THESE MESSAGES ARE PRINTED AFTER THE ASSOCIATED GENERAL MESSAGE.

EXTENDED ERROR MESSAGES CONTAIN SUPPLEMENTARY ERROR INFORMATION SUCH AS REGISTER CONTENTS OR GOOD/BAD DATA. THESE ARE ALWAYS PRINTED UNLESS THE 'IER', 'IBR' OR 'IXR' FLAGS ARE SET (SECTION 2.3). THESE MESSAGES ARE PRINTED AFTER THE ASSOCIATED GENERAL ERROR MESSAGE AND AN ASSOCIATED BASIC ERROR MESSAGES.

### 3.2 SPECIFIC ERROR MESSAGES

#### 3.2.1 MESSAGES DURING INITIALIZATION

DURING INITIALIZATION, THERE ARE 3 ERROR CONDITIONS WHICH CAN BE DETECTED AND WHICH CAUSE THE DRIVE TO BE DROPPED:

CAUSE1 = ' (NOT POWERED UP)'  
CAUSE2 = ' (NOT AN ML11 UNIT)'  
CAUSE3 = ' (OPERATOR SELECTED TEST LIMITS INCORRECTLY)'

THE 'CAUSE3' MESSAGE IS THEN FOLLOWED BY  
EITHER: '!TOP SECTOR OF XXXXXX EXCEEDS SYSTEM LIMIT OF YYYYYY'  
OR: '!LOW SECTOR OF XXXXXX EXCEEDS TOP SECTOR OF YYYYYY'

#### 3.2.2 MESSAGES DURING TESTING

AFTER INITIALIZATION, THERE ARE 4 ERROR DIAGNOSES WHICH ARE POSSIBLE:

MSG1 = '--> RUN ML11 LOGIC TEST'

THIS MESSAGE APPEARS WHEN AN ML11 SYSTEM ERROR OCCURS. THE USER IS INSTRUCTED TO RUN THE LOGIC TEST TO SEE IF THAT PROGRAM CAN ISOLATE THE REASON FOR THE FAILURE. NOTE THAT THIS MESSAGE IS NOT USED FOR DATA ERRORS.

MSG2 = '--> RUN ML11 PROM MAINTENANCE PROGRAM'

THIS MESSAGE APPEARS WHENEVER ANY OF THE FOLLOWING CONDITIONS IS TRUE:

- A) AN ECC HARD ERROR (ECH OR UNC) IS DETECTED
- B) AN ARRAY REACHES ITS HARD OR SOFT ERROR THRESHOLD
- C) DURING A PERFORMANCE SUMMARY, IF AN ARRAY HAS REACHED OR EXCEEDED ITS HARD OR SOFT ERROR THRESHOLD.

MSG3 = 'SOFT ERROR'

A SOFT ERROR IS A CORRECTABLE DATA ERROR WHICH CAN BE ELIMINATED BY REWRITING AND REREADING.

MSG4 = 'HARD ERROR'

A HARD ERROR IS A CORRECTABLE DATA ERROR WHICH PERSISTS AFTER A REWRITE AND A REREAD.

ANY WRITE COMMAND HAS 3 POSSIBLE ERROR CALLS ASSOCIATED WITH IT:

ERROR POSITION	ERROR TYPE	DIAGNOSTIC MESSAGE	CAUSE OF ERROR/ACTION
1ST	ERRDF	MSG1	ALL 6 RETRIES FAILED FOR AN ML11 SYSTEM ERROR WHICH WAS ORIGINALLY CONSIDERED TO BE NON-FATAL. DROP THE DRIVE.
2ND	ERRDF	MSG1	CONTROLLER FATAL ERROR. DROP THE DRIVE.
3RD	ERRDF	MSG1	DRIVE FATAL ERROR. DROP THE DRIVE.

ANY READ OR WRITE CHECK COMMAND HAS 8 ERROR CALLS ASSOCIATED WITH IT, 5 FOR SYSTEM ERRORS (WHICH RESULT IN DROPPING THE DRIVE) AND 3 FOR HARD AND SOFT DATA ERRORS (WHICH ARE COUNTED ON A PER ARRAY BASIS):

THE SYSTEM ERRORS:

ERROR POSITION	ERROR TYPE	DIAGNOSTIC MESSAGE	CAUSE OF ERROR/ACTION
1ST	ERRDF	MSG1	AFTER A SUCCESSFUL READ COMMAND, THE WRITE AND READ BUFFERS ARE COMPARED. IF THEY DO NOT MATCH PERFECTLY, THEN THE ECC LOGIC FAILED. DROP THE DRIVE.
2ND	ERRDF	MSG1	ALL 6 RETRIES FAILED FOR AN ML11 SYSTEM ERROR WHICH WAS ORIGINALLY CONSIDERED TO BE NON-FATAL. DROP THE DRIVE.
3RD	ERRDF	MSG1	CONTROLLER FATAL ERROR. DROP THE DRIVE.
4TH	ERRDF	MSG1	DRIVE FATAL ERROR. DROP THE DRIVE.
5TH	ERRDF	MSG2	ECC HARD ERROR DETECTED. DROP THE DRIVE.

THE DATA ERRORS:

ERROR POSITION	ERROR TYPE	DIAGNOSTIC MESSAGE	CAUSE OF ERROR/ACTION
6TH	ERRHRD	MSG4	HARD ERROR. DURING ERROR CLASSIFICATION, RETRY FAILED AGAIN (AS IF A DATA ERROR IN THE SAME BIT POSITION). UPDATE THE HARD COUNT FOR THE ASSOCIATED ARRAY AND CHECK THE NEW COUNT AGAINST THE THRESHOLD.
7TH	ERRSOFT	MSG3	SOFT ERROR. DURING ERROR CLASSIFICATION, RETRY FAILED AGAIN (AS IF A DATA ERROR IN A DIFFERENT BIT POSITION). UPDATE THE SOFT COUNT FOR THE ASSOCIATED ARRAY AND CHECK THE NEW COUNT AGAINST THE THRESHOLD.
8TH	ERRSOFT	MSG3	SOFT ERROR. DURING ERROR CLASSIFICATION, RETRY DID NOT PRODUCE ANOTHER DATA ERROR (THE RETRY MAY PASS, OR IT MAY FAIL FOR SOME NEW REASON). UPDATE THE SOFT COUNT FOR THE ASSOCIATED ARRAY AND CHECK THE NEW COUNT AGAINST THE THRESHOLD.

THERE IS A CORRESPONDENCE BETWEEN EVERY NUMBER AND THE PLACE IN THE PROGRAM WHERE THE ERROR WAS DETECTED:

A) THE MOST SIGNIFICANT DIGIT IDENTIFIES THE OPTION NUMBER.

EXCEPTION: THE COMMAND INTEGRITY ROUTINE IS THOUGHT OF AS OPTION 0, AND SO ALL ITS ERROR NUMBERS WILL APPEAR TO ONLY BE 1 OR 2 SIGNIFICANT DIGITS. TAKE THE 0 TO BE A SIGNIFICANT DIGIT (SEE EXAMPLES).

B) THE 2 LEAST SIGNIFICANT DIGITS ARE THE POSITIONAL NUMBERS TO IDENTIFY WHERE THE ERROR CALL IS LOCATED WITHIN THE OPTION.

C) FOR OPTION 5, THE SECOND MOST SIGNIFICANT DIGIT IS USED TO IDENTIFY WHICH RANDOM TEST IS RUNNING.

EXAMPLES:

- 1) ERROR NUMBER: 6 => COMMAND INTEGRITY ROUTINE (OPTION 0), ERROR 06
- 2) ERROR NUMBER: 208 => OPTION 2, ERROR 08
- 3) ERROR NUMBER: 5311 => OPTION 5, RAND3, ERROR 11

BELOW IS A SUMMARY OF THE ACTUAL ERROR CALLS IN EACH TEST OPTION:

THE 'INTEGRITY' ROUTINE:

!WRITE COMMAND:

```
!ERRDF(1,MSG1,0):  !**** INTEGRITY ROUTINE ERROR 01 ****
!ERRDF(2,MSG1,0):  !**** INTEGRITY ROUTINE ERROR 02 ****
!ERRDF(3,MSG1,0):  !**** INTEGRITY ROUTINE ERROR 03 ****
```

!READ COMMAND:

```
!ERRDF(4,MSG1,0):  !**** INTEGRITY ROUTINE ERROR 04 ****
!ERRDF(5,MSG1,0):  !**** INTEGRITY ROUTINE ERROR 05 ****
!ERRDF(6,MSG1,0):  !**** INTEGRITY ROUTINE ERROR 06 ****
!ERRDF(7,MSG1,0):  !**** INTEGRITY ROUTINE ERROR 07 ****
!ERRDF(8,MSG2,0):  !**** INTEGRITY ROUTINE ERROR 08 ****
!ERRHRD(9,MSG4,0): !**** INTEGRITY ROUTINE ERROR 09 ****
!ERRSOFT(10,MSG3,0):!**** INTEGRITY ROUTINE ERROR 10 ****
!ERRSOFT(11,MSG3,1):!**** INTEGRITY ROUTINE ERROR 10 ****
```

!WRITE CHECK COMMAND:

```
!ERRDF(12,MSG1,0): !**** INTEGRITY ROUTINE ERROR 12 ****
!ERRDF(13,MSG1,0): !**** INTEGRITY ROUTINE ERROR 13 ****
!ERRDF(14,MSG1,0): !**** INTEGRITY ROUTINE ERROR 14 ****
!ERRDF(15,MSG2,0): !**** INTEGRITY ROUTINE ERROR 15 ****
!ERRHRD(16,MSG4,0):!**** INTEGRITY ROUTINE ERROR 16 ****
!ERRSOFT(17,MSG3,8):!**** INTEGRITY ROUTINE ERROR 17 ****
!ERRSOFT(18,MSG3,1):!**** INTEGRITY ROUTINE ERROR 18 ****
```

OPTION 1:

!WRITE COMMAND:

```
!ERRDF(101,MSG1,0): !**** OPTION 1 ERROR 01 ****
!ERRDF(102,MSG1,0): !**** OPTION 1 ERROR 02 ****
!ERRDF(103,MSG1,0): !**** OPTION 1 ERROR 03 ****
```

!CHECK OR READ:

```
!ERRDF(104,MSG1,0): !**** OPTION 1 ERROR 04 ****
!ERRDF(105,MSG1,0): !**** OPTION 1 ERROR 05 ****
!ERRDF(106,MSG1,0): !**** OPTION 1 ERROR 06 ****
!ERRDF(107,MSG1,0): !**** OPTION 1 ERROR 07 ****
!ERRDF(108,MSG2,0): !**** OPTION 1 ERROR 08 ****
!ERRHRD(109,MSG4,0): !**** OPTION 1 ERROR 09 ****
!ERRSOFT(110,MSG3,0): !**** OPTION 1 ERROR 10 ****
!ERRSOFT(111,MSG3,0): !**** OPTION 1 ERROR 11 ****
```

OPTION 2:

!WRITE COMMAND:

!ERRDF(201,MSG1,0); !\*\*\*\* OPTION 2 ERROR 01 \*\*\*\*  
!ERRDF(202,MSG1,0); !\*\*\*\* OPTION 2 ERROR 02 \*\*\*\*  
!ERRDF(203,MSG1,0); !\*\*\*\* OPTION 2 ERROR 03 \*\*\*\*

!CHECK OR READ:

!ERRDF(204,MSG1,0); !\*\*\*\* OPTION 2 ERROR 04 \*\*\*\*  
!ERRDF(205,MSG1,0); !\*\*\*\* OPTION 2 ERROR 05 \*\*\*\*  
!ERRDF(206,MSG1,0); !\*\*\*\* OPTION 2 ERROR 06 \*\*\*\*  
!ERRDF(207,MSG1,0); !\*\*\*\* OPTION 2 ERROR 07 \*\*\*\*  
!ERRDF(208,MSG2,0); !\*\*\*\* OPTION 2 ERROR 08 \*\*\*\*  
!ERRHRD(209,MSG4,0); !\*\*\*\* OPTION 2 ERROR 09 \*\*\*\*  
!ERRSOFT(210,MSG3,0); !\*\*\*\* OPTION 2 ERROR 10 \*\*\*\*  
!ERRSOFT(211,MSG3,1); !\*\*\*\* OPTION 2 ERROR 10 \*\*\*\*

!LOOP CHECK OR READ:

!ERRDF(212,MSG1,0); !\*\*\*\* OPTION 2 ERROR 12 \*\*\*\*  
!ERRDF(213,MSG1,0); !\*\*\*\* OPTION 2 ERROR 13 \*\*\*\*  
!ERRDF(214,MSG1,0); !\*\*\*\* OPTION 2 ERROR 14 \*\*\*\*  
!ERRDF(215,MSG1,0); !\*\*\*\* OPTION 2 ERROR 15 \*\*\*\*  
!ERRDF(216,MSG2,0); !\*\*\*\* OPTION 2 ERROR 16 \*\*\*\*  
!ERRHRD(217,MSG4,0); !\*\*\*\* OPTION 2 ERROR 17 \*\*\*\*  
!ERRSOFT(218,MSG3,0); !\*\*\*\* OPTION 2 ERROR 18 \*\*\*\*  
!ERRSOFT(219,MSG3,0); !\*\*\*\* OPTION 2 ERROR 19 \*\*\*\*

OPTION 3:

!WRITE COMMAND:

!ERRDF(301,MSG1,0); !\*\*\*\* OPTION 3 ERROR 01 \*\*\*\*  
!ERRDF(302,MSG1,0); !\*\*\*\* OPTION 3 ERROR 02 \*\*\*\*  
!ERRDF(303,MSG1,0); !\*\*\*\* OPTION 3 ERROR 03 \*\*\*\*

!CHECK OR READ:

!ERRDF(304,MSG1,0); !\*\*\*\* OPTION 3 ERROR 04 \*\*\*\*  
!ERRDF(305,MSG1,0); !\*\*\*\* OPTION 3 ERROR 05 \*\*\*\*  
!ERRDF(306,MSG1,0); !\*\*\*\* OPTION 3 ERROR 06 \*\*\*\*  
!ERRDF(307,MSG1,0); !\*\*\*\* OPTION 3 ERROR 07 \*\*\*\*  
!ERRDF(308,MSG2,0); !\*\*\*\* OPTION 3 ERROR 08 \*\*\*\*  
!ERRHRD(309,MSG4,0); !\*\*\*\* OPTION 3 ERROR 09 \*\*\*\*  
!ERRSOFT(310,MSG3,0); !\*\*\*\* OPTION 3 ERROR 10 \*\*\*\*  
!ERRSOFT(311,MSG3,0); !\*\*\*\* OPTION 3 ERROR 11 \*\*\*\*



```
! OPTION 4:

!MARCHING UP:

!WRITE DATA:

!ERRDF(401,MSG1,0): !**** OPTION 4 ERROR 01 ****
!ERRDF(402,MSG1,0): !**** OPTION 4 ERROR 02 ****
!ERRDF(403,MSG1,0): !**** OPTION 4 ERROR 03 ****

!MARCHING UP:

!CHECK OR READ DATA:

!ERRDF(404,MSG1,0): !**** OPTION 4 ERROR 04 ****
!ERRDF(405,MSG1,0): !**** OPTION 4 ERROR 05 ****
!ERRDF(406,MSG1,0): !**** OPTION 4 ERROR 06 ****
!ERRDF(407,MSG1,0): !**** OPTION 4 ERROR 07 ****
!ERRDF(408,MSG2,0): !**** OPTION 4 ERROR 08 ****
!ERRHRD(409,MSG4,0): !**** OPTION 4 ERROR 09 ****
!ERRSOFT(410,MSG3,0): !**** OPTION 4 ERROR 10 ****
!ERRSOFT(411,MSG3,0): !**** OPTION 4 ERROR 11 ****

!WRITE COMP:

!ERRDF(412,MSG1,0): !**** OPTION 4 ERROR 12 ****
!ERRDF(413,MSG1,0): !**** OPTION 4 ERROR 13 ****
!ERRDF(414,MSG1,0): !**** OPTION 4 ERROR 14 ****

!MARCHING DOWN:

!CHECK OR READ COMP:

!ERRDF(415,MSG1,0): !**** OPTION 4 ERROR 15 ****
!ERRDF(416,MSG1,0): !**** OPTION 4 ERROR 16 ****
!ERRDF(417,MSG1,0): !**** OPTION 4 ERROR 17 ****
!ERRDF(418,MSG1,0): !**** OPTION 4 ERROR 18 ****
!ERRDF(419,MSG2,0): !**** OPTION 4 ERROR 19 ****
!ERRHRD(420,MSG4,0): !**** OPTION 4 ERROR 20 ****
!ERRSOFT(421,MSG3,0): !**** OPTION 4 ERROR 21 ****
!ERRSOFT(422,MSG3,0): !**** OPTION 4 ERROR 22 ****

!WRITE DATA:

!ERRDF(423,MSG1,0): !**** OPTION 4 ERROR 23 ****
!ERRDF(424,MSG1,0): !**** OPTION 4 ERROR 24 ****
!ERRDF(425,MSG1,0): !**** OPTION 4 ERROR 25 ****
```

!MARCHING UP:

!CHECK OR READ DATA:

```
!ERRDF(426,MSG1,0):  !**** OPTION 4 ERROR 26 ****
!ERRDF(427,MSG1,0):  !**** OPTION 4 ERROR 27 ****
!ERRDF(428,MSG1,0):  !**** OPTION 4 ERROR 28 ****
!ERRDF(429,MSG1,0):  !**** OPTION 4 ERROR 29 ****
!ERRDF(430,MSG2,0):  !**** OPTION 4 ERROR 30 ****
!ERRHRD(431,MSG4,0): !**** OPTION 4 ERROR 31 ****
!ERRSOFT(432,MSG3,0): !**** OPTION 4 ERROR 32 ****
!ERRSOFT(433,MSG3,0): !**** OPTION 4 ERROR 33 ****
```

! OPTION 5:

!'RAND1' ROUTINE:

!WRITE COMMAND:

```
!ERRDF(5101,MSG1,0): !**** OPTION 5, RAND1 ERROR 01 ****
!ERRDF(5102,MSG1,0): !**** OPTION 5, RAND1 ERROR 02 ****
!ERRDF(5103,MSG1,0): !**** OPTION 5, RAND1 ERROR 03 ****
```

!CHECK OR READ:

```
!ERRDF(5104,MSG1,0): !**** OPTION 5, RAND1 ERROR 04 ****
!ERRDF(5105,MSG1,0): !**** OPTION 5, RAND1 ERROR 05 ****
!ERRDF(5106,MSG1,0): !**** OPTION 5, RAND1 ERROR 06 ****
!ERRDF(5107,MSG1,0): !**** OPTION 5, RAND1 ERROR 07 ****
!ERRDF(5108,MSG2,0): !**** OPTION 5, RAND1 ERROR 08 ****
!ERRHRD(5109,MSG4,0): !**** OPTION 5, RAND1 ERROR 09 ****
!ERRSOFT(5110,MSG3,0): !**** OPTION 5, RAND1 ERROR 10 ****
!ERRSOFT(5111,MSG3,0): !**** OPTION 5, RAND1 ERROR 11 ****
```

!'RAND2' ROUTINE:

!WRITE COMMAND:

```
!ERRDF(5201,MSG1,0): !**** OPTION 5, RAND2 ERROR 01 ****
!ERRDF(5202,MSG1,0): !**** OPTION 5, RAND2 ERROR 02 ****
!ERRDF(5203,MSG1,0): !**** OPTION 5, RAND2 ERROR 03 ****
```

!CHECK OR READ:

```
!ERRDF(5204,MSG1,0): !**** OPTION 5, RAND2 ERROR 04 ****
!ERRDF(5205,MSG1,0): !**** OPTION 5, RAND2 ERROR 05 ****
!ERRDF(5206,MSG1,0): !**** OPTION 5, RAND2 ERROR 06 ****
!ERRDF(5207,MSG1,0): !**** OPTION 5, RAND2 ERROR 07 ****
!ERRDF(5208,MSG2,0): !**** OPTION 5, RAND2 ERROR 08 ****
!ERRHRD(5209,MSG4,0): !**** OPTION 5, RAND2 ERROR 09 ****
!ERRSOFT(5210,MSG3,0): !**** OPTION 5, RAND2 ERROR 10 ****
!ERRSOFT(5211,MSG3,0): !**** OPTION 5, RAND2 ERROR 11 ****
```

!'RAND3' ROUTINE:

!WRITE COMMAND:

!ERRDF(5301,MSG1,0); !\*\*\*\* OPTION 5, RAND3 ERROR 01 \*\*\*\*  
!ERRDF(5302,MSG1,0); !\*\*\*\* OPTION 5, RAND3 ERROR 02 \*\*\*\*  
!ERRDF(5303,MSG1,0); !\*\*\*\* OPTION 5, RAND3 ERROR 03 \*\*\*\*

!CHECK OR READ:

!ERRDF(5304,MSG1,0); !\*\*\*\* OPTION 5, RAND3 ERROR 04 \*\*\*\*  
!ERRDF(5305,MSG1,0); !\*\*\*\* OPTION 5, RAND3 ERROR 05 \*\*\*\*  
!ERRDF(5306,MSG1,0); !\*\*\*\* OPTION 5, RAND3 ERROR 06 \*\*\*\*  
!ERRDF(5307,MSG1,0); !\*\*\*\* OPTION 5, RAND3 ERROR 07 \*\*\*\*  
!ERRDF(5308,MSG2,0); !\*\*\*\* OPTION 5, RAND3 ERROR 08 \*\*\*\*  
!ERRHRD(5309,MSG4,0); !\*\*\*\* OPTION 5, RAND3 ERROR 09 \*\*\*\*  
!ERRSOFT(5310,MSG3,0); !\*\*\*\* OPTION 5, RAND3 ERROR 10 \*\*\*\*  
!ERRSOFT(5311,MSG3,0); !\*\*\*\* OPTION 5, RAND3 ERROR 11 \*\*\*\*

!'RAND4' ROUTINE:

!WRITE COMMAND:

!ERRDF(5401,MSG1,0); !\*\*\*\* OPTION 5, RAND4 ERROR 01 \*\*\*\*  
!ERRDF(5402,MSG1,0); !\*\*\*\* OPTION 5, RAND4 ERROR 02 \*\*\*\*  
!ERRDF(5403,MSG1,0); !\*\*\*\* OPTION 5, RAND4 ERROR 03 \*\*\*\*

!CHECK OR READ:

!ERRDF(5404,MSG1,0); !\*\*\*\* OPTION 5, RAND4 ERROR 04 \*\*\*\*  
!ERRDF(5405,MSG1,0); !\*\*\*\* OPTION 5, RAND4 ERROR 05 \*\*\*\*  
!ERRDF(5406,MSG1,0); !\*\*\*\* OPTION 5, RAND4 ERROR 06 \*\*\*\*  
!ERRDF(5407,MSG1,0); !\*\*\*\* OPTION 5, RAND4 ERROR 07 \*\*\*\*  
!ERRDF(5408,MSG2,0); !\*\*\*\* OPTION 5, RAND4 ERROR 08 \*\*\*\*  
!ERRHRD(5409,MSG4,0); !\*\*\*\* OPTION 5, RAND4 ERROR 09 \*\*\*\*  
!ERRSOFT(5410,MSG3,0); !\*\*\*\* OPTION 5, RAND4 ERROR 10 \*\*\*\*  
!ERRSOFT(5411,MSG3,0); !\*\*\*\* OPTION 5, RAND4 ERROR 11 \*\*\*\*

#### 4.0 PERFORMANCE AND PROGRESS REPORTS

AT THE END OF EACH PASS, THE PASS COUNT IS GIVEN ALONG WITH A SUMMARY WHICH SHOWS THE DIAGNOSTIC'S PERFORMANCE SINCE IT WAS STARTED. THE SAME PROGRESS REPORT CAN BE OBTAINED BY STOPPING THE PROGRAM'S EXECUTION (VIA A ^C) AND BY ISSUING THE 'PRINT' COMMAND. A TYPICAL REPORT FOR 2 DRIVES IS SHOWN BELOW:

##### PERFORMANCE SUMMARY

###### NUMBER OF MBYTES TRANSFERRED:

1028 MBYTES WRITTEN  
250 MBYTES READ  
1145 MBYTES WRITE CHECKED

LOGICAL UNIT: 0 DRIVE: 1 SERIAL #: 1234

SOFT ERROR COUNT: 9  
ARRAY 3: 9  
HARD ERROR COUNT: 11  
ARRAY 0: 6  
ARRAY 10: 2  
ARRAY 15: 3  
TRANSFER RETRIES: 0

LOGICAL UNIT: 1 DRIVE: 1 SERIAL #: 9876  
DRIVE DROPPED (CONTROLLER FATAL ERROR)

SOFT ERROR COUNT: 100  
ARRAY 1: 9  
ARRAY 13: 10 --> RUN ML11 PROM MAINTENANCE PROGRAM  
ARRAY 14: 1  
ARRAY 15: 80 --> RUN ML11 PROM MAINTENANCE PROGRAM  
HARD ERROR COUNT: 2  
ARRAY 14: 1  
ARRAY 15: 1  
TRANSFER RETRIES: 0

#### 5.0 DEVICE INFORMATION TABLES

AT THE START OF THE PROGRAM, AN AUTOMATIC CHECK OF THE SYSTEM CONFIGURATION IS MADE AND DEVICE INFORMATION SIMILAR TO THE FOLLOWING IS PRINTED FOR EACH UNIT.

LOGICAL UNIT: 0 DRIVE: 0 SERIAL #: 1234  
ML11-A SECTORS UNDER TEST: 000000 TO 017777  
TRANSFER RATE: 1 MBYTES/SECOND CSR ADDRESS: 176400

#### 6.0 TEST SUMMARIES

THERE IS JUST ONE HARDWARE TEST IN THE EXERCISER, AND ITS PURPOSE IS TO ACT AS A SCHEDULER TO CALL THE SUBROUTINES WHICH ACTUALLY PERFORM ALL OF THE TEST CODE. THE SUBROUTINES, CALLED OPTIONS, ARE DESCRIBED BELOW:

## 6.1 OPTION 1 (OPT1)

PURPOSE: TO CHECK ADDRESSES USING DATA = SECTOR NUMBER.  
TRANSFERS ARE 4K WORDS IN LENGTH, AND ALL SECTORS  
ARE TESTED.

THE CODE FOR 'OPT1' IN BRIEF:

```
BEGIN 1 (START OF ROUTINE)
SAY ROUTINE IS RUNNING
INCR COMPLEMENT FLAG FROM 0 TO 1
: BEGIN 2 (START OF COMPLEMENT FLAG SELECTION LOOP)
: INCR LOGICAL UNIT FROM 0 TO LAST
: : BEGIN 3 (START OF LOGICAL UNIT SELECTION LOOP)
: : TESTLOOP:
: : : BEGIN 4 (START OF LOOP THAT COMPLETELY TESTS ONE UNIT)
: : : IF UNIT IS ACTIVE
: : : THEN
: : : : BEGIN 5 (START OF TEST FOR AN ACTIVE UNIT)
: : : : INITIALIZE WRITE AND READ BUFFER POINTERS
: : : : SECTOR = LOWEST
: : : : WHILE SECTOR LEQ HIGHEST DO
: : : : : BEGIN 6 (START OF SECTOR SELECTION LOOP)
: : : : : GET WRDCNT
: : : : : GENERATE THE PATTERN
: : : : : SET UP BUFFER POINTERS BEFORE TRANSFER
: : : : : WRITE
: : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
: : : : : CHOOSE WHETHER TO WRITE CHECK OR READ
: : : : : DO THE WRITE CHECK OR READ
: : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
: : : : : CHANGE BUFFER POINTERS AFTER TRANSFER
: : : : : UPDATE SECTOR NUMBER BY NUMBER OF SECTORS IN PREVIOUS TRANSFER
: : : : : END 6 (END OF SECTOR SELECTION LOOP)
: : : : END 5 (END OF TEST FOR AN ACTIVE UNIT)
: : : END 4 (END OF TESTLOOP)
: : END 3 (END OF LOGICAL UNIT SELECTION LOOP)
: END 2 (END OF COMPLEMENT FLAG SELECTION LOOP)
RETURN
END 1 (END OF ROUTINE)
```

## 6.2 OPTION 2 (OPT2)

PURPOSE: TO CHECK ON DATA RELIABILITY USING THE PATTERNS FROM  
THE PATTERN TABLE.

THE CODE FOR 'OPT2' IN BRIEF:

```
BEGIN 1 (START OF ROUTINE)
SAY THE ROUTINE IS RUNNING
CHOOSE A MAXIMUM PATTERN NUMBER
INCR COUNT FROM 1 TO (2*MAX)
: BEGIN 2 (START OF PATTERN SELECTION LOOP)
: GENERATE THE PATTERN
: INCR LUN FROM 0 TO LAST
: : BEGIN 3 (START OF LOGICAL UNIT SELECTION LOOP)
: : TESTLOOP:
: : : BEGIN 4 (START OF LOOP THAT COMPLETELY TESTS 1 UNIT)
: : : IF UNIT IS ACTIVE
: : : THEN
: : : : BEGIN 5 (START OF TEST FOR AN ACTIVE UNIT)
: : : : INITIALIZE THE WRITE AND READ BUFFER POINTERS
: : : : SECTOR = LOWEST
: : : : WHILE SECTOR LEQ HIGHEST DO
: : : : : BEGIN 6 (START OF SECTOR SELECTION LOOP)
: : : : : GET_WRCNT
: : : : : SET_UP BUFFER POINTERS BEFORE TRANSFER
: : : : : WRITE
: : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
: : : : : CHOOSE WHETHER TO WRITE CHECK OR READ
: : : : : DO THE WRITE CHECK OR READ
: : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
: : : : : CHANGE BUFFER POINTERS AFTER TRANSFER
: : : : : UPDATE SECTOR NUMBER BY NUMBER OF SECTORS IN PREVIOUS TRANSFER
: : : : : END 6 (END OF SECTOR SELECTION LOOP)
: : : : : END 5 (END OF TEST FOR AN ACTIVE UNIT)
: : : : END 4 (END OF TESTLOOP)
: : : END 3 (END OF LOGICAL UNIT SELECTION LOOP)
: IF NOT THE QUICK VERIFY PASS THEN 'LOOP READ' (DESCRIBED BELOW)
: END 2 (END OF PATTERN SELECTION LOOP)
RETURN
END 1 (END OF ROUTINE)
```

THE 'LOOP READ' CODE IN BRIEF:

```
BEGIN 11 (START OF LOOP READING SECTION)
INCR LUN FROM 0 TO LAST
: BEGIN 12 (START OF LOGICAL UNIT SELECTION LOOP)
: TESTLOOP2:
: : BEGIN 13 (START OF LOOP THAT COMPLETELY TESTS 1 UNIT)
: : IF UNIT IS ACTIVE
: : THEN
: : : BEGIN 14 (START OF TEST FOR AN ACTIVE UNIT)
: : : INITIALIZE THE WRITE AND READ BUFFER POINTERS
: : : SFCTOR = LOWEST
: : : WHILE SECTOR LEQ HIGHEST DO
: : : : BEGIN 15 (START OF SECTOR SELECTION LOOP)
: : : : GET_WRCNT
: : : : SET UP BUFFER POINTERS BEFORE TRANSFER
: : : : CHOOSE WHETHER TO WRITE CHECK OR READ
: : : : INCR KOUNT FROM 1 TO TIMES
: : : : : BEGIN 16 (START OF COUNTING LOOP FOR LOOP READING)
: : : : : DO THE WRITE CHECK OR READ
: : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP2)
: : : : : END 16 (END OF COUNTING LOOP FOR LOOP READING)
: : : : CHANGE BUFFER POINTERS AFTER TRANSFER
: : : : UPDATE SECTOR NUMBER BY # SECTORS IN PREVIOUS TRANSFER
: : : : END 15 (END OF SECTOR SELECTION LOOP)
: : : END 14 (END OF TEST FOR AN ACTIVE UNIT)
: : END 13 (END OF TESTLOOP2)
: END 12 (END OF LOGICAL UNIT SELECTION LOOP)
END 11 (END OF LOOP READING SECTION)
```

### 6.3 OPTION 3 (OPT3)

PURPOSE: TO DO A UNIQUE DATA CHECK ON ALL AVAILABLE UNITS.

THE CODE FOR 'OPT3' IN BRIEF:

```
BEGIN 1 (START OF ROUTINE)
SAY ROUTINE IS RUNNING
INCR COMPLEMENT FLAG FROM 0 TO 1
: BEGIN 2 (START OF COMPLEMENT FLAG SELECTION LOOP)
: GENERATE THE PATTERN
: INCR IUN FROM 0 TO LAST
: : BEGIN 3 (START OF LOGICAL UNIT SELECTION LOOP)
: : TESTLOOP:
: : : BEGIN 4 (START OF LOOP THAT COMPLETELY TESTS 1 UNIT)
: : : IF UNIT IS ACTIVE
: : : THEN
: : : : BEGIN 5 (START OF TEST FOR AN ACTIVE UNIT)
: : : : INITIALIZE THE WRITE AND READ BUFFER POINTERS
: : : : SECTOR = LOWEST
: : : : WHILE SECTOR LEQ HIGHEST DO
: : : : : BEGIN 6 (START OF SECTOR SELECTION LOOP)
: : : : : GET_WRCNT
: : : : : SET_UP BUFFER POINTERS BEFORE TRANSFER
: : : : : WRITE
: : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
: : : : : CHOOSE WHETHER TO WRITE CHECK OR READ
: : : : : DO THE WRITE CHECK OR READ
: : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
: : : : : CHANGE BUFFER POINTERS AFTER TRANSFER
: : : : : UPDATE SECTOR NUMBER BY NUMBER OF SECTORS IN PREVIOUS TRANSFER
: : : : : END 6 (END OF SECTOR SELECTION LOOP)
: : : : : END 5 (END OF TEST FOR AN ACTIVE UNIT)
: : : : END 4 (END OF TESTLOOP)
: : : END 3 (END OF LOGICAL UNIT SELECTION LOOP)
: END 2 (END OF COMPLEMENT FLAG SELECTION LOOP)
RETURN
END 1 (END OF ROUTINE)
```



6.4 OPTION 4 (OPT4)

PURPOSE: TO LOOK FOR INTERACTIONS BETWEEN SECTORS  
 USING A MARCH TEST.

THE CODE FOR 'OPT4' IN BRIEF:

```

BEGIN 1 (START OF ROUTINE)
  SAY ROUTINE IS RUNNING
  WORD COUNT = 256
  GENERATE A BUFFER OF DATA
  GENERATE A BUFFER OF COMP
  INITIALIZE POINTERS TO 4 BUFFERS FOR WRITE/READ DATA/COMP
  INCR LUN FROM 0 TO LAST
  : BEGIN 2 (START OF LOGICAL UNIT SELECTION LOOP)
  : TESTLOOP:
  : : BEGIN 3 (START OF LOOP THAT COMPLETELY TESTS 1 UNIT)
  : : IF UNIT IS ACTIVE
  : : THEN
  : : : BEGIN 4 (START OF TEST FOR AN ACTIVE UNIT)
  : : : INCR SECTOR FROM LOWEST TO HIGHEST
  : : : : BEGIN 4A (START OF 1ST SECTOR SELECTION LOOP)
  : : : : WRITE 'DATA'
  : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
  : : : : END 4A (END OF 1ST SECTOR SELECTION LOOP)
  : : : : CHOOSE WHETHER TO WRITE CHECK OR READ 'DATA'
  : : : : INCR SECTOR FROM LOWEST TO HIGHEST
  : : : : BEGIN 4B (START OF 2ND SECTOR SELECTION LOOP)
  : : : : DO THE WRITE CHECK OR READ OF 'DATA'
  : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
  : : : : WRITE 'COMP'
  : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
  : : : : END 4B (END OF 2ND SECTOR SELECTION LOOP)
  : : : : CHOOSE WHETHER TO WRITE CHECK OR READ 'COMP'
  : : : : DECR SECTOR FROM HIGHEST TO LOWEST
  : : : : BEGIN 4C (START OF 3RD SECTOR SELECTION LOOP)
  : : : : DO THE WRITE CHECK OR READ OF 'COMP'
  : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
  : : : : WRITE DATA
  : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
  : : : : END 4C (END OF 3RD SECTOR SELECTION LOOP)
  : : : : CHOOSE WHETHER TO WRITE CHECK OR READ 'DATA'
  : : : : INCR SECTOR FROM LOWEST TO HIGHEST
  : : : : BEGIN 4D (START OF 4TH SECTOR SELECTION LOOP)
  : : : : DO THE WRITE CHECK OR READ OF 'DATA'
  : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
  : : : : END 4D (END OF 4TH SECTOR SELECTION LOOP)
  : : : END 4 (END OF TEST FOR AN ACTIVE UNIT)
  : : END 3 (END OF TESTLOOP)
  : END 2 (END OF LOGICAL UNIT SELECTION LOOP)
  RETURN
END 1 (END OF ROUTINE)
  
```

## 6.5 OPTION 5 (OPT5)

PURPOSE: TO EXERCISE THE ML11 SYSTEMS UNDER TEST IN A RANDOM MANNER, SO AS TO SIMULATE THE FLEXIBILITY OF TESTING THAT WOULD BE DONE BY AN OPERATING SYSTEM.

THERE ARE 4 RANDOM TESTS WHICH ARE CALLED BY 'OPT5' TO ACCOMPLISH ALL TESTING. IT IS THE RESPONSIBILITY OF THIS ROUTINE TO DECIDE HOW MANY TIMES THOSE 4 RANDOM TESTS WILL BE EXECUTED. REFER TO 'RAND1' TO 'RAND4' BELOW FOR MORE INFORMATION.

### 6.5.1 RAND1 ROUTINE

PURPOSE: TO TEST USING RANDOM DATA

THE CODE FOR 'RAND1' IN BRIEF:

```
BEGIN 1 (START OF ROUTINE)
SAY ROUTINE IS RUNNING
INCR COUNT FROM 1 TO REPEAT
: BEGIN 2 (START OF REPEAT LOOP FOR THE ROUTINE)
: GENERATE THE RANDOM PATTERN
: INCR LUN FROM 0 TO LAST
: : BEGIN 3 (START OF LOGICAL UNIT SELECTION LOOP)
: : TESTLOOP:
: : : BEGIN 4 (START OF LOOP THAT COMPLETELY TESTS 1 UNIT)
: : : IF UNIT IS ACTIVE
: : : THEN
: : : : BEGIN 5 (START OF TEST FOR AN ACTIVE UNIT)
: : : : INITIALIZE THE WRITE AND READ BUFFER POINTERS
: : : : SECTOR = LOWEST
: : : : WHILE SECTOR LEQ HIGHEST DO
: : : : : BEGIN 6 (START OF SECTOR SELECTION LOOP)
: : : : : GET WRDCNT
: : : : : SET-UP BUFFER POINTERS BEFORE TRANSFER
: : : : : WRITE
: : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
: : : : : CHOOSE WHETHER TO WRITE CHECK OR READ
: : : : : DO THE WRITE CHECK OR READ
: : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
: : : : : CHANGE BUFFER POINTERS AFTER TRANSFER
: : : : : UPDATE SECTOR NUMBER BY NUMBER OF SECTORS IN PREVIOUS TRANSFER
: : : : : END 6 (END OF SECTOR SELECTION LOOP)
: : : : : END 5 (END OF TEST FOR AN ACTIVE UNIT)
: : : : END 4 (END OF TESTLOOP)
: : : END 3 (END OF LOGICAL UNIT SELECTION LOOP)
: END 2 (END OF REPEAT LOOP FOR THIS ROUTINE)
RETURN
END 1 (END OF ROUTINE)
```

### 6.5.2 RAND2 ROUTINE

PURPOSE: TO TEST USING RANDOM DATA AND WORD COUNTS

THE CODE FOR 'RAND2' IN BRIEF:

```
BEGIN 1 (START OF ROUTINE)
SAY ROUTINE IS RUNNING
INCR COUNT FROM 1 TO REPEAT
: BEGIN 2 (START OF REPEAT LOOP FOR THE ROUTINE)
: INCR LUN FROM 0 TO LAST
: : BEGIN 3 (START OF LOGICAL UNIT SELECTION LOOP)
: : GENERATE THE RANDOM PATTERN
: : TESTLOOP:
: : : BEGIN 4 (START OF LOOP THAT COMPLETELY TESTS 1 UNIT)
: : : IF UNIT IS ACTIVE
: : : THEN
: : : : BEGIN 5 (START OF TEST FOR AN ACTIVE UNIT)
: : : : SECTOR = LOWEST
: : : : INITIALIZE THE WRITE AND READ BUFFER POINTERS
: : : : WHILE SECTOR LEQ HIGHEST DO
: : : : : BEGIN 6 (START OF A PASS THROUGH ALL SECTORS)
: : : : : CHOOSE A RANDOM WORD COUNT
: : : : : SET UP BUFFER POINTERS BEFORE TRANSFER
: : : : : CALCULATE NEXT STARTING SECTOR (BASED ON WORD COUNT)
: : : : : IF NEXT STARTING SECTOR GTR HIGHEST
: : : : : : THEN ADJUST THE WORD COUNT AND NEXT SECTOR SO THEY FIT
: : : : : : : WITHIN THE TESTABLE SECTOR LIMITS
: : : : : WRITE
: : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
: : : : : CHOOSE WHETHER TO WRITE CHECK OR READ
: : : : : DO THE WRITE CHECK OR READ
: : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
: : : : : SECTOR = THE CALCULATED NEXT STARTING SECTOR
: : : : : CHANGE BUFFER POINTERS AFTER TRANSFER
: : : : : END 6 (END OF A PASS THROUGH ALL SECTORS)
: : : : : END 5 (END OF TEST FOR AN ACTIVE UNIT)
: : : : : END 4 (END TESTLOOP)
: : : : : END 3 (END OF LOGICAL UNIT SELECTION LOOP)
: : : : : END 2 (END OF REPEAT LOOP FOR THIS ROUTINE)
RETURN
END 1 (END OF ROUTINE)
```

### 6.5.3 RAND3 ROUTINE

PURPOSE: TO TEST USING RANDOM DATA, WORD COUNTS AND SECTORS

THE CODE FOR 'RAND3' IN BRIEF:

```
BEGIN 1 (START OF ROUTINE)
SAY ROUTINE IS RUNNING
INCR COUNT FROM 1 TO REPEAT
: BEGIN 2 (START OF REPEAT LOOP FOR THIS ROUTINE)
: GENERATE THE RANDOM PATTERN
: INCR IUN FROM 0 TO LAST
: : BEGIN 3 (START OF LOGICAL UNIT SELECTION LOOP)
: : INITIALIZE BUFFER POINTERS
: : TESTLOOP:
: : : BEGIN 4 (START OF LOOP THAT COMPLETELY TESTS 1 UNIT)
: : : IF UNIT IS ACTIVE
: : : THEN
: : : : BEGIN 5 (START OF TEST FOR AN ACTIVE UNIT)
: : : : : TIMES = (HIGHEST - LOWEST)/2 + 1
: : : : : INCR KOUNT FROM 1 TO TIMES
: : : : : BEGIN 6 (START OF COUNTING LOOP FOR SECTOR SELECTION)
: : : : : : CHOOSE A RANDOM WORD COUNT
: : : : : : SET UP BUFFER POINTERS BEFORE TRANSFER
: : : : : : CHOOSE A RANDOM SECTOR
: : : : : : CALCULATE WHERE TRANSFER WILL END (BASED ON WORD COUNT)
: : : : : : IF CALCULATED VALUE GTR HIGHEST
: : : : : : THEN ADJUST THE WORD COUNT SO IT FITS
: : : : : : : : WITHIN THE TESTABLE SECTOR LIMITS
: : : : : : WRITE
: : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE LABEL)
: : : : : : CHOOSE WHETHER TO WRITE CHECK OR READ
: : : : : : DO THE WRITE CHECK OR READ
: : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE LABEL)
: : : : : : CHANGE BUFFER POINTERS AFTER TRANSFER
: : : : : : END 6 (END OF COUNTING LOOP FOR SECTOR SELECTION)
: : : : : END 5 (END OF TEST FOR AN ACTIVE UNIT)
: : : : END 4 (END OF TESTLOOP)
: : : END 3 (END OF LOGICAL UNIT SELECTION LOOP)
: : END 2 (END OF REPEAT LOOP FOR THIS ROUTINE)
RETURN
END 1 (END OF ROUTINE)
```

#### 6.5.4 RAND4 ROUTINE

PURPOSE: TO TEST USING RANDOM DATA, WORD COUNTS, SECTORS  
AND UNITS

THE CODE FOR 'RAND4' IN BRIEF:

```
BEGIN 1 (START OF ROUTINE)
SAY ROUTINE IS RUNNING
INCR COUNT FROM 1 TO REPEAT
: BEGIN 2 (START OF REPEAT LOOP FOR THIS ROUTINE)
: GENERATE THE RANDOM PATTERN
: TIMES = NUMBER OF UNITS * 4
: INCR KOUNT FROM 1 TO TIMES
: : BEGIN 3 (START OF COUNTING LOOP FOR UNIT SELECTION)
: : CHOOSE A RANDOM LOGICAL UNIT WHICH IS ACTIVE
: : INITIALIZE BUFFER POINTERS
: : TESTLOOP:
: : : BEGIN 4 (START OF LOOP THAT COMPLETELY TESTS 1 UNIT)
: : : INCR KOUNT2 FROM 1 TO 10
: : : : BEGIN 5 (START OF COUNTING LOOP FOR SECTOR SELECTION)
: : : : CHOOSE A RANDOM WORD COUNT
: : : : SET UP BUFFER POINTERS BEFORE TRANSFER
: : : : CHOOSE A RANDOM SECTOR
: : : : CALCULATE WHERE TRANSFER WILL END (BASED ON WORD COUNT)
: : : : IF CALCULATED VALUE GTR HIGHEST
: : : : THEN ADJUST THE WORD COUNT SO IT FITS
: : : : : WITHIN THE TESTABLE SECTOR LIMITS
: : : : WRITE
: : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
: : : : CHOOSE WHETHER TO WRITE CHECK OR READ
: : : : DO THE WRITE CHECK OR READ
: : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
: : : : CHANGE BUFFER POINTERS AFTER TRANSFER
: : : : END 5 (END OF COUNTING LOOP FOR SECTOR SELECTION)
: : : END 4 (END OF TESTLOOP)
: : END 3 (END OF COUNTING LOOP FOR UNIT SELECTION)
: END 2 (END OF REPEAT LOOP FOR THIS ROUTINE)
RETURN
END 1 (END OF ROUTINE)
```

1  
33  
35  
36 002000  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
64  
65

```
.SBTTL PROGRAM HEADER AND TABLES
      .ENABL ABS,AMA
      .      =      2000
      BGNMOD
      :++
      : THE PROGRAM HEADER IS THE INTERFACE BETWEEN
      : THE DIAGNOSTIC PROGRAM AND THE SUPERVISOR.
      :--
      POINTER ALL
      HEADER CZMLB,A,0,1800.,1
```

103  
132  
115  
114  
102  
000  
000  
000  
101  
060  
000000  
003410  
002264  
002440  
002210  
002222  
075012  
000000  
000000  
000001  
000000  
002204  
000000  
000000

```
LSNAME::
        .ASCII /C/
        .ASCII /Z/
        .ASCII /M/
        .ASCII /L/
        .ASCII /B/
        .BYTE 0
        .BYTE 0
        .BYTE 0
LSREV::
        .ASCII /A/
LSDEPO::
        .ASCII /O/
LSUNIT::
        .WORD T$PTHV
LSTIML::
        .WORD 1800.
LSHPCP::
        .WORD L$HARD
LSSPCP::
        .WORD L$SOFT
LSHPTP::
        .WORD L$HW
LSSPTP::
        .WORD L$SW
LSLADP::
        .WORD L$LAST
L$STA::
        .WORD 0
L$CO::
        .WORD 0
L$DTYP::
        .WORD 1
L$APT::
        .WORD 0
L$DTP::
        .WORD L$DISPATCH
L$PRIO::
        .WORD 0
L$ENVI::
        .WORD 0
```

000000

003  
003

000000  
000000

000000

002122

005362

000000

000000

005540

005526

000000

002130

104035

002172

036004

074776

005374

004014

000000

000000

000000

66  
77  
78  
79  
80

115 114 061

⋮

NAMES OF DEVICES SUPPORTED BY THIS PROGRAM  
DEV TYP <ML11>

LSEXP1:: .WORD 0  
LSMREV:: .BYTE CSREVISION  
          .BYTE CSREDIT  
LSEF:: .WORD 0  
          .WORD 0  
LSSPC:: .WORD 0  
LSDEVP:: .WORD LSDVTYP  
LSREPP:: .WORD LSRPT  
LSEXP4:: .WORD 0  
LSEXP5:: .WORD 0  
LSAUT:: .WORD LSAU  
LSDUT:: .WORD LSDU  
LSLUN:: .WORD 0  
LSDESP:: .WORD LSDESC  
LSLOAD:: .WORD EMT ESLOAD  
LSETP:: .WORD LSERRTBL  
LSICP:: .WORD LSINIT  
LSCCP:: .WORD LSCLEAN  
LSACP:: .WORD LSAUTO  
LSPRT:: .WORD LSPROT  
LSTEST:: .WORD 0  
LSDLY:: .WORD 0  
LSHIME:: .WORD 0

LSDVTYP:: .ASCIZ /ML11/  
          .EVEN

82  
83  
84  
85  
  
86  
87  
88  
89  
90  
91  
92  
93  
94  
  
95  
96  
97  
98  
99  
100  
101  
  
102  
109  
110  
111  
112  
113  
114  
115  
116  
117  
  
118  
128  
129  
130  
131  
132  
133  
134

103 132 115

000000  
000000  
000000  
000000

000001  
073024

000004

:  
: TEST DESCRIPTION  
:  
: DESCRIPT <CZMLB ML11 PERFORMANCE EXERCISER>

:  
: THE GLOBAL ERROR TABLE (INFORMATION  
: USED IN A CALL TO THE MACRO 'ERROR')  
:

ERRTBL  
ERRTYP:: .WORD 0  
ERRNBR:: .WORD 0  
ERRMSG:: .WORD 0  
ERRBLK:: .WORD 0

:++  
: THE DISPATCH TABLE CONTAINS THE STARTING ADDRESS OF EACH TEST.  
: IT IS USED BY THE SUPERVISOR TO DISPATCH TO EACH TEST.  
:--

DISPATCH 1

:++  
: THE DEFAULT HARDWARE P-TABLE CONTAINS DEFAULT VALUES OF  
: THE TEST-DEVICE PARAMETERS. THE STRUCTURE OF THIS TABLE  
: IS IDENTICAL TO THE STRUCTURE OF THE HARDWARE P-TABLES,  
: AND IS USED AS A 'TEMPLATE' FOR BUILDING THE P-TABLES.  
:--

BGNHW DFPTBL

.WORD 176400 ;CSR ADDRESS  
.WORD 204 ;RH VECTOR ADDRESS  
.WORD 5 ;BR LEVEL FOR INTERRUPT  
.WORD 0 ;ML11 DRIVE NUMBER

ENDHW

LSDESC:: .ASCIZ /CZMLB ML11 PERFORMANCE EXER  
.EVEN

LSERRTBL::

.WORD 1  
LSDISPATCH::  
.WORD T1

.WORD L10000-LSHW/2  
LSHW::  
DFPTBL::

L10000:



136  
137  
138  
139  
140  
141  
142  
143

;++  
: THE DEFAULT SOFTWARE P-TABLE CONTAINS VARIOUS DATA USED BY THE  
: PROGRAM AS OPERATIONAL PARAMETERS. THESE PARAMETERS ARE SET  
: UP AT ASSEMBLY TIME AND MAY BE VARIED BY THE OPERATOR AT RUN  
: TIME.  
:--

000020

BGNSW SFPTBL

.WORD L10001-L\$\$W/2

L\$\$W::  
SFPTBL::

144  
152  
153 000000  
154  
155 000000  
156  
157 000000  
158 000000  
159 000000  
160 000000  
161 000000  
162 000000  
163 000000  
164 000000  
165 000001  
166 000000  
167 000000  
168  
169 000000  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188 000000  
189  
190 000000  
191  
192  
193

LIMIT:: .WORD 0 ;LIMIT RANGE OF SECTORS TO BE TESTED  
; 0 = NO 1 = YES  
RANGE:: .WORD 0 ;DOES OPERATOR KNOW EXACT SECTOR NUMBERS?  
; 0 = NO 1 = YES  
LSECT:: .WORD 0 ;LOW SECTOR NUMBER  
TSECT:: .WORD 0 ;TOP SECTOR NUMBER  
ONLY:: .WORD 0 ;ONLY BOARD TO TEST (BOARD #'S ARE 0 TO 15)  
DROPNE:: .WORD 0 ;DROP ANY OPTIONS? 0 = NO 1 = YES  
DROP1:: .WORD 0 ;DROP OPTION #1? 0 = NO 1 = YES  
DROP2:: .WORD 0 ;DROP OPTION #2? 0 = NO 1 = YES  
DROP3:: .WORD 0 ;DROP OPTION #3? 0 = NO 1 = YES  
DROP4:: .WORD 0 ;DROP OPTION #4? 0 = NO 1 = YES  
MARPAT:: .WORD 1 ;PATTERN NUMBER USED FOR MARCH TEST  
DROPS:: .WORD 0 ;DROP OPTION #5? 0 = NO 1 = YES  
REFRESH:: .WORD 0 ;ENABLE MARGINING? 0 = NO (LEAVE DISABLED)  
; 1 = YES (ENABLE IT)  
ECCDIS:: .WORD 0 ;DISABLE ECC? 0 = NO (LEAVE ENABLED)  
; 1 = YES (DISABLE IT)

\*\*\*\*\* IMPORTANT NOTE \*\*\*\*\*  
: THE FOLLOWING 2 SOFTWARE PARAMETERS ;  
: HAVE NON-STANDARD DEFAULTS. ERRORS ;  
: AND END OF PASS PERFORMANCE SUMMARY ;  
: REPORTS WILL \*\* N O T \*\* BE PRINTED ;  
: UNLESS THE OPERATOR SPECIFICALLY ;  
: REQUESTS THE PRINTOUTS VIA SUITABLE ;  
: ANSWERS TO THE SOFTWARE QUESTIONS. ;  
: ;  
: THIS OPERATING FEATURE WAS INCLUDED ;  
: IN THE ML11 PERFORMANCE EXERCISER'S ;  
: FUNCTIONAL SPECIFICATION AT THE RE- ;  
: QUEST OF FIELD SERVICE. ;  
\*\*\*\*\*

EOPSUM:: .WORD 0 ;ENABLE EOP SUMMARIES? 0 = NO PRINTOUT  
; 1 = YES  
ERROUT:: .WORD 0 ;ENABLE ERROR PRINTOUTS? 0 = NO PRINTOUT  
; 1 = YES

ENDSW

L10001:

219  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256

```

:++
: THE HARDWARE PARAMETER CODING SECTION CONTAINS MACROS
: THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES. THE
: MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
: INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES. THE
: MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
: WITH THE OPERATOR.
:--

```

```

000022          BGNHRD
                                     .WORD L10002-L$HARD/2
L$HARD::

```

257  
267  
268  
269

```

000031          GPRMA   QH1,0,0,0,177777,YES
002330
000000
177777
                                     .WORD   T$CODE
                                     .WORD   QH1
                                     .WORD   T$LOLIM
                                     .WORD   T$HILIM

```

270  
271  
272

```

001031          GPRMA   QH2,2,0,0,377,YES
002345
000000
000377
                                     .WORD   T$CODE
                                     .WORD   QH2
                                     .WORD   T$LOLIM
                                     .WORD   T$HILIM

```

271  
272  
273  
274

```

002032          GPRMD   QH3,4,0,7,1,7,YES
002367
000007
000001
000007
                                     .WORD   T$CODE
                                     .WORD   QH3
                                     .WORD   7
                                     .WORD   T$LOLIM
                                     .WORD   T$HILIM

```

272  
273  
274

```

003032          GPRMD   QH4,6,0,7,0,7,YES
002417
000007
000000
000007
                                     .WORD   T$CODE
                                     .WORD   QH4
                                     .WORD   7
                                     .WORD   T$LOLIM
                                     .WORD   T$HILIM

```

273  
274

```

ENDHRD
                                     .EVEN
L10002:

```

275  
282  
283  
284  
285  
286  
287  
288  
289

```

103    123    122    QH1:  .ASCIZ  /CSR ADDRESS?/
111    116    124    QH2:  .ASCIZ  /INTERRUPT VECTOR?/
102    122    040    QH3:  .ASCIZ  /BR LEVEL FOR INTERRUPT?/
104    122    111    QH4:  .ASCIZ  /DRIVE NUMBER?/
                                     .EVEN

```

291  
292  
293  
294  
295  
296  
297  
298  
299  
300

```

:++
: THE SOFTWARE PARAMETER CODING SECTION CONTAINS MACROS
: THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES. THE
: MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
: INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES. THE
: MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
: WITH THE OPERATOR.
:--

```

000103

BGNSFT

.WORD L10003-L\$SOFT/2  
L\$SOFT::

301  
310  
311

000130  
002646  
000001

GPRML QS1,0,1,YES

.WORD T\$CODE  
.WORD QS1  
.WORD 1

312

025044

XFERF 6\$

.WORD T\$CODE

313

001130  
002713  
000001

GPRML QS2,2,1,YES

.WORD T\$CODE  
.WORD QS2  
.WORD 1

314

014044

XFERF 5\$

.WORD T\$CODE

315

002032  
003000  
177777  
000000  
177777

GPRMD QS3,4,0,177777,0,177777,YES

.WORD T\$CODE  
.WORD QS3  
.WORD 177777  
.WORD T\$LOLIM  
.WORD T\$HILIM

316

003032  
003033  
177777  
000000  
177777

GPRMD QS4,6,0,177777,0,177777,YES

.WORD T\$CODE  
.WORD QS4  
.WORD 177777  
.WORD T\$LOLIM  
.WORD T\$HILIM

317

006004

XFER 6\$

.WORD T\$CODE

318

004052  
003065  
000037  
000000  
000017

5\$: GPRMD QS5,10,D,37,0,15.,YES

.WORD T\$CODE  
.WORD QS5  
.WORD 37  
.WORD T\$LOLIM  
.WORD T\$HILIM

319

005130  
003132  
000001

6\$: GPRML QS6,12,1,YES

.WORD T\$CODE  
.WORD QS6  
.WORD 1

320

007024

XFERT 7\$

.WORD T\$CODE

322		GPRMD	QS11,24,D,77,1,10.,YES		
	012052			.WORD	T\$CODE
	003532			.WORD	QS11
	000077			.WORD	77
	000001			.WORD	T\$LLOLIM
	000012			.WORD	T\$HILIM
323		XFER	13\$		
	026004			.WORD	T\$CODE
324		7\$: GPRML	QS7,14,1,YES		
	006130			.WORD	T\$CODE
	003436			.WORD	QS7
	000001			.WORD	1
325		GPRML	QS8,16,1,YES		
	007130			.WORD	T\$CODE
	003455			.WORD	QS8
	000001			.WORD	1
326		GPRML	QS9,20,1,YES		
	010130			.WORD	T\$CODE
	003474			.WORD	QS9
	000001			.WORD	1
327		GPRML	QS10,22,1,YES		
	011130			.WORD	T\$CODE
	003513			.WORD	QS10
	000001			.WORD	1
328		XFERT	12\$		
	006024			.WORD	T\$CODE
329		GPRMD	QS11,24,D,77,1,10.,YES		
	012052			.WORD	T\$CODE
	003532			.WORD	QS11
	000077			.WORD	77
	000001			.WORD	T\$LLOLIM
	000012			.WORD	T\$HILIM
330		12\$: GPRML	QS12,26,1,YES		
	013130			.WORD	T\$CODE
	003614			.WORD	QS12
	000001			.WORD	1
331		13\$: GPRML	QS13,30,1,YES		
	014130			.WORD	T\$CODE
	003633			.WORD	QS13
	000001			.WORD	1
332		GPRML	QS14,32,1,YES		
	015130			.WORD	T\$CODE
	003665			.WORD	QS14
	000001			.WORD	1
333		GPRML	QS15,34,1,YES		
	016130			.WORD	T\$CODE
	003717			.WORD	QS15
	000001			.WORD	1
334		GPRML	QS16,36,1,YES		
	017130			.WORD	T\$CODE
	003764			.WORD	QS16
	000001			.WORD	1
335		.EVEN			
336					
337		ENDSF T			

L10003: .EVEN

```

345 114 111 115 QS1: .ASCIZ /LIMIT RANGE OF SECTORS TO BE TESTED?/
346 104 117 040 QS2: .ASCIZ /DO YOU KNOW THE EXACT RANGE OF SECTORS TO BE TESTED?/
347 106 111 122 QS3: .ASCIZ /FIRST SECTOR TO BE TESTED?/
348 114 101 123 QS4: .ASCIZ /LAST SECTOR TO BE TESTED?/
349 127 110 111 QS5: .ASCIZ /WHICH BOARD (0-15) SHOULD BE TESTED?/
350 124 110 105 QS6: .ASCII /THE PROGRAM OPTIONS INCLUDE:/
351 012 015 040 .ASCII <12><15>/ 1. ADDRESS CHECK/
352 012 015 040 .ASCII <12><15>/ 2. PATTERN TEST/
353 012 015 040 .ASCII <12><15>/ 3. UNIQUE DATA CHECK/
354 012 015 040 .ASCII <12><15>/ 4. MARCH TEST/
355 012 015 040 .ASCII <12><15>/ 5. RANDOMNESS TESTS/
356 012 015 104 .ASCIZ <12><15>/DO YOU WANT TO DROP ANY OF THESE FROM THE EXERCISER?/
357 104 122 117 QS7: .ASCIZ /DROP OPTION 1?/ -
358 104 122 117 QS8: .ASCIZ /DROP OPTION 2?/
359 104 122 117 QS9: .ASCIZ /DROP OPTION 3?/
360 104 122 117 QS10: .ASCIZ /DROP OPTION 4?/
361 120 101 124 QS11: .ASCIZ /PATTERN NUMBER (1-10) TO BE USED WITH MARCH TEST?/
362 104 122 117 QS12: .ASCIZ /DROP OPTION 5?/
363 105 116 101 QS13: .ASCIZ /ENABLE REFRESH MARGINING?/
364 104 111 123 QS14: .ASCIZ /DISABLE ERROR CORRECTION?/
365 105 116 101 QS15: .ASCIZ /ENABLE END OF PASS SUMMARY PRINTOUT?/
366 105 116 101 QS16: .ASCIZ /ENABLE ERROR PRINTOUTS?/
367 .EVEN
368
369
370
371
372
373
374
375
376

```

```

:++
: THE PROTECTION TABLE IS USED BY THE RUNTIME
: SERVICES TO PROTECT THE LOAD MEDIA.
:--

```

```

377 BGNPROT
378 177777 -1 ;OFFSET INTO P-TABLE FOR CSR ADDRESS
379 177777 -1 ;OFFSET INTO P-TABLE FOR MASSBUS ADDRESS
380 177777 -1 ;OFFSET INTO P-TABLE FOR DRIVE NUMBER
381
382 ENDPROT
383
397
398
399 $PATCH: .BLKW 32. ;THIS LEAVES ROOM FOR THE 22 ML-11 REGISTERS TOO
400
407
408 ENDMOD
409
410
411
412
425

```

```
4 :OTS.MAC CONTAINS THE FOLLOWING MODULES AND ROUTINES:
5 :
6 :   MODULE B16PG1
7 :     ROUTINE BL$GT1 -- TO FETCH A FIELD OUT OF A VALUE
8 :
9 :   MODULE B16PG2
10 :    ROUTINE BL$GT2 -- TO FETCH A FIELD OUT OF A VALUE GIVEN THE ADDRESS
11 :
12 :   MODULE B16PG3
13 :    ROUTINE BL$PU1 -- TO STORE A NEW VALUE INTO A FIELD IN AN OLD VALUE
14 :
15 :   MODULE B16PG4
16 :    ROUTINE BL$PU2 -- TO STORE A NEW VALUE INTO A FIELD GIVEN THE ADDRESS
17 :
18 :   MODULE B16MUL
19 :    ROUTINE BL$MUL -- TO MULTIPLY TWO SIGNED INTEGERS
20 :    ROUTINE DIVMOD --
21 :    ROUTINE BL$DIV -- TO DIVIDE TWO SIGNED INTEGERS
22 :    ROUTINE BL$MOD --
23 :    ROUTINE BL$SHF --
24 :
25 :   MODULE B16SAV
26 :    ROUTINE $$SAVE2 -- TO SAVE 2 REGISTERS (R2, R1)
27 :    ROUTINE $$SAVE3 -- TO SAVE 3 REGISTERS (R3, R2, R1)
28 :    ROUTINE $$SAVE4 -- TO SAVE 4 REGISTERS (R4, R3, R2, R1)
29 :    ROUTINE $$SAVE5 -- TO SAVE 5 REGISTERS (R5, R4, R3, R2, R1)
```

1389

:\\ \\ \\ \\ \\ \\ \\ LEAVE OTS.MAC WITH THE LISTING TURNED ON \\ \\ \\ \\ \\ \\ \\

```

1      .SBTTL 'RANDOM NUMBER GENERATOR'
2
3      ;ROUTINE TO GENERATE 16-BIT PSEUDO RANDOM NUMBERS
4
5      ;INPUTS:                NONE
6      ;IMPLIED INPUTS:       SEED1, SEED2, SEED3 are global values which will
7      ;                       be used by, but not input to the generator.
8      ;OUTPUTS:              'RANDOM'--WORD CONTAINING RANDOM 16-BIT INTEGER VALUE
9      ;IMPLIED OUTPUTS:      NONE
10     ;CALLING SEQUENCE:     JSR PC,RN
11
12     005256 010046          RN::  MOV     R0,-(SP)
13     005260 013700 005346  MOV     SEED2,R0
14     005264 000241          CLC
15     005266 005337 005344  DEC     SEED1
16     005272 006100          ROL     R0
17     005274 006100          ROL     R0
18     005276 063700 005344  ADD     SEED1,R0
19     005302 063700 005350  ADD     SEED3,R0
20     005306 010037 005346  MOV     R0,SEED2
21     005312 006100          ROL     R0
22     005314 006100          ROL     R0
23     005316 063700 005350  ADD     SEED3,R0
24     005322 006100          ROL     R0
25     005324 006100          ROL     R0
26     005326 010037 005350  MOV     R0,SEED3
27     005332 013737 005346 005352  MOV     SEED2,RANDOM      ;SAVE NEW R.N.
28     005340 012600          MOV     (SP)+,R0
29     005342 000207          RTS     PC
30
31
32
33     005344 000000          SEED1:: .WORD 0
34     005346 001233          SEED2:: .WORD 1233
35     005350 007622          SEED3:: .WORD 7622
36
37     005352 000000          RANDOM:: .WORD 0
    
```



18-Sep-1980 09:39:38  
 18-Sep-1980 09:39:31

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX3.BLI.1 (1)

```

6 ;MLX3
7 ;
8
9      0001  MODULE MLX3 =
10     0002  BEGIN
11     0003
12     0004  REQUIRE 'BLSMAC.REQ';
13     0943
14     0944  %SBTTL 'REPORT CODING SECTION'
15     0945
16     0946  EXTERNAL ROUTINE
17     0947      EOP: NOVALUE;
18     0948
19     0949  BGNRPT;
20     0950
21     0951  EOP();
22     0952
23     0953  RETURN;
24     0954
25     0955  ENDRPT;
29
30             .TITLE  MLX3
31
32             .GLOBL  EOP
33
34
35             .SBTTL  LRPT REPORT CODING SECTION
36
40 005354 004737 073040      LRPT:  JSR    PC,EOP      ;      0951
41 005360 000207              RTS    PC          ;      0947
42
43             ; Routine Size: 3 words
44             ; Maximum stack depth per invocation: 0 words
49
50
54
55             .SBTTL  L$RPT REPORT CODING SECTION
59 005362 004737 005354  L$RPT:: JSR    PC,LRPT      ;      0953
    
```

61  
62  
63  
64  
65  
66  
67  
68  
73  
74

;MLX3  
;

REPORT CODING SECTION

18-Sep-1980 09:39:38 TOPS  
18-Sep-1980 09:39:31 PA:<

005366 104425  
005370 000207

TRAP 25  
RTS PC

; Routine Size: 4 words  
; Maximum stack depth per invocation: 0 words

76 :MLX3

18-Sep-1980 09:39:38

TOPS-20 Bliss-16 V2(206)

77 :

AUTODROP SECTION

18-Sep-1980 09:39:31

PA:<ROSFN>MLX3.BLI.1 (2)

78 :

79 :

0956 %SBTTL 'AUTODROP SECTION'

80 :

0957

81 :

0958

!++

82 :

0959

THIS SECTION IS OPTIONALLY EXECUTED IMMEDIATELY AFTER THE INITIALIZATION  
 CODE IF THE /ADR FLAG WAS SET. THE INTENT IS TO EXAMINE THE UNITS UNDER  
 TEST TO SEE IF THEY WILL RESPOND. THOSE THAT DON'T RESPOND ARE DROPPED.

83 :

0960

84 :

0961

85 :

0962

86 :

0963

THIS FUNCTION IS AN INTEGRAL PART OF THE INITIALIZATION CODE ITSELF, AND  
 THEREFORE A SEPARATE AUTODROP SECTION IS REDUNDANT AND NOT PROVIDED.

87 :

0964

88 :

0965

!--

89 :

0966

90 :

0967

BGNAUTO;

91 :

0968

92 :

0969

RETURN;

93 :

0970

94 :

0971

ENDAUTO;

98 :

99 :

.SBTTL LAUTO AUTODROP SECTION

103 005372 000207

LAUTO: RTS PC ;

0955

104 :

; Routine Size: 1 word  
 ; Maximum stack depth per invocation: 0 words

105 :

106 :

111 :

112 :

116 :

117 :

.SBTTL L\$AUTO AUTODROP SECTION

121 005374 004737 005372

L\$AUTO::JSR PC,LAUTO ;

0969

122 :

005400 104461

TRAP 61

123 :

005402 000207

RTS PC

124 :

125 :

; Routine Size: 4 words  
 ; Maximum stack depth per invocation: 0 words

126 :

132 :MLX3

18-Sep-1980 09:39:38  
 18-Sep-1980 09:39:31

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX3.BLI.1 (3)

```

133 : DROP UNIT SECTION
134 :
135 : 0972 %SBTTL 'DROP UNIT SECTION'
136 : 0973
137 : 0974 !++
138 : 0975 ! THE DROP-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DRIVE TO NO
139 : 0976 ! LONGER BE TESTED. WHEN THIS HAPPENS, THE FOLLOWING ACTIONS WILL BE
140 : 0977 ! TAKEN:
141 : 0978
142 : 0979 ! (1) SET THE DRIVE STATUS TO INACTIVE AND SET THE DRIVE'S DROP FLAG.
143 : 0980
144 : 0981 ! (2) DECREMENT THE NUMBER OF ACTIVE DRIVES, AND IF ZERO ABORT PASS.
145 : 0982 !--
146 : 0983
147 : 0984 BGNDU;
148 : 0985
149 : 0986 EXTERNAL
150 : 0987 L$LUN,
151 : 0988 DRIVE_STATUS:BITVECTOR[8],
152 : 0989 DROPT_DRIVES:BITVECTOR[8],
153 : 0990 NUM_DRIVES;
154 : 0991
155 : 0992
156 : 0993 LITERAL
157 : 0994 INACTIVE = 0,
158 : 0995 ACTIVE = 1;
159 : 0996
160 : 0997
161 : 0998 DRIVE_STATUS[.L$LUN] = INACTIVE;
162 : 0999 DROPT_DRIVES[.L$LUN] = ACTIVE;
163 : 1000
164 : 1001 NUM_DRIVES = .NUM_DRIVES - 1;
165 : 1002
166 : 1003 IF .NUM_DRIVES EQL 0
167 : 1004 THEN D0CLN;
168 : 1005
169 : 1006 RETURN;
170 : 1007
171 : 1008 ENDDU;
  
```

```

.GLOBL L$LUN, DRIVE.STATUS, DROPT.DRIVES
.GLOBL NUM.DRIVES
  
```

```

175 :
176 :
177 :
178 :
179 :
180 : .SBTTL LDU DROP UNIT SECTION
184 005404 013700 002074 LDU: MOV L$LUN,RO ;
185 005410 006200 ASR RO
186 005412 006200 ASR RO
  
```

0998

4  
 18-Sep-1980 09:39:38 TOPS  
 18-Sep-1980 09:39:31 PA:<

```

188 ;MLX3
189 ;
190
191 005414 006200 ASR R0
192 005416 062700 032460 ADD #DRIVE.STATUS,R0
193 005422 010046 MOV R0,-(SP)
194 005424 013746 002074 MOV L$LUN,-(SP)
195 005430 042716 177770 BIC #177770,(SP)
196 005434 012746 000001 MOV #1,-(SP)
197 005440 005046 CLR -(SP)
198 005442 004737 004502 JSR PC,BL$PU2
199 005446 013700 002074 MOV L$LUN,R0 ;
200 005452 006200 ASR R0
201 005454 006200 ASR R0
202 005456 006200 ASR R0
203 005460 062700 032462 ADD #DROPT.DRIVES,R0
204 005464 010016 MOV R0,(SP)
205 005466 013746 002074 MOV L$LUN,-(SP)
206 005472 042716 177770 BIC #177770,(SP)
207 005476 012746 000001 MOV #1,-(SP)
208 005502 011646 MOV (SP),-(SP)
209 005504 004737 004502 JSR PC,BL$PU2
210 005510 005337 032474 DEC NUM.DRIVES ;
211 005514 001001 BNE 1$ ;
212 005516 104444 TRAP 44 ;
213 005520 062706 000016 1$: ADD #16,SP ;
214 005524 000207 RTS PC ;
215
216 ; Routine Size: 41 words
217 ; Maximum stack depth per invocation: 7 words
222
223
227
228 .SBTTL L$DU DROP UNIT SECTION
232 005526 004737 005404 L$DU:: JSR PC,LDU ;
233 005532 104453 TRAP 53 ;
234 005534 000207 RTS PC ;
235
236 ; Routine Size: 4 words
237 ; Maximum stack depth per invocation: 0 words
  
```

0999

1001  
 1003  
 1004  
 0971

1006

```

243 ;MLX3
244 :
245 :
246 : 1009 %SBTTL 'ADD UNIT SECTION'
247 : 1010
248 : 1011 !++
249 : 1012 ! THE ADD UNIT SECTION CONTAINS ANY CODE THE PROGRAMMER WISHES
250 : 1013 ! TO BE EXECUTED IN CONJUNCTION WITH THE ADDING OF A UNIT BACK
251 : 1014 ! TO THE TEST CYCLE. SINCE THE INITIALIZATION CODE WILL HAVE
252 : 1015 ! TO BE EXECUTED IMMEDIATELY AFTER AN 'ADD', THERE IS NO NEED
253 : 1016 ! FOR ANY CODE IN THIS SECTION.
254 : 1017 !--
255 : 1018
256 : 1019 BGNAU;
257 : 1020
258 : 1021 RETURN;
259 : 1022
260 : 1023 ENDAU;
264
265
269 005536 000207 LAU: .SBTTL LAU ADD UNIT SECTION ; 1008
270 : RTS PC
271 : Routine Size: 1 word
272 : Maximum stack depth per invocation: 0 words
277
278
282
283
287 005540 004737 005536 L$AU:: .SBTTL L$AU ADD UNIT SECTION ; 1021
288 005544 104452 : JSR PC,LAU
289 005546 000207 : TRAP 52
: RTS PC
290
291 : Routine Size: 4 words
292 : Maximum stack depth per invocation: 0 words
297 : 1024
  
```

18-Sep-1980 09:39:38  
 18-Sep-1980 09:39:31

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX3.BLI.1 (4)

299 ;MLX3

300 ;

ADD UNIT SECTION

301

302 : 1025

303 : 1026 END

304 : 1027

305 : 1028 ELUDOM

309

: OTS external references  
.GLOBL BL&PU2

310

311

312

313

314

315

316

317

: Size: 62 code + 0 data words  
: Run Time: 00:02.7  
: Elapsed Time: 00:08.9  
: Memory Used: 17 pages  
: Compilation Complete

318

319

320

321

322

323

18-Sep-1980 09:39:38

18-Sep-1980 09:39:31

TOPS-20 Bliss-16 V2(206)

PA:<ROSEN>MLX3.BLI.1 (4)

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (1)

6 ;MLX4

```
7 ;
8 ;
9 0001 MODULE MLX4 =
10 0002 BEGIN
11 0003
12 0004 REQUIRE 'BLSMAC.REQ';
13 1488
14 1489 %SBTTL 'VARIABLES AND CONSTANTS'
15 1490
16 1491 LITERAL
17 1492
18 1493 ONE = 1, !NUMBER OF TIMES TO RETRY FOR DATA ERROR
19 1494 SIX = 6, !NUMBER OF TIMES TO RETRY FOR SYSTEM ERROR
20 1495 NUM_PATS = 10, !NUMBER OF REGULAR PATTERNS
21 1496 NUM_REGS = 22, !NUMBER OF ML11 REGISTERS
22 1497 TIMES_TO_LOOP = 2, !LOOP READING CONSTANT
23 1498 BUFSIZ = 256*8, !2048 16-BIT WORDS IN A FULL BUFFER
24 1499 !256 = NUMBER OF WORDS IN A SECTOR
25 1500 !8 = NUMBER OF SECTORS IN THE BUFFER
26 1501
27 1502
28 1503 ! ML11 DRIVE TYPES:
29 1504
30 1505
31 1506 ML11A = %0'000110', !16K CHIPS
32 1507 ML11B = %0'000111', !64K CHIPS
33 1508
34 1509 !
35 1510 ! FUNCTION CODES:
36 1511
37 1512
38 1513 DRV_CLR = %0'11',
39 1514 WC_CMD = %0'51',
40 1515 WR_CMD = %0'61',
41 1516 RD_CMD = %0'71',
42 1517
43 1518 !
44 1519 ! STATUS CODES:
45 1520
46 1521
47 1522 INACTIVE = 0,
48 1523 ACTIVE = 1,
49 1524
50 1525 !
51 1526 ! ERROR THRESHOLD VALUES:
52 1527
53 1528
54 1529 S16K_LIMIT = 10, !SOFT ERROR, 16K ARRAYS
55 1530 H16K_LIMIT = 10, !HARD ERROR, 16K ARRAYS
56 1531
57 1532 S64K_LIMIT = 10, !SOFT ERROR, 64K ARRAYS
58 1533 H64K_LIMIT = 10, !HARD ERROR, 64K ARRAYS
```



60 :MLX4  
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 :  
103 :  
104 :  
105 :  
106 :  
107 :  
108 :  
109 :  
110 :  
111 :  
112 :  
113 :  
114 :

VARIABLES AND CONSTANTS

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (2)

SUMMARY OF ERROR CODES:

THE 'INTEGRITY' ROUTINE:

!WRITE COMMAND:

!ERRDF(1,MSG1,0); !\*\*\*\* INTEGRITY ROUTINE ERROR 01 \*\*\*\*  
!ERRDF(2,MSG1,0); !\*\*\*\* INTEGRITY ROUTINE ERROR 02 \*\*\*\*  
!ERRDF(3,MSG1,0); !\*\*\*\* INTEGRITY ROUTINE ERROR 03 \*\*\*\*

!READ COMMAND:

!ERRDF(4,MSG1,0); !\*\*\*\* INTEGRITY ROUTINE ERROR 04 \*\*\*\*  
!ERRDF(5,MSG1,0); !\*\*\*\* INTEGRITY ROUTINE ERROR 05 \*\*\*\*  
!ERRDF(6,MSG1,0); !\*\*\*\* INTEGRITY ROUTINE ERROR 06 \*\*\*\*  
!ERRDF(7,MSG1,0); !\*\*\*\* INTEGRITY ROUTINE ERROR 07 \*\*\*\*  
!ERRDF(8,MSG2,0); !\*\*\*\* INTEGRITY ROUTINE ERROR 08 \*\*\*\*  
!ERRHRD(9,MSG4,0); !\*\*\*\* INTEGRITY ROUTINE ERROR 09 \*\*\*\*  
!ERRSOFT(10,MSG3,0); !\*\*\*\* INTEGRITY ROUTINE ERROR 10 \*\*\*\*  
!ERRSOFT(11,MSG3,1); !\*\*\*\* INTEGRITY ROUTINE ERROR 10 \*\*\*\*

!WRITE CHECK COMMAND:

!ERRDF(12,MSG1,0); !\*\*\*\* INTEGRITY ROUTINE ERROR 12 \*\*\*\*  
!ERRDF(13,MSG1,0); !\*\*\*\* INTEGRITY ROUTINE ERROR 13 \*\*\*\*  
!ERRDF(14,MSG1,0); !\*\*\*\* INTEGRITY ROUTINE ERROR 14 \*\*\*\*  
!ERRDF(15,MSG2,0); !\*\*\*\* INTEGRITY ROUTINE ERROR 15 \*\*\*\*  
!ERRHRD(16,MSG4,0); !\*\*\*\* INTEGRITY ROUTINE ERROR 16 \*\*\*\*  
!ERRSOFT(17,MSG3,8); !\*\*\*\* INTEGRITY ROUTINE ERROR 17 \*\*\*\*  
!ERRSOFT(18,MSG3,1); !\*\*\*\* INTEGRITY ROUTINE ERROR 18 \*\*\*\*

OPTION 1:

!WRITE COMMAND:

!ERRDF(101,MSG1,0); !\*\*\*\* OPTION 1 ERROR 01 \*\*\*\*  
!ERRDF(102,MSG1,0); !\*\*\*\* OPTION 1 ERROR 02 \*\*\*\*  
!ERRDF(103,MSG1,0); !\*\*\*\* OPTION 1 ERROR 03 \*\*\*\*

!CHECK OR READ:

!ERRDF(104,MSG1,0); !\*\*\*\* OPTION 1 ERROR 04 \*\*\*\*  
!ERRDF(105,MSG1,0); !\*\*\*\* OPTION 1 ERROR 05 \*\*\*\*  
!ERRDF(106,MSG1,0); !\*\*\*\* OPTION 1 ERROR 06 \*\*\*\*  
!ERRDF(107,MSG1,0); !\*\*\*\* OPTION 1 ERROR 07 \*\*\*\*  
!ERRDF(108,MSG2,0); !\*\*\*\* OPTION 1 ERROR 08 \*\*\*\*

116 :MLX4

23-Oct-1980 15:01:43

TOPS-20 Bliss-16 V2(206)

117 :

VARIABLES AND CONSTANTS

23-Oct-1980 14:57:05

PA:<ROSEN>MLX4.BLI.1 (2)

118

119 :

1586

!ERRHRD(109,MSG4,0); !\*\*\*\* OPTION 1 ERROR 09 \*\*\*\*

120 :

1587

!ERRSOFT(110,MSG3,0); !\*\*\*\* OPTION 1 ERROR 10 \*\*\*\*

121 :

1588

.ERRSOFT(111,MSG3,0); !\*\*\*\* OPTION 1 ERROR 11 \*\*\*\*

123 :MLX4  
124 :  
125 :  
126 :  
127 :  
128 :  
129 :  
130 :  
131 :  
132 :  
133 :  
134 :  
135 :  
136 :  
137 :  
138 :  
139 :  
140 :  
141 :  
142 :  
143 :  
144 :  
145 :  
146 :  
147 :  
148 :  
149 :  
150 :  
151 :  
152 :  
153 :  
154 :  
155 :  
156 :  
157 :  
158 :  
159 :  
160 :  
161 :  
162 :  
163 :  
164 :  
165 :  
166 :  
167 :  
168 :  
169 :  
170 :  
171 :  
172 :  
173 :  
174 :  
175 :  
176 :  
177 :

VARIABLES AND CONSTANTS

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 v2(206)  
PA:<ROSEN>MLX4.BLI.1 (3)

! OPTION 2:

.WRITE COMMAND:

!ERRDF(201,MSG1,0); :\*\*\*\* OPTION 2 ERROR 01 \*\*\*\*  
!ERRDF(202,MSG1,0); :\*\*\*\* OPTION 2 ERROR 02 \*\*\*\*  
!ERRDF(203,MSG1,0); :\*\*\*\* OPTION 2 ERROR 03 \*\*\*\*

!CHECK OR READ:

!ERRDF(204,MSG1,0); :\*\*\*\* OPTION 2 ERROR 04 \*\*\*\*  
!ERRDF(205,MSG1,0); :\*\*\*\* OPTION 2 ERROR 05 \*\*\*\*  
!ERRDF(206,MSG1,0); :\*\*\*\* OPTION 2 ERROR 06 \*\*\*\*  
!ERRDF(207,MSG1,0); :\*\*\*\* OPTION 2 ERROR 07 \*\*\*\*  
!ERRDF(208,MSG2,0); :\*\*\*\* OPTION 2 ERROR 08 \*\*\*\*  
!ERRHRD(209,MSG4,0); :\*\*\*\* OPTION 2 ERROR 09 \*\*\*\*  
!ERRSOFT(210,MSG3,0); :\*\*\*\* OPTION 2 ERROR 10 \*\*\*\*  
!ERRSOFT(211,MSG3,1); :\*\*\*\* OPTION 2 ERROR 10 \*\*\*\*

!LOOP CHECK OR READ:

!ERRDF(212,MSG1,0); :\*\*\*\* OPTION 2 ERROR 12 \*\*\*\*  
!ERRDF(213,MSG1,0); :\*\*\*\* OPTION 2 ERROR 13 \*\*\*\*  
!ERRDF(214,MSG1,0); :\*\*\*\* OPTION 2 ERROR 14 \*\*\*\*  
!ERRDF(215,MSG1,0); :\*\*\*\* OPTION 2 ERROR 15 \*\*\*\*  
!ERRDF(216,MSG2,0); :\*\*\*\* OPTION 2 ERROR 16 \*\*\*\*  
!ERRHRD(217,MSG4,0); :\*\*\*\* OPTION 2 ERROR 17 \*\*\*\*  
!ERRSOFT(218,MSG3,0); :\*\*\*\* OPTION 2 ERROR 18 \*\*\*\*  
!ERRSOFT(219,MSG3,0); :\*\*\*\* OPTION 2 ERROR 19 \*\*\*\*

! OPTION 3:

!WRITE COMMAND:

!ERRDF(301,MSG1,0); :\*\*\*\* OPTION 3 ERROR 01 \*\*\*\*  
!ERRDF(302,MSG1,0); :\*\*\*\* OPTION 3 ERROR 02 \*\*\*\*  
!ERRDF(303,MSG1,0); :\*\*\*\* OPTION 3 ERROR 03 \*\*\*\*

!CHECK OR READ:

!ERRDF(304,MSG1,0); :\*\*\*\* OPTION 3 ERROR 04 \*\*\*\*  
!ERRDF(305,MSG1,0); :\*\*\*\* OPTION 3 ERROR 05 \*\*\*\*  
!ERRDF(306,MSG1,0); :\*\*\*\* OPTION 3 ERROR 06 \*\*\*\*  
!ERRDF(307,MSG1,0); :\*\*\*\* OPTION 3 ERROR 07 \*\*\*\*  
!ERRDF(308,MSG2,0); :\*\*\*\* OPTION 3 ERROR 08 \*\*\*\*  
!ERRHRD(309,MSG4,0); :\*\*\*\* OPTION 3 ERROR 09 \*\*\*\*  
!ERRSOFT(310,MSG3,0); :\*\*\*\* OPTION 3 ERROR 10 \*\*\*\*  
!ERRSOFT(311,MSG3,0); :\*\*\*\* OPTION 3 ERROR 11 \*\*\*\*

179 :MLX4

VARIABLES AND CONSTANTS

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (3)

```
180 :  
181 :  
182 : 1641  
183 : 1642  
184 : OPTION 4:  
185 : 1643  
186 : 1644  
187 : 1645  
188 : 1646  
189 : 1647  
190 : 1648  
191 : 1649  
192 : !ERRDF(401,MSG1,0); !**** OPTION 4 ERROR 01 ****  
193 : !ERRDF(402,MSG1,0); !**** OPTION 4 ERROR 02 ****  
194 : !ERRDF(403,MSG1,0); !**** OPTION 4 ERROR 03 ****  
195 : 1650  
196 : 1651  
197 : 1652  
198 : 1653  
199 : 1654  
200 : 1655  
201 : 1656  
202 : 1657  
203 : 1658  
204 : 1659  
205 : 1660  
206 : 1661  
207 : 1662  
208 : 1663  
209 : 1664  
210 : 1665  
211 : 1666  
212 : 1667  
213 : 1668  
214 : 1669  
215 : 1670  
216 : 1671  
217 : 1672  
218 : 1673  
219 : 1674  
220 : 1675  
221 : 1676  
222 : 1677  
223 : 1678  
224 : 1679  
225 : 1680  
226 : 1681  
227 : 1682  
228 : 1683  
229 : 1684  
230 : 1685  
231 : 1686  
232 : 1687  
233 : 1688  
234 : 1689  
235 : 1690  
236 : 1691  
237 : 1692
```

!MARCHING UP:  
!WRITE DATA:  
!ERRDF(401,MSG1,0); !\*\*\*\* OPTION 4 ERROR 01 \*\*\*\*  
!ERRDF(402,MSG1,0); !\*\*\*\* OPTION 4 ERROR 02 \*\*\*\*  
!ERRDF(403,MSG1,0); !\*\*\*\* OPTION 4 ERROR 03 \*\*\*\*  
!MARCHING UP:  
!CHECK OR READ DATA:  
!ERRDF(404,MSG1,0); !\*\*\*\* OPTION 4 ERROR 04 \*\*\*\*  
!ERRDF(405,MSG1,0); !\*\*\*\* OPTION 4 ERROR 05 \*\*\*\*  
!ERRDF(406,MSG1,0); !\*\*\*\* OPTION 4 ERROR 06 \*\*\*\*  
!ERRDF(407,MSG1,0); !\*\*\*\* OPTION 4 ERROR 07 \*\*\*\*  
!ERRDF(408,MSG2,0); !\*\*\*\* OPTION 4 ERROR 08 \*\*\*\*  
!ERRHRD(409,MSG4,0); !\*\*\*\* OPTION 4 ERROR 09 \*\*\*\*  
!ERRSOFT(410,MSG3,0); !\*\*\*\* OPTION 4 ERROR 10 \*\*\*\*  
!ERRSOFT(411,MSG3,0); !\*\*\*\* OPTION 4 ERROR 11 \*\*\*\*  
!WRITE COMP:  
!ERRDF(412,MSG1,0); !\*\*\*\* OPTION 4 ERROR 12 \*\*\*\*  
!ERRDF(413,MSG1,0); !\*\*\*\* OPTION 4 ERROR 13 \*\*\*\*  
!ERRDF(414,MSG1,0); !\*\*\*\* OPTION 4 ERROR 14 \*\*\*\*  
!MARCHING DOWN:  
!CHECK OR READ COMP:  
!ERRDF(415,MSG1,0); !\*\*\*\* OPTION 4 ERROR 15 \*\*\*\*  
!ERRDF(416,MSG1,0); !\*\*\*\* OPTION 4 ERROR 16 \*\*\*\*  
!ERRDF(417,MSG1,0); !\*\*\*\* OPTION 4 ERROR 17 \*\*\*\*  
!ERRDF(418,MSG1,0); !\*\*\*\* OPTION 4 ERROR 18 \*\*\*\*  
!ERRDF(419,MSG2,0); !\*\*\*\* OPTION 4 ERROR 19 \*\*\*\*  
!ERRHRD(420,MSG4,0); !\*\*\*\* OPTION 4 ERROR 20 \*\*\*\*  
!ERRSOFT(421,MSG3,0); !\*\*\*\* OPTION 4 ERROR 21 \*\*\*\*  
!ERRSOFT(422,MSG3,0); !\*\*\*\* OPTION 4 ERROR 22 \*\*\*\*  
!WRITE DATA:  
!ERRDF(423,MSG1,0); !\*\*\*\* OPTION 4 ERROR 23 \*\*\*\*  
!ERRDF(424,MSG1,0); !\*\*\*\* OPTION 4 ERROR 24 \*\*\*\*  
!ERRDF(425,MSG1,0); !\*\*\*\* OPTION 4 ERROR 25 \*\*\*\*  
!MARCHING UP:

235 ;MLX4

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (3)

VARIABLES AND CONSTANTS

```
236 :
237 :
238 : 1693
239 : 1694          !CHECK OR READ DATA:
240 : 1695
241 : 1696          !ERRDF(426,MSG1,0);  !**** OPTION 4 ERROR 26 ****
242 : 1697          !ERRDF(427,MSG1,0);  !**** OPTION 4 ERROR 27 ****
243 : 1698          !ERRDF(428,MSG1,0);  !**** OPTION 4 ERROR 28 ****
244 : 1699          !ERRDF(429,MSG1,0);  !**** OPTION 4 ERROR 29 ****
245 : 1700          !ERRDF(430,MSG2,0);  !**** OPTION 4 ERROR 30 ****
246 : 1701          !ERRHRD(431,MSG4,0);  !**** OPTION 4 ERROR 31 ****
247 : 1702          !ERRSOFT(432,MSG3,0); !**** OPTION 4 ERROR 32 ****
248 : 1703          !ERRSOFT(433,MSG3,0); !**** OPTION 4 ERROR 33 ****
249 : 1704
250 : 1705
251 : 1706
252 : 1707          ! OPTION 5:
253 : 1708
254 : 1709
255 : 1710          !'RAND1' ROUTINE:
256 : 1711
257 : 1712          !WRITE COMMAND:
258 : 1713
259 : 1714          !ERRDF(5101,MSG1,0); !**** OPTION 5, RAND1 ERROR 01 ****
260 : 1715          !ERRDF(5102,MSG1,0); !**** OPTION 5, RAND1 ERROR 02 ****
261 : 1716          !ERRDF(5103,MSG1,0); !**** OPTION 5, RAND1 ERROR 03 ****
262 : 1717
263 : 1718          !CHECK OR READ:
264 : 1719
265 : 1720          !ERRDF(5104,MSG1,0); !**** OPTION 5, RAND1 ERROR 04 ****
266 : 1721          !ERRDF(5105,MSG1,0); !**** OPTION 5, RAND1 ERROR 05 ****
267 : 1722          !ERRDF(5106,MSG1,0); !**** OPTION 5, RAND1 ERROR 06 ****
268 : 1723          !ERRDF(5107,MSG1,0); !**** OPTION 5, RAND1 ERROR 07 ****
269 : 1724          !ERRDF(5108,MSG2,0); !**** OPTION 5, RAND1 ERROR 08 ****
270 : 1725          !ERRHRD(5109,MSG4,0); !**** OPTION 5, RAND1 ERROR 09 ****
271 : 1726          !ERRSOFT(5110,MSG3,0); !**** OPTION 5, RAND1 ERROR 10 ****
272 : 1727          !ERRSOFT(5111,MSG3,0); !**** OPTION 5, RAND1 ERROR 11 ****
273 : 1728
274 : 1729          !'RAND2' ROUTINE:
275 : 1730
276 : 1731          !WRITE COMMAND:
277 : 1732
278 : 1733          !ERRDF(5201,MSG1,0); !**** OPTION 5, RAND2 ERROR 01 ****
279 : 1734          !ERRDF(5202,MSG1,0); !**** OPTION 5, RAND2 ERROR 02 ****
280 : 1735          !ERRDF(5203,MSG1,0); !**** OPTION 5, RAND2 ERROR 03 ****
281 : 1736
282 : 1737          !CHECK OR READ:
283 : 1738
284 : 1739          !ERRDF(5204,MSG1,0); !**** OPTION 5, RAND2 ERROR 04 ****
285 : 1740          !ERRDF(5205,MSG1,0); !**** OPTION 5, RAND2 ERROR 05 ****
286 : 1741          !ERRDF(5206,MSG1,0); !**** OPTION 5, RAND2 ERROR 06 ****
287 : 1742          !ERRDF(5207,MSG1,0); !**** OPTION 5, RAND2 ERROR 07 ****
288 : 1743          !ERRDF(5208,MSG2,0); !**** OPTION 5, RAND2 ERROR 08 ****
289 : 1744          !ERRHRD(5209,MSG4,0); !**** OPTION 5, RAND2 ERROR 09 ****
```

291 ;MLX4

23-Oct-1980 15:01:43

TOPS-20 Bliss-16 V2(206)

292 ;

VARIABLES AND CONSTANTS

23-Oct-1980 14:57:05

PA:<ROSEN>MLX4.BLI.1 (3)

293

294

1745

.ERRSOFT(5210,MSG3,0); !\*\*\*\* OPTION 5, RAND2 ERROR 10 \*\*\*\*

295

1746

!ERRSOFT(5211,MSG3,0); !\*\*\*\* OPTION 5, RAND2 ERROR 11 \*\*\*\*

296

1747

297

1748

. 'RAND3' ROUTINE:

298

1749

299

1750

!WRITE COMMAND:

300

1751

301

1752

!ERRDF(5301,MSG1,0); !\*\*\*\* OPTION 5, RAND3 ERROR 01 \*\*\*\*

302

1753

.ERRDF(5302,MSG1,0); !\*\*\*\* OPTION 5, RAND3 ERROR 02 \*\*\*\*

303

1754

.ERRDF(5303,MSG1,0); !\*\*\*\* OPTION 5, RAND3 ERROR 03 \*\*\*\*

304

1755

305

1756

!CHECK OR READ:

306

1757

307

1758

!ERRDF(5304,MSG1,0); !\*\*\*\* OPTION 5, RAND3 ERROR 04 \*\*\*\*

308

1759

!ERRDF(5305,MSG1,0); !\*\*\*\* OPTION 5, RAND3 ERROR 05 \*\*\*\*

309

1760

!ERRDF(5306,MSG1,0); !\*\*\*\* OPTION 5, RAND3 ERROR 06 \*\*\*\*

310

1761

!ERRDF(5307,MSG1,0); !\*\*\*\* OPTION 5, RAND3 ERROR 07 \*\*\*\*

311

1762

!ERRDF(5308,MSG2,0); !\*\*\*\* OPTION 5, RAND3 ERROR 08 \*\*\*\*

312

1763

!ERRHRD(5309,MSG4,0); !\*\*\*\* OPTION 5, RAND3 ERROR 09 \*\*\*\*

313

1764

!ERRSOFT(5310,MSG3,0); !\*\*\*\* OPTION 5, RAND3 ERROR 10 \*\*\*\*

314

1765

!ERRSOFT(5311,MSG3,0); !\*\*\*\* OPTION 5, RAND3 ERROR 11 \*\*\*\*

315

1766

316

1767

! 'RAND4' ROUTINE:

317

1768

318

1769

!WRITE COMMAND:

319

1770

320

1771

!ERRDF(5401,MSG1,0); !\*\*\*\* OPTION 5, RAND4 ERROR 01 \*\*\*\*

321

1772

!ERRDF(5402,MSG1,0); !\*\*\*\* OPTION 5, RAND4 ERROR 02 \*\*\*\*

322

1773

!ERRDF(5403,MSG1,0); !\*\*\*\* OPTION 5, RAND4 ERROR 03 \*\*\*\*

323

1774

324

1775

!CHECK OR READ:

325

1776

326

1777

!ERRDF(5404,MSG1,0); !\*\*\*\* OPTION 5, RAND4 ERROR 04 \*\*\*\*

327

1778

!ERRDF(5405,MSG1,0); !\*\*\*\* OPTION 5, RAND4 ERROR 05 \*\*\*\*

328

1779

!ERRDF(5406,MSG1,0); !\*\*\*\* OPTION 5, RAND4 ERROR 06 \*\*\*\*

329

1780

!ERRDF(5407,MSG1,0); !\*\*\*\* OPTION 5, RAND4 ERROR 07 \*\*\*\*

330

1781

!ERRDF(5408,MSG2,0); !\*\*\*\* OPTION 5, RAND4 ERROR 08 \*\*\*\*

331

1782

!ERRHRD(5409,MSG4,0); !\*\*\*\* OPTION 5, RAND4 ERROR 09 \*\*\*\*

332

1783

!ERRSOFT(5410,MSG3,0); !\*\*\*\* OPTION 5, RAND4 ERROR 10 \*\*\*\*

333

1784

!ERRSOFT(5411,MSG3,0); !\*\*\*\* OPTION 5, RAND4 ERROR 11 \*\*\*\*

334

1785

335

1786

336

1787

337

1788

338

1789

CODES FOR WHY A UNIT WAS DRGPPED:

339

1790

340

1791

341

1792

CODE\_1 = 1, !DRIVE NOT POWERED UP

342

1793

CODE\_2 = 2, !DRIVE NOT AN ML11 UNIT

343

1794

CODE\_3 = 3, !OPERATOR SET TEST LIMITS INCORRECTLY

344

1795

CODE\_4 = 4, !FAILED ALL RETRIES FOR A NON-FATAL ERROR

345

1796

CODE\_5 = 5, !CONTROLLER FATAL ERROR

```
347 :MLX4
348 :
349 :
350 : 1797 CODE_6 = 6, !DRIVE FATAL ERROR
351 : 1798 CODE_7 = 7, !ECC HARD ERROR
352 : 1799 CODE_8 = 8, !ECC LOGIC FAILED TO DETECT ERROR
353 : 1800
354 : 1801
355 : 1802 OWN
356 : 1803
357 : 1804 WBUFF :VECTOR[BUFSIZ] VOLATILE, !IMPORTANT -- WBUFF AND RBUFF MUST
358 : 1805 RBUFF :VECTOR[BUFSIZ] VOLATILE, .BE CONTIGUOUS AND IN THAT ORDER.!
359 : 1806
360 : 1807 WPTR : VOLATILE,
361 : 1808 RPTR : VOLATILE,
362 : 1809
363 : 1810 QUICK : VOLATILE,
364 : 1811 PATTERN : VOLATILE,
365 : 1812 DATA_COUNT : VOLATILE,
366 : 1813 COMP_COUNT : VOLATILE,
367 : 1814
368 : 1815 EOP_COUNT : VOLATILE,
369 : 1816
370 : 1817 BASE_ADDR,
371 : 1818 VEC,
372 : 1819 BR_LEVEL,
373 : 1820
374 : 1821 SOFTS: BLOCKVECTOR[8,16], !COUNTS FOR 8 LUNS, 16 ARRAYS EACH
375 : 1822 HARDS: BLOCKVECTOR[8,16], !COUNTS FOR 8 LUNS, 16 ARRAYS EACH
376 : 1823 TRIES: BLOCKVECTOR[8,16], !COUNTS FOR 8 LUNS, 16 ARRAYS EACH
377 : 1824
378 : 1825 WR_COUNT, !# BYTES TRANSFERED VIA 'WRITE'
379 : 1826 WR_THOUSANDS, !THOUSANDS OF BYTES
380 : 1827 WR_MILLIONS, !MILLIONS OF BYTES
381 : 1828
382 : 1829 RD_COUNT, !# BYTES TRANSFERED VIA 'READ'
383 : 1830 RD_THOUSANDS,
384 : 1831 RD_MILLIONS,
385 : 1832
386 : 1833 WC_COUNT, !# BYTES TRANSFERED VIA 'WRITE CHECK'
387 : 1834 WC_THOUSANDS,
388 : 1835 WC_MILLIONS,
389 : 1836
390 : 1837 I_AM_DONE,
391 : 1838 RETRYING,
392 : 1839 BOARD,
393 : 1840 BANK;
394 : 1841
395 : 1842
396 : 1843
397 : 1844 GLOBAL
398 : 1845
399 : 1846 ML_REG:VECTOR[NUM REGS] VOLATILE,
400 : 1847 PTABLE_ADDR:VECTOR[8] VOLATILE,
401 : 1848 DRIVE_STATUS:BITVECTOR[8] VOLATILE,
```

```

403 ;MLX4
404 :
405 :
406 : 1849 DROPT DRIVES:BITVECTOR[8] VOLATILE,
407 : 1850 WHY_DROPT:VECTOR[8,BYTE],
408 : 1851 NUM_DRIVES,
409 : 1852 LOW_SECT:VECTOR[8] VOLATILE,
410 : 1853 TOP_SECT:VECTOR[8] VOLATILE;
411 : 1854
412 : 1855
413 : 1856
414 : 1857 MACRO
415 : 1858
416 : 1859
417 : 1860 : TO CALCULATE THE TEST RANGES FOR A PARTICULAR LOGICAL UNIT:
418 : 1861 :
419 : 1862
420 : 1863 LOWEST = .LOW_SECT[.LUN]%, !THE FIRST SECTOR TO TEST
421 : 1864 HIGHEST = .TOP_SECT[.LUN]%, !THE LAST SECTOR TO TEST
422 : 1865
423 : 1866
424 : 1867 : ADDRESS IN MAIN MEMORY THAT CONTAINS THE PHYSICAL DRIVE
425 : 1868 : NUMBER THAT CORRESPONDS TO A PARTICULAR LOGICAL UNIT:
426 : 1869
427 : 1870
428 : 1871 DRIVE = (.PTABLE_ADDR[.LUN] + 6)%,
429 : 1872
430 : 1873
431 : 1874
432 : 1875 : ML-11 REGISTER NAMES:
433 : 1876
434 : 1877
435 : 1878 MLCS1 = .ML_REG[0]%, !CONTROL AND STATUS REGISTER 1
436 : 1879 MLWC = .ML_REG[1]%, !WORD COUNT REGISTER
437 : 1880 MLBA = .ML_REG[2]%, !UNIBUS ADDRESS REGISTER
438 : 1881 MLDA = .ML_REG[3]%, !DESIRED ADDRESS REGISTER
439 : 1882 MLCS2 = .ML_REG[4]%, !CONTROL AND STATUS REGISTER 2
440 : 1883 MLDS = .ML_REG[5]%, !DRIVE STATUS REGISTER
441 : 1884 MLER = .ML_REG[6]%, !ERROR REGISTER
442 : 1885 MLAS = .ML_REG[7]%, !ATTENTION SUMMARY REGISTER
443 : 1886 MLPA = .ML_REG[8]%, !PROM ADDRESS REGISTER
444 : 1887 MLDB = .ML_REG[9]%, !DATA BUFFER REGISTER
445 : 1888 MLMA = .ML_REG[10]%, !MAINTENANCE REGISTER
446 : 1889 MLDT = .ML_REG[11]%, !DRIVE TYPE REGISTER
447 : 1890 MLSN = .ML_REG[12]%, !SERIAL NUMBER REGISTER
448 : 1891 MLE1 = .ML_REG[13]%, !ECC REGISTER 1
449 : 1892 MLE2 = .ML_REG[14]%, !ECC REGISTER 2
450 : 1893 MLD1 = .ML_REG[15]%, !DATA DIAGNOSTIC REGISTER 1
451 : 1894 MLD2 = .ML_REG[16]%, !DATA DIAGNOSTIC REGISTER 2
452 : 1895 MLEE = .ML_REG[17]%, !ECC ERROR REGISTER
453 : 1896 MLEL = .ML_REG[18]%, !ECC ERROR LOCATION REGISTER
454 : 1897 MLPD = .ML_REG[19]%, !PROM DATA REGISTER
455 : 1898 MLBAE = .ML_REG[20]%, !BUS ADDRESS EXTENSION REGISTER
456 : 1899 MLCS3 = .ML_REG[21]%, !CONTROL AND STATUS REGISTER 3
457 : 1900

```

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (3)



459 :MLX4

VARIABLES AND CONSTANTS

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (3)

460 :  
461 :  
462 : 1901  
463 : 1902  
464 : 1903  
465 : 1904  
466 : 1905  
467 : 1906  
468 : 1907  
469 : 1908  
470 : 1909  
471 : 1910  
472 : 1911  
473 : 1912  
474 : 1913  
475 : 1914  
476 : 1915  
477 : 1916  
478 : 1917  
479 : 1918  
480 : 1919  
481 : 1920  
482 : 1921  
483 : 1922  
484 : 1923  
485 : 1924  
486 : 1925  
487 : 1926  
488 : 1927  
489 : 1928  
490 : 1929  
491 : 1930  
492 : 1931  
493 : 1932  
494 : 1933  
495 : 1934  
496 : 1935  
497 : 1936  
498 : 1937  
499 : 1938  
500 : 1939  
501 : 1940  
502 : 1941  
503 : 1942  
504 : 1943  
505 : 1944  
506 : 1945  
507 : 1946  
508 : 1947  
509 : 1948  
510 : 1949  
511 : 1950  
512 : 1951  
513 : 1952

BIT ASSIGNMENTS:

MLCS1 BITS:

SC	= (MLCS1)<15,1>X,	!SPECIAL CONDITION
TRE	= (MLCS1)<14,1>X,	!TRANSFER ERROR
MCPE	= (MLCS1)<13,1>X,	!MASSBUS CONTROL BUS PARITY ERROR
DVA	= (MLCS1)<11,1>X,	!DRIVE AVAILABLE
RDY	= (MLCS1)<7,1>X,	!READY
IE	= (MLCS1)<6,1>X,	!INTERRUPT ENABLE
FUNC	= (MLCS1)<0,6>X,	!FUNCTION (COMMAND) CODE AND GO BIT

MLCS2 BITS:

DLT	= (MLCS2)<15,1>X,	!DATA LATE
WCE	= (MLCS2)<14,1>X,	!WRITE CHECK ERROR
PE	= (MLCS2)<13,1>X,	!PARITY ERROR
NED	= (MLCS2)<12,1>X,	!NON-EXISTENT DRIVE
NEM	= (MLCS2)<11,1>X,	!NON-EXISTENT MEMORY
PGE	= (MLCS2)<10,1>X,	!PROGRAM ERROR
MXF	= (MLCS2)<9,1>X,	!MISSED TRANSFER
MDPE	= (MLCS2)<8,1>X,	!MASSBUS DATA BUS PARITY ERROR
ORDY	= (MLCS2)<7,1>X,	!OUTPUT READY
IRDY	= (MLCS2)<6,1>X,	!INPUT READY
CLR	= (MLCS2)<5,1>X,	!CONTROLLER CLEAR
PAT	= (MLCS2)<4,1>X,	!PARITY TEST
BAI	= (MLCS2)<3,1>X,	!UNIBUS ADDRESS INCREMENT INHIBIT
UNIT	= (MLCS2)<0,3>X,	!UNIT SELECT

MLDS BITS:

ATA	= (MLDS)<15,1>X,	!ATTENTION ACTIVE
ERR	= (MLDS)<14,1>X,	!ERROR SUMMARY
MOL	= (MLDS)<12,1>X,	!MEDIUM ON LINE
LBT	= (MLDS)<10,1>X,	!LAST BLOCK TRANSFERRED
DPR	= (MLDS)<8,1>X,	!DRIVE PRESENT
DRY	= (MLDS)<7,1>X,	!DRIVE READY
VV	= (MLDS)<6,1>X,	!VOLUME VALID

MLER BITS:

515 :MLX4

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (3)

VARIABLES AND CONSTANTS

516 :  
517 :  
518 : 1953  
519 : 1954  
520 : 1955  
521 : 1956  
522 : 1957  
523 : 1958  
524 : 1959  
525 : 1960  
526 : 1961  
527 : 1962  
528 : 1963  
529 : 1964  
530 : 1965  
531 : 1966  
532 : 1967  
533 : 1968  
534 : 1969  
535 : 1970  
536 : 1971  
537 : 1972  
538 : 1973  
539 : 1974  
540 : 1975  
541 : 1976  
542 : 1977  
543 : 1978  
544 : 1979  
545 : 1980  
546 : 1981  
547 : 1982  
548 : 1983  
549 : 1984  
550 : 1985  
551 : 1986  
552 : 1987  
553 : 1988  
554 : 1989  
555 : 1990  
556 : 1991  
557 : 1992  
558 : 1993  
559 : 1994  
560 : 1995  
561 : 1996  
562 : 1997  
563 : 1998  
564 : 1999  
565 : 2000  
566 : 2001  
567 : 2002  
568 : 2003  
569 : 2004

DCK = (MLER)<15,1>%  
UNS = (MLER)<14,1>%  
OPI = (MLER)<13,1>%  
IAE = (MLER)<10,1>%  
AOE = (MLER)<9,1>%  
ECH = (MLER)<6,1>%  
DPAR = (MLER)<5,1>%  
CPAR = (MLER)<3,1>%  
RMR = (MLER)<2,1>%  
ILR = (MLER)<1,1>%  
ILF = (MLER)<0,1>%

!DATA CHECK  
!DRIVE UNSAFE  
!OPERATION INCOMPLETE  
!INVALID ADDRESS ERROR  
!ADDRESS OVERFLOW ERROR  
!ECC HARD ERROR  
!DATA PARITY ERROR  
!CONTROL PARITY ERROR  
!REGISTER MODIFICATION REFUSED  
!ILLEGAL REGISTER  
!ILLEGAL FUNCTION

MLMR BITS:

SZ = (MLMR)<11,5>%  
ARR\_TYP = (MLMR)<10,1>%  
TRT = (MLMR)<8,2>%  
REF\_MAR = (MLMR)<7,1>%  
PROM\_RW = (MLMR)<6,1>%  
PROM\_DIS = (MLMR)<5,1>%  
DAT\_CLK = (MLMR)<4,1>%  
DAT\_DM = (MLMR)<3,1>%  
DCK\_EN = (MLMR)<2,1>%  
ECC\_DIS = (MLMR)<1,1>%  
ECC\_DM = (MLMR)<0,1>%

!SYSTEM SIZE (# ARRAY CARDS)  
!ARRAY TYPE (0=16K;1=64K CHIPS)  
!TRANSFER RATE  
!REFRESH MARGIN  
!PROM READ/WRITE  
!PROM DISABLE  
!DATA CLOCK  
!DATA DIAGNOSTIC MODE  
!DATA CHECK ENABLE  
!ECC DISABLE  
!ECC DIAGNOSTIC MODE

MLSN BITS:

SN3 = (MLSN)<12,4>%  
SN2 = (MLSN)<8,4>%  
SN1 = (MLSN)<4,4>%  
SNO = (MLSN)<0,4>%

!HIGH ORDER DECADE  
!THIRD DECADE  
!SECOND DECADE  
!LOW ORDER DECADE

MLEE BITS:

UNC = (MLEE)<15,1>%  
SGL = (MLEE)<14,1>%  
CRC = (MLEE)<13,1>%  
CHAN = (MLEE)<6,6>%  
EFUN = (MLEE)<0,6>%

!UNCORRECTABLE ERROR  
!SINGLE ERROR  
!CRC ERROR  
!CHANNEL IN ERROR  
!ERROR FUNCTION

EXTERNAL

571 ;MLX4  
572 :  
573 :  
574 :  
575 :  
576 :  
577 :  
578 :  
579 :  
580 :  
581 :  
582 :  
583 :  
584 :  
585 :  
586 :  
587 :  
588 :  
589 :  
590 :  
591 :  
592 :  
593 :  
594 :  
595 :  
596 :  
597 :  
598 :  
599 :  
600 :  
601 :  
602 :

VARIABLES AND CONSTANTS

! HEADER INFORMATION:

LSUNIT, LSLUN,

! ALL OF THE SOFTWARE P-TABLE LOCATIONS:

LIMIT, RANGE, LSECT, TSECT, ONLY,  
DROPNE, DROP1, DROP2, DROP3, DROP4, DROPS,  
MARPAT, REFRESH, ECCDIS, EOPSUM, ERPOUT,

! RANDOM NUMBER GENERATION VALUES:

SEED1, SEED2, SEED3, RANDOM;

EXTERNAL ROUTINE

RN : NOVALUE; !FOR 16-BIT RANDOM NUMBER GENERATION

EQUALS

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (3)

604 :MLX4  
 605 :  
 606 :  
 607 :  
 608 :  
 609 :  
 610 :  
 611 :  
 612 :  
 613 :  
 614 :  
 615 :  
 616 :  
 617 :  
 618 :  
 619 :  
 620 :  
 621 :  
 622 :  
 623 :  
 624 :  
 625 :  
 626 :  
 627 :  
 628 :  
 629 :  
 630 :  
 631 :  
 632 :  
 633 :  
 634 :  
 635 :  
 636 :  
 637 :  
 638 :  
 639 :  
 640 :  
 641 :  
 642 :  
 643 :  
 644 :  
 645 :  
 646 :  
 647 :

VARIABLES AND CONSTANTS

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA-<ROSEN>MLX4.BLI.1 (4)

2034  
 2035  
 2036  
 2037  
 2038  
 2039  
 2040  
 2041  
 2042  
 2043  
 2044  
 2045  
 2046  
 2047  
 2048  
 2049  
 2050  
 2051  
 2052  
 2053  
 2054  
 2055  
 2056  
 2057  
 2058  
 2059  
 2060  
 2061  
 2062  
 2063  
 2064  
 2065  
 2066  
 2067  
 2068  
 2069  
 2070  
 2071  
 2072  
 2073  
 2074

THE PATTERN TABLE:

REGULAR PATTERNS

PATTERN NUMBER	THE TWO ASSOCIATED COUNTS		VALUE OF DATA	VALUE OF COMP
1	0 NIBBLES OF DATA,	1024 NIBBLES OF COMP	0101	1010
2	1 NIBBLE OF DATA,	1 NIBBLE OF COMP	0101	1010
3	4 NIBBLES OF DATA,	4 NIBBLES OF COMP	0101	1010
4	9 NIBBLES OF DATA,	9 NIBBLES OF COMP	0101	1010
5	1024 NIBBLES OF DATA,	1024 NIBBLES OF COMP	0101	1010
6	0 NIBBLES OF DATA,	1024 NIBBLES OF COMP	0000	1111
7	1 NIBBLE OF DATA,	1 NIBBLE OF COMP	0000	1111
8	4 NIBBLES OF DATA,	4 NIBBLES OF COMP	0000	1111
9	9 NIBBLES OF DATA,	9 NIBBLES OF COMP	0000	1111
10	1024 NIBBLES OF DATA,	1024 NIBBLES OF COMP	0000	1111

COMPLEMENT PATTERNS

PATTERN NUMBER	THE TWO ASSOCIATED COUNTS		VALUE OF DATA	VALUE OF COMP
- 1	0 NIBBLES OF DATA,	1024 NIBBLES OF COMP	1010	0101
- 2	1 NIBBLE OF DATA,	1 NIBBLE OF COMP	1010	0101
- 3	4 NIBBLES OF DATA,	4 NIBBLES OF COMP	1010	0101
- 4	9 NIBBLES OF DATA,	9 NIBBLES OF COMP	1010	0101
- 5	1024 NIBBLES OF DATA,	1024 NIBBLES OF COMP	1010	0101
- 6	0 NIBBLES OF DATA,	1024 NIBBLES OF COMP	1111	0000
- 7	1 NIBBLE OF DATA,	1 NIBBLE OF COMP	1111	0000
- 8	4 NIBBLES OF DATA,	4 NIBBLES OF COMP	1111	0000
- 9	9 NIBBLES OF DATA,	9 NIBBLES OF COMP	1111	0000
-10	1024 NIBBLES OF DATA,	1024 NIBBLES OF COMP	1111	0000

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (5)

649 :MLX4  
650 :  
651 :  
652 :  
653 :  
654 :  
655 :  
656 :  
657 :  
658 :  
659 :  
660 :  
661 :  
662 :  
663 :  
664 :  
665 :  
666 :  
667 :  
668 :  
669 :  
670 :  
671 :  
672 :  
673 :  
674 :  
675 :  
676 :  
677 :  
678 :  
679 :  
680 :  
681 :  
682 :  
683 :  
684 :  
685 :  
686 :  
687 :  
688 :  
689 :  
690 :  
691 :  
692 :  
693 :  
694 :

VARIABLES AND CONSTANTS

2075 FIELD  
2076 PATMAP =  
2077 SET  
2078 COUNT1 = [0,0,16,0],  
2079 COUNT2 = [1,0,16,0]  
2080 TES;

GLOBAL

2084 PATTABL: BLOCKVECTOR [NUM\_PATS/2,2] FIELD(PATMAP)  
2085 PRESET (  
2086 [0,COUNT1] = %DECIMAL'0' ; !FOR PATTERNS 1, -1, 6, -6  
2087 [0,COUNT2] = %DECIMAL'1024' ;  
2089 [1,COUNT1] = %DECIMAL'1' ; !FOR PATTERNS 2, -2, 7, -7  
2090 [1,COUNT2] = %DECIMAL'1' ;  
2092 [2,COUNT1] = %DECIMAL'4' ; !FOR PATTERNS 3, -3, 8, -8  
2093 [2,COUNT2] = %DECIMAL'4' ;  
2095 [3,COUNT1] = %DECIMAL'9' ; !FOR PATTERNS 4, -4, 9, -9  
2096 [3,COUNT2] = %DECIMAL'9' ;  
2098 [4,COUNT1] = %DECIMAL'1024' ; !FOR PATTERNS 5, -5, 10, -10  
2099 [4,COUNT2] = %DECIMAL'1024' ;  
2100 );  
2101 );

BIND

! :  
! : DEFINITIONS OF LOCATIONS WITHIN THE WRITE AND READ BUFFERS:  
! :

2110 WDBUFF = WBUFF, !256-WORD WRITE DATA BUFFER  
2111 WCBUFF = WBUFF + 512, !256-WORD WRITE COMP BUFFER  
2112  
2113 RDBUFF = RBUFF, !256-WORD READ DATA BUFFER  
2114 RCBUFF = RBUFF + 512, !256-WORD READ COMP BUFFER  
2115  
2116 END\_WBUFF = (WBUFF + BUFSIZ\*2), !JUST BEYOND END OF FULL WRITE BUFFER  
2117 END\_RBUFF = (RBUFF + BUFSIZ\*2), !JUST BEYOND END OF FULL READ BUFFER

```

696 :MLX4
697 :
698 :
699 : 2118 %SBTTL 'MESSAGES AND PRINT FORMATS'
700 : 2119
701 : 2120
702 : 2121
703 : 2122
704 : 2123
705 : 2124
706 : 2125
707 : 2126
708 : 2127
709 : 2128
710 : 2129
711 : 2130
712 : 2131
713 : 2132
714 : 2133
715 : 2134
716 : 2135
717 : 2136
718 : 2137
719 : 2138
720 : 2139
721 : 2140
722 : 2141
723 : 2142
724 : 2143
725 : 2144
726 : 2145
727 : 2146
728 : 2147
729 : 2148
730 : 2149
731 : 2150
732 : 2151
733 : 2152
734 : 2153
735 : 2154
736 : 2155
737 : 2156
738 : 2157
739 : 2158
740 : 2159
741 : 2160
742 : 2161
743 : 2162
744 : 2163
745 : 2164
746 : 2165
747 : 2166
748 : 2167
749 : 2168
750 : 2169

MESSAGES AND PRINT FORMATS

SPECIAL PURPOSE FORMATS (WITH SAMPLE PRINTOUTS):

FMT1A = UPLIT(%ASCIZ'%N%T%S2%D2'),
!'PATTERN NUMBER XX'

FMT1B = UPLIT(%ASCIZ'%S7%T%S%A-%D2'),
!' PATTERN NUMBER -XX'

FMT2 = UPLIT(%ASCIZ'%S2%T'),
!' (NOT POWERED UP)',
!' (NOT AN ML11 UNIT)',
!' (OPERATOR SELECTED TEST LIMITS INCORRECTLY)',
!' (ALL RETRIES FAILED FOR A NON-FATAL ERROR)',
!' (CONTROLLER FATAL ERROR)',
!' (DRIVE FATAL ERROR)',
!' (ECC HARD ERROR)',
!' UP',
!' DOWN',
!' --> RUN ML11 PROM MAINTENANCE PROGRAM'

FMT3 = UPLIT(%ASCIZ'%N%A**%S%T%S%T%D3%S%A**'),
!'** END PASS X **'

FMT4A = UPLIT(%ASCIZ'%N2%T%A:%S%D1%S4%T%A:%S%D1'),
!'LOGICAL UNIT: X DRIVE: Y'

FMT4B = UPLIT(%ASCIZ'%S4%T%A:%S%06'),
!' SERIAL #: ZZZZZZ',
!' CSR ADDRESS: YXXXXX'

FMT4C = UPLIT(%ASCIZ'%S4%T%A:%S%D1%D1%D1%D1'),
!' SERIAL #: DDDD'

FMT5 = UPLIT(%ASCIZ'%N%T%S3%ASECTORS UNDER TEST:%S%06%S%ATO%S%06'),
!'ML11-X SECTORS UNDEP TEST: XXXXXX TO YYYYYY'

FMT6 = UPLIT(%ASCIZ'%N%ABEGAN %D4%A WORD%S%T%A AT%S%T%S%06'),
!'BEGAN YYYY WORD WRITE AT SECTOR ZZZZZZ',
!'BEGAN YYYY WORD READ AT SECTOR ZZZZZZ',
!'BEGAN YYYY WORD WRITE CHECK AT SECTOR ZZZZZZ'

FMT7 = UPLIT(%ASCIZ'%N%S2%T%A:%S%D5'),
!' SOFT ERROR COUNT: DDDDD',
!' HARD ERROR COUNT: DDDDD',
!' TRANSFER RETRIES: DDDDD'

FMT8 = UPLIT(%ASCIZ'%N%ATTRANSFER RATE: %T%A MBYTES/SECOND'),
!'TRANSFER RATE: X MBYTES/SECOND'

```

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (6)

752 :MLX4

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (6)

MESSAGES AND PRINT FORMATS

753 :  
754 :  
755 : 2170  
756 : 2171  
757 : 2172  
758 : 2173  
759 : 2174  
760 : 2175  
761 : 2176  
762 : 2177  
763 : 2178  
764 : 2179  
765 : 2180  
766 : 2181  
767 : 2182  
768 : 2183  
769 : 2184  
770 : 2185  
771 : 2186  
772 : 2187  
773 : 2188  
774 : 2189  
775 : 2190  
776 : 2191  
777 : 2192  
778 : 2193  
779 : 2194  
780 : 2195  
781 : 2196  
782 : 2197  
783 : 2198  
784 : 2199  
785 : 2200  
786 : 2201  
787 : 2202  
788 : 2203  
789 : 2204  
790 : 2205  
791 : 2206  
792 : 2207  
793 : 2208  
794 : 2209  
795 : 2210  
796 : 2211  
797 : 2212  
798 : 2213  
799 : 2214  
800 : 2215  
801 : 2216  
802 : 2217  
803 : 2218  
804 : 2219  
805 : 2220  
806 : 2221

FMT9 = UPLIT(%ASCIZ'XNXS4%AARRAY%D3%A:XS%D5'),  
! ' ARRAY XX: YYYYY'

FMT10A = UPLIT(%ASCIZ'XNX%A:XS2XTXS%06XS2%ABOARD%D3XS2%ABANK%D2'),  
! 'FAILED: SECTOR XXXXXX BOARD YY BANK Z

FMT10B = UPLIT(%ASCIZ'XS2%ABIT%D3'),  
! ' BIT 00'

FMT11 = UPLIT(%ASCIZ'XNX%T%XS%06%A EXCEEDS XT%XS%06'),  
! 'TOP SECTOR OF XXXXXX EXCEEDS SYSTEM LIMIT OF YYYYYY'  
! 'LOW SECTOR OF XXXXXX EXCEEDS TOP SECTOR OF YYYYYY'

FMT12A = UPLIT(%ASCIZ'XNX%AGOOD DATA: %06%A AT LOCATION %06'),  
! 'GOOD DATA: XXXXXX AT LOCATION YYYYYY'

FMT12B = UPLIT(%ASCIZ'XNX%ABAD DATA: %06%A AT LOCATION %06'),  
! 'BAD DATA: XXXXXX AT LOCATION YYYYYY'

FMT13 = UPLIT(%ASCIZ'XNX%AARRAY%D3XS2XT'),  
! 'ARRAY XX --> RUN ML11 PROM MAINTENANCE PROGRAM'

FMT14 = UPLIT(%ASCIZ'XNX%D5XSXT'),  
! 'XXXXX MBYTES WRITTEN'  
! 'XXXXX MBYTES READ'  
! 'XXXXX MBYTES WRITE CHECKED'

FMT15 = UPLIT(%ASCIZ'XS2XT'),  
! ' ECH'  
! ' NED'  
! (ANY OF THE ERROR BITS)

CRLF = UPLIT(%ASCIZ'XN'),

! :  
! : MESSAGE MAPS:  
! :  
SAY1 = UPLIT(%ASCIZ'XNX%T'),  
SAY2 = UPLIT(%ASCIZ'XNX%T%XSXT'),  
SAY3 = UPLIT(%ASCIZ'XNX%T%XSXT%XSXT'),  
SAY4 = UPLIT(%ASCIZ'XNX%T%XSXT%XSXT%XSXT'),  
SAY5 = UPLIT(%ASCIZ'XNX%T%XSXT%XSXT%XSXT%XSXT'),

! :  
! : WORDS:  
! :  
WRD2 = UPLIT(%ASCIZ'BEGIN'),  
WRD3 = UPLIT(%ASCIZ'END'),  
WRD4 = UPLIT(%ASCIZ'PASS'),

808 :MLX4

MESSAGES AND PRINT FORMATS

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
PA:<ROSER>MLX4.BLI.1 (6)

809 :  
810 :  
811 : 2222 WRD6 = UPLIT(%ASCIZ'ML11-A'),  
812 : 2223 WRD7 = UPLIT(%ASCIZ'ML11-B'),  
813 : 2224 WRD11 = UPLIT(%ASCIZ'DRIVE'),  
814 : 2225 WRD15 = UPLIT(%ASCIZ'SECTOR'),  
815 : 2226 WRD16 = UPLIT(%ASCIZ'WRITE'),  
816 : 2227 WRD17 = UPLIT(%ASCIZ'READ'),  
817 : 2228 WRD18 = UPLIT(%ASCIZ'RETRY'),  
818 : 2229 WRD19 = UPLIT(%ASCIZ'SUCCEEDED'),  
819 : 2230 WRD20 = UPLIT(%ASCIZ'FAILED'),  
820 : 2231 WRD21 = UPLIT(%ASCIZ'DROPPED'),  
821 : 2232 WRD24 = UPLIT(%ASCIZ'UP'),  
822 : 2233 WRD25 = UPLIT(%ASCIZ'DOWN'),  
823 : 2234 WRD34 = UPLIT(%ASCIZ'RUNNING'),  
824 : 2235 WRD35 = UPLIT(%ASCIZ'MARCHING: '),  
825 : 2236 WRD36 = UPLIT(%ASCIZ'ENABLED'),  
826 : 2237 WRD37 = UPLIT(%ASCIZ'DISABLED'),  
827 : 2238 WRD38 = UPLIT(%ASCIZ'ECC'),  
828 : 2239 WRD40 = UPLIT(%ASCIZ'WITH'),  
829 : 2240 WRD41 = UPLIT(%ASCIZ'AND'),  
830 : 2241  
831 : 2242  
832 : 2243  
833 : 2244 : ML-11 BITS:  
834 : 2245  
835 : 2246  
836 : 2247  
837 : 2248 MLB2 = UPLIT(%ASCIZ'NED'),  
838 : 2249 MLB3 = UPLIT(%ASCIZ'NEM'),  
839 : 2250 MLB4 = UPLIT(%ASCIZ'PGE'),  
840 : 2251 MLB5 = UPLIT(%ASCIZ'DLT'),  
841 : 2252 MLB6 = UPLIT(%ASCIZ'WCE'),  
842 : 2253 MLB7 = UPLIT(%ASCIZ'PE'),  
843 : 2254 MLB8 = UPLIT(%ASCIZ'MXF'),  
844 : 2255 MLB9 = UPLIT(%ASCIZ'MDPE'),  
845 : 2256 MLB10 = UPLIT(%ASCIZ'MCPE'),  
846 : 2257 MLB11 = UPLIT(%ASCIZ'UNS'),  
847 : 2258 MLB12 = UPLIT(%ASCIZ'IAE'),  
848 : 2259 MLB13 = UPLIT(%ASCIZ'AOE'),  
849 : 2260 MLB14 = UPLIT(%ASCIZ'RMR'),  
850 : 2261 MLB15 = UPLIT(%ASCIZ'ILR'),  
851 : 2262 MLB16 = UPLIT(%ASCIZ'ILF'),  
852 : 2263 MLB17 = UPLIT(%ASCIZ'OPI'),  
853 : 2264 MLB18 = UPLIT(%ASCIZ'DPAR'),  
854 : 2265 MLB19 = UPLIT(%ASCIZ'CPAR'),  
855 : 2266 MLB20 = UPLIT(%ASCIZ'DCK'),  
856 : 2267 MLB21 = UPLIT(%ASCIZ'ECH'),  
857 : 2268 MLB22 = UPLIT(%ASCIZ'CRC'),  
858 : 2269 MLB23 = UPLIT(%ASCIZ'SGL'),  
859 : 2270 MLB24 = UPLIT(%ASCIZ'UNC'),  
860 : 2271  
861 : 2272  
862 : 2273



864 :MLX4

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (6)

-----MESSAGES AND PRINT FORMATS

866  
867 :  
868 :  
869 :  
870 :  
871 :  
872 :  
873 :  
874 :  
875 :  
876 :  
877 :  
878 :  
879 :  
880 :  
881 :  
882 :  
883 :  
884 :  
885 :  
886 :  
887 :  
888 :  
889 :  
890 :  
891 :  
892 :  
893 :  
894 :  
895 :  
896 :  
897 :  
898 :  
899 :  
900 :  
901 :  
902 :  
903 :  
904 :  
905 :  
906 :  
907 :  
908 :  
909 :  
910 :  
911 :  
912 :  
913 :  
914 :  
915 :  
916 :  
917 :  
918 :

2274  
2275  
2276  
2277  
2278  
2279  
2280  
2281  
2282  
2283  
2284  
2285  
2286  
2287  
2288  
2289  
2290  
2291  
2292  
2293  
2294  
2295  
2296  
2297  
2298  
2299  
2300  
2301  
2302  
2303  
2304  
2305  
2306  
2307  
2308  
2309  
2310  
2311  
2312  
2313  
2314  
2315  
2316  
2317  
2318  
2319  
2320  
2321  
2322  
2323  
2324  
2325

ROUTINE NAMES:

RTN0 = UPLIT(%ASCIZ'COMMAND INTEGRITY ROUTINE'),  
RTN1 = UPLIT(%ASCIZ'OPT1'),  
RTN2 = UPLIT(%ASCIZ'OPT2'),  
RTN3 = UPLIT(%ASCIZ'OPT3'),  
RTN4 = UPLIT(%ASCIZ'OPT4'),  
RTN5 = UPLIT(%ASCIZ'OPT5'),  
RTN5A = UPLIT(%ASCIZ'RAND1'), !RANDOM DATA  
RTN5B = UPLIT(%ASCIZ'RAND2'), !DATA & WORD COUNTS  
RTN5C = UPLIT(%ASCIZ'RAND3'), !DATA, WORD COUNTS & SECTORS  
RTN5D = UPLIT(%ASCIZ'RAND4'), !DATA, WORD COUNTS, SECTORS & UNITS

PHRASES:

PHR1 = UPLIT(%ASCIZ'WRITE CHECK'),  
PHR2 = UPLIT(%ASCIZ'QUICK VERIFY'),  
PHR3 = UPLIT(%ASCIZ'REFRESH MARGINING'),  
PHR4 = UPLIT(%ASCIZ'CSR ADDRESS'),  
PHR5 = UPLIT(%ASCIZ'SOFT ERROR COUNT'),  
PHR6 = UPLIT(%ASCIZ'TRANSFER RETRIES'),  
PHR7 = UPLIT(%ASCIZ'LOGICAL UNIT'),  
PHR8 = UPLIT(%ASCIZ'SERIAL #'),  
PHR9 = UPLIT(%ASCIZ'PATTERN NUMBER'),  
PHR10 = UPLIT(%ASCIZ'ERROR BITS SET:'),  
PHR11 = UPLIT(%ASCIZ'SC SET BUT NO SYSTEM ERRORS FOUND'),  
PHR12 = UPLIT(%ASCIZ'NUMBER OF MBYTES TRANSFERED:'),  
PHR13 = UPLIT(%ASCIZ'MBYTES WRITE CHECKED'),  
PHR14 = UPLIT(%ASCIZ'TOP SECTOR OF'),  
PHR15 = UPLIT(%ASCIZ'LOW SECTOR OF'),  
PHR16 = UPLIT(%ASCIZ'SYSTEM LIMIT OF'),  
PHR17 = UPLIT(%ASCIZ'PERFORMANCE SUMMARY'),  
PHR18 = UPLIT(%ASCIZ'HARD ERROR COUNT'),  
PHR19 = UPLIT(%ASCIZ'MBYTES WRITTEN'),  
PHR20 = UPLIT(%ASCIZ'MBYTES READ'),

TRANSFER RATES:

TRT00 = UPLIT(%ASCIZ'2'),  
TRT01 = UPLIT(%ASCIZ'1'),  
TRT10 = UPLIT(%ASCIZ'.5'),  
TRT11 = UPLIT(%ASCIZ'.25'),

920 :MLX4

23-Oct-1980 15:01:43

TOPS-20 Bliss-16 V2(206)

921 :

MESSAGES AND PRINT FORMATS

23-Oct-1980 14:57:05

PA:<ROSEN>MLX4.BLI.1 (6)

922 :

923 :

2326

! DROP MESSAGES:

924 :

2327

925 :

2328

926 :

2329

927 :

2330

928 :

2331

929 :

2332

930 :

2333

931 :

2334

932 :

2335

933 :

2336

934 :

2337

935 :

2338

936 :

2339

937 :

2340

938 :

2341

939 :

2342

940 :

2343

941 :

2344

942 :

2345

943 :

2346

944 :

2347

945 :

2348

CAUSE1 = UPLIT(%ASCIZ'(NOT POWERED UP)'),  
CAUSE2 = UPLIT(%ASCIZ'(NOT AN ML11 UNIT)'),  
CAUSE3 = UPLIT(%ASCIZ'(OPERATOR SELECTED TEST LIMITS INCORRECTLY)'),  
CAUSE4 = UPLIT(%ASCIZ'(ALL RETRIES FAILED FOR A NON-FATAL ERROR)'),  
CAUSE5 = UPLIT(%ASCIZ'(CONTROLLER FATAL ERROR)'),  
CAUSE6 = UPLIT(%ASCIZ'(DRIVE FATAL ERROR)'),  
CAUSE7 = UPLIT(%ASCIZ'(ECC HARD ERROR)'),  
CAUSE8 = UPLIT(%ASCIZ'(ECC LOGIC FAILURE)'),

! DIAGNOSES:

MSG0 = UPLIT(%ASCIZ'INTERRUPT DID NOT OCCUR, BUT THE TRANSFER IS COMPLETE'),  
MSG1 = UPLIT(%ASCIZ'--> RUN ML11 LOGIC TEST'),  
MSG2 = UPLIT(%ASCIZ'--> RUN ML11 PROM MAINTENANCE PROGRAM'),  
MSG3 = UPLIT(%ASCIZ'SOFT ERROR'),  
MSG4 = UPLIT(%ASCIZ'HARD ERROR'),  
MSG5 = UPLIT(%ASCIZ'ECC LOGIC FAILED TO DETECT DATA ERROR');

```

947 :MLX4
948 :
949 :
950 :      2349 %SBTTL 'ML11 INTERRUPT SERVICE ROUTINE'
951 :      2350
952 :      2351 BGNSRV(SERVICE)
953 :      2352
954 :      2353 I_AM_DONE = ACTIVE;
955 :      2354
956 :      2355 ENDSRV
960
961           .TITLE MLX4
962
963

```

```

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)
23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (7)

```

1595 010706	WBUFF: .BLKW	4000
1596 020706	RBUFF: .BLKW	4000
1597 030706	WPTR: .BLKW	1
1598 030710	RPTR: .BLKW	1
1599 030712	QUICK: .BLKW	1
1600 030714	PATTERN: .BLKW	1
1601 030716	DATA.COUNT:	
1602 030716	.BLKW	1
1603 030720	COMP.COUNT:	
1604 030720	.BLKW	1
1605 030722	EOP.COUNT:	
1606 030722	.BLKW	1
1607 030724	BASE.ADDR:	
1608 030724	.BLKW	1
1609 030726	VEC: .BLKW	1
1610 030730	BR.LEVEL:	
1611 030730	.BLKW	1
1612 030732	SOFTS: .BLKW	200
1613 031332	HARDS: .BLKW	200
1614 031732	TRIES: .BLKW	200
1615 032332	WR.COUNT:	
1616 032332	.BLKW	1
1617 032334	WR.THOUSANDS:	

23-Oct-1980 15:01:43 TOPS  
 23-Oct-1980 14:57:05 PA:<

```

1619 ;MLX4
1620 ;
1621 ;
1622 032334 .BLKW 1
1623 032336 WR.MILLIONS:
1624 032336 .BLKW 1
1625 032340 RD.COUNT:
1626 032340 .BLKW 1
1627 032342 RD.THOUSANDS:
1628 032342 .BLKW 1
1629 032344 RD.MILLIONS:
1630 032344 .BLKW 1
1631 032346 WC.COUNT:
1632 032346 .BLKW 1
1633 032350 WC.THOUSANDS:
1634 032350 .BLKW 1
1635 032352 WC.MILLIONS:
1636 032352 .BLKW 1
1637 032354 I.AM.DONE:
1638 032354 .BLKW 1
1639 032356 RETRYING:
1640 032356 .BLKW 1
1641 032360 BOARD: .BLKW 1
1642 032362 BANK: .BLKW 1
1643
1644
1645
1646 032364 ML.REG: .BLKW 26
1647 032440 PTABLE.ADDR:
1648 032440 .BLKW 10
1649 032460 DRIVE.STATUS:
1650 032460 .BLKB 1
1651 .EVEN
1652 032462 DROPT.DRIVES:
1653 032462 .BLKB 1
1654 .EVEN
1655 032464 WHY.DROPT:
1656 032464 .BLKW 4
1657 032474 NUM.DRIVES:
1658 032474 .BLKW 1
1659 032476 LOW.SECT:
1660 032476 .BLKW 10
1661 032516 TOP.SECT:
1662 032516 .BLKW 10
1663 032536 000000 PATTBL: .WORD 0
1664 032540 002000 .WORD 2000
1665 032542 000001 .WORD 1
1666 032544 000001 .WORD 1
1667 032546 000004 .WORD 4
1668 032550 000004 .WORD 4
1669 032552 000011 .WORD 11
1670 032554 000011 .WORD 11
1671 032556 002000 .WORD 2000
1672 032560 002000 .WORD 2000
  
```

23-Oct-1980 15:01:43 TOPS  
 23-Oct-1980 14:57:05 PA:<

1674  
 1675  
 1676  
 1677  
 1678  
 1679  
 1680  
 1681  
 1682  
 1683  
 1684  
 1685  
 1686  
 1687  
 1688  
 1689  
 1690  
 1691  
 1692  
 1693  
 1694  
 1695  
 1696  
 1697  
 1698  
 1699  
 1700  
 1701  
 1702  
 1703  
 1704  
 1705  
 1706  
 1707  
 1708  
 1709  
 1710  
 1711  
 1712  
 1713  
 1714  
 1715  
 1716  
 1717  
 1718  
 1719  
 1720  
 1721  
 1722  
 1723  
 1724  
 1725  
 1726  
 1727  
 1728

:MLX4  
 :

ML11 INTERRUPT SERVICE ROUTINE

.GLOBL L\$UNIT, L\$LUN, LIMIT, RANGE, LSECT  
 .GLOBL TSECT, ONLY, DROPNE, DROP1, DROP2  
 .GLOBL DROP3, DROP4, DROP5, MARPAT, REFRESH  
 .GLOBL ECCDIS, EOPSUM, ERR0UT, SEED1  
 .GLOBL SEED2, SEED3, RANDOM, RN

100000	BIT15==	-100000
040000	BIT14==	40000
020000	BIT13==	20000
010000	BIT12==	10000
004000	BIT11==	4000
002000	BIT10==	2000
001000	BIT09==	1000
000400	BIT08==	400
000200	BIT07==	200
000100	BIT06==	100
000040	BIT05==	40
000020	BIT04==	20
000010	BIT03==	10
000004	BIT02==	4
000002	BIT01==	2
000001	BIT00==	1
001000	BIT9==	1000
000400	BIT8==	400
000200	BIT7==	200
000100	BIT6==	100
000040	BIT5==	40
000020	BIT4==	20
000010	BIT3==	10
000004	BIT2==	4
000002	BIT1==	2
000001	BIT0==	1
000040	EF.START==	40
000037	EF.RESTART==	37
000036	EF.CONTINUE==	36
000035	EF.NEW==	35
000034	EF.PWR==	34
000340	PRI07==	340
000300	PRI06==	300
000240	PRI05==	240
000200	PRI04==	200
000140	PRI03==	140
000100	PRI02==	100
000040	PRI01==	40
000000	PRI00==	0
000004	EVL==	4
000010	LOT==	10
000020	ADR==	20
000040	IDU==	40

23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<

```

1730      ;MLX4
1731      ;
1732
1733      000100      ISR==          100
1734      000200      UAM==         200
1735      000400      BOE==         400
1736      001000      PNT==        1000
1737      002000      PRI==        2000
1738      004000      IXE==         4000
1739      010000      IBE==        10000
1740      020000      IER==        20000
1741      040000      LOE==        40000
1742      100000      HOE==       -100000
1743      010706      WDBUFF=       WBUFF
1744      011706      WCBUFF=       WBUFF+1000
1745      020706      RDBUFF=       RBUFF
1746      021706      RCBUFF=       RBUFF+1000
1747      020706      END.WBUFF=    WBUFF+10000
1748      030706      END.RBUFF=    RBUFF+10000

```

ML11 INTERRUPT SERVICE ROUTINE

1878  
1879  
1880  
1881  
1885  
1886  
1887  
1888  
1889  
1890  
1895  
1896

.SBTTL SERVICE ML11 INTERRUPT SERVICE ROUTINE

SERVICE::

MOV #1,I.AM.DONE ;  
RTI ;

2353  
2351

; Routine Size: 4 words  
; Maximum stack depth per invocation: 0 words



1898 :MLX4

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (8)

1899 : ONCE-ONLY CODE

1900 :  
1901 : 2356 %SBTTL 'ONCE-ONLY CODE'

1902 : 2357  
1903 : 2358 ROUTINE CLRTBLS: NOVALUE =  
1904 : 2359 BEGIN !\* 1 \*

1905 : 2360  
1906 : 2361 !\*\*

1907 : 2362 ROUTINE: CLRTBLS

1908 : 2363  
1909 : 2364 PURPOSE: THIS ROUTINE IS CALLED BY THE INITIALIZATION CODE WHEN  
1910 : 2365 A 'START' OR 'RESTART' COMMAND HAS BEEN USED TO BEGIN  
1911 : 2366 THE PROGRAM. THE PURPOSES OF THE ROUTINE ARE:

1912 : 2367 (1) TO INITIALIZE STATISTICS TABLES

1913 : 2368 (2) TO SET ALL DRIVE AND ARRAY STATUS LOCATIONS TO ACTIVE  
1914 : 2369  
1915 : 2370 NOTE: EVEN IF A UNIT WAS DROPPED DURING A PREVIOUS  
1916 : 2371 RUN, THE START OR RESTART COMMAND GUARANTEES  
1917 : 2372 THAT THE DRIVE WILL ONCE AGAIN BE TESTABLE.

1918 : 2373  
1919 : 2374 (3) TO ENABLE THE RUNNING OF ALL TEST OPTIONS IF THE  
1920 : 2375 OPERATOR WANTED THEM ALL TO RUN.  
1921 : 2376  
1922 : 2377 !--

1923 : 2378  
1924 : 2379 !+  
1925 : 2380 THIS IS THE CODE FOR PURPOSE 1:  
1926 : 2381 !-

1927 : 2382  
1928 : 2383 INCR LUN FROM 0 TO .LSUNIT - 1) DO  
1929 : 2384 BEGIN

1930 : 2385 INCR ARRAY FROM 0 TO 15 DO  
1931 : 2386 BEGIN  
1932 : 2387 SOFTS[.LUN,.ARRAY,0,16,0] = 0;  
1933 : 2388 HARDS[.LUN,.ARRAY,0,16,0] = 0;  
1934 : 2389 TRIES[.LUN,.ARRAY,0,16,0] = 0;

1935 : 2390 END;  
1936 : 2391 END;

1937 : 2392  
1938 : 2393 EOP\_COUNT = 0;  
1939 : 2394 NUM\_DRIVES = 0;  
1940 : 2395 RETRYING = INACTIVE;

1941 : 2396  
1942 : 2397 WR\_COUNT = 0;  
1943 : 2398 WR\_THOUSANDS = 0;  
1944 : 2399 WR\_MILLIONS = 0;

1945 : 2400  
1946 : 2401 RD\_COUNT = 0;  
1947 : 2402 RD\_THOUSANDS = 0;  
1948 : 2403 RD\_MILLIONS = 0;

1949 : 2404  
1950 : 2405 WC\_COUNT = 0;  
1951 : 2406 WC\_THOUSANDS = 0;  
1952 : 2407 WC\_MILLIONS = 0;

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (8)

```

1954 :MLX4
1955 :
1956 :
1957 : 2408
1958 : 2409
1959 : 2410
1960 : 2411
1961 : 2412
1962 : 2413 INCR LUN FROM 0 TO (.L$UNIT - 1) DO
1963 : 2414 BEGIN !* 2 *
1964 : 2415 L$LUN = .LUN;
1965 : 2416 LOW SECT[.LUN] = 0;
1966 : 2417 DRIVE_STATUS[.LUN] = ACTIVE;
1967 : 2418 DROPT_DRIVES[.LUN] = INACTIVE;
1968 : 2419 WHY_DROPT[.LUN] = 0;
1969 : 2420 END; !* 2 *
1970 : 2421
1971 : 2422
1972 : 2423
1973 : 2424
1974 : 2425
1975 : 2426 IF .DROPNE EQL 0
1976 : 2427 THEN
1977 : 2428 BEGIN !* 3 *
1978 : 2429 DROP1 = 0;
1979 : 2430 DROP2 = 0;
1980 : 2431 DROP3 = 0;
1981 : 2432 DROP4 = 0;
1982 : 2433 DROP5 = 0;
1983 : 2434 END; !* 3 *
1984 : 2435
1985 : 2436 RETURN;
1986 : 2437
1987 : 2438 END; !* 1 *
  
```

```

1991
1992
1996 032572 004137 005202 .SBTTL CLRTBLS ONCE-ONLY CODE
1997 032576 013704 002012 CLRTBLS:JSR R1,$SAVE4
1998 032602 005003 MOV L$UNIT,R4
1999 032604 000424 CLR R3
2000 032606 010300 BR 3$
2001 032610 006300 1$: MOV R3,R0
2002 032612 006300 ASL R0
2003 032614 006300 ASL R0
2004 032616 006300 ASL R0
2005 032620 005001 CLR R1
2006 032622 010002 2$: MOV R0,R2
2007 032624 060102 ADD R1,R2
2008 032626 006302 ASL R2
  
```

2358  
 2383  
 : LUN  
 : LUN,\*  
 2387  
 : ARRAY  
 : ARRAY,\*  
 2385  
 2387



2066  
2067  
2068  
2069 033072 005037 002236  
2070 033076 005037 002240  
2071 033102 005037 002242  
2072 033106 005037 002244  
2073 033112 005037 002250  
2074 033116 000207  
2075  
2076  
2077  
2082  
2083

;MLX4  
;  
ONCE-ONLY CODE  
CLR DROP1  
CLR DROP2  
CLR DROP3  
CLR DROP4  
CLR DROP5  
6S: RTS PC

23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<  
2429  
2430  
2431  
2432  
2433  
2358

; Routine Size: 107 words  
; Maximum stack depth per invocation: 12 words

```

2085 ;MLX4
2086 :
2087 :
2088 : 2439 ROUTINE INIT_ADDRESSES(PLOC): NOVALUE =
2089 : 2440 BEGIN !* 1 *
2090 : 2441
2091 : 2442 !*+
2092 : 2443 ! ROUTINE: INIT_ADDRESSES(PLOC)
2093 : 2444
2094 : 2445 ! PURPOSE: THIS ROUTINE IS CALLED ONLY ONCE DURING THE INITIALIZATION
2095 : 2446 ! CODE, EVEN IF THERE IS MORE THAN ONE DRIVE PRESENT. THE
2096 : 2447 ! PURPOSES OF THE ROUTINE ARE:
2097 : 2448
2098 : 2449 ! (1) TO OBTAIN HARDWARE P-TABLE INFORMATION FROM
2099 : 2450 ! MAIN MEMORY WHICH PERTAINS TO ALL DRIVES.
2100 : 2451
2101 : 2452 ! (2) TO SET UP THE ADDRESSES FOR THE 22 ML-11 REGISTERS.
2102 : 2453
2103 : 2454 ! (3) TO SET UP THE INTERRUPT SERVICE ROUTINE AT THE
2104 : 2455 ! CORRECT PRIORITY, AND LOWER THE CPU PRIORITY TO
2105 : 2456 ! ALLOW INTERRUPTS TO OCCUR.
2106 : 2457
2107 : 2458 ! ARGUMENT: PLOC = THE POINTER TO THE LOCATION IN MAIN MEMORY WHERE
2108 : 2459 ! THE HARDWARE P-TABLE IS TO BE FOUND.
2109 : 2460 !--
2110 : 2461
2111 : 2462 LOCAL
2112 : 2463 TEMP, PRIORITY, OFFSET;
2113 : 2464
2114 : 2465 !+
2115 : 2466 ! THIS IS THE CODE FOR PURPOSE 1:
2116 : 2467 !-
2117 : 2468
2118 : 2469 BASE_ADDR = (.PLOC + 0);
2119 : 2470 VEC = (.PLOC + 2);
2120 : 2471 BR_LEVEL = (.PLOC + 4);
2121 : 2472 PRIORITY = .BR_LEVEL^5;
2122 : 2473
2123 : 2474 IF .ECCDIS
2124 : 2475 THEN TEMP = WRD37 !ECC IS DISABLED
2125 : 2476 ELSE TEMP = WRD36; !ECC IS ENABLED (NORMAL OPERATION)
2126 : 2477
2127 : 2478 PRINTB(CRLF);
2128 : 2479 PRINTB(SAY5,WRD34,WRD40,WRD38,.TEMP,WRD41);
2129 : 2480 !'RUNNING WITH ECC ENABLED/DISABLED AND'
2130 : 2481
2131 : 2482 IF .REFRESH
2132 : 2483 THEN TEMP = WRD36 !REFRESH MARGINING IS ENABLED
2133 : 2484 ELSE TEMP = WRD37; !REFRESH MARGINING IS DISABLED (NORMAL OPERATION)
2134 : 2485
2135 : 2486 PRINTB(SAY3,WRD40,PHR3,.TEMP);
2136 : 2487 !'WITH REFRESH MARGINING FNABLED/DISABLED'
2137 : 2488 PRINTB(CRLF);
2138 : 2489
2139 : 2490 !+

```

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (9)

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (9)

```

2141 :MLX4
2142 : ONCE-ONLY CODE
2143 :
2144 : 2491 :THIS IS THE CODE FOR PURPOSE 2:
2145 : 2492 :-
2146 : 2493
2147 : 2494 OFFSET = 0;
2148 : 2495
2149 : 2496 INCR COUNT FROM 0 TO (NUM_REGS - 1) DO
2150 : 2497 BEGIN !* 2 *
2151 : 2498 ML REG[.COUNT] = .BASE_ADDR + .OFFSET;
2152 : 2499 OFFSET = .OFFSET + 2;
2153 : 2500 END: !* 2 *
2154 : 2501
2155 : 2502
2156 : 2503 :THIS IS THE CODE FOR PURPOSE 3:
2157 : 2504 :-
2158 : 2505
2159 : 2506 SETPRI(PRI00);
2160 : 2507 SETVEC(.VEC,SERVICE,.PRIORITY);
2161 : 2508
2162 : 2509 RETURN;
2163 : 2510
2164 : 2511 END: !* 1 *
  
```

```

2168
2169 .SBTTL INIT.ADDRESSES ONCE-ONLY CODE
2173 033120 INIT.ADDRESSES:
2174 033120 004137 005202 JSR R1,$SAVE4 ;
2175 033124 016602 000014 MOV 14(SP),R2 ; PLOC,*
2176 033130 011237 030724 MOV (R2),BASE_ADDR ;
2177 033134 016237 000002 030726 MOV 2(R2),VEC ;
2178 033142 016237 000004 030730 MOV 4(R2),BR.LEVEL ;
2179 033150 013746 030730 MOV BR.LEVEL,-(SP) ;
2180 033154 012746 000005 MOV #5,-(SP) ;
2181 033160 004737 005102 JSR PC,BL$SHF ;
2182 033164 010003 MOV R0,R3 ; *,PRIORITY
2183 033166 032737 000001 002254 BIT #1,ECCDIS ;
2184 033174 001403 BEQ 1$ ;
2185 033176 012701 007022 MOV #WRD37,R1 ; *,TEMP
2186 033202 000402 BR 2$ ;
2187 033204 012701 007012 1$: MOV #WRD36,R1 ; *,TEMP
2188 033210 012716 006510 2$: MOV #CRLF,(SP) ;
2189 033214 012746 000001 MOV #1,-(SP) ;
2190 033220 010600 MOV SP,R0 ; SP,*
2191 033222 104414 TRAP 14 ;
2192 033224 012716 007046 MOV #WRD41,(SP) ;
2193 033230 010146 MOV R1,-(SP) ; TEMP,*
2194 033232 012746 007034 MOV #WRD38,-(SP) ;
2195 033236 012746 007040 MOV #WRD40,-(SP) ;
  
```

Address	OpCode	Operand 1	Operand 2	Operand 3	Comment	Line No.
2197					:MLX4	
2198					:	
2199					ONCE-ONLY CODE	
2200	MOV	#WRD34,-(SP)				
2201	MOV	#SAY5,-(SP)				
2202	MOV	#6,-(SP)				
2203	MOV	SP,R0			: SP,*	
2204	TRAP	14				
2205	BIT	#1,REFRESH			:	2482
2206	BEQ	3\$				
2207	MOV	#WRD36,R1			: *,TEMP	2483
2208	BR	4\$			:	2482
2209	MOV	#WRD37,R1			: *,TEMP	2484
2210	MOV	R1,(SP)			: TEMP,*	2486
2211	MOV	#PHR3,-(SP)				
2212	MOV	#WRD40,-(SP)				
2213	MOV	#SAY3,-(SP)				
2214	MOV	#4,-(SP)				
2215	MOV	SP,R0			: SP,*	
2216	TRAP	14				
2217	MOV	#CRLF,(SP)			:	2488
2218	MOV	#1,-(SP)				
2219	MOV	SP,R0			: SP,*	
2220	TRAP	14				
2221	CLR	R2			: OFFSET	2494
2222	CLR	R0			: COUNT	2496
2223	MOV	R0,R1			: COUNT,*	2498
2224	ASL	R1				
2225	MOV	BASE.ADDR,R4				
2226	ADD	R2,R4			: OFFSET,*	
2227	MOV	R4,ML.REG(R1)				
2228	ADD	#2,R2			: *,OFFSET	2499
2229	INC	R0			: COUNT	2496
2230	CMP	R0,#25			: COUNT,*	
2231	BLE	5\$				
2232	CLR	R0			:	2506
2233	TRAP	41				
2234	MOV	R3,(SP)			: PRIORITY,*	2507
2235	MOV	#SERVICE,-(SP)				
2236	MOV	VEC,-(SP)				
2237	MOV	#3,-(SP)				
2238	TRAP	37				
2239	ADD	#42,SP			:	2439
2240	RTS	PC				

2241  
 2242  
 2243  
 2248  
 2249

: Routine Size: 103 words  
 : Maximum stack depth per invocation: 22 words

```

2251 :MLX4
2252 :
2253 :
2254 : 2512 %SBTTL 'DRIVE IDENTIFICATION ROUTINES'
2255 : 2513
2256 : 2514 ROUTINE SAYWHO(LUN): NOVALUE =
2257 : 2515 BEGIN
2258 : 2516
2259 : 2517 !++
2260 : 2518 ROUTINE: SAYWHO(LUN)
2261 : 2519
2262 : 2520 PURPOSE: TO PRINT OUT AN IDENTIFICATION LINE WHICH INCLUDES:
2263 : 2521 LOGICAL UNIT: X DRIVE: Y SERIAL #: ZZZZ
2264 : 2522
2265 : 2523 THE UNIT'S SERIAL NUMBER IN EITHER BCD OR OCTAL FORMAT.
2266 : 2524 BCD WILL BE PRINTED AS LONG AS THE DIGITS ARE ALL VALID.
2267 : 2525 !--
2268 : 2526
2269 : 2527
2270 : 2528 LOCAL
2271 : 2529 D3,D2,D1,D0;
2272 : 2530
2273 : 2531
2274 : 2532 D3 = .SN3;
2275 : 2533 D2 = .SN2;
2276 : 2534 D1 = .SN1;
2277 : 2535 D0 = .SN0;
2278 : 2536
2279 : 2537 PRINTB(FMT4A,PHR7,.LUN,WRD11,.DRIVE);
2280 : 2538 !'LOGICAL UNIT: X DRIVE: Y'
2281 : 2539
2282 : 2540 IF ((.D3 GTR 9) OR (.D2 GTR 9) OR (.D1 GTR 9) OR (.D0 GTR 9))
2283 : 2541 THEN PRINTB(FMT4B,PHR8,.MLSN)
2284 : 2542 !' SERIAL #: ZZZZZZ'
2285 : 2543 ELSE PRINTB(FMT4C,PHR8,.D3,.D2,.D1,.D0);
2286 : 2544 !' SERIAL #: DDDD'
2287 : 2545
2288 : 2546 RETURN;
2289 : 2547
2290 : 2548 END;
  
```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (10)

```

2294 :
2295 :
2299 033436 004137 005222 SAYWHO: .SBTTL SAYWHO DRIVE IDENTIFICATION ROUTINES
2300 033442 017705 176746 JSR R1,$SAVE5
2301 033446 006205 ASR R5
2302 033450 006205 ASR R5
2303 033452 006205 ASR R5
2304 033454 006205 ASR R5
2305 033456 000305 SWAB R5
  
```

2514  
 2532



23-Oct-1980 15:01:43 TOPS  
 23-Oct-1980 14:57:05 PA:<

Address	Hex	Hex	Hex	Label	Code	Comments	Line
2307				.MLX4			
2308				:			
2309					DRIVE IDENTIFICATION ROUTINES		
2310	033460	042705	177760	BIC	#177760,R5	: *,D3	
2311	033464	017704	176724	MOV	@ML.REG+30,R4	: *,D2	2533
2312	033470	000304		SWAB	R4	: D2	
2313	033472	042704	177760	BIC	#177760,R4	: *,D2	
2314	033476	117701	176712	MOVB	@ML.REG+30,R1	: *,D1	2534
2315	033502	006201		ASR	R1	: D1	
2316	033504	006201		ASR	R1	: D1	
2317	033506	006201		ASR	R1	: D1	
2318	033510	006201		ASR	R1	: D1	
2319	033512	042701	177760	BIC	#177760,R1	: *,D1	
2320	033516	117702	176672	MOVB	@ML.REG+30,R2	: *,D0	2535
2321	033522	042702	177760	BIC	#177760,R2	: *,D0	
2322	033526	016603	000016	MOV	16(SP),R3	: LUN,*	2537
2323	033532	006303		ASL	R3		
2324	033534	016303	032440	MOV	PTABLE.ADDR(R3),R3		
2325	033540	016346	000006	MOV	6(R3),-(SP)		
2326	033544	012746	006662	MOV	#WRD11,-(SP)		
2327	033550	016646	000022	MOV	22(SP),-(SP)	: LUN,*	
2328	033554	012746	007472	MOV	#PHR7,-(SP)		
2329	033560	012746	005640	MOV	#FMT4A,-(SP)		
2330	033564	012746	000005	MOV	#5,-(SP)		
2331	033570	010600		MOV	SP,R0	: SP,*	
2332	033572	104414		TRAP	14		
2333	033574	020527	000011	CMP	R5,#11	: D3,*	2540
2334	033600	003011		BGT	1\$		
2335	033602	020427	000011	CMP	R4,#11	: D2,*	
2336	033606	003006		BGT	1\$		
2337	033610	020127	000011	CMP	R1,#11	: D1,*	
2338	033614	003003		BGT	1\$		
2339	033616	020227	000011	CMP	R2,#11	: D0,*	
2340	033622	003413		BLE	2\$		
2341	033624	017746	176564	1\$: MOV	@ML.REG+30,-(SP)	:	2541
2342	033630	012746	007510	MOV	#PHR8,-(SP)		
2343	033634	012746	005674	MOV	#FMT4B,-(SP)		
2344	033640	012746	000003	MOV	#3,-(SP)		
2345	033644	010600		MOV	SP,R0	: SP,*	
2346	033646	104414		TRAP	14		
2347	033650	000416		BR	3\$		2540
2348	033652	010246		2\$: MOV	R2,-(SP)	: D0,*	2543
2349	033654	010146		MOV	R1,-(SP)	: D1,*	
2350	033656	010446		MOV	R4,-(SP)	: D2,*	
2351	033660	010546		MOV	R5,-(SP)	: D3,*	
2352	033662	012746	007510	MOV	#PHR8,-(SP)		
2353	033666	012746	005712	MOV	#FMT4C,-(SP)		
2354	033672	012746	000006	MOV	#6,-(SP)		
2355	033676	010600		MOV	SP,R0	: SP,*	
2356	033700	104414		TRAP	14		
2357	033702	062706	000006	ADD	#6,SP		
2358	033706	062706	000024	3\$: ADD	#24,SP	:	2514
2359	033712	000207		RTS	PC		
2360							
2361							

: Routine Size: 87 words

2363  
2364  
2365  
2366  
2371  
2372

;MLX4  
;  
DRIVE IDENTIFICATION ROUTINES  
; Maximum stack depth per invocation: 19 words

23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<

```

2374 :MLX4
2375 :
2376 :
2377 : 2549 ROUTINE CONFIG(LUN): NOVALUE =
2378 : 2550 BEGIN !* 1 *
2379 : 2551
2380 : 2552 !++
2381 : 2553 ROUTINE: CONFIG(LUN)
2382 : 2554
2383 : 2555 PURPOSE: CONFIG IS CALLED BY THE INITIALIZATION CODE FOR EACH DRIVE
2384 : 2556 WHICH SUCCESSFULLY RESPONDS TO A REQUEST FOR ITS HARDWARE
2385 : 2557 P-TABLE. THE PURPOSES OF THIS ROUTINE ARE:
2386 : 2558
2387 : 2559 (1) TO CHECK THAT THE DRIVE IS POWERED UP.
2388 : 2560
2389 : 2561 (2) TO VERIFY THAT THE DRIVE IS AN ML11 UNIT.
2390 : 2562
2391 : 2563 (3) TO VERIFY THAT THE OPERATOR DEFINED TEST RANGES ARE
2392 : 2564 WITHIN THE CALCULATED SYSTEM SIZE.
2393 : 2565
2394 : 2566 (4) TO PRINT DRIVE IDENTIFICATION INFORMATION.
2395 : 2567
2396 : 2568 ARGUMENT: LUN = THE LOGICAL UNIT NUMBER, COUNTING FROM 0 TO
2397 : 2569 NUMBER OF DRIVES MINUS 1.
2398 : 2570 !--
2399 : 2571
2400 : 2572
2401 : 2573 LOCAL
2402 : 2574 TYPE, TEMP, TOP;
2403 : 2575
2404 : 2576 !+
2405 : 2577 !THIS IS THE CODE FOR PURPOSE 1:
2406 : 2578 !-
2407 : 2579
2408 : 2580 UNIT = .DRIVE;
2409 : 2581 IF .DPR NEQ 1
2410 : 2582 THEN
2411 : 2583 BEGIN
2412 : 2584 PRINTB(FMT4A,PHR7,.LUN,WRD11,.DRIVE);
2413 : 2585 !'LOGICAL UNIT: X DRIVE: Y'
2414 : 2586 PRINTB(SAY2,WRD11,WRD21);
2415 : 2587 !'DRIVE DROPPED'
2416 : 2588 PRINTB(FMT2,CAUSE1);
2417 : 2589 !' (NOT POWERED UP)'
2418 : 2590 WHY DROPT[.LUN] = CODE_1;
2419 : 2591 DODD(.LUN);
2420 : 2592 RETURN;
2421 : 2593 END;
2422 : 2594
2423 : 2595 !+
2424 : 2596 !THIS IS THE CODE FOR PURPOSE 2:
2425 : 2597 !-
2426 : 2598
2427 : 2599 SAYWHO(.LUN);
2428 : 2600 !'LOGICAL UNIT: X DRIVE: Y SERIAL #: ZZZZ'

```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (11)

```

2430 :MLX4
2431 :
2432 :
2433 : 2601
2434 : 2602 TYPE = .MLDT;
2435 : 2603
2436 : 2604 IF ((.TYPE NEQ ML11A) AND (.TYPE NEQ ML11B))
2437 : 2605 THEN
2438 : 2606 BEGIN
2439 : 2607 PRINTB(SAY2,WRD11,WRD21);
2440 : 2608 !'DRIVE DROPPED'
2441 : 2609 PRINTB(FMT2,CAUSE2);
2442 : 2610 !' (NOT AN ML11 UNIT)'
2443 : 2611 WHY DROPT[.LUN] = CODE_2;
2444 : 2612 DODD(.LUN);
2445 : 2613 RETURN;
2446 : 2614 END
2447 : 2615 ELSE
2448 : 2616 BEGIN !* 3 *
2449 : 2617 IF .ARR_TYP EQL 0
2450 : 2618 THEN
2451 : 2619 BEGIN !* 4 *
2452 : 2620 TYPE = WRD6;
2453 : 2621 TOP = 1;
2454 : 2622 END !* 4 *
2455 : 2623 ELSE
2456 : 2624 BEGIN !* 5 *
2457 : 2625 TYPE = WRD7;
2458 : 2626 TOP = 4;
2459 : 2627 END; !* 5 *
2460 : 2628
2461 : 2629 !+
2462 : 2630 ! THIS IS THE CODE FOR PURPOSE 3:
2463 : 2631 !-
2464 : 2632
2465 : 2633 TOP_SECT[.LUN] = (.TOP) * (512) * (.SZ) - 1;
2466 : 2634 IF .LIMIT
2467 : 2635 THEN
2468 : 2636 !+
2469 : 2637 ! THE OPERATOR HAS CHOSEN LIMITS:
2470 : 2638 !-
2471 : 2639 BEGIN !* 6 *
2472 : 2640 IF .RANGE EQL 0
2473 : 2641 THEN
2474 : 2642 !+
2475 : 2643 ! THE BOARD NUMBER (0-15) IS CONTAINED IN 'ONLY'
2476 : 2644 !-
2477 : 2645 BEGIN !* 10 *
2478 : 2646 IF .TYPE
2479 : 2647 THEN
2480 : 2648 !+
2481 : 2649 ! THE CHIPS ARE 64K
2482 : 2650 !-
2483 : 2651 BEGIN !* 7 *
2484 : 2652 LSECT = 2048 * .ONLY;

```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (11)

2486 :MLX4  
 2487 :  
 2488 :  
 2489 :  
 2490 :  
 2491 :  
 2492 :  
 2493 :  
 2494 :  
 2495 :  
 2496 :  
 2497 :  
 2498 :  
 2499 :  
 2500 :  
 2501 :  
 2502 :  
 2503 :  
 2504 :  
 2505 :  
 2506 :  
 2507 :  
 2508 :  
 2509 :  
 2510 :  
 2511 :  
 2512 :  
 2513 :  
 2514 :  
 2515 :  
 2516 :  
 2517 :  
 2518 :  
 2519 :  
 2520 :  
 2521 :  
 2522 :  
 2523 :  
 2524 :  
 2525 :  
 2526 :  
 2527 :  
 2528 :  
 2529 :  
 2530 :  
 2531 :  
 2532 :  
 2533 :  
 2534 :  
 2535 :  
 2536 :  
 2537 :  
 2538 :  
 2539 :  
 2540 :

DRIVE IDENTIFICATION ROUTINES

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (11)

```

    TSECT = .LSECT + 2047;
  END      !* 7 *
ELSE
  !+
  !THE CHIPS ARE 16K
  !-
  BEGIN   !* 8 *
  LSECT = 512 * .ONLY;
  TSECT = .LSECT + 511;
  END;    !* 8 *

END;     !* 10 *

IF .TSECT LEQA .TOP_SECT[.LUN]
THEN TOP_SECT[.LUN] = .TSECT
ELSE
  BEGIN
  PRINTB(SAY2,WRD11,WRD21);
  !'DRIVE DROPPED'
  PRINTB(FMT2,CAUSE3);
  !' (OPERATOR SELECTED TEST LIMITS INCORRECTLY)'
  PRINTB(FMT11,PHR14,.TSECT,PHR16,.TOP_SECT[.LUN]);
  !'TOP SECTOR OF XXXXXX EXCEEDS SYSTEM LIMIT OF YYYYYY'
  WHY DROPT[.LUN] = CODE_3;
  DOD[.LUN];
  RETURN;
  END;

IF .LSECT LEQA .TSECT
THEN LOW_SECT[.LUN] = .LSECT
ELSE
  BEGIN
  PRINTB(SAY2,WRD11,WRD21);
  !'DRIVE DROPPED'
  PRINTB(FMT2,CAUSE3);
  !' (OPERATOR SELECTED TEST LIMITS INCORRECTLY)'
  PRINTB(FMT11,PHR15,.LSECT,PHR14,.TSECT);
  !'LOW SECTOR OF XXXXXX EXCEEDS TOP SECTOR OF YYYYYY'
  WHY DROPT[.LUN] = CODE_3;
  DOD[.LUN];
  RETURN;
  END;

END;     !* 6 *

!+
!THIS IS THE CODE FOR PURPOSE 4:
!-

PRINTB(FMT5,.TYPE,LOWEST,HIGHEST);
!'ML11-X SECTORS UNDER TEST: XXXXXX TO YYYYYY'
SELECTONE .TRT OF
  
```

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (11)

```

2542 ;MLX4
2543 ;
2544
2545 : 2705 SET
2546 : 2706 [0] : TEMP = TRT00;
2547 : 2707 [1] : TEMP = TRT01;
2548 : 2708 [2] : TEMP = TRT10;
2549 : 2709 [3] : TEMP = TRT11;
2550 : 2710 TES;
2551 : 2711 PRINTB(FMT8,.TEMP),
2552 : 2712 !'TRANSFER RATE: X MBYTES/SECOND'
2553 : 2713 PRINTB(FMT4B,PHR4,.BASE_ADDR);
2554 : 2714 !' CSR ADDRESS: XXXXXX'
2555 : 2715 END; !* 3 *
2556 : 2716
2557 : 2717 RETURN;
2558 : 2718
2559 : 2719 END; !* 1 *
  
```

```

2563
2564 .SBTTL CONFIG DRIVE IDENTIFICATION ROUTINES
2568 033714 004137 005222 CONFIG: JSR R1,$SAVE5 ; 2549
2569 033720 016603 000016 MOV 16(SP),R3 ; LUN,* 2580
2570 033724 010304 MOV R3,R4
2571 033726 006304 ASL R4
2572 033730 016401 032440 MOV PTABLE.ADDR(R4),R1
2573 033734 016105 000006 MOV 6(R1),R5
2574 033740 042705 177770 BIC #177770,R5
2575 033744 142777 000007 176422 BICB #7,@ML.REG+10
2576 033752 150577 176416 BISB R5,@ML.REG+10
2577 033756 032777 000400 176412 BIT #400,@ML.REG+12 ; 2581
2578 033764 001051 BNE 1$ ;
2579 033766 016401 032440 MOV PTABLE.ADDR(R4),R1 ; 2584
2580 033772 016146 000006 MOV 6(R1),-(SP)
2581 033776 012746 006662 MOV #WRD11,-(SP)
2582 034002 010346 MOV R3,-(SP)
2583 034004 012746 007472 MOV #PHR7,-(SP)
2584 034010 012746 005640 MOV #FMT4A,-(SP)
2585 034014 012746 000005 MOV #5,-(SP)
2586 034020 010600 MOV SP,R0 ; SP,*
2587 034022 104414 TRAP 14
2588 034024 012716 006744 MOV #WRD21,(SP) ; 2586
2589 034030 012746 006662 MOV #WRD11,-(SP)
2590 034034 012746 006522 MOV #SAY2,-(SP)
2591 034040 012746 000003 MOV #3,-(SP)
2592 034044 010600 MOV SP,R0 ; SP,*
2593 034046 104414 TRAP 14
2594 034050 012716 010102 MOV #CAUSE1,(SP) ; 2588
2595 034054 012746 005602 MOV #FMT2,-(SP)
2596 034060 012746 000002 MOV #2,-(SP)
  
```

2598									23-Oct-1980 15:01:43	TOPS
2599									23-Oct-1980 14:57:05	PA:<
2600										
2601	034064	010600								
2602	034066	104414								
2603	034070	112763	000001	032464						2590
2604	034076	010300								2591
2605	034100	104451								
2606	034102	062706	000026							2581
2607	034106	000207								2583
2608	034110	010346			1\$:					2599
2609	034112	004737	033436							
2610	034116	017702	176270							2602
2611	034122	020227	000110							2604
2612	034126	001435								
2613	034130	020227	000111							
2614	034134	001432								
2615	034136	012746	006744							2607
2616	034142	012746	006662							
2617	034146	012746	006522							
2618	034152	012746	000003							
2619	034156	010600								
2620	034160	104414								
2621	034162	012716	010124							2609
2622	034166	012746	005602							
2623	034172	012746	000002							
2624	034176	010600								
2625	034200	104414								
2626	034202	112763	000002	032464						2611
2627	034210	010300								2612
2628	034212	104451								
2629	034214	062706	000016							2604
2630	034220	000207								2606
2631	034222	032777	002000	176160	2\$:					2617
2632	034230	001005								
2633	034232	012702	006642							2620
2634	034236	012701	000001							2621
2635	034242	000404								2617
2636	034244	012702	006652		3\$:					2625
2637	034250	012701	000004							2626
2638	034254	012705	032516		4\$:					2633
2639	034260	060405								
2640	034262	010146								2616
2641	034264	017701	176120							
2642	034270	006201								
2643	034272	006201								
2644	034274	006201								
2645	034276	000301								
2646	034300	042701	177740							
2647	034304	010146								
2648	034306	004737	004632							
2649	034312	000300								2633
2650	034314	105000								
2651	034316	006300								
2652	034320	010001								

2654												23-Oct-1980 15:01:43	TOPS
2655												23-Oct-1980 14:57:05	PA:<
2656													
2657	034322	005301				DEC	R1						
2658	034324	010115				MOV	R1,(R5)						
2659	034326	032737	000001	002222		BIT	#1,LIMIT	:					2634
2660	034334	001574				BEQ	11\$	:					
2661	034336	005737	002224			TST	RANGE	:					2640
2662	034342	001036				BNE	6\$	:					
2663	034344	032702	000001			BIT	#1,R2	:		*	,TYPE		2646
2664	034350	001417				BEQ	5\$	:					
2665	034352	013700	002232			MOV	ONLY,R0	:					2652
2666	034356	000300				SWAB	R0						
2667	034360	105000				CLRB	R0						
2668	034362	006300				ASL	R0						
2669	034364	006300				ASL	R0						
2670	034366	006300				ASL	R0						
2671	034370	010037	002226			MOV	R0,LSECT						
2672	034374	010037	002230			MOV	R0,TSECT	:		LSECT,*			2653
2673	034400	062737	003777	002230		ADD	#3777,TSECT						
2674	034406	000414				BR	6\$	:					2646
2675	034410	013700	002232		5\$:	MOV	ONLY,R0	:					2660
2676	034414	000300				SWAB	R0						
2677	034416	105000				CLRB	R0						
2678	034420	006300				ASL	R0						
2679	034422	010037	002226			MOV	R0,LSECT						
2680	034426	010037	002230			MOV	R0,TSECT	:		LSECT,*			2661
2681	034432	062737	000777	002230		ADD	#777,TSECT						
2682	034440	023715	002230		6\$:	CMP	TSECT,(R5)	:					2666
2683	034444	101003				BHI	7\$	:					
2684	034446	013715	002230			MOV	TSECT,(R5)	:					2667
2685	034452	000445				BR	8\$	:					2666
2686	034454	012746	006744		7\$:	MOV	#WRD21,-(SP)	:					2670
2687	034460	012746	006662			MOV	#WRD11,-(SP)						
2688	034464	012746	006522			MOV	#SAY2,-(SP)						
2689	034470	012746	000003			MOV	#3,-(SP)						
2690	034474	010600				MOV	SP,R0	:		SP,*			
2691	034476	104414				TRAP	14						
2692	034500	012716	010150			MOV	#CAUSE3,(SP)	:					2672
2693	034504	012746	005602			MOV	#FMT2,-(SP)						
2694	034510	012746	000002			MOV	#2,-(SP)						
2695	034514	010600				MOV	SP,R0	:		SP,*			
2696	034516	104414				TRAP	14						
2697	034520	011516				MOV	(R5),(SP)	:					2674
2698	034522	012746	007744			MOV	#PHR16,-(SP)						
2699	034526	013746	002230			MOV	TSECT,-(SP)						
2700	034532	012746	007710			MOV	#PHR14,-(SP)						
2701	034536	012746	006276			MOV	#FMT11,-(SP)						
2702	034542	012746	000005			MOV	#5,-(SP)						
2703	034546	010600				MOV	SP,R0	:		SP,*			
2704	034550	104414				TRAP	14						
2705	034552	112763	000003	032464		MOVB	#3,WHY.DROPT(R3)	:					2676
2706	034560	010300				MOV	R3,R0	:					2677
2707	034562	104451				TRAP	51	:					
2708	034564	000455				BR	10\$	:					2660



Address	OpCode	Op1	Op2	Op3	Op4	Label	Instruction	Comments	Line
2710						;MLX4			
2711						;			
2712							DRIVE IDENTIFICATION ROUTINES		
2713	034566	023737	002226	002230	8\$:		CMP LSECT,TSECT		2681
2714	034574	101004					BHI 9\$		
2715	034576	013764	002226	032476			MOV LSECT,LOW.SECT(R4)		2682
2716	034604	000450					BR 11\$		2681
2717	034606	012746	006744		9\$:		MOV #WRD21,-(SP)		2685
2718	034612	012746	006662				MOV #WRD11,-(SP)		
2719	034616	012746	006522				MOV #SAY2,-(SP)		
2720	034622	012746	000003				MOV #3,-(SP)		
2721	034626	010600					MOV SP,R0	: SP,*	
2722	034630	104414					TRAP 14		
2723	034632	012716	010150				MOV #CAUSE3,(SP)		2687
2724	034636	012746	005602				MOV #FMT2,-(SP)		
2725	034642	012746	000002				MOV #2,-(SP)		
2726	034646	010600					MOV SP,R0	: SP,*	
2727	034650	104414					TRAP 14		
2728	034652	013716	002230				MOV TSECT,(SP)		2689
2729	034656	012746	007710				MOV #PHR14,-(SP)		
2730	034662	013746	002226				MOV LSECT,-(SP)		
2731	034666	012746	007726				MOV #PHR15,-(SP)		
2732	034672	012746	006276				MOV #FMT11,-(SP)		
2733	034676	012746	000005				MOV #5,-(SP)		
2734	034702	010600					MOV SP,R0	: SP,*	
2735	034704	104414					TRAP 14		
2736	034706	112763	000003	032464			MOVB #3,WHY.DROPT(R3)		2691
2737	034714	010300					MOV R3,R0		2692
2738	034716	104451					TRAP 51		
2739	034720	062706	000034		10\$:		ADD #34,SP		2681
2740	034724	000207					RTS PC		2684
2741	034726	011516			11\$:		MOV (R5),(SP)		2702
2742	034730	016446	032476				MOV LOW.SECT(R4),-(SP)		
2743	034734	010246					MOV R2,-(SP)	: TYPE,*	
2744	034736	012746	005742				MOV #FMT5,-(SP)		
2745	034742	012746	000004				MOV #4,-(SP)		
2746	034746	010600					MOV SP,R0	: SP,*	
2747	034750	104414					TRAP 14		
2748	034752	017701	175432				MOV @ML.REG+24,R1		2704
2749	034756	000301					SWAB R1		
2750	034760	042701	177774				BIC #177774,R1		
2751	034764	001003					BNE 12\$		
2752	034766	012700	010066				MOV #TRT00,R0	: *,TEMP	2706
2753	034772	000421					BR 15\$		2704
2754	034774	020127	000001		12\$:		CMP R1,#1		
2755	035000	001003					BNE 13\$		
2756	035002	012700	010070				MOV #TRT01,R0	: *,TEMP	2707
2757	035006	000413					BR 15\$		2704
2758	035010	020127	000002		13\$:		CMP R1,#2		
2759	035014	001003					BNE 14\$		
2760	035016	012700	010072				MOV #TRT10,R0	: *,TEMP	2708
2761	035022	000405					BR 15\$		2704
2762	035024	020127	000003		14\$:		CMP R1,#3		
2763	035030	001002					BNE 15\$		
2764	035032	012700	010076				MOV #TRT11,R0	: *,TEMP	2709

```

2766 ;MLX4
2767 ;
2768 ;
2769 035036 010016 15$: MOV R0,(SP) ; TEMP,* 2711
2770 035040 012746 006110 MOV #FMTB,-(SP)
2771 035044 012746 000002 MOV #2,-(SP)
2772 035050 010600 MOV SP,R0 ; SP,*
2773 035052 104414 TRAP 14
2774 035054 013716 030724 MOV BASE.ADDR,(SP) ; 2713
2775 035060 012746 007412 MOV #PHR4,-(SP)
2776 035064 012746 005674 MOV #FMT4B,-(SP)
2777 035070 012746 000003 MOV #3,-(SP)
2778 035074 010600 MOV SP,R0 ; SP,*
2779 035076 104414 TRAP 14
2780 035100 062706 000030 ADD #30,SP ; 2549
2781 035104 000207 RTS PC
2782
2783 ; Routine Size: 317 words
2784 ; Maximum stack depth per invocation: 20 words
2789
2790

```

23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<

```

2792 :MLX4
2793 :
2794 :
2795 : 2720 ROUTINE DECODE(SECT): NOVALUE =
2796 : 2721 BEGIN
2797 : 2722
2798 : 2723 !++
2799 : 2724 ROUTINE: DECODE(SECT)
2800 : 2725
2801 : 2726 PURPOSE: TO INTERPRET THE FIELDS OF THE FAILING SECTOR
2802 : 2727 FOR BOTH 16K AND 64K CHIPS.
2803 : 2728
2804 : 2729 ARGUMENT: SECT = FAILING SECTOR NUMBER
2805 : 2730
2806 : 2731 RESULTS: (1) BANK = WHICH ROW OF CHIPS (0-4)
2807 : 2732
2808 : 2733 (2) BOARD = WHICH ARRAY MODULE (0-15)
2809 : 2734 !--
2810 : 2735
2811 : 2736 IF .MLDT
2812 : 2737 THEN !THE CHIPS ARE 64K
2813 : 2738 BEGIN
2814 : 2739 BANK = .SECT<9,2>;
2815 : 2740 BOARD = .SECT<11,4>;
2816 : 2741 END
2817 : 2742 ELSE !THE CHIPS ARE 16K
2818 : 2743 BEGIN
2819 : 2744 BANK = .SECT<7,2>;
2820 : 2745 BOARD = .SECT<9,4>;
2821 : 2746 END;
2822 : 2747
2823 : 2748 RETURN;
2824 : 2749
2825 : 2750 END;
  
```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (12)

2830				.SBTTL	DECODE DRIVE IDENTIFICATION ROUTINES	
2834	035106	032777	000001	175276	DECODE: BIT #1,@ML.REG+26	2736
2835	035114	001415			BEQ 1\$	
2836	035116	016600	000002		MOV 2(SP),R0 ; SECT,*	2739
2837	035122	006200			ASR R0	
2838	035124	000300			SWAB R0	
2839	035126	042700	177774		BIC #177774,R0	
2840	035132	010037	032362		MOV R0,BANK	
2841	035136	016600	000002		MOV 2(SP),R0 ; SECT,*	2740
2842	035142	006200			ASR R0	
2843	035144	006200			ASR R0	
2844	035146	000412			BR 2\$	
2845	035150	016600	000002	1\$:	MOV 2(SP),R0 ; SECT,*	2744
2846	035154	006100			ROL R0	

```
2848      ;MLX4
2849      ;
2850      ;
2851 035156 000300
2852 035160 042700 177774
2853 035164 010037 032362
2854 035170 016600 000002
2855 035174 006200
2856 035176 000300
2857 035200 042700 177760
2858 035204 010037 032360
2859 035210 000207
2860
2861      ; Routine Size: 34 words
2862      ; Maximum stack depth per invocation: 0 words
2863
2864
2865
2866
2867
2868
```

DRIVE IDENTIFICATION ROUTINES

```

      SWAB  R0
      BIC   #177774,R0
      MOV   R0,BANK
      MOV   2(SP),R0      ; SECT,*
2855 28:  ASR   R0
      SWAB  R0
      BIC   #177760,R0
      MOV   R0,BOARD
      RTS   PC           ;
2745
2720
```

23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<

```

2870 ;MLX4
2871 :
2872 :
2873 : 2751 ROUTINE ISOLATE : NOVALUE =
2874 : 2752 BEGIN
2875 : 2753
2876 : 2754 !++
2877 : 2755 ROUTINE: ISOLATE
2878 : 2756
2879 : 2757 PURPOSE: UPON THE DETECTION OF A DATA ERROR (EITHER RECOVERABLE OR
2880 : 2758 NOT) TO EXAMINE THE SECTOR ADDRESS CONTAINED IN THE ECC
2881 : 2759 ERROR LOCATION REGISTER AND PINPOINT THE LOCATION OF THE
2882 : 2760 ERROR. THE SECTOR, BOARD (0-15) AND BANK (0-3) WILL BE
2883 : 2761 REPORTED.
2884 : 2762 !--
2885 : 2763
2886 : 2764 DECODE(.MLEL);
2887 : 2765 IF .ERROUT
2888 : 2766 THEN PRINTB(FMT10A,WRD20,WRD15,.MLEL,.BOARD,.BANK);
2889 : 2767 !'FAILED: SECTOR XXXXXX BOARD YY BANK Z'
2890 : 2768 RETURN;
2891 : 2769 END;
  
```

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (13)

```

2895
2896 .SBTTL ISOLATE DRIVE IDENTIFICATION ROUTINES
2900 035212 017746 175212 ISOLATE:MOV @ML.REG+44,-(SP) ; 2764
2901 035216 004737 035106 JSR PC,DECODE ;
2902 035222 032737 000001 002260 BIT #1,ERROUT ; 2765
2903 035230 001422 BEQ 1$ ;
2904 035232 013746 032362 MOV BANK,-(SP) ; 2766
2905 035236 013746 032360 MOV BOARD,-(SP)
2906 035242 017746 175162 MOV @ML.REG+44,-(SP)
2907 035246 012746 006670 MOV #WRD15,-(SP)
2908 035252 012746 006734 MOV #WRD20,-(SP)
2909 035256 012746 006206 MOV #FMT10A,-(SP)
2910 035262 012746 000006 MOV #6,-(SP)
2911 035266 010600 MOV SP,R0 ; SP,*
2912 035270 104414 TRAP 14
2913 035272 062706 000016 ADD #16,SP
2914 035276 005726 1$: TST (SP)+ ; 2751
2915 035300 000207 RTS PC ;
  
```

; Routine Size: 28 words  
 ; Maximum stack depth per invocation: 8 words

2916  
 2917  
 2918  
 2923

```

2925 ;MLX4
2926 : DRIVE IDENTIFICATION ROUTINES
2927 :
2928 : 2770 ROUTINE UP_HARD_COUNT(LUN,ARRAY) : NOVALUE =
2929 : 2771 BEGIN
2930 : 2772
2931 : 2773 !++
2932 : 2774 : ROUTINE: UP_HARD_COUNT(LUN,ARRAY)
2933 : 2775 :
2934 : 2776 : PURPOSE: TO INCREMENT THE HARD ERROR COUNT FOR THE GIVEN
2935 : 2777 : ARRAY, AND TO SEE IF THE HARD ERROR THRESHOLD HAS
2936 : 2778 : BEEN REACHED.
2937 : 2779 :
2938 : 2780 : ARGUMENTS: LUN = LOGICAL UNIT WHICH HAS THE HARD ERROR
2939 : 2781 : ARRAY = THE BOARD NUMBER (0-15)
2940 : 2782 :--
2941 : 2783
2942 : 2784 HARDS[.LUN,.ARRAY,0,16,0] = .HARDS[.LUN,.ARRAY,0,16,0] + 1;
2943 : 2785
2944 : 2786 IF (((.ARR_TYP EQL 1) AND (.HARDS[.LUN,.ARRAY,0,16,0] EQL H64K_LIMIT))
2945 : 2787 OR ((.ARR_TYP EQL 0) AND (.HARDS[.LUN,.ARRAY,0,16,0] EQL H16K_LIMIT)))
2946 : 2788 THEN
2947 : 2789 BEGIN
2948 : 2790 SAYWHO(.LUN);
2949 : 2791 PRINTB(FMT13,.ARRAY,MSG2);
2950 : 2792 !'ARRAY XX --> RUN ML11 PROM MAINTENANCE PROGRAM'
2951 : 2793 END;
2952 : 2794
2953 : 2795 RETURN;
2954 : 2796 END;

```

```

2958
2959 .SBTTL UP.HARD.COUNT DRIVE IDENTIFICATION ROUTINES
2963 035302 UP.HARD.COUNT:
2964 035302 016600 000004 MOV 4(SP),R0 ; LUN,* 2784
2965 035306 006300 ASL R0
2966 035310 006300 ASL R0
2967 035312 006300 ASL R0
2968 035314 006300 ASL R0
2969 035316 066600 000002 ADD 2(SP),R0 ; ARRAY,*
2970 035322 006300 ASL R0
2971 035324 062700 031332 ADD #HARDS,R0
2972 035330 005210 INC (R0)
2973 035332 032777 002000 175050 BIT #2000,@ML.REG+24 ; 2786
2974 035340 001403 BEQ 1$
2975 035342 021027 000012 CMP (R0),#12
2976 035346 001407 BEQ 2$
2977 035350 032777 002000 175032 1$: BIT #2000,@ML.REG+24 ; 2787
2978 035356 001023 BNE 3$
2979 035360 021027 000012 CMP (R0),#12

```

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 v2(206)  
PA:<ROSEN>MLX4.BLI.1 (14)

```

2981 ;MLX4
2982 ;
2983
2984 035364 001020
2985 035366 016646 000004
2986 035372 004737 033436
2987 035376 012716 010542
2988 035402 016646 000004
2989 035406 012746 006446
2990 035412 012746 000003
2991 035416 010600
2992 035420 104414
2993 035422 062706 000010
2994 035426 000207
2995
2996 ; Routine Size: 43 words
2997 ; Maximum stack depth per invocation: 4 words
3002
3003

```

23-Oct-1980 15:01:43 TOPS  
 23-Oct-1980 14:57:05 PA:<

DRIVE IDENTIFICATION ROUTINES

```

2$: BNE 3$
    MOV 4(SP),-(SP) ; LUN,*
    JSR PC,SAYWHO
    MOV #MSG2,(SP) ;
    MOV 4(SP),-(SP) ; ARRAY,*
    MOV #FMT13, -(SP)
    MOV #3, -(SP)
    MOV SP,R0 ; SP,*
    TRAP 14
    ADD #10,SP
3$: RTS PC

```

2790  
 2791  
 2789  
 2770

```

3005 ;MLX4
3006 ; DRIVE IDENTIFICATION ROUTINES
3007
3008 : 2797 ROUTINE UP_SOFT_COUNT(LUN,ARRAY) : NOVALUE =
3009 : 2798 BEGIN
3010 : 2799
3011 : 2800 !++
3012 : 2801 ROUTINE: UP_SOFT_COUNT(LUN,ARRAY)
3013 : 2802
3014 : 2803 PURPOSE: TO INCREMENT THE SOFT ERROR COUNT FOR THE GIVEN
3015 : 2804 ARRAY, AND TO SEE IF THE SOFT ERROR THRESHOLD HAS
3016 : 2805 BEEN REACHED.
3017 : 2806
3018 : 2807 ARGUMENTS: LUN = LOGICAL UNIT WHICH HAS THE SOFT ERROR
3019 : 2808 ARRAY = THE BOARD NUMBER (0-15)
3020 : 2809 !--
3021 : 2810
3022 : 2811 SOFTS[.LUN,.ARRAY,0,16,0] = .SOFTS[.LUN,.ARRAY,0,16,0] + 1;
3023 : 2812
3024 : 2813 IF (((.ARR_TYP EQL 1) AND (.SOFTS[.LUN,.ARRAY,0,16,0] EQL S64K_LIMIT))
3025 : 2814 OR ((.ARR_TYP EQL 0) AND (.SOFTS[.LUN,.ARRAY,0,16,0] EQL S16K_LIMIT)))
3026 : 2815 THEN
3027 : 2816 BEGIN
3028 : 2817 SAYWHO(.LUN);
3029 : 2818 PRINTB(FMT13,.ARRAY,MSG2);
3030 : 2819 !'ARRAY XX --> RUN ML11 PROM MAINTENANCE PROGRAM'
3031 : 2820 END;
3032 : 2821
3033 : 2822 RETURN;
3034 : 2823 END;
3038
3039

```

```

3039 .SBTTL UP.SOFT.COUNT DRIVE IDENTIFICATION ROUTINES
3043 035430 UP.SOFT.COUNT:
3044 035430 016600 000004 MOV 4(SP),R0 ; LUN,* 2811
3045 035434 006300 ASL R0
3046 035436 006300 ASL R0
3047 035440 006300 ASL R0
3048 035442 006300 ASL R0
3049 035444 066600 000002 ADD 2(SP),R0 ; ARRAY,*
3050 035450 006300 ASL R0
3051 035452 062700 030732 ADD #SOFTS,R0
3052 035456 005210 INC (R0)
3053 035460 032777 002000 174722 BIT #2000,@ML.REG+24 ; 2813
3054 035466 001403 BEQ 1$
3055 035470 021027 000012 CMP (R0),#12
3056 035474 001407 BEQ 2$
3057 035476 032777 002000 174704 1$: BIT #2000,@ML.REG+24 ; 2814
3058 035504 001023 BNE 3$
3059 035506 021027 000012 CMP (R0),#12

```



```
3061 :MLX4
3062 :
3063 :
3064 035512 001020
3065 035514 016646 000004
3066 035520 004737 033436
3067 035524 012716 010542
3068 035530 016646 000004
3069 035534 012746 006446
3070 035540 012746 000003
3071 035544 010600
3072 035546 104414
3073 035550 062706 000010
3074 035554 000207
3075
3076 ; Routine Size: 43 words
3077 ; Maximum stack depth per invocation: 4 words
3082
3083
```

DRIVE IDENTIFICATION ROUTINES

2\$: BNE 3\$  
MOV 4(SP),-(SP) : LUN,\* 2817  
JSR PC,SAYWHO  
MOV #MSG2,(SP) :  
MOV 4(SP),-(SP) : ARRAY,\* 2818  
MOV #FMT1\$,-(SP)  
MOV #3,-(SP)  
MOV SP,R0 : SP,\*  
TRAP 14  
ADD #10,SP :  
3\$: RTS PC : 2816  
2797

23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<

3085 :MLX4  
 3086 :  
 3087 :  
 3088 :  
 3089 :  
 3090 :  
 3091 :  
 3092 :  
 3093 :  
 3094 :  
 3095 :  
 3096 :  
 3097 :  
 3098 :  
 3099 :  
 3100 :  
 3101 :  
 3102 :  
 3103 :  
 3104 :  
 3105 :  
 3106 :  
 3107 :  
 3108 :  
 3109 :  
 3110 :  
 3111 :  
 3112 :  
 3113 :  
 3114 :  
 3115 :  
 3116 :  
 3117 :  
 3118 :  
 3119 :  
 3120 :  
 3121 :  
 3122 :  
 3123 :  
 3124 :  
 3125 :  
 3126 :  
 3127 :  
 3128 :  
 3129 :  
 3130 :  
 3131 :  
 3132 :  
 3133 :  
 3134 :  
 3135 :  
 3136 :  
 3137 :  
 3138 :  
 3139 :

INITIALIZATION CODE

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 v2(206)  
 PA:<ROSEN>MLX4.BLI.1 (16)

2824 %SBTTL 'INITIALIZATION CODE'

2826 BGNINIT; !\* 1 \*

! INITIALIZATION CODE IS EXECUTED AT THE BEGINNING OF EACH  
 PASS, WHEN POWER DOWN/POWER UP HAS OCCURRED, OR WHEN THE  
 OPERATOR HAS ISSUED A START, RESTART OR CONTINUE COMMAND.

! DURING INITIALIZATION, THE 'GPHARD' MACRO IS USED TO GET  
 P-TABLE INFORMATION FOR THE LOGICAL UNIT UNDER TEST. THE  
 NUMBER OF UNITS AVAILABLE FOR TESTING IS CONTAINED IN A  
 HEADER LOCATION ('L\$UNIT').

! THE CODE WITHIN THE INITIALIZATION SECTION IS DIVIDED  
 INTO THREE CATEGORIES:

- 1) ONCE-ONLY CODE WHICH IS EXECUTED ONLY ON THE VERY FIRST PASS (I.E., WHEN THE OPERATOR HAS JUST TYPED IN THE 'START' COMMAND;
- 2) CODE WHICH SHOULD NOT BE EXECUTED ON THE FIRST PASS, BUT WHICH IS TO BE RUN ON EVERY SUBSEQUENT PASS;
- 3) COMMON CODE WHICH IS TO BE EXECUTED ON EVERY PASS (INCLUDING THE FIRST).

2851 LOCAL  
 2852 PLOC;

2855 IF ((READEF(EF\_START)) OR (READEF(EF\_RESTART)))  
 2856 THEN !THIS IS CATEGORY 1 CODE

2857 BEGIN !\* 2 \*

2858 QUICK = 1;

2859 CLRTBLS();

2860 INCR LUN FROM 0 TO (.L\$UNIT-1) DO

2861 BEGIN !\* 3 \*

2862 L\$LUN = .LUN;

2863 IF GPHARD(.LUN,PLOC) NEQ 0

2864 THEN

2865 BEGIN !\* 4 \*

2866 PTABLE ADDR[.LUN] = .PLOC;

2867 IF .NUM DRIVES EQL 0

2868 THEN INIT\_ADDRESSES(.PLOC);

2869 NUM DRIVES = .NUM DRIVES + 1;

2870 DRIVE STATUS[.LUN] = ACTIVE;

2871 CONFIG(.LUN);

2872 END !\* 4 \*

2873 ELSE

2874 DRIVE STATUS[.LUN] = INACTIVE;

2875 END; !\* 3 \*

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (16)

```

3141 ;MLX4
3142 :      INITIALIZATION CODE
3143 :
3144 :      2876      END      !* 2 *
3145 :      2877
3146 :      2878      ELSE      !THIS IS CATEGORY 2 CODE
3147 :      2879      QUICK = 0;
3148 :      2880
3149 :      2881      !CATEGORY 3 CODE WOULD GO HERE, BUT THERE IS NONE.
3150 :      2882
3151 :      2883      ENDINIT;  !* 1 *
3155 :
  
```

```

3156 :      .SBTTL  LINIT INITIALIZATION CODE
3160 035556 004137 005202      LINIT: JSR      R1,$SAVE4      ;
3161 035562 012700 000040      MOV      #40,R0      ;
3162 035566 104447      TRAP     47      ;
3163 035570 103404      BCS     1$      ;
3164 035572 012700 000037      MOV      #37,R0      ;
3165 035576 104447      TRAP     47      ;
3166 035600 103076      BHIS    7$      ;
3167 035602 012737 000001 030712 1$: MOV      #1,QUICK      ;
3168 035610 004737 032572      JSR     PC,CLRTBLS    ;
3169 035614 013704 002012      MOV     L$UNIT,R4    ;
3170 035620 005003      CLR     R3      ; LUN
3171 035622 000462      BR      6$      ;
3172 035624 010337 002074      2$: MOV     R3,L$LUN    ; LUN,*
3173 035630 010302      MOV     R3,R2      ; LUN,*
3174 035632 006202      ASR     R2      ;
3175 035634 006202      ASR     R2      ;
3176 035636 006202      ASR     R2      ;
3177 035640 062702 032460      ADD     #DRIVE.STATUS,R2
3178 035644 010300      MOV     R3,R0      ; LUN,*
3179 035646 104442      TRAP     42      ;
3180 035650 010001      MOV     R0,R1      ; *,PLOC
3181 035652 001432      BEQ     4$      ;
3182 035654 010300      MOV     R3,R0      ; LUN,*
3183 035656 006300      ASL     R0      ;
3184 035660 010160 032440      MOV     R1,PTABLE.ADDR(R0) ; PLOC,*
3185 035664 005737 032474      TST     NUM.DRIVES    ;
3186 035670 001004      BNE     3$      ;
3187 035672 010146      MOV     R1,-(SP)    ; PLOC,*
3188 035674 004737 033120      JSR     PC,INIT.ADDRESSES
3189 035700 005726      TST     (SP)+      ;
3190 035702 005237 032474      3$: INC     NUM.DRIVES    ;
3191 035706 010246      MOV     R2,-(SP)    ;
3192 035710 010346      MOV     R3,-(SP)    ; LUN,*
3193 035712 042716 177770      BIC     #177770,(SP)
3194 035716 012746 000001      MOV     #1,-(SP)
3195 035722 011646      MOV     (SP),-(SP)
  
```

```

3197 ;MLX4
3198 ;
3199 ;
3200 035724 004737 004502 JSR PC,BL$PU2
3201 035730 010316 MOV R3,(SP) ; LUN,* 2871
3202 035732 004737 033714 JSR PC,CONFIG
3203 035736 000411 BR 5$ ; 2863
3204 035740 010246 4$: MOV R2,-(SP) ; 2874
3205 035742 010346 MOV R3,-(SP) ; LUN,*
3206 035744 042716 177770 BIC #177770,(SP)
3207 035750 012746 000001 MOV #1,-(SP)
3208 035754 005046 CLR -(SP)
3209 035756 004737 004502 JSR PC,BL$PU2
3210 035762 062706 000010 5$: ADD #10,SP ; 2861
3211 035766 005203 INC R3 ; LUN 2860
3212 035770 020304 6$: CMP R3,R4 ; LUN,*
3213 035772 002714 BLT 2$
3214 035774 000207 RTS PC ; 2855
3215 035776 005037 030712 7$: CLR QUICK ; 2879
3216 036002 000207 RTS PC ; 2823
3217
3218 ; Routine Size: 75 words
3219 ; Maximum stack depth per invocation: 9 words
3224
3225
3229
3230 ;.SBTTL L$INIT INITIALIZATION CODE
3234 036004 004737 035556 L$INIT::JSR PC,LINIT ; 2879
3235 036010 104411 TRAP 11
3236 036012 000207 RTS PC
3237
3238 ; Routine Size: 4 words
3239 ; Maximum stack depth per invocation: 0 words
3244
3245
  
```

23-Oct-1980 15:01:43 TOPS  
 23-Oct-1980 14:57:05 PA:<

```

3247 :MLX4
3248 :
3249 :
3250 : 2884 %SBTTL 'GENERATOR FOR OPTION 1'
3251 : 2885
3252 : 2886 ROUTINE GEN1(SECTOR,COMP_FLAG): NOVALUE =
3253 : 2887 BEGIN !* 1 *
3254 : 2888
3255 : 2889 !**
3256 : 2890 ROUTINE: GEN1(SECTOR,COMP_FLAG)
3257 : 2891
3258 : 2892 PURPOSE: TO GENERATE THE ENTIRE 4K-WORD WRITE BUFFER FOR OPTION 1.
3259 : 2893 THE GENERATION IS IN 16 GROUPS OF 256 WORDS (THE SIZE
3260 : 2894 OF 1 SECTOR). EACH GROUP OF 256 WORDS RECEIVES THE
3261 : 2895 SECTOR ADDRESS WHERE THE WORDS WILL BE TRANSFERRED.
3262 : 2896
3263 : 2897 ARGUMENTS. (1) SECTOR - CURRENT SECTOR NUMBER.
3264 : 2898
3265 : 2899 (2) COMP_FLAG - AN INDICATOR WHICH IS USED TO DECIDE
3266 : 2900 WHETHER THE LOCATION IN THE WRITE BUFFER
3267 : 2901 SHOULD BE COMPLEMENTED.
3268 : 2902 !--
3269 : 2903
3270 : 2904 LOCAL
3271 : 2905 OFFSET;
3272 : 2906
3273 : 2907 OFFSET = 0;
3274 : 2908
3275 : 2909 IF .COMP_FLAG
3276 : 2910 THEN !GENERATE COMPLEMENT DATA
3277 : 2911 INCR SECT FROM (.SECTOR) TO ((.SECTOR) + 15) DO
3278 : 2912 BEGIN
3279 : 2913 BREAK;
3280 : 2914 INCR COUNT FROM 1 TO 256 DO
3281 : 2915 BEGIN !* 2A *
3282 : 2916 (WBUF + (.OFFSET)) = NOT (.SECT); !COMPLEMENT DATA
3283 : 2917 OFFSET = .OFFSET + 2;
3284 : 2918 END; !* 2A *
3285 : 2919 END
3286 : 2920
3287 : 2921 ELSE !GENERATE REGULAR DATA
3288 : 2922 INCR SECT FROM (.SECTOR) TO ((.SECTOR) + 15) DO
3289 : 2923 BEGIN
3290 : 2924 BREAK;
3291 : 2925 INCR COUNT FROM 1 TO 256 DO
3292 : 2926 BEGIN !* 2B *
3293 : 2927 (WBUF + (.OFFSET)) = (.SECT); !REGULAR DATA
3294 : 2928 OFFSET = .OFFSET + 2;
3295 : 2929 END; !* 2B *
3296 : 2930 END;
3297 : 2931
3298 : 2932 RETURN;
3299 : 2933
3300 : 2934 END; !* 1 *
  
```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (17)

```

3308
3309          .SBTTL  GEN1 GENERATOR FOR OPTION 1
3313 036014 004137 005202          GEN1: JSR    R1,$SAVE4          ;
3314 036020 005002                   CLR    R2                    ; OFFSET
3315 036022 016601 000016          MOV    16(SP),R1             ; SECTOR,*
3316 036026 010104                   MOV    R1,R4
3317 036030 062704 000017          ADD    #17,R4
3318 036034 032766 000001 000014  BIT    #1,14(SP)           ; *,COMP.FLAG
3319 036042 0C1424                   BEQ    4$
3320 036044 010103                   MOV    R1,R3                ; *,SECT
3321 036046 005303                   DEC    R3                    ; SECT
3322 036050 000415                   BR     3$
3323 036052 104422          1$: TRAP  22                    ;
3324 036054 010301                   MOV    R3,R1                ; SECT,*
3325 036056 005101                   COM    R1
3326 036060 012700 000001          MOV    #1,R0                ; *,COUNT
3327 036064 010162 010706          2$: MOV    R1,WBUFF(R2)     ; *,*(OFFSET)
3328 036070 062702 000002          ADD    #2,R2                ; *,OFFSET
3329 036074 005200                   INC    R0                    ; COUNT
3330 036076 020027 000400          CMP    R0,#400              ; COUNT,*
3331 036102 003770                   BLE    2$
3332 036104 005203          3$: INC    R3                    ; SECT
3333 036106 020304                   CMP    R3,R4                ; SECT,*
3334 036110 003760                   BLE    1$
3335 036112 000207                   RTS    PC                    ;
3336 036114 005301          4$: DEC    R1                    ;
3337 036116 000413                   BR     7$
3338 036120 104422          5$: TRAP  22                    ;
3339 036122 012700 000001          MOV    #1,R0                ; *,COUNT
3340 036126 010162 010706          6$: MOV    R1,WBUFF(R2)     ; SECT,*(OFFSET)
3341 036132 062702 000002          ADD    #2,R2                ; *,OFFSET
3342 036136 005200                   INC    R0                    ; COUNT
3343 036140 020027 000400          CMP    R0,#400              ; COUNT,*
3344 036144 003770                   BLE    6$
3345 036146 005201          7$: INC    R1                    ; SECT
3346 036150 020104                   CMP    R1,R4                ; SECT,*
3347 036152 003762                   BLE    5$
3348 036154 000207                   RTS    PC                    ;
3349
3350
3351
3356
  
```

```

; Routine Size: 49 words
; Maximum stack depth per invocation: 5 words
  
```

```

3358 ;MLX4
3359 :
3360 :
3361 : 2935 %SBTTL 'PATTERN NUMBER SELECTION'
3362 : 2936
3363 : 2937 ROUTINE SELPAT : NOVALUE =
3364 : 2938 BEGIN
3365 : 2939
3366 : 2940 !++
3367 : 2941 ROUTINE: SELPAT
3368 : 2942
3369 : 2943 PURPOSE: TO SELECT THE NEXT PATTERN NUMBER IN OPTION 2.
3370 : 2944
3371 : 2945 THIS ROUTINE AUTOMATICALLY COMPLEMENTS THE CURRENT
3372 : 2946 PATTERN NUMBER AND EXAMINES THE SIGN OF THE RESULT.
3373 : 2947 IF THE RESULT IS NEGATIVE, THEN IT IS TAKEN TO BE
3374 : 2948 THE PATTERN NUMBER OF A COMPLEMENT PATTERN AND A
3375 : 2949 'RETURN' IS EXECUTED. IF, HOWEVER, THE RESULT IS
3376 : 2950 POSITIVE, THEN AN ENTIRELY NEW PATTERN NUMBER IS
3377 : 2951 REQUIRED. THE NEW PATTERN NUMBER IS OBTAINED BY
3378 : 2952 INCREMENTING THE OLD.
3379 : 2953 !--
3380 : 2954
3381 : 2955
3382 : 2956 PATTERN = -(.PATTERN);
3383 : 2957
3384 : 2958 IF .PATTERN GEQ 0
3385 : 2959 THEN PATTERN = .PATTERN + 1;
3386 : 2960
3387 : 2961 RETURN;
3388 : 2962
3389 : 2963 END;
3393 :
3394 :
3398 036156 005437 030714 .SBTTL SELPAT PATTERN NUMBER SELECTION
3399 036162 005737 030714 SELPAT: NEG PATTERN : 2956
3400 036166 002402 BLT 1$ : 2958
3401 036170 005237 030714 INC PATTERN : 2959
3402 036174 000207 1$: RTS PC : 2937
3403 :
3404 : ; Routine Size: 8 words
3405 : ; Maximum stack depth per invocation: 0 words
3410 :
3411 :
  
```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 v2(206)  
 PA:<ROSEN>MLX4.BLI.1 (18)

```

3413 :MLX4
3414 :
3415 :
3416 : 2964 %SBTTL 'USING THE PATTERN TABLE'
3417 : 2965
3418 : 2966 ROUTINE GETCNTS(PATTERN) =
3419 : 2967 BEGIN
3420 : 2968
3421 : 2969
3422 : 2970
3423 : 2971
3424 : 2972
3425 : 2973
3426 : 2974
3427 : 2975
3428 : 2976
3429 : 2977
3430 : 2978
3431 : 2979
3432 : 2980
3433 : 2981
3434 : 2982
3435 : 2983
3436 : 2984
3437 : 2985
3438 : 2986
3439 : 2987
3440 : 2988
3441 : 2989
3442 : 2990
3443 : 2991
3444 : 2992
3445 : 2993
3446 : 2994
3447 : 2995
3448 : 2996
3449 : 2997
3450 : 2998
3451 : 2999
3452 : 3000
3453 : 3001
3454 : 3002
3455 : 3003
3456 : 3004
3457 : 3005
3458 : 3006
3459 : 3007
3460 : 3008
3461 : 3009
3462 : 3010
3463 : 3011
3464 : 3012
3465 : 3013
3466 : 3014
3467 : 3015

USING THE PATTERN TABLE

ROUTINE GETCNTS(PATTERN) =
BEGIN
++
ROUTINE: GETCNTS(PATTERN)
PURPOSE: TO POINT TO THE APPROPRIATE BLOCK OF THE PATTERN
TABLE AND GRAB THE VALUES WHICH ARE REQUIRED FOR THE
PATTERN GENERATOR FOR OPTIONS 2 AND 4.
ARGUMENT: PATTERN = THE CURRENT PATTERN NUMBER.
RESULTS: (1) 'VALUE' RECEIVES THE VALUE WHICH IS RETURNED FOR
THE SUBROUTINE. IT IS THE 4-BIT BINARY AMOUNT WHICH
WILL BE USED AS THE CONTENTS OF A NIBBLE OF COMPLEMENT
DATA.
(2) 'DATA_COUNT' RECEIVES THE NUMBER OF NIBBLES OF DATA.
(3) 'COMP_COUNT' RECEIVES THE NUMBER OF NIBBLES OF
COMPLEMENT DATA.
--
LOCAL
PATNUM, VALUE;
SELECTONE .PATTERN OF
SET
[1 TO 5] : BEGIN
PATNUM = .PATTERN - 1;
VALUE = %B'1010';
END;
[-5 TO -1] : BEGIN
PATNUM = -(.PATTERN) - 1;
VALUE = %B'0101';
END;
[6 TO 10] : BEGIN
PATNUM = .PATTERN - 6;
VALUE = %B'1111';
END;
[-10 TO -6] : BEGIN
PATNUM = -(.PATTERN) - 6;
VALUE = %B'0000';
END;
TES;

```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (19)



23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (19)

```

3469 ;MLX4
3470 :
3471 :
3472 : 3016 DATA_COUNT = .PATTBL[.PATNUM,COUNT1];
3473 : 3017 COMP_COUNT = .PATTBL[.PATNUM,COUNT2];
3474 : 3018
3475 : 3019 RETURN .VALUE;
3476 : 3020
3477 : 3021 END;
3481 :
3482 :
  
```

```

3486 036176 004137 005150          .SBTTL GETCNTS USING THE PATTERN TABLE
3487 036202 016600 000010 GETCNTS:JSR R1,$SAVE2          ; 2966
3488 036206 003410          MOV 10(SP),R0          ; PATTERN,* 2992
3489 036210 020027 000005          BLE 1$
3490 036214 003005          CMP R0,#5
3491 036216 010001          BGT 1$
3492 036220 005301          MOV R0,R1          ; *,PATNUM 2996
3493 036222 012702 000012          DEC R1          ; PATNUM
3494 036226 000441          MOV #12,R2          ; *,VALUE 2997
3495 036230 020027 177773          BR 4$          ; 2992
3496 036234 002410          1$: CMP R0,#-5
3497 036236 005700          BLT 2$
3498 036240 002006          TST R0
3499 036242 012701 177777          BGE 2$
3500 036246 160001          MOV #-1,R1          ; *,PATNUM 3001
3501 036250 012702 000005          SUB R0,R1          ; *,PATNUM
3502 036254 000426          MOV #5,R2          ; *,VALUE 3002
3503 036256 020027 000006          BR 4$          ; 2992
3504 036262 002411          2$: CMP R0,#6
3505 036264 020027 000012          BLT 3$
3506 036270 003006          CMP R0,#12
3507 036272 010001          BGT 3$
3508 036274 162701 000006          MOV R0,R1          ; *,PATNUM 3006
3509 036300 012702 000017          SUB #6,R1          ; *,PATNUM
3510 036304 000412          MOV #17,R2          ; *,VALUE 3007
3511 036306 020027 177766          BR 4$          ; 2992
3512 036312 002407          3$: CMP R0,#-12
3513 036314 020027 177772          BLT 4$
3514 036320 003004          CMP R0,#-6
3515 036322 012701 177772          BGT 4$
3516 036326 160001          MOV #-6,R1          ; *,PATNUM 3011
3517 036330 005002          SUB R0,R1          ; *,PATNUM
3518 036332 006301          CLR R2          ; VALUE 3012
3519 036334 006301          4$: ASL R1          ; 3016
3520 036336 016137 032536 030716          ASL R1
3521 036344 016137 032540 030720          MOV PATTBL(R1),DATA_COUNT
3522 036352 010200          MOV PATTBL+2(R1),COMP_COUNT
3523 036354 000207          MOV R2,R0          ; VALUE,* 3017
          RTS PC          ; 2967
          ; 2966
  
```

3525  
3526  
3527  
3528  
3529  
3530  
3535  
3536

;MLX4  
;

USING THE PATTERN TABLE

; Routine Size: 56 words  
; Maximum stack depth per invocation: 3 words

23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<

```

3538 :MLX4
3539 :
3540 :
3541 : 3022 %SBTTL 'NIBBLE GENERATOR'
3542 : 3023
3543 : 3024 ROUTINE FILLER(BUFFER,WRDCNT,VALUE): NOVALUE =
3544 : 3025 BEGIN !* 1 *
3545 : 3026
3546 : 3027 !**
3547 : 3028 ROUTINE: FILLER(BUFFER,WRDCNT,VALUE)
3548 : 3029
3549 : 3030 PURPOSE: TO LOAD A CHOSEN WRITE BUFFER, ONE NIBBLE AT A TIME, WITH A
3550 : 3031 PARTICULAR PATTERN, FOR A SPECIFIED NUMBER OF 16-BIT WORDS.
3551 : 3032 THIS ROUTINE IS THE HEART OF THE GENERATOR FOR OPTIONS 2 AND 4.
3552 : 3033
3553 : 3034 THE 'FILLER' ROUTINE IS A LOOP WHICH WILL ALTERNATE BETWEEN
3554 : 3035 THE TWO COUNTS AND FILL THE APPROPRIATE NUMBER OF NIBBLES OF
3555 : 3036 THE CHOSEN BUFFER WITH THE CONTENTS OF 'VALUE'. EVERY TIME
3556 : 3037 A COUNT IS EXHAUSTED, THE OTHER COUNT IS STARTED UP AGAIN,
3557 : 3038 AND EACH TIME A NEW COUNT IS BEGUN, 'VALUE' IS COMPLEMENTED.
3558 : 3039 NOTE THAT THE FIRST COUNT THAT WILL BE USED WILL ALWAYS BE
3559 : 3040 'DATA_COUNT', SO THAT THE BUFFER WILL BE REPETITIONS OF:
3560 : 3041 DATA, COMP, DATA, COMP ...
3561 : 3042
3562 : 3043 ARGUMENTS: (1) BUFFER = THE STARTING ADDRESS OF THE CHOSEN WRITE BUFFER.
3563 : 3044
3564 : 3045 (2) WRDCNT = THE NUMBER OF WORDS OF PATTERN TO BE GENERATED.
3565 : 3046
3566 : 3047 (3) VALUE = NIBBLE OF DATA TO BE PLACED IN THE WRITE BUFFER
3567 : 3048 --
3568 : 3049
3569 : 3050 LOCAL
3570 : 3051 NIBBLE, OFFSET, FLAG, THE_COUNT;
3571 : 3052
3572 : 3053 FLAG = 0;
3573 : 3054 OFFSET = 0;
3574 : 3055 NIBBLE = 0;
3575 : 3056
3576 : 3057 WHILE 1 DO
3577 : 3058 BEGIN !* 2 *
3578 : 3059 FLAG = NOT .FLAG; !FLIP THE FLAG
3579 : 3060 VALUE = NOT .VALUE; !AND THE DATA
3580 : 3061
3581 : 3062 IF .FLAG EQL 0 !CHOOSE A COUNT
3582 : 3063 THEN
3583 : 3064 THE_COUNT = .COMP_COUNT
3584 : 3065 ELSE
3585 : 3066 THE_COUNT = .DATA_COUNT;
3586 : 3067
3587 : 3068 DECR COUNT FROM .THE_COUNT TO 1 DO !EXHAUST THE COUNT
3588 : 3069 BEGIN !* 3 *
3589 : 3070 (.BUFFER + (.OFFSET))<.NIBBLE,4,0> = .VALUE; !LOAD THE NIBBLE
3590 : 3071 NIBBLE = (((.NIBBLE) + 4) MOD 16); !CHANGE NIBBLE POINTER
3591 : 3072 IF .NIBBLE EQL 0
3592 : 3073 THEN
  
```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (20)

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (20)

```

3594 :MLX4
3595 :
3596 :
3597 : 3074 BEGIN !* 4 *
3598 : 3075 BREAK;
3599 : 3076 OFFSET = .OFFSET + 2;
3600 : 3077 WRDCNT = .WRDCNT - 1;
3601 : 3078 IF .WRDCNT EQL 0
3602 : 3079 THEN RETURN;
3603 : 3080 END; !* 4 *
3604 : 3081 END; !* 3 *
3605 : 3082 END; !* 2 *
3606 : 3083 END; !* 1 *
3610 :
3611 :
  
```

```

3615 036356 004137 005222 FILLER: .SBTTL FILLER NIBBLE GENERATOR
3616 036362 005001 JSR R1,$SAVE5
3617 036364 005005 CLR R1
3618 036366 005002 CLR R5
3619 036370 005101 CLR R2
3620 036372 005166 000016 1$: COM R1
3621 036376 005701 COM 16(SP)
3622 036400 001003 TST R1
3623 036402 013704 030720 BNE 2$
3624 036406 000402 BR 3$
3625 036410 013704 030716 2$: MOV DATA.COUNT,R4
3626 036414 010403 3$: MOV R4,R3
3627 036416 003764 BLE 1$
3628 036420 010546 4$: MOV R5,-(SP)
3629 036422 066616 000024 ADD 24(SP),(SP)
3630 036426 010246 MOV R2,-(SP)
3631 036430 012746 000004 MOV #4,-(SP)
3632 036434 016646 000024 MOV 24(SP),-(SP)
3633 036440 004737 004502 JSR PC,BL$PU2
3634 036444 010216 MOV R2,(SP)
3635 036446 062716 000004 ADD #4,(SP)
3636 036452 012746 000020 MOV #20,-(SP)
3637 036456 004737 005070 JSR PC,BL$MOD
3638 036462 010002 MOV R0,R2
3639 036464 001011 BNE 6$
3640 036466 104422 TRAP 22
3641 036470 062705 000002 ADD #2,R5
3642 036474 005366 000032 DEC 32(SP)
3643 036500 001003 BNE 6$
3644 036502 062706 000012 ADD #12,SP
3645 036506 000207 5$: RTS PC
3646 036510 062706 000012 6$: ADD #12,SP
3647 036514 005303 DEC R3
3648 036516 001340 BNE 4$
  
```

```

:
: FLAG 3024
: OFFSET 3053
: NIBBLE 3054
: FLAG 3055
: VALUE 3059
: FLAG 3060
: FLAG 3062
: *,THE.COUNT 3064
: 3062
: *,THE.COUNT 3066
: THE.COUNT,COUNT 3068
:
: OFFSET,* 3070
: BUFFER,*
: NIBBLE,*
:
: VALUE,*
: NIBBLE,* 3071
:
: *,NIBBLE 3072
:
: *,OFFSET 3074
: WRDCNT 3076
: 3077
: 3078
: 3024
: 3079
: 3069
: COUNT 3068
  
```

3650  
3651  
3652  
3653 036520 000723  
3654  
3655  
3656  
3661  
3662

:MLX4  
;  
NIBBLE GENERATOR  
BR 1\$ ;  
; Routine Size: 50 words  
; Maximum stack depth per invocation: 11 words

23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<  
3057

3664 :MLX4  
 3665 :  
 3666 :  
 3667 : 23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 3668 : 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (21)

```

3667 : 3084 %SBTTL 'GENERATOR FOR OPTION 2'
3668 : 3085
3669 : 3086 ROUTINE GEN2 : NOVALUE =
3670 : 3087 BEGIN
3671 : 3088
3672 : 3089 !++
3673 : 3090 ROUTINE: GEN2
3674 : 3091
3675 : 3092 PURPOSE: TO GENERATE THE ENTIRE WRITE BUFFER (WHICH
3676 : 3093 WILL BE USED IN CONJUNCTION WITH OPTION 2)
3677 : 3094 BASED ON PATTERN TABLE ENTRIES.
3678 : 3095 !--
3679 : 3096
3680 : 3097 LOCAL
3681 : 3098 VALUE;
3682 : 3099
3683 : 3100 VALUE = GETCNTS(.PATTERN);
3684 : 3101 FILLER(WBUFF,BUFSIZ,.VALUE);
3685 : 3102 RETURN;
3686 : 3103 END;
  
```

```

3690
3691 .SBTTL GEN2 GENERATOR FOR OPTION 2
3695 036522 013746 030714 GEN2: MOV PATTERN,-(SP) ; 3100
3696 036526 004737 036176 JSR PC,GETCNTS ;
3697 036532 012716 010706 MOV #WBUFF,(SP) ; 3101
3698 036536 012746 004000 MOV #4000,-(SP) ;
3699 036542 010046 MOV R0,-(SP) ; VALUE,*
3700 036544 004737 036356 JSR PC,FILLER ;
3701 036555 062706 000006 ADD #6,SP ; 3086
3702 036554 000207 RTS PC ;
  
```

3703  
 3704 : Routine Size: 14 words  
 3705 : Maximum stack depth per invocation: 3 words  
 3710  
 3711

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (22)

```

3713 :MLX4
3714 : GENERATOR FOR OPTION 3
3715 :
3716 : 3104 %SBTTL 'GENERATOR FOR OPTION 3'
3717 : 3105
3718 : 3106 ROUTINE GEN3(COMP_FLAG): NOVALUE =
3719 : 3107 BEGIN
3720 : 3108
3721 : 3109 !++
3722 : 3110 ! ROUTINE: GEN3(COMP_FLAG)
3723 : 3111 !
3724 : 3112 ! PURPOSE: TO GENERATE THE ENTIRE WRITE BUFFER WHICH WILL BE MADE
3725 : 3113 ! UP OF WORDS WHICH CONTAIN THE UNIQUE COUNT FROM ONE TO
3726 : 3114 ! BUFSIZ.
3727 : 3115 !
3728 : 3116 ! ARGUMENT: COMP_FLAG - THIS IS AN INDICATOR OF WHETHER OR NOT TO
3729 : 3117 ! COMPLEMENT EVERY WORD IN THE WRITE BUFFER.
3730 : 3118 !--
3731 : 3119
3732 : 3120 LOCAL
3733 : 3121 OFFSET;
3734 : 3122
3735 : 3123 OFFSET = 0;
3736 : 3124
3737 : 3125 IF .COMP_FLAG
3738 : 3126 THEN !GENERATE COMPLEMENT DATA
3739 : 3127 INCR COUNT FROM 1 TO BUFSIZ DO
3740 : 3128 BEGIN
3741 : 3129 (WBUFF + (.OFFSET)) = NOT (.COUNT); !COMPLEMENT DATA
3742 : 3130 OFFSET = .OFFSET + 2;
3743 : 3131 END
3744 : 3132
3745 : 3133 ELSE !GENERATE REGULAR DATA
3746 : 3134 INCR COUNT FROM 1 TO BUFSIZ DO
3747 : 3135 BEGIN
3748 : 3136 (WBUFF + (.OFFSET)) = (.COUNT); !REGULAR DATA
3749 : 3137 OFFSET = .OFFSET + 2;
3750 : 3138 END;
3751 : 3139
3752 : 3140 RETURN;
3753 : 3141
3754 : 3142 END;
3758
3759

```

```

3763 036556 010146 GEN3: .SBTTL GEN3 GENERATOR FOR OPTION 3
3764 036560 005001 MOV R1,-(SP) ; 3106
3765 036562 032766 000001 000004 CLR R1 ; OFFSET 3123
3766 036570 001415 BIT #1,4(SP) ; *.COMP.FLAG 3125
3767 036572 012700 000001 BEQ 2$
MOV #1,R0 ; *.COUNT 3127

```

```

3769 ;MLX4
3770 ;
3771 ;
3772 036576 010061 010706 1$: MOV R0,WBUFF(R1) ; COUNT,*(OFFSET) 3129
3773 036602 005161 010706 COM WBUFF(R1) ; *(OFFSET)
3774 036606 062701 000002 ADD #2,R1 ; *,OFFSET 3130
3775 036612 005200 INC R0 ; COUNT 3127
3776 036614 020027 C04000 CMP R0,#4000 ; COUNT,*
3777 036620 003766 BLE 1$ ;
5778 036622 000412 BR 4$ ; 3125
3779 036624 012700 000001 2$: MOV #1,R0 ; *,COUNT 3134
3780 036630 010061 010706 3$: MOV R0,WBUFF(R1) ; COUNT,*(OFFSET) 3136
3781 036634 062701 000002 ADD #2,R1 ; *,OFFSET 3137
3782 036640 005200 INC R0 ; COUNT 3134
3783 036642 020027 004000 CMP R0,#4000 ; COUNT,*
3784 036646 003770 BLE 3$ ;
3785 036650 012601 4$: MOV (SP)+,R1 ; 3106
3786 036652 000207 RTS PC ;
3787 ;
3788 ; Routine Size: 31 words
3789 ; Maximum stack depth per invocation: 2 words
3794 ;
3795 ;
  
```



3797 :MLX4  
 3798 : GENERATOR FOR OPTION 4  
 3799 : 23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (23)

```

3800 : 3143 %SBTTL 'GENERATOR FOR OPTION 4'
3801 : 3144
3802 : 3145
3803 : 3146 ROUTINE GEN4(MARPAT,BUFFER) : NOVALUE =
3804 : 3147 BEGIN
3805 : 3148
3806 : 3149 !**
3807 : 3150 ROUTINE: GEN4(MARPAT,BUFFER)
3808 : 3151
3809 : 3152 PURPOSE: THIS IS THE GENERATOR WHICH IS USED FOR OPTION 4, THE
3810 : 3153 MARCH TEST. IT BEGINS AT THE START OF THE CHOSEN
3811 : 3154 BUFFER AND GENERATES 256 WORDS IN THE EXACT WAY THAT
3812 : 3155 OPTION 2 DOES.
3813 : 3156
3814 : 3157 ARGUMENTS: (1) MARPAT - THE MARCH PATTERN NUMBER
3815 : 3158
3816 : 3159 (2) BUFFER - THE STARTING ADDRESS OF THE CHOSEN BUFFER
3817 : 3160 !--
3818 : 3161
3819 : 3162 LOCAL
3820 : 3163 VALUE;
3821 : 3164
3822 : 3165 VALUE = GETCNTS(.MARPAT);
3823 : 3166 FILLER(.BUFFER,256,.VALUE);
3824 : 3167 RETURN;
3825 : 3168 END;
  
```

```

3830 : .SBTTL GEN4 GENERATOR FOR OPTION 4
3834 036654 016646 000004 GEN4: MOV 4(SP),-(SP) ; MARPAT,* 3165
3835 036660 004737 036176 JSR PC,GETCNTS
3836 036664 016616 000004 MOV 4(SP),(SP) ; BUFFER,* 3166
3837 036670 012746 000400 MOV #400,-(SP)
3838 036674 010046 MOV R0,-(SP) ; VALUE,*
3839 036676 004737 036356 JSR PC,FILLER
3840 036702 062706 000006 ADD #6,SP ;
3841 036706 000207 RTS PC 3146
  
```

3842 :  
 3843 : ; Routine Size: 14 words  
 3844 : ; Maximum stack depth per invocation: 3 words  
 3849 :  
 3850 :

```

3852 :MLX4
3853 :
3854 :
3855 : 3169 %SBTTL 'GENERATOR FOR OPTION 5'
3856 : 3170
3857 : 3171 ROUTINE GEN5 : NOVALUE =
3858 : 3172 BEGIN
3859 : 3173
3860 : 3174 !++
3861 : 3175 ! ROUTINE: GEN5
3862 : 3176 !
3863 : 3177 ! PURPOSE: THIS IS THE GENERATOR WHICH IS USED FOR OPTION 5.
3864 : 3178 ! IT BEGINS AT THE START OF THE WRITE BUFFER, AND GENERATES
3865 : 3179 ! A FULL BUFFER OF RANDOM DATA. THE FIRST 3 WORDS OF THE BUFFER
3866 : 3180 ! RECEIVE THE 3 SEEDS WHICH WERE USED IN THE GENERATION.
3867 : 3181 !--
3868 : 3182 LOCAL
3869 : 3183 OFFSET;
3870 : 3184
3871 : 3185 (WBUFF + 0) = .SEED1; !THE FIRST 3 WORDS OF THE
3872 : 3186 (WBUFF + 2) = .SEED2; !WRITE BUFFER RECEIVE THE
3873 : 3187 (WBUFF + 4) = .SEED3; !3 GENERATING SEEDS.
3874 : 3188 OFFSET = 6;
3875 : 3189
3876 : 3190 INCR COUNT FROM 4 TO BUFSIZ DO
3877 : 3191 BEGIN
3878 : 3192 RN(); !NOTE: USING SIGNED VALUES (FULL 16 BITS)
3879 : 3193
3880 : 3194 BREAK;
3881 : 3195 (WBUFF + .OFFSET) = .RANDOM;
3882 : 3196 OFFSET = .OFFSET + 2;
3883 : 3197 END;
3884 : 3198
3885 : 3199 RETURN;
3886 : 3200 END;
  
```

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (24)

```

3890 :
3891 :
3895 036710 004137 005150          GEN5: .SBTTL GEN5 GENERATOR FOR OPTION 5
3896 036714 013737 005344 010706 JSR      R1,$SAVE2
3897 036722 013737 005346 010710 MOV     SEED1,WBUFF
3898 036730 013737 005350 010712 MOV     SEED2,WBUFF+2
3899 036736 012701 000006          MOV     SEED3,WBUFF+4
3900 036742 012702 000004          MOV     #6,R1
3901 036746 004737 005256          1$: JSR   PC,RN
3902 036752 104422          TRAP   22
3903 036754 013761 005352 010706 MOV     RANDOM,WBUFF(R1)
3904 036762 062701 000002          ADD    #2,R1
3905 036766 005202          INC   R2
3906 036770 020227 004000          CMP   R2,#4000
  
```

3171  
 3185  
 3186  
 3187  
 3188  
 3190  
 3192  
 3195  
 3196  
 3190

3908  
3909  
3910  
3911 036774 003764  
3912 036776 000207  
3913  
3914  
3915  
3920  
3921

;MLX4  
;  
GENERATOR FOR OPTION 5  
  
BLE 1\$  
RTS PC ;  
  
; Routine Size: 28 words  
; Maximum stack depth per invocation: 3 words

23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<

3171

```
3923 :MLX4
3924 :
3925 :
3926 : 3201 %SBTTL 'SYSTEM ERROR DETECTOR'
3927 : 3202
3928 : 3203 ROUTINE SYSERR(LUN) =
3929 : 3204 BEGIN !* 1 *
3930 : 3205
3931 : 3206 !++
3932 : 3207 ROUTINE: SYSERR(LUN)
3933 : 3208
3934 : 3209 PURPOSE: (1) TO SCAN FOR SYSTEM ERRORS AFTER A TRANSFER HAS ENDED.
3935 : 3210
3936 : 3211 (2) PRINT ERROR MESSAGES IF APPROPRIATE:
3937 : 3212
3938 : 3213 A) ERROR MESSAGES DURING RETRIES WILL NOT
3939 : 3214 BE PRINTED.
3940 : 3215
3941 : 3216 B) THE OPERATOR HAS THE ABILITY TO INHIBIT THE
3942 : 3217 PRINTING OF DATA ERRORS, BUT NOT SYSTEM ERRORS.
3943 : 3218
3944 : 3219 (3) DECIDE ON A VALUE TO RETURN WHICH WILL INDICATE THE
3945 : 3220 RELATIVE SEVERITY OF THE ERROR.
3946 : 3221
3947 : 3222 ARGUMENT: LUN = THE CURRENT LOGICAL UNIT NUMBER
3948 : 3223 [0 TO NUMBER OF DRIVES MINUS 1].
3949 : 3224
3950 : 3225 RESULT: VALUE RETURNED FOR THE SUBROUTINE SHOWS TYPE OF ERROR:
3951 : 3226 0 = NO ERRORS AT ALL
3952 : 3227 1 = RETRY ALLOWED FOR THESE TYPES OF ERRORS
3953 : 3228 2 = FATAL CONTROLLER ERROR
3954 : 3229 3 = FATAL DRIVE ERROR
3955 : 3230 4 = UNCORRECTABLE DATA ERROR
3956 : 3231 5 = CORRECTABLE DATA ERROR
3957 : 3232
3958 : 3233 FIGURE OUT WHICH VALUE TO RETURN BY THE FOLLOWING SCHEME:
3959 : 3234
3960 : 3235 (1) IF THERE IS A CHOICE BETWEEN DATA ERROR AND SYSTEM ERROR,
3961 : 3236 FIRST CHOOSE THE DATA ERROR.
3962 : 3237 (2) IF THERE IS A CHOICE BETWEEN FATAL AND NON-FATAL ERROR,
3963 : 3238 FIRST CHOOSE THE FATAL ERROR.
3964 : 3239 (3) IF THERE IS A CHOICE BETWEEN CONTROLLER ERROR AND DRIVE ERROR,
3965 : 3240 FIRST CHOOSE CONTROLLER ERROR.
3966 : 3241 !--
3967 : 3242
3968 : 3243 LABEL
3969 : 3244 PURPOSE_2_CODE;
3970 : 3245
3971 : 3246 LOCAL
3972 : 3247 ERR2,ERR3,ERR4,ERR5,ERR6,ERR7;
3973 : 3248
3974 : 3249 ERR2 = INACTIVE;
3975 : 3250 ERR3 = INACTIVE;
3976 : 3251 ERR4 = INACTIVE;
3977 : 3252 ERR5 = INACTIVE;
```

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (25)

```

3979 :MLX4
3980 :
3981 :
3982 : 3253 SYSTEM ERROR DETECTOR
3983 : 3254 ERR6 = INACTIVE;
3984 : 3255 ERR7 = INACTIVE;
3985 : 3256
3986 : 3257 !+
3987 : 3258 ! THIS IS THE CODE FOR PURPOSE 1:
3988 : 3259 -
3989 : 3260 IF .SC
3990 : 3261 THEN !THERE ARE SOME ERRORS -- FIND OUT WHAT KIND
3991 : 3262 BEGIN !* 2 *
3992 : 3263 IF ((.NED) OR (.NEM) OR (.PGE))
3993 : 3264 THEN ERR2 = ACTIVE;
3994 : 3265 IF ((.DLT) OR (.PE) OR (.MXF) OR (.MDPE) OR (.MCPE))
3995 : 3266 THEN ERR3 = ACTIVE;
3996 : 3267 IF ((.WCE) AND (.ECCDIS EQL 0))
3997 : 3268 THEN ERR3 = ACTIVE;
3998 : 3269 IF ((.UNS) OR (.IAE) OR (.AOE) OR (.RMR) OR (.ILR) OR (.ILF))
3999 : 3270 THEN ERR4 = ACTIVE;
4000 : 3271 IF ((.OPI) OR (.DPAR) OR (.CPAR))
4001 : 3272 THEN ERR5 = ACTIVE;
4002 : 3273 IF ((.DCK) OR (.CRC) OR (.SGL))
4003 : 3274 THEN ERR7 = ACTIVE;
4004 : 3275 IF ((.WCE) AND (.ECCDIS EQL 1))
4005 : 3276 THEN ERR7 = ACTIVE;
4006 : 3277 IF ((.ECH) OR (.UNC))
4007 : 3278 THEN ERR6 = ACTIVE;
4008 : 3279 END !* 2 *
4009 : 3280 ELSE !THERE ARE NO ERRORS -- RETURN SUCCESSFULLY
4010 : 3281 RETURN 0; !NO ERRORS AT ALL
4011 : 3282
4012 : 3283 !+
4013 : 3284 ! THIS IS THE CODE FOR PURPOSE 2:
4014 : 3285
4015 : 3286 LABEL:
4016 : 3287 BEGIN 3
4017 : 3288 IF RETRYING
4018 : 3289 THEN LEAVE LABEL (NO PRINTOUTS AT ALL)
4019 : 3290 ELSE
4020 : 3291 BEGIN 4
4021 : 3292 IF ERROR PRINTOUTS ARE ALLOWED
4022 : 3293 THEN
4023 : 3294 BEGIN 5A
4024 : 3295 IDENTIFY THE LOGICAL UNIT
4025 : 3296 SCAN FOR & PRINT DATA ERRORS
4026 : 3297 END 5A
4027 : 3298 ELSE
4028 : 3299 BEGIN 5B
4029 : 3300 IF THERE ARE SYSTEM ERRORS TO REPORT
4030 : 3301 THEN IDENTIFY THE LOGICAL UNIT
4031 : 3302 ELSE LEAVE LABEL
4032 : 3303 END 5B
4033 : 3304 SCAN FOR & PRINT SYSTEM ERRORS
  
```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (25)

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (25)

```

4035 :MLX4
4036 :
4037 :
4038 :          3305          |          END 4
4039 :          3306          |          END 3
4040 :          3307          |          |
4041 :          3308          |          |
4042 :          3309 PURPOSE_2_CODE:
4043 :          3310 BEGIN !* 3 *
4044 :          3311 IF .RETRYING
4045 :          3312 THEN LEAVE PURPOSE_2_CODE !(NO PRINTOUTS AT ALL)
4046 :          3313 ELSE
4047 :          3314 BEGIN !* 4 *
4048 :          3315 IF .ERR0UT
4049 :          3316 THEN
4050 :          3317 BEGIN !* 5A *
4051 :          3318 SAYWHO(.LUN);
4052 :          3319 PRINTB(SAY1,PHR10);
4053 :          3320 !'ERROR BITS SET:
4054 :          3321 IF .ERR7
4055 :          3322 THEN
4056 :          3323 BEGIN
4057 :          3324 IF .DCK
4058 :          3325 THEN PRINTB(FMT15,MLB20);
4059 :          3326 IF .CRC
4060 :          3327 THEN PRINTB(FMT15,MLB22)
4061 :          3328 IF .SGL
4062 :          3329 THEN PRINTB(FMT15,MLB23);
4063 :          3330 IF ((.WCE) AND (.ECCDIS EQL 1))
4064 :          3331 THEN PRINTB(FMT15,MLB6);
4065 :          3332 END;
4066 :          3333 IF .ERR6
4067 :          3334 THEN
4068 :          3335 BEGIN
4069 :          3336 IF .ECH
4070 :          3337 THEN PRINTB(FMT15,MLB21);
4071 :          3338 IF .UNC
4072 :          3339 THEN PRINTB(FMT15,MLB24);
4073 :          3340 END;
4074 :          3341 END !* 5A *
4075 :          3342 ELSE
4076 :          3343 BEGIN !* 5B *
4077 :          3344 IF ((.ERR2) OR (.ERR3) OR (.ERR4) OR (.ERR5))
4078 :          3345 THEN
4079 :          3346 BEGIN
4080 :          3347 SAYWHO(.LUN);
4081 :          3348 PRINTB(SAY1,PHR10);
4082 :          3349 !'ERROR BITS SET:
4083 :          3350 END
4084 :          3351 ELSE LEAVE PURPOSE_2_CODE;
4085 :          3352 END; !* 5B *
4086 :          3353 IF .ERR2
4087 :          3354 THEN
4088 :          3355 BEGIN
4089 :          3356 IF .NED
  
```

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (25)

```
4091 :MLX4
4092 :
4093 :
4094 : 3357     THEN PRINTB(FMT15,MLB2);
4095 : 3358     IF .NEM
4096 : 3359     THEN PRINTB(FMT15,MLB3);
4097 : 3360     IF .PGE
4098 : 3361     THEN PRINTB(FMT15,MLB4);
4099 : 3362     END;
4100 : 3363     IF .ERR3
4101 : 3364     THEN
4102 : 3365         BEGIN
4103 : 3366             IF .DLT
4104 : 3367             THEN PRINTB(FMT15,MLB5);
4105 : 3368             IF ((.WCE) AND (.ECCDIS EQL 0))
4106 : 3369             THEN PRINTB(FMT15,MLB6);
4107 : 3370             IF .PE
4108 : 3371             THEN PRINTB(FMT15,MLB7);
4109 : 3372             IF .MXF
4110 : 3373             THEN PRINTB(FMT15,MLB8);
4111 : 3374             IF .MDPE
4112 : 3375             THEN PRINTB(FMT15,MLB9);
4113 : 3376             IF .MCPE
4114 : 3377             THEN PRINTB(FMT15,MLB10);
4115 : 3378             END;
4116 : 3379     IF .ERR4
4117 : 3380     THEN
4118 : 3381         BEGIN
4119 : 3382             IF .UNS
4120 : 3383             THEN PRINTB(FMT15,MLB11);
4121 : 3384             IF .IAE
4122 : 3385             THEN PRINTB(FMT15,MLB12);
4123 : 3386             IF .AOE
4124 : 3387             THEN PRINTB(FMT15,MLB13);
4125 : 3388             IF .RMR
4126 : 3389             THEN PRINTB(FMT15,MLB14);
4127 : 3390             IF .ILR
4128 : 3391             THEN PRINTB(FMT15,MLB15);
4129 : 3392             IF .ILF
4130 : 3393             THEN PRINTB(FMT15,MLB16);
4131 : 3394             END;
4132 : 3395     IF .ERR5
4133 : 3396     THEN
4134 : 3397         BEGIN
4135 : 3398             IF .OPI
4136 : 3399             THEN PRINTB(FMT15,MLB17);
4137 : 3400             IF .DPAR
4138 : 3401             THEN PRINTB(FMT15,MLB18);
4139 : 3402             IF .CPAR
4140 : 3403             THEN PRINTB(FMT15,MLB19);
4141 : 3404             END;
4142 : 3405     END;      !* 4 *
4143 : 3406     END;      !* 3 *
4144 : 3407
4145 : 3408     !*
```

```

4147 ;MLX4
4148 :
4149 :
4150 : 3409 ! THIS IS THE CODE FOR PURPOSE 3:
4151 : 3410 !-
4152 : 3411
4153 : 3412 IF .ERR6 !UNRECOVERABLE DATA ERROR (NO RETRY ALLOWED)
4154 : 3413 THEN RETURN 4;
4155 : 3414
4156 : 3415 IF .ERR7 !RECOVERABLE DATA ERROR (RETRY TO CLASSIFY)
4157 : 3416 THEN RETURN 5;
4158 : 3417
4159 : 3418 IF .ERR2 !CONTROLLER FATAL ERROR
4160 : 3419 THEN RETURN 2;
4161 : 3420
4162 : 3421 IF .ERR4 !DRIVE FATAL ERROR
4163 : 3422 THEN RETURN 3;
4164 : 3423
4165 : 3424 IF (.ERR3 OR .ERR5) .NON-FATAL ERRORS
4166 : 3425 THEN RETURN 1
4167 : 3426 ELSE
4168 : 3427 BEGIN
4169 : 3428 IF ((.ERR0UT) AND (NOT .RETRYING))
4170 : 3429 THEN PRINTB(SAY1,PHR11);
4171 : 3430 !'SC SET BUT NO SYSTEM ERRORS FOUND'
4172 : 3431 RETURN 1;
4173 : 3432 END;
4174 : 3433
4175 : 3434 END; !* 1 *
4179 :

```

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (25)

```

4180 :
4184 037000 004137 005222 SYSERR: .SBTTL SYSERR SYSTEM ERROR DETECTOR
4185 037004 005005 JSR R1,$SAVE5 ;
4186 037006 005004 CLR R5 ; ERR2
4187 037010 005001 CLR R4 ; ERR3
4188 037012 005002 CLR R1 ; ERR4
4189 037014 005046 CLR R2 ; ERR5
4190 037016 005003 CLR -(SP) ; ERR6
4191 037020 005777 173340 CLR R3 ; ERR7
4192 037024 100402 TST @ML.REG ;
4193 037026 000137 041220 BMI 1$ ;
4194 037032 032777 010000 173334 1$: JMP 49$ ;
4195 037040 001010 BNE 2$ ;
4196 037042 032777 004000 173324 BIT #10000,@ML.REG+10 ;
4197 037050 001004 BNE 2$ ;
4198 037052 032777 002000 173314 BIT #4000,@ML.REG+10 ;
4199 037060 001402 BEQ 3$ ;
4200 037062 012705 000001 2$: MOV #1,R5 ; *,ERR2
4201 037066 005777 173302 3$: TST @ML.REG+10 ;

```

3203  
3249  
3250  
3251  
3252  
3253  
3254  
3260  
3263  
3264  
3265



4203					:MLX4					
4204					:			SYSTEM ERROR DETECTOR		
4205										
4206	037072	100420				BMI	4\$			
4207	037074	032777	020000	173272		BIT	#20000,@ML.REG+10			
4208	037102	001014				BNE	4\$			
4209	037104	032777	001000	173262		BIT	#1000,@ML.REG+10			
4210	037112	001010				BNE	4\$			
4211	037114	032777	000400	173252		BIT	#400,@ML.REG+10			
4212	037122	001004				BNE	4\$			
4213	037124	032777	020000	173232		BIT	#20000,@ML.REG			
4214	037132	001402				BEQ	5\$			
4215	037134	012704	000001		4\$:	MOV	#1,R4	: * ,ERR3		3266
4216	037140	032777	040000	173226	5\$:	BIT	#40000,@ML.REG+10	:		3267
4217	037146	001405				BEQ	6\$			
4218	037150	005737	002254			TST	ECCDIS			
4219	037154	001002				BNE	6\$			
4220	037156	012704	000001			MOV	#1,R4	: * ,ERR3		3268
4221	037162	032777	040000	173210	6\$:	BIT	#40000,@ML.REG+14	:		3269
4222	037170	001024				BNE	7\$			
4223	037172	032777	002000	173200		BIT	#2000,@ML.REG+14			
4224	037200	001020				BNE	7\$			
4225	037202	032777	001000	173170		BIT	#1000,@ML.REG+14			
4226	037210	001014				BNE	7\$			
4227	037212	132777	000004	173160		BITB	#4,@ML.REG+14			
4228	037220	001010				BNE	7\$			
4229	037222	132777	000002	173150		BITB	#2,@ML.REG+14			
4230	037230	001004				BNE	7\$			
4231	037232	132777	000001	173140		BITB	#1,@ML.REC+14			
4232	037240	001402				BEQ	8\$			
4233	037242	012701	000001		7\$:	MOV	#1,R1	: * ,ERR4		3270
4234	037246	032777	020000	173124	8\$:	BIT	#20000,@ML.REG+14	:		3271
4235	037254	001010				BNE	9\$			
4236	037256	132777	000040	173114		BITB	#40,@ML.REG+14			
4237	037264	001004				BNE	9\$			
4238	037266	132777	000010	173104		BITB	#10,@ML.REG+14			
4239	037274	001402				BEQ	10\$			
4240	037276	012702	000001		9\$:	MOV	#1,R2	: * ,ERR5		3272
4241	037302	005777	173072		10\$:	TST	@ML.REG+14	:		3273
4242	037306	100410				BMI	11\$			
4243	037310	032777	020000	173110		BIT	#20000,@ML.REG+42			
4244	037316	001004				BNE	11\$			
4245	037320	032777	040000	173100		BIT	#40000,@ML.REG+42			
4246	037326	001402				BEQ	12\$			
4247	037330	012703	000001		11\$:	MOV	#1,R3	: * ,ERR7		3274
4248	037334	032777	040000	173032	12\$:	BIT	#40000,@ML.REG+10	:		3275
4249	037342	001406				BEQ	13\$			
4250	037344	023727	002254	000001		CMP	ECCDIS,#1			
4251	037352	001002				BNE	13\$			
4252	037354	012703	000001			MOV	#1,R3	: * ,ERR7		3276
4253	037360	132777	000100	173012	13\$:	BITB	#100,@ML.REG+14	:		3277
4254	037366	001003				BNE	14\$			
4255	037370	005777	173032			TST	@ML.REG+42			
4256	037374	100002				BPL	15\$			
4257	037376	012716	000001		14\$:	MOV	#1,(SP)	: * ,ERR6		3273

Address	Label	Op1	Op2	Op3	Op4	Op5	Op6	Op7	Op8	Op9	Date/Time	Page
4259											23-Oct-1980 15:01:43	TOPS
4260											23-Oct-1980 14:57:05	PA:<
4261												
4262	037402	032737	000001	032356	15\$:	BIT	#1,RETRYING					3311
4263	037410	001402				BEQ	17\$					3312
4264	037412	000137	041062		16\$:	JMP	43\$					
4265	037416	032737	000001	002260	17\$:	BIT	#1,ERR0UT					3315
4266	037424	001552				BEQ	23\$					
4267	037426	016646	000020			MOV	20(SP),-(SP)					3318
4268	037432	004737	033436			JSR	PC,SAYWHO					
4269	037436	012716	007542			MOV	#PHR10,(SP)					3319
4270	037442	012746	006514			MOV	#SAY1,-(SP)					
4271	037446	012746	000002			MOV	#2,-(SP)					
4272	037452	010600				MOV	SP,RO					
4273	037454	104414				TRAP	14					
4274	037456	032703	000001			BIT	#1,R3					3321
4275	037462	001473				BEQ	21\$					
4276	037464	005777	172710			TST	@ML.REG+14					3324
4277	037470	100012				BPL	18\$					
4278	037472	012746	007172			MOV	#MLB20,-(SP)					3325
4279	037476	012746	006502			MOV	#FMT15,-(SP)					
4280	037502	012746	000002			MOV	#2,-(SP)					
4281	037506	010600				MOV	SP,RO					
4282	037510	104414				TRAP	14					
4283	037512	062706	000006			ADD	#6,SP					
4284	037516	032777	020000	172702	18\$:	BIT	#20000,@ML.REG+42					3326
4285	037524	001412				BEQ	19\$					
4286	037526	012746	007202			MOV	#MLB22,-(SP)					3327
4287	037532	012746	006502			MOV	#FMT15,-(SP)					
4288	037536	012746	000002			MOV	#2,-(SP)					
4289	037542	010600				MOV	SP,RO					
4290	037544	104414				TRAP	14					
4291	037546	062706	000006			ADD	#6,SP					
4292	037552	032777	040000	172646	19\$:	BIT	#40000,@ML.REG+42					3328
4293	037560	001412				BEQ	20\$					
4294	037562	012746	007206			MOV	#MLB23,-(SP)					3329
4295	037566	012746	006502			MOV	#FMT15,-(SP)					
4296	037572	012746	000002			MOV	#2,-(SP)					
4297	037576	010600				MOV	SP,RO					
4298	037600	104414				TRAP	14					
4299	037602	062706	000006			ADD	#6,SP					
4300	037606	032777	040000	172560	20\$:	BIT	#40000,@ML.REG+10					3330
4301	037614	001416				BEQ	21\$					
4302	037616	023727	002254	000001		CMP	ECCDIS,#1					
4303	037624	001012				BNE	21\$					
4304	037626	012746	007072			MOV	#MLB6,-(SP)					3331
4305	037632	012746	006502			MOV	#FMT15,-(SP)					
4306	037636	012746	000002			MOV	#2,-(SP)					
4307	037642	010600				MOV	SP,RO					
4308	037644	104414				TRAP	14					
4309	037646	062706	000006			ADD	#6,SP					
4310	037652	032766	000001	000006	21\$:	BIT	#1,6(SP)					3333
4311	037660	001464				BEQ	25\$					
4312	037662	132777	000100	172510		BITB	#100,@ML.REG+14					3336
4313	037670	001412				BEQ	22\$					

23-Oct-1980 15:01:43 TOPS  
 23-Oct-1980 14:57:05 PA:c

Address	OpCode	Operand 1	Operand 2	Operand 3	Instruction	Comments	Line No.
4315							
4316							
4317							
4318	037672	012746	007176		MOV #MLB21,-(SP)		3337
4319	037676	012746	006502		MOV #FMT15,-(SP)		
4320	037702	012746	000002		MOV #2,-(SP)		
4321	037706	010600			MOV SP,R0	: SP,*	
4322	037710	104414			TRAP 14		
4323	037712	062706	000006		ADD #6,SP		
4324	037716	005777	172504	22\$:	TST @ML.REG+42		3338
4325	037722	100043			BPL 25\$		
4326	037724	012746	007212		MOV #MLB24,-(SP)		3339
4327	037730	012746	006502		MOV #FMT15,-(SP)		
4328	037734	012746	000002		MOV #2,-(SP)		
4329	037740	010600			MOV SP,R0	: SP,*	
4330	037742	104414			TRAP 14		
4331	037744	062706	000006		ADD #6,SP		
4332	037750	000430			BR 25\$		3317
4333	037752	032705	000001	23\$:	BIT #1,R5	: *,ERR2	3344
4334	037756	001011			BNE 24\$		
4335	037760	032704	000001		BIT #1,R4	: *,ERR3	
4336	037764	001006			BNE 24\$		
4337	037766	032701	000001		BIT #1,R1	: *,ERR4	
4338	037772	001003			BNE 24\$		
4339	037774	032702	000001		BIT #1,R2	: *,ERR5	
4340	040000	001604			BEQ 16\$		
4341	040002	016646	000020	24\$:	MOV 20(SP),-(SP)	: LUN,*	3347
4342	040006	004737	033436		JSR PC,SAYWHO		
4343	040012	012716	007542		MOV #PHR10,(SP)		3348
4344	040016	012746	006514		MOV #SAY1,-(SP)		
4345	040022	012746	000002		MOV #2,-(SP)		
4346	040026	010600			MOV SP,R0	: SP,*	
4347	040030	104414			TRAP 14		
4348	040032	062706	000006	25\$:	ADD #6,SP		3346
4349	040036	032705	000001		BIT #1,R5	: *,ERR2	3353
4350	040042	001452			BEQ 28\$		
4351	040044	032777	010000	172322	BIT #10000,@ML.REG+10		3356
4352	040052	001412			BEQ 26\$		
4353	040054	012746	007052		MOV #MLB2,-(SP)		3357
4354	040060	012746	006502		MOV #FMT15,-(SP)		
4355	040064	012746	000002		MOV #2,-(SP)		
4356	040070	010600			MOV SP,R0	: SP,*	
4357	040072	104414			TRAP 14		
4358	040074	062706	000006		ADD #6,SP		
4359	040100	032777	004000	172266	26\$:	BIT #4000,@ML.REG+10	3358
4360	040106	001412			BEQ 27\$		
4361	040110	012746	007056		MOV #MLB3,-(SP)		3359
4362	040114	012746	006502		MOV #FMT15,-(SP)		
4363	040120	012746	000002		MOV #2,-(SP)		
4364	040124	010600			MOV SP,R0	: SP,*	
4365	040126	104414			TRAP 14		
4366	040130	062706	000006		ADD #6,SP		
4367	040134	032777	002000	172232	27\$:	BIT #2000,@ML.REG+10	3360
4368	040142	001412			BEQ 28\$		
4369	040144	012746	007062		MOV #MLB4,-(SP)		3361

23-Oct-1980 15:01:43 TOPS  
 23-Oct-1980 14:57:05 PA:<

Address	Hex	Hex	Hex	Label	Instruction	Comment	Address
4371							
4372				;MLX4			
4373				:			
4374	040150	012746	006502		MOV #FMT15,-(SP)		
4375	040154	012746	000002		MOV #2,-(SP)		
4376	040160	010600			MOV SP,R0	; SP,*	
4377	040162	104414			TRAP 14		
4378	040164	062706	000006		ADD #6,SP		
4379	040170	032704	000001	28\$:	BIT #1,R4	; *,ERR3	3363
4380	040174	001526			BEQ 34\$		
4381	040176	005777	172172		TST @ML.REG+10	:	3366
4382	040202	100012			BPL 29\$		
4383	040204	012746	007066		MOV #MLB5,-(SP)	:	3367
4384	040210	012746	006502		MOV #FMT15,-(SP)		
4385	040214	012746	000002		MOV #2,-(SP)		
4386	040220	010600			MOV SP,R0	; SP,*	
4387	040222	104414			TRAP 14		
4388	040224	062706	000006		ADD #6,SP		
4389	040230	032777	040000	172136 29\$:	BIT #40000,@ML.REG+10	:	3368
4390	040236	001415			BEQ 30\$		
4391	040240	005737	002254		TST ECCDIS		
4392	040244	001012			BNE 30\$		
4393	040246	012746	007072		MOV #MLB6,-(SP)	:	3369
4394	040252	012746	006502		MOV #FMT15,-(SP)		
4395	040256	012746	000002		MOV #2,-(SP)		
4396	040262	010600			MOV SP,R0	; SP,*	
4397	040264	104414			TRAP 14		
4398	040266	062706	000006		ADD #6,SP		
4399	040272	032777	020000	172074 30\$:	BIT #20000,@ML.REG+10	:	3370
4400	040300	001412			BEQ 31\$		
4401	040302	012746	007076		MOV #MLB7,-(SP)	:	3371
4402	040306	012746	006502		MOV #FMT15,-(SP)		
4403	040312	012746	000002		MOV #2,-(SP)		
4404	040316	010600			MOV SP,R0	; SP,*	
4405	040320	104414			TRAP 14		
4406	040322	062706	000006		ADD #6,SP		
4407	040326	032777	001000	172040 31\$:	BIT #1000,@ML.REG+10	:	3372
4408	040334	001412			BEQ 32\$		
4409	040336	012746	007102		MOV #MLB8,-(SP)	:	3373
4410	040342	012746	006502		MOV #FMT15,-(SP)		
4411	040346	012746	000002		MOV #2,-(SP)		
4412	040352	010600			MOV SP,R0	; SP,*	
4413	040354	104414			TRAP 14		
4414	040356	062706	000006		ADD #6,SP		
4415	040362	032777	000400	172004 32\$:	BIT #400,@ML.REG+10	:	3374
4416	040370	001412			BEQ 33\$		
4417	040372	012746	007106		MOV #MLB9,-(SP)	:	3375
4418	040376	012746	006502		MOV #FMT15,-(SP)		
4419	040402	012746	000002		MOV #2,-(SP)		
4420	040406	010600			MOV SP,R0	; SP,*	
4421	040410	104414			TRAP 14		
4422	040412	062706	000006		ADD #6,SP		
4423	040416	032777	020000	171740 33\$:	BIT #20000,@ML.REG	:	3376
4424	040424	001412			BEQ 34\$		
4425	040426	012746	007114		MOV #MLB10,-(SP)	:	3377

Address	OpCode	Operand1	Operand2	Operand3	Operand4	Label	Instruction	Comments	Line
4427									
4428									
4429									
4430	040432	012746	006502				MOV #FMT15,-(SP)		
4431	040436	012746	000002				MOV #2,-(SP)		
4432	040442	010600					MOV SP,R0	; SP,*	
4433	040444	104414					TRAP 14		
4434	040446	062706	000006				ADD #6,SP		
4435	040452	032701	000001	34\$:			BIT #1,R1	; *,ERR4	3379
4436	040456	001524					BEQ 40\$		
4437	040460	032777	040000	171712			BIT #40000,@ML.REG+14		3382
4438	040466	001412					BEQ 35\$		
4439	040470	012746	007122				MOV #MLB11,-(SP)		3383
4440	040474	012746	006502				MOV #FMT15,-(SP)		
4441	040500	012746	000002				MOV #2,-(SP)		
4442	040504	010600					MOV SP,R0	; SP,*	
4443	040506	104414					TRAP 14		
4444	040510	062706	000006				ADD #6,SP		
4445	040514	032777	002000	171656	35\$:		BIT #2000,@ML.REG+14		3384
4446	040522	001412					BEQ 36\$		
4447	040524	012746	007126				MOV #MLB12,-(SP)		3385
4448	040530	012746	006502				MOV #FMT15,-(SP)		
4449	040534	012746	000002				MOV #2,-(SP)		
4450	040540	010600					MOV SP,R0	; SP,*	
4451	040542	104414					TRAP 14		
4452	040544	062706	000006				ADD #6,SP		
4453	040550	032777	001000	171622	36\$:		BIT #1000,@ML.REG+14		3386
4454	040556	001412					BEQ 37\$		
4455	040560	012746	007132				MOV #MLB13,-(SP)		3387
4456	040564	012746	006502				MOV #FMT15,-(SP)		
4457	040570	012746	000002				MOV #2,-(SP)		
4458	040574	010600					MOV SP,R0	; SP,*	
4459	040576	104414					TRAP 14		
4460	040600	062706	000006				ADD #6,SP		
4461	040604	132777	000004	171566	37\$:		BITB #4,@ML.REG+14		3388
4462	040612	001412					BEQ 38\$		
4463	040614	012746	007136				MOV #MLB14,-(SP)		3389
4464	040620	012746	006502				MOV #FMT15,-(SP)		
4465	040624	012746	000002				MOV #2,-(SP)		
4466	040630	010600					MOV SP,R0	; SP,*	
4467	040632	104414					TRAP 14		
4468	040634	062706	000006				ADD #6,SP		
4469	040640	132777	000002	171532	38\$:		BITB #2,@ML.REG+14		3390
4470	040646	001412					BEQ 39\$		
4471	040650	012746	007142				MOV #MLB15,-(SP)		3391
4472	040654	012746	006502				MOV #FMT15,-(SP)		
4473	040660	012746	000002				MOV #2,-(SP)		
4474	040664	010600					MOV SP,R0	; SP,*	
4475	040666	104414					TRAP 14		
4476	040670	062706	000006				ADD #6,SP		
4477	040674	132777	000001	171476	39\$:		BITB #1,@ML.REG+14		3392
4478	040702	001412					BEQ 40\$		
4479	040704	012746	007146				MOV #MLB16,-(SP)		3393
4480	040710	012746	006502				MOV #FMT15,-(SP)		
4481	040714	012746	000002				MOV #2,-(SP)		

23-Oct-1980 15:01:43 TOPS  
 23-Oct-1980 14:57:05 PA:<

23-Oct-1980 15:01:43 TOPS  
 23-Oct-1980 14:57:05 PA:<

```

4483 ;MLX4
4484 ;
4485
4486 040720 010600 MOV SP,R0 ; SP,*
4487 040722 104414 TRAP 14
4488 040724 062706 000006 ADD #6,SP
4489 040730 032702 000001 40$: BIT #1,R2 ; *,ERR5 3395
4490 040734 001452 BEQ 43$
4491 040736 032777 020000 171434 BIT #20000,@ML.REG+14 ; 3398
4492 040744 001412 BEQ 41$ ; 3399
4493 040746 012746 007152 MOV #MLB17,-(SP) ;
4494 040752 012746 006502 MOV #FMT15,-(SP) ;
4495 040756 012746 000002 MOV #2,-(SP) ;
4496 040762 010600 MOV SP,R0 ; SP,*
4497 040764 104414 TRAP 14
4498 040766 062706 000006 ADD #6,SP
4499 040772 132777 000040 171400 41$: BITB #40,@ML.REG+14 ; 3400
4500 041000 001412 BEQ 42$ ; 3401
4501 041002 012746 007156 MOV #MLB18,-(SP) ;
4502 041006 012746 006502 MOV #FMT15,-(SP) ;
4503 041012 012746 000002 MOV #2,-(SP) ;
4504 041016 010600 MOV SP,R0 ; SP,*
4505 041020 104414 TRAP 14
4506 041022 062706 000006 ADD #6,SP
4507 041026 132777 000010 171344 42$: BITB #10,@ML.REG+14 ; 3402
4508 041034 001412 BEQ 43$ ; 3403
4509 041036 012746 007164 MOV #MLB19,-(SP) ;
4510 041042 012746 006502 MOV #FMT15,-(SP) ;
4511 041046 012746 000002 MOV #2,-(SP) ;
4512 041052 010600 MOV SP,R0 ; SP,*
4513 041054 104414 TRAP 14
4514 041056 062706 000006 ADD #6,SP
4515 041062 032716 000001 43$: BIT #1,(SP) ; *,ERR6 3412
4516 041066 001403 BEQ 44$ ; 3413
4517 041070 012700 000004 MOV #4,R0 ; 3415
4518 041074 000452 BR 50$ ;
4519 041076 006003 44$: ROR R3 ; ERR7 3416
4520 041100 103003 BCC 45$ ; 3418
4521 041102 012700 000005 MOV #5,R0 ; 3419
4522 041106 000445 BR 50$ ;
4523 041110 006005 45$: ROR R5 ; ERR2 3421
4524 041112 103003 BCC 46$ ; 3422
4525 041114 012700 000002 MOV #2,R0 ; 3424
4526 041120 000440 BR 50$ ;
4527 041122 006001 46$: ROR R1 ; ERR4 3428
4528 041124 103003 BCC 47$ ;
4529 041126 012700 000003 MOV #3,R0 ;
4530 041132 000433 BR 50$ ;
4531 041134 006004 47$: ROR R4 ; ERR3 3428
4532 041136 103424 BLO 48$ ;
4533 041140 006002 ROR R2 ; ERR5
4534 041142 103422 BLO 48$ ;
4535 041144 032737 000001 002260 BIT #1,ERROUT ;
4536 041152 001416 BEQ 48$ ;
4537 041154 032737 000001 032356 BIT #1,RETRYING

```

```
4539      ;MLX4
4540      ;
4541
4542 041162 001012      BNE      48$
4543 041164 012746 007562  MOV     #PHR11,-(SP)      ;
4544 041170 012746 006514  MOV     #SAY1,-(SP)      ;
4545 041174 012746 000002  MOV     #2,-(SP)        ;
4546 041200 010600      MOV     SP,R0           ; SP,+
4547 041202 104414      TRAP    14
4548 041204 062706 000006  ADD     #6,SP
4549 041210 012704 000001 48$:   MOV     #1,R4           ;
4550 041214 010400      MOV     R4,R0           ;
4551 041216 000401      BR      50$
4552 041220 005000 49$:   CLR     R0           ;
4553 041222 005726 50$:   TST     (SP)+
4554 041224 000207      RTS     PC
4555
4556      ; Routine Size: 587 words
4557      ; Maximum stack depth per invocation: 13 words
4562
4563
```

23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<

3429

3424

3204

3203

```

4565 :MLX4
4566 :
4567 :
4568 : 3435 %SBTTL 'DATA COMPARISON ROUTINE'
4569 : 3436
4570 : 3437 ROUTINE DOUBLE_CHECK(W_POINTER,R_POINTER,COUNT) =
4571 : 3438 BEGIN * 1 *
4572 : 3439
4573 : 3440 !*+
4574 : 3441 ROUTINE: DOUBLE_CHECK(W_POINTER,R_POINTER,COUNT)
4575 : 3442
4576 : 3443 PURPOSE: TO DOUBLE CHECK THE ECC DETECTION LOGIC AFTER A
4577 : 3444 SUCCESSFUL READ COMMAND. THE WRITE AND READ BUFFERS
4578 : 3445 ARE COMPARED, AND NO ERRORS SHOULD BE FOUND UNDER
4579 : 3446 NORMAL CIRCUMSTANCES.
4580 : 3447
4581 : 3448 ARGUMENTS: W_POINTER = POINTER TO THE WRITE BUFFER
4582 : 3449 R_POINTER = POINTER TO THE READ BUFFER
4583 : 3450 COUNT = THE NUMBER OF WORDS TO COMPARE
4584 : 3451
4585 : 3452 RESULTS: VALUE RETURNED IS EITHER:
4586 : 3453 0 = NO ERRORS WERE FOUND
4587 : 3454 N > 0 = ADDRESS IN WRITE BUFFER WHERE ERROR WAS FOUND
4588 : 3455 !--
4589 : 3456
4590 : 3457 LOCAL
4591 : 3458 VALUE, OFFSET, GOOD, BAD;
4592 : 3459
4593 : 3460 LABEL
4594 : 3461 LOOP;
4595 : 3462
4596 : 3463 OFFSET = 0;
4597 : 3464 VALUE = 0;
4598 : 3465
4599 : 3466 LOOP:
4600 : 3467 BEGIN !* 2 *
4601 : 3468 INCR I FROM 1 TO .COUNT DO
4602 : 3469 BEGIN !* 3 *
4603 : 3470 GOOD = (.W_POINTER + .OFFSET);
4604 : 3471 BAD = (.R_POINTER + .OFFSET);
4605 : 3472 IF .GOOD EQ .BAD
4606 : 3473 THEN OFFSET = .OFFSET + 2
4607 : 3474 ELSE
4608 : 3475 BEGIN !* 4 *
4609 : 3476 VALUE = (.W_POINTER + .OFFSET); !ADDRESS OF GOOD
4610 : 3477 LEAVE LOOP;
4611 : 3478 END; !* 4 *
4612 : 3479 END; !* 3 *
4613 : 3480 END; !* 2 *
4614 : 3481
4615 : 3482 RETURN .VALUE; !EITHER 0 OR THE ADDRESS OF GOOD DATA
4616 : 3483 END; !* 1 *

```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (26)



```

4621 ;MLX4
4622 ;
4623 ;
4624 ;
4625 ;
4629 041226 .SBTTL DOUBLE.CHECK DATA COMPARISON ROUTINE
4630 041226 004137 005222 DOUBLE.CHECK:
4631 041232 005746 JSR R1,$SAVE5 ;
4632 041234 005003 TST -(SP) ;
4633 041236 005001 CLR R3 ; OFFSET
4634 041240 005002 CLR R1 ; VALUE
4635 041242 000417 CLR R2 ; I
4636 041244 010304 BR 3$ ;
4637 041246 066604 000024 1$: MOV R3,R4 ; OFFSET,*
4638 041252 011416 ADD 24(SP),R4 ; W.POINTER,*
4639 041254 010305 MOV (R4),(SP) ; *,GOOD
4640 041256 066605 000022 MOV R3,R5 ; OFFSET,*
4641 041262 011500 ADD 22(SP),R5 ; R.POINTER,*
4642 041264 021600 MOV (R5),R0 ; *,BAD
4643 041266 001003 CMP (SP),R0 ; *,BAD
4644 041270 062703 000002 ADD #2,R3 ; *,OFFSET
4645 041274 000402 BR 3$ ;
4646 041276 010401 2$: MOV R4,R1 ; *,VALUE
4647 041300 000404 BR 4$ ;
4648 041302 005202 3$: INC R2 ; I
4649 041304 020266 000020 CMP R2,20(SP) ; I,COUNT
4650 041310 003755 BLE 1$ ;
4651 041312 010100 4$: MOV R1,R0 ; VALUE,*
4652 041314 005726 TST (SP)+ ;
4653 041316 000207 RTS PC ;
4654 ;
4655 ; Routine Size: 29 words
4656 ; Maximum stack depth per invocation: 7 words.
4661 ;
4662 ;

```

23-Oct-1980 15:01:43 TOPS  
 23-Oct-1980 14:57:05 PA:<

3437  
 3463  
 3464  
 3468  
 3470  
 3471  
 3472  
 3473  
 3472  
 3476  
 3477-  
 3468  
 3438  
 3437

```

4664 :MLX4
4665 :
4666 :
4667 : 3484 %SBTTL 'COMMAND INITIATION AND TERMINATION'
4668 : 3485
4669 : 3486 ROUTINE WAITER: NOVALUE =
4670 : 3487 BEGIN
4671 : 3488
4672 : 3489 !+
4673 : 3490 ! THIS ROUTINE DOES NOTHING, BUT IT IS CALLED BY 'START IT' TO
4674 : 3491 ! WASTE TIME WHILE WAITING TO BEGIN OR END AN ML11 TRANSFER.
4675 : 3492 !-
4676 : 3493
4677 : 3494 BREAK;
4678 : 3495 RETURN;
4679 : 3496 END;
4683 :
4684 :
4688 041320 104422
4689 041322 000207
4690 :
4691 :
4692 :
4697 :
4698 :

```

23-Oct-1980 15:01:43 (OPS-20 Bliss-16 V2(206)  
23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (27)

```

.SBTTL WAITER COMMAND INITIATION AND TERMINATION
WAITER: TRAP 22
RTS PC

```

; Routine Size: 2 words  
; Maximum stack depth per invocation: 0 words

3487  
3486

```

4700 :MLX4
4701 :
4702 :
4703 : 3497 ROUTINE START_IT(COMMAND,LUN,WRDCNT,BUFFER,SECTOR) =
4704 : 3498 BEGIN
4705 : 3499
4706 : 3500 |**
4707 : 3501 | ROUTINE: START_IT(COMMAND,LUN,WRDCNT,BUFFER,SECTOR)
4708 : 3502 |
4709 : 3503 | PURPOSE: TO INITIATE A TRANSFER TO OR FROM THE ML11,
4710 : 3504 | TO WAIT FOR THE TRANSFER TO COMPLETE, AND TO CALL
4711 : 3505 | THE 'SYSERR' ROUTINE TO LOOK FOR RESULTING ERRORS.
4712 : 3506 |
4713 : 3507 | ARGUMENTS: (1) COMMAND - THE FUNCTION CODE FOR THE TYPE OF TRANSFER
4714 : 3508 | DESIRED. THIS CODE WILL BE SENT TO THE MLCS1
4715 : 3509 | REGISTER TO START THE OPERATION, SINCE IT
4716 : 3510 | ALSO CONTAINS THE GO BIT.
4717 : 3511 |
4718 : 3512 | (2) LUN - LOGICAL UNIT NUMBER
4719 : 3513 |
4720 : 3514 | (3) WRDCNT - NUMBER OF 16-BIT WORDS TO TRANSFER
4721 : 3515 |
4722 : 3516 | (4) BUFFER - THE ADDRESS IN MAIN MEMORY OF THE SELECTED
4723 : 3517 | WRITE OR READ BUFFER
4724 : 3518 |
4725 : 3519 | (5) SECTOR THE TRANSFER'S STARTING ADDRESS IN THE ML11
4726 : 3520 |
4727 : 3521 | RESULTS: VALUES RETURNED ARE IDENTICAL TO THOSE DEFINED ABOVE IN
4728 : 3522 | THE 'SYSERR' ROUTINE.
4729 : 3523 | --
4730 : 3524
4731 : 3525 LOCAL
4732 : 3526 TEMP, READY_BIT, RTN;
4733 : 3527
4734 : 3528 LABEL
4735 : 3529 LOOP;
4736 : 3530
4737 : 3531
4738 : 3532 ; WAIT FOR DRIVE READY:
4739 : 3533 ;
4740 : 3534
4741 : 3535 UNIT = .DRIVE; !SELECT THE UNIT
4742 : 3536 READY_BIT = 0;
4743 : 3537 UNTIL READY_BIT NEQ 0 DO !WAIT FOR DRIVE READY
4744 : 3538 BEGIN
4745 : 3539 WAITER();
4746 : 3540 READY_BIT = .MLCS1 AND BIT7;
4747 : 3541 END;
4748 : 3542
4749 : 3543
4750 : 3544 ; INITIALIZE BEFORE EACH TRANSFER:
4751 : 3545 ;
4752 : 3546
4753 : 3547 CLR = 1; !CONTROLLER CLEAR
4754 : 3548 MLCS1 = DRV_CLR; !DRIVE CLEAR

```

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (28)

```

4756 :MLX4
4757 :
4758 :
4759 : 3549 UNIT = .DRIVE;          !PUT BACK THE DRIVE NUMBER
4760 : 3550
4761 : 3551
4762 : 3552          ! SET UP THE REQUIRED ENABLE/DISABLE BITS:
4763 : 3553
4764 : 3554
4765 : 3555 DCK_EN = 1;          !ALLOW REPORTING OF DATA CHECK ERRORS
4766 : 3556 IF .REFRESH
4767 : 3557 THEN REF_MAR = 1;      !TURN ON REFRESH MARGINING IF OPERATOR SELECTED IT
4768 : 3558 IF .ECCDIS
4769 : 3559 THEN ECC_DIS = 1;    !TURN OFF ECC IF OPERATOR SELECTED IT
4770 : 3560
4771 : 3561
4772 : 3562          ! SEND REQUIRED INFORMATION TO DEVICE REGISTERS:
4773 : 3563
4774 : 3564
4775 : 3565 MLWC = -(.WRDCNT);      !WORD COUNT REGISTER
4776 : 3566 MLBA = .BUFFER;        !BUS ADDRESS REGISTER (ADDRESS IN MAIN MEMORY)
4777 : 3567 MLDA = .SECTOR;        !DESIRED SECTOR ADDRESS REGISTER (ADDRESS IN ML11)
4778 : 3568
4779 : 3569
4780 : 3570          ! GO:
4781 : 3571
4782 : 3572
4783 : 3573 I_AM_DONE = INACTIVE;
4784 : 3574 TEMP = .COMMAND + BIT6;    !SET THE INTERRUPT ENABLE BIT WHILE LOADING THE
4785 : 3575 MLCS1 = .TEMP;            !CONTROL AND STATUS REGISTER WITH THE COMMAND
4786 : 3576
4787 : 3577
4788 : 3578          ! WAIT FOR DRIVE TO FINISH:
4789 : 3579
4790 : 3580
4791 : 3581 LOOP:
4792 : 3582 BEGIN
4793 : 3583   READY_BIT = 0;
4794 : 3584   UNTIL .I_AM_DONE DO
4795 : 3585     BEGIN
4796 : 3586       WAITER();
4797 : 3587       READY_BIT = .MLCS1 AND BIT7;
4798 : 3588       IF ((.READY_BIT NEQ 0) AND (.I_AM_DONE EQL INACTIVE))
4799 : 3589         THEN
4800 : 3590           BEGIN
4801 : 3591             IF (NOT .RETRYING)
4802 : 3592               THEN
4803 : 3593                 BEGIN
4804 : 3594                   SAYWHO(.LUN);
4805 : 3595                   IF .COMMAND EQL WR_CMD
4806 : 3596                     THEN RTN = WRD16;
4807 : 3597                   IF .COMMAND EQL RD_CMD
4808 : 3598                     THEN RTN = WRD17;
4809 : 3599                   IF .COMMAND EQL WC_CMD
4810 : 3600                     THEN RTN = PHR1;

```

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (28)

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 v2(206)  
23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (28)

```

4812 ;MLX4
4813 :
4814 :
4815 : 3601 PRINTB(FMT6,,WRDCNT,,RTN,WRD15,,SECTOR);
4816 : 3602 !'BEGAN YYYY WORD (???) AT SECTOR ZZZZZZ'
4817 : 3603 !WHERE (???) IS EITHER 'WRITE', 'READ' OR 'WRITE CHECK'
4818 : 3604 PRINTB(SAY1,MSG0);
4819 : 3605 !INTERRUPT DID NOT OCCUR, BUT THE TRANSFER IS COMPLETE
4820 : 3606 END;
4821 : 3607 LEAVE LOOP;
4822 : 3608 END;
4823 : 3609 END.
4824 : 3610 END;
4825 : 3611
4826 : 3612 RETURN SYSERR(.LUN); !PASS ALONG THE ERROR VALUES THAT 'SYSERR' OBTAINED.
4827 : 3613
4828 : 3614 END;

```

```

4832
4833 .SBTTL START.IT COMMAND INITIATION AND TERMINATION
4837 041324 START.IT:
4838 041324 004137 005202 JSR R1,$SAVE4 ;
4839 041330 016604 000022 MOV 22(SP),R4 ; LUN,* 3497
4840 041334 010402 MOV R4,R2 ; 3535
4841 041336 006302 ASL R2
4842 041340 016201 032440 MOV PTABLE.ADDR(R2),R1
4843 041344 016103 000006 MOV 6(R1),R3
4844 041350 042703 177770 BIC #177770,R3
4845 041354 142777 000007 171012 BICB #7,@ML.REG+10
4846 041362 150377 171006 BISB R3,@ML.REG+10
4847 041366 005003 CLR R3 ; READY.BIT 3536
4848 041370 020707 CMP PC,PC ; 3537
4849 041372 001007 1$: BNE 2$
4850 041374 004737 041320 JSR PC,WAITER ; 3539
4851 041400 017703 170760 MOV @ML.REG,R3 ; *,READY.BIT 3540
4852 041404 042703 177577 BIC #177577,R3 ; *,READY.BIT
4853 041410 000770 BR 1$ ; 3537
4854 041412 152777 000040 170754 2$: BISB #40,@ML.REG+10 ; 3547
4855 041420 012777 000011 170736 MOV #11,@ML.REG ; 3548
4856 041426 016201 032440 MOV PTABLE.ADDR(R2),R1 ; 3549
4857 041432 016102 000006 MOV 6(R1),R2
4858 041436 042702 177770 BIC #177770,R2
4859 041442 142777 000007 170724 BICB #7,@ML.REG+10
4860 041450 150277 170720 BISB R2,@ML.REG+10
4861 041454 152777 000004 170726 BISB #4,@ML.REG+24 ; 3555
4862 041462 032737 000001 002252 BIT #1,REFRESH ; 3556
4863 041470 001403 BEQ 3$
4864 041472 152777 000200 170710 BISB #200,@ML.REG+24 ; 3557
4865 041500 032737 000001 002254 3$: BIT #1,ECCDIS ; 3558
4866 041506 001403 BEQ 4$

```

23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<

Address	Hex	Hex	Hex	Hex	Label	Code	Comment	Address
4868					:MLX4			
4869					:		COMMAND INITIATION AND TERMINATION	
4870					:			
4871	041510	152777	000002	170672		BISB	#2,@ML.REG+24	3559
4872	041516	016677	000020	170642	4\$:	MOV	20(SP),@ML.REG+2	3565
4873	041524	005477	170636			NEG	@ML.REG+2	
4874	041530	016677	000016	170632		MOV	16(SP),@ML.REG+4	3566
4875	041536	016677	000014	170626		MOV	14(SP),@ML.REG+6	3567
4876	041544	005037	032354			CLR	I.AM.DONE	3573
4877	041550	016601	000024			MOV	24(SP),R1	3574
4878	041554	010102				MOV	R1,R2	
4879	041556	062702	000100			ADD	#100,R2	
4880	041562	010277	170576			MOV	R2,@ML.REG	3575
4881	041566	005003				CLR	R3	3583
4882	041570	032737	000001	032354	5\$:	BIT	#1,I.AM.DONE	3584
4883	041576	001067				BNE	9\$	
4884	041600	004737	041320			JSR	PC,WAITER	3586
4885	041604	017703	170554			MOV	@ML.REG,R3	3587
4886	041610	042703	177577			BIC	#177577,R3	
4887	041614	001765				BEQ	5\$	3588
4888	041616	005737	032354			TST	I.AM.DONE	
4889	041622	001362				BNE	5\$	
4890	041624	032737	000001	032356		BIT	#1,RETRYING	3591
4891	041632	001051				BNE	9\$	
4892	041634	010446				MOV	R4,-(SP)	3594
4893	041636	004737	033436			JSR	PC,SAYWHO	
4894	041642	020127	000061			CMP	R1,#61	3595
4895	041646	001002				BNE	6\$	
4896	041650	012702	006700			MOV	#WORD16,R2	3596
4897	041654	020127	000071		6\$:	CMP	R1,#71	3597
4898	041660	001002				BNE	7\$	
4899	041662	012702	006706			MOV	#WORD17,R2	3598
4900	041666	020127	000051		7\$:	CMP	R1,#51	3599
4901	041672	001002				BNE	8\$	
4902	041674	012702	007336			MOV	#PHR1,R2	3600
4903	041700	016616	000016		8\$:	MOV	16(SP),(SP)	3601
4904	041704	012746	006670			MOV	#WORD15,-(SP)	
4905	041710	010246				MOV	R2,-(SP)	
4906	041712	016646	000026			MOV	26(SP),-(SP)	
4907	041716	012746	006020			MOV	#FMT6,-(SP)	
4908	041722	012746	000005			MOV	#5,-(SP)	
4909	041726	010600				MOV	SP,R0	
4910	041730	104414				TRAP	14	
4911	041732	012716	010424			MOV	#MSG0,(SP)	3604
4912	041736	012746	006514			MOV	#SAY1,-(SP)	
4913	041742	012746	000002			MOV	#2,-(SP)	
4914	041746	010600				MOV	SP,R0	
4915	041750	104414				TRAP	14	
4916	041752	062706	000020			ADD	#20,SP	3593
4917	041756	010446			9\$:	MOV	R4,-(SP)	3612
4918	041760	004737	037000			JSR	PC,SYSERR	
4919	041764	005726				TST	(SP)+	3497
4920	041766	000207				RTS	PC	
4921								
4922								

: Routine Size: 146 words

4924  
4925  
4926  
4927  
4932  
4933

:MLX4  
:  
COMMAND INITIATION AND TERMINATION  
; Maximum stack depth per invocation: 13 words

23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<

```

4935 ;MLX4
4936 :
4937 :
4938 : 3615 %SBTTL 'COUNTING BYTES TRANSFERED'
4939 : 3616
4940 : 3617 ROUTINE UP_WR_COUNT(WORD_COUNT) : NOVALUE =
4941 : 3618 BEGIN
4942 : 3619
4943 : 3620 WR_COUNT = .WR_COUNT + (.WORD_COUNT * 2);
4944 : 3621
4945 : 3622 IF .WR_COUNT GEQ 20000
4946 : 3623 THEN
4947 : 3624 BEGIN
4948 : 3625 WR_THOUSANDS = .WR_THOUSANDS + 20;
4949 : 3626 WR_COUNT = .WR_COUNT - 20000;
4950 : 3627 IF .WR_THOUSANDS GEQ 1000
4951 : 3628 THEN
4952 : 3629 BEGIN
4953 : 3630 WR_THOUSANDS = .WR_THOUSANDS - 1000;
4954 : 3631 WR_MILLIONS = .WR_MILLIONS + 1;
4955 : 3632 IF .WR_MILLIONS LSS 0
4956 : 3633 THEN WR_MILLIONS = 0;
4957 : 3634 END;
4958 : 3635 END;
4959 : 3636
4960 : 3637 RETURN;
4961 : 3638 END;
  
```

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (29)

```

4966 .SBTTL UP.WR.COUNT COUNTING BYTES TRANSFERED
4970 041770 UP.WR.COUNT:
4971 041770 016600 000002 MOV 2(SP),RO ; WORD.COUNT,* 3620
4972 041774 006300 ASL RO
4973 041776 063700 032332 ADD WR.COUNT,RO
4974 042002 010037 032332 MOV RO,WR.COUNT
4975 042006 020027 047040 CMP RO,#47040 ; WR.COUNT,* 3622
4976 042012 002422 BLT 1$
4977 042014 062737 000024 032334 ADD #24,WR.THOUSANDS ; 3625
4978 042022 162737 047040 032332 SUB #47040,WR.COUNT ; 3626
4979 042030 023727 032334 001750 CMP WR.THOUSANDS,#1750 ; 3627
4980 042036 002410 BLT 1$
4981 042040 162737 001750 032334 SUB #1750,WR.THOUSANDS ; 3630
4982 042046 005237 032336 INC WR.MILLIONS ; 3631
4983 042052 100002 BPL 1$ ; 3632
4984 042054 005037 032336 CLR WR.MILLIONS ; 3633
4985 042060 000207 1$: RTS PC ; 3617
4986
4987 ; Routine Size: 29 words
4988 ; Maximum stack depth per invocation: 0 words
  
```



4997  
4998

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (30)

```

5000 :MLX4
5001 :
5002 :
5003 : 3639 ROUTINE UP_RD_COUNT(WORD_COUNT) : NOVALUE =
5004 : 3640 BEGIN
5005 : 3641
5006 : 3642 RD_COUNT = .RD_COUNT + (.WORD_COUNT * 2);
5007 : 3643
5008 : 3644 IF .RD_COUNT GEQ 20000
5009 : 3645 THEN
5010 : 3646 BEGIN
5011 : 3647 RD_THOUSANDS = .RD_THOUSANDS + 20;
5012 : 3648 RD_COUNT = .RD_COUNT - 20000;
5013 : 3649 IF .RD_THOUSANDS GEQ 1000
5014 : 3650 THEN
5015 : 3651 BEGIN
5016 : 3652 RD_THOUSANDS = .RD_THOUSANDS - 1000;
5017 : 3653 RD_MILLIONS = .RD_MILLIONS + 1;
5018 : 3654 IF .RD_MILLIONS LSS 0
5019 : 3655 THEN RD_MILLIONS = 0;
5020 : 3656 END;
5021 : 3657 END;
5022 : 3658
5023 : 3659 RETURN;
5024 : 3660 END;
  
```

Address	Label	Hex	Hex	Hex	Hex	Instruction	Comment	Address
5028								
5029						.SBTTL	UP.RD.COUNT COUNTING BYTES TRANSFERED	
5033	042062					UP.RD.COUNT:		
5034	042062	016600	000002			MOV	2(SP),RO ; WORD.COUNT,*	3642
5035	042066	006300				ASL	RO	
5036	042070	063700	032340			ADD	RD.COUNT,RO	
5037	042074	010037	032340			MOV	RO,RD.COUNT	
5038	042100	020027	047040			CMP	RO,#47040 ; RD.COUNT,*	3644
5039	042104	002422				BLT	1\$	
5040	042106	062737	000024	032342		ADD	#24,RD.THOUSANDS ;	3647
5041	042114	162737	047040	032340		SUB	#47040,RD.COUNT ;	3648
5042	042122	023727	032342	001750		CMP	RD.THOUSANDS,#1750 ;	3649
5043	042130	002410				BLT	1\$	
5044	042132	162737	001750	032342		SUB	#1750,RD.THOUSANDS ;	3652
5045	042140	005237	032344			INC	RD.MILLIONS ;	3653
5046	042144	100002				BPL	1\$ ;	3654
5047	042146	005037	032344			CLR	RD.MILLIONS ;	3655
5048	042152	000207				RTS	PC ;	3639
5049								
5050								
5051								

; Routine Size: 29 words  
 ; Maximum stack depth per invocation: 0 words

5060  
5061

```

5063 ;MLX4
5064 ; COUNTING BYTES TRANSFERED
5065 ;
5066 ; 3661 ROUTINE UP_WC_COUNT(WORD_COUNT) : NOVALUE =
5067 ; 3662 BEGIN
5068 ; 3663
5069 ; 3664 WC_COUNT = .WC_COUNT + (.WORD_COUNT * 2);
5070 ; 3665
5071 ; 3666 IF .WC_COUNT GEQ 20000
5072 ; 3667 THEN
5073 ; 3668 BEGIN
5074 ; 3669 WC_THOUSANDS = .WC_THOUSANDS + 20;
5075 ; 3670 WC_COUNT = .WC_COUNT - 20000;
5076 ; 3671 IF .WC_THOUSANDS GEQ 1000
5077 ; 3672 THEN
5078 ; 3673 BEGIN
5079 ; 3674 WC_THOUSANDS = .WC_THOUSANDS - 1000;
5080 ; 3675 WC_MILLIONS = .WC_MILLIONS + 1;
5081 ; 3676 IF .WC_MILLIONS LSS 0
5082 ; 3677 THEN WC_MILLIONS = 0;
5083 ; 3678 END;
5084 ; 3679 END;
5085 ; 3680
5086 ; 3681 RETURN;
5087 ; 3682 END;
  
```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (31)

		.SBTTL		UP.WC.COUNT COUNTING BYTES TRANSFERED		
		UP.WC.COUNT:				
5096	042154			MOV	2(SP),RO	; WORD.COUNT,* 3664
5097	042154	016600	000002	ASL	RO	
5098	042160	006300		ADD	WC.COUNT,RO	
5099	042162	063700	032346	MOV	RO,WC.COUNT	
5100	042166	010037	032346	CMP	RO,#47040	; WC.COUNT,* 3666
5101	042172	020027	047040	BLT	1\$	
5102	042176	002422		ADD	#24,WC.THOUSANDS	; 3669
5103	042200	062737	000024 032350	SUB	#47040,WC.COUNT	; 3670
5104	042206	162737	047040 032346	CMP	WC.THOUSANDS,#1750	; 3671
5105	042214	023727	032350 001750	BLT	1\$	
5106	042222	002410		SUB	#1750,WC.THOUSANDS	; 3674
5107	042224	162737	001750 032350	INC	WC.MILLIONS	; 3675
5108	042232	005237	032352	BPL	1\$	; 3676
5109	042236	100002		CLR	WC.MILLIONS	; 3677
5110	042240	005037	032352	RTS	PC	; 3661
5111	042244	000207	1\$:			
5112						
5113						
5114						

; Routine Size: 29 words  
 ; Maximum stack depth per invocation: 0 words

MLX4 MACRO M1113 24-OCT-80 09:59 PAGE 192  
UP.WC.COUNT COUNTING BYTES TRANSFERED

SEQ 0147

5123  
5124

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (32)

```

5126 ;MLX4
5127 :
5128 :
5129 : 3683 %SBTTL 'ML11 TRANSFER COMMANDS'
5130 : 3684
5131 : 3685 ROUTINE WRITE(LUN,WRDCNT,BUFFER,SECTOR) =
5132 : 3686 BEGIN
5133 : 3687
5134 : 3688 !++
5135 : 3689 ROUTINE: WPITE(LUN,WRDCNT,BUFFER,SECTOR)
5136 : 3690
5137 : 3691 PURPOSE: TO TRANSFER INFORMATION FROM MAIN MEMORY TO THE ML11
5138 : 3692 AND TO DECIDE ON THE ADVISABILITY OF A RETRY.
5139 : 3693
5140 : 3694 ARGUMENTS: SEE 'START_IT' ROUTINE ABOVE FOR DETAILS
5141 : 3695
5142 : 3696 RESULTS: UPON TERMINATION OF THE TRANSFER, ERROR DETECTION OCCURS
5143 : 3697 AND THE CONTENTS OF 'VALUE' BECOMES IMPORTANT IN TERMS OF
5144 : 3698 WHETHER THE TRANSFER WAS COMPLETELY SUCCESSFUL OR WHETHER A
5145 : 3699 RETRY SHOULD OR SHOULD NOT BE PERMITTED.
5146 : 3700
5147 : 3701 THE VALUES RETURNED FOR THIS SUBROUTINE ARE THE SAME AS
5148 : 3702 FOR THE SYSERR ROUTINE ABOVE.
5149 : 3703 !--
5150 : 3704
5151 : 3705 LOCAL
5152 : 3706 VALUE, COMMAND;
5153 : 3707
5154 : 3708 COMMAND = WR_CMD;
5155 : 3709 VALUE = START_IT(.COMMAND,.LUN,.WRDCNT,.BUFFER,.SECTOR);
5156 : 3710
5157 : 3711 !F (NOT .RETRYING)
5158 : 3712 THEN UP_WR_COUNT(.WRDCNT);
5159 : 3713
5160 : 3714 IF .VALUE EQL 0
5161 : 3715 THEN RETURN 0; !NO ERRORS AT ALL
5162 : 3716
5163 : 3717 IF ((.ERROUT) AND (NOT .RETRYING))
5164 : 3718 THEN PRINTB(FMT6,.WRDCNT,WRD16,WRD15,.SECTOR);
5165 : 3719 !'BEGAN YYYY WORD WRITE AT SECTOR ZZZZZZ'
5166 : 3720
5167 : 3721 RETURN .VALUE;
5168 : 3722
5169 : 3723 END;
5170 :
5171 :
5172 :
5173 :
5174 :

```

```

5178 042246 004137 005150 WRITE: .SBTTL WRITE ML11 TRANSFER COMMANDS ; 3685
5179 042252 012700 000061 JSR R1,$SAVE2 ; * ,COMMAND 3708
5180 042256 010046 MOV #61,R0 ; COMMAND,* 3709
MOV R0,-(SP) ;

```

23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<

```

5182 ;MLX4
5183 ;
5184 ML11 TRANSFER COMMANDS
5185 042260 016646 000020 MOV 20(SP),-(SP) ; LUN,*
5186 042264 016601 000020 MOV 20(SP),R1 ; WRDCNT,*
5187 042270 010146 MOV R1,-(SP)
5188 042272 016646 000020 MOV 20(SP),-(SP) ; BUFFER,*
5189 042276 016646 000020 MOV 20(SP),-(SP) ; SECTOR,*
5190 042302 004737 041324 JSR PC,START.IT
5191 042306 010002 MOV R0,R2 ; *,VALUE
5192 042310 032737 000001 C32356 BIT #1,RETRYING ;
5193 042316 001004 BNE 1$ ; 3711
5194 042320 010146 MOV R1,-(SP) ; 3712
5195 042322 004737 041770 JSR PC,UP.WR.COUNT
5196 042326 005726 TST (SP)+
5197 042330 005702 1$: TST R2 ; VALUE 3714
5198 042332 001003 BNE 2$ ;
5199 042334 062706 000012 ADD #12,SP ; 3715
5200 042340 000433 BR 4$ ;
5201 042342 032737 000001 002260 2$: BIT #1,ERROUT ; 3717
5202 042350 001423 BEQ 3$ ;
5203 042352 032737 000001 032356 BIT #1,RETRYING
5204 042360 001017 BNE 3$ ;
5205 042362 016646 000022 MOV 22(SP),-(SP) ; SECTOR,* 3718
5206 042366 012746 006670 MOV #WRD15,-(SP)
5207 042372 012746 006700 MOV #WRD16,-(SP)
5208 042376 010146 MOV R1,-(SP)
5209 042400 012746 006020 MOV #FMT6,-(SP)
5210 042404 012746 000005 MOV #5,-(SP)
5211 042410 010600 MOV SP,R0 ; SP,*
5212 042412 104414 TRAP 14
5213 042414 062706 000014 ADD #14,SP
5214 042420 062706 000012 3$: ADD #12,SP ; 3685
5215 042424 010200 MOV R2,R0 ; VALUE,* 3686
5216 042426 000207 RTS PC
5217 042430 005000 4$: CLR R0 ; 3685
5218 042432 000207 RTS PC
5219
5220 ; Routine Size: 59 words
5221 ; Maximum stack depth per invocation: 14 words
5226
5227

```

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (33)

```

5229 ;MLX4
5230 :
5231 :
5232 : 3724 ROUTINE READ(LUN,WRDCNT,BUFFER,SECTOR) =
5233 : 3725 BEGIN
5234 : 3726
5235 : 3727 !**
5236 : 3728 ! ROUTINE: READ(LUN,WRDCNT,BUFFER,SECTOR)
5237 : 3729
5238 : 3730 ! PURPOSE: TO TRANSFER INFORMATION FROM THE ML11 TO MAIN MEMORY
5239 : 3731 ! USING A 'READ' COMMAND.
5240 : 3732
5241 : 3733 ! ARGUMENTS: SEE 'START_IT' ROUTINE ABOVE FOR DETAILS
5242 : 3734
5243 : 3735 ! RESULTS: UPON TERMINATION OF THE TRANSFER, ERPOP DETECTION OCCURS
5244 : 3736 ! AND THE CONTENTS OF 'VALUE' BECOMES IMPORTANT IN TERMS OF
5245 : 3737 ! WHETHER THE TRANSFER WAS COMPLETELY SUCCESSFUL OR WHETHER A
5246 : 3738 ! RETRY SHOULD OR SHOULD NOT BE PERMITTED.
5247 : 3739
5248 : 3740 ! THE VALUES RETURNED FOR THIS SUBROUTINE ARE THE SAME AS
5249 : 3741 ! FOR THE SYSERR ROUTINE ABOVE.
5250 : 3742 !--
5251 : 3743
5252 : 3744 LOCAL
5253 : 3745 VALUE, COMMAND;
5254 : 3746
5255 : 3747 COMMAND = RD CMD;
5256 : 3748 VALUE = START_IT(.COMMAND,.LUN,.WRDCNT,.BUFFER,.SECTOR);
5257 : 3749
5258 : 3750 IF (NOT .RETRYING)
5259 : 3751 THEN UP_RD_COUNT(.WRDCNT);
5260 : 3752
5261 : 3753 IF .VALUE EQL 0
5262 : 3754 THEN RETURN 0; !NO ERRORS AT ALL
5263 : 3755
5264 : 3756 IF ((.ERROUT) AND (NOT .RETRYING))
5265 : 3757 THEN PRINTB(FMT6,.WRDCNT,WRD17,WRD15,.SECTOR);
5266 : 3758 !'BEGAN YYYY WORD READ AT SECTOR ZZZZZZ'
5267 : 3759
5268 : 3760 RETURN .VALUE;
5269 : 3761
5270 : 3762 END;

```

```

5275
5279 042434 004137 005150 READ: .SBTTL READ ML11 TRANSFER COMMANDS : 3724
5280 042440 012700 000071 JSR R1,$SAVE2 : * ,COMMAND 3747
5281 042444 010046 MOV #71,R0 : COMMAND, * 3748
5282 042446 016646 000020 MOV R0,-(SP) : LUN, *
5283 042452 016601 000020 MOV 20(SP),-(SP) : WRDCNT, *

```



23-Oct-1980 15:01:43 TOPS  
 23-Oct-1980 14:57:05 PA:<

```

5285 ;MLX4
5286 ;
5287
5288 042456 010146 MOV R1,-(SP)
5289 042460 016646 000020 MOV 20(SP),-(SP) ; BUFFER,*
5290 042464 016646 000020 MOV 20(SP),-(SP) ; SECTOR,*
5291 042470 004737 041324 JSR PC,START.IT
5292 042474 010002 MOV R0,R2 ; *,VALUE
5293 042476 032737 000001 032356 BIT #1,RETRYING ;
5294 042504 001004 BNE 1$ ;
5295 042506 010146 MOV R1,-(SP) ;
5296 042510 004737 042062 JSR PC,UP.RD.COUNT ;
5297 042514 005726 TST (SP)+ ;
5298 042516 005702 1$: TST R2 ; VALUE 3753
5299 042520 001003 BNE 2$ ;
5300 042522 062706 000012 ADD #12,SP ;
5301 042526 000433 BR 4$ ;
5302 042530 032737 000001 002260 2$: BIT #1,ERROUT ;
5303 042536 001423 BEQ 3$ ;
5304 042540 032737 000001 032356 BIT #1,RETRYING ;
5305 042546 001017 BNE 3$ ;
5306 042550 016646 000022 MOV 22(SP),-(SP) ; SECTOR,* 3757
5307 042554 012746 006670 MOV #WRD15,-(SP)
5308 042560 012746 006706 MOV #WRD17,-(SP)
5309 042564 010146 MOV R1,-(SP)
5310 042566 012746 006020 MOV #FMT6,-(SP)
5311 042572 012746 000005 MOV #5,-(SP)
5312 042576 010600 MOV SP,R0 ; SP,*
5313 042600 104414 TRAP 14
5314 042602 062706 000014 ADD #14,SP
5315 042606 062706 000012 3$: ADD #12,SP ;
5316 042612 010200 MOV R2,R0 ; VALUE,* 3724
5317 042614 000207 RTS PC ; 3725
5318 042616 005000 4$: CLR R0 ; 3724
5319 042620 000207 RTS PC
5320
5321 ; Routine Size: 59 words
5322 ; Maximum stack depth per invocation: 14 words
5327
5328

```

```

5330 ;MLX4
5331 ;
5332 ;
5333 ; 3763 ROUTINE CHECK(LUN,WRDCNT,BUFFER,SECTOR) =
5334 ; 3764 BEGIN
5335 ; 3765
5336 ; 3766 !++
5337 ; 3767 ROUTINE: CHECK(LUN,WRDCNT,BUFFER,SECTOR)
5338 ; 3768
5339 ; 3769 PURPOSE: TO TRANSFER INFORMATION FROM THE ML11 TO MAIN MEMORY
5340 ; 3770 USING A 'WRITE CHECK' COMMAND.
5341 ; 3771
5342 ; 3772 ARGUMENTS: SEE 'START_IT' ROUTINE ABOVE FOR DETAILS
5343 ; 3773
5344 ; 3774 RESULTS: UPON TERMINATION OF THE TRANSFER, ERROR DETECTION OCCURS
5345 ; 3775 AND THE CONTENTS OF 'VALUE' BECOMES IMPORTANT IN TERMS OF
5346 ; 3776 WHETHER THE TRANSFER WAS COMPLETELY SUCCESSFUL OR WHETHER A
5347 ; 3777 RETRY SHOULD OR SHOULD NOT BE PERMITTED.
5348 ; 3778
5349 ; 3779 THE VALUES RETURNED FOR THIS SUBROUTINE ARE THE SAME AS
5350 ; 3780 FOR THE SYSERR ROUTINE ABOVE.
5351 ; 3781 !--
5352 ; 3782
5353 ; 3783 LOCAL
5354 ; 3784 VALUE, COMMAND;
5355 ; 3785
5356 ; 3786 COMMAND = WC_CMD;
5357 ; 3787 VALUE = START_IT(.COMMAND,.LUN,.WRDCNT,.BUFFER,.SECTOR);
5358 ; 3788
5359 ; 3789 IF (NOT .RETRYING)
5360 ; 3790 THEN UP_WC_COUNT(.WRDCNT);
5361 ; 3791
5362 ; 3792 IF .VALUE EQL 0
5363 ; 3793 THEN RETURN 0; !NO ERRORS AT ALL
5364 ; 3794
5365 ; 3795 IF ((.ERROUT) AND (NOT .RETRYING))
5366 ; 3796 THEN PRINTB(FMT6,.WRDCNT,PHR1,WRD15,.SECTOR);
5367 ; 3797 !'BEGAN YYYY WORD WRITE CHECK AT SECTOR ZZZZZZ'
5368 ; 3798
5369 ; 3799 RETURN .VALUE;
5370 ; 3800
5371 ; 3801 END;
5375 ;
5376 ;
5380 042622 004137 005150 CHECK: .SBTTL CHECK ML11 TRANSFER COMMANDS
5381 042626 012700 000051 JSR R1,$SAVE2
5382 042632 010046 000020 MOV #51,R0
5383 042634 016646 000020 MOV R0,-(SP)
5384 042640 016601 000020 MOV 20(SP),-(SP)

```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (34)

3763  
 3786  
 3787

```

:
: *,COMMAND
: COMMAND,*
: LUN,*
: WRDCNT,*

```

5386									23-Oct-1980 15:01:43	TOPS
5387									23-Oct-1980 14:57:05	PA:<
5388										
5389	042644	010146								
5390	042646	016646	000020							
5391	042652	016646	000020							
5392	042656	004737	041324							
5393	042662	010002								
5394	042664	032737	000001	032356						3789
5395	042672	001004								
5396	042674	010146								3790
5397	042676	004737	042154							
5398	042702	005726								
5399	042704	005702			1\$:					3792
5400	042706	001003								
5401	042710	062706	000012							3793
5402	042714	000433								
5403	042716	032737	000001	002260	2\$:					3795
5404	042724	001423								
5405	042726	032737	000001	032356						
5406	042734	001017								
5407	042736	016646	000022							3796
5408	042742	012746	006670							
5409	042746	012746	007336							
5410	042752	010146								
5411	042754	012746	006020							
5412	042760	012746	000005							
5413	042764	010600								
5414	042766	104414								
5415	042770	062706	000014							
5416	042774	062706	000012		3\$:					3763
5417	043000	010200								3764
5418	043002	000207								
5419	043004	005000			4\$:					3763
5420	043006	000207								
5421										
5422										
5423										
5428										
5429										

; MLX4  
 ;  
 ML11 TRANSFER COMMANDS  
 MOV R1,-(SP)  
 MOV 20(SP),-(SP) ; BUFFER,\*  
 MOV 20(SP),-(SP) ; SECTOR,\*  
 JSR PC,START.IT  
 MOV R0,R2 ; \*,VALUE  
 BIT #1,RETRYING ;  
 BNE 1\$ ;  
 MCV R1,-(SP) ;  
 JSR PC,UP.WC.COUNT  
 TST (SP)+  
 TST R2 ; VALUE  
 BNE 2\$ ;  
 ADD #12,SP ;  
 BR 4\$ ;  
 BIT #1,ERROUT ;  
 BEQ 3\$ ;  
 BIT #1,RETRYING ;  
 BNE 3\$ ;  
 MOV 22(SP),-(SP) ; SECTOR,\*  
 MOV #WRD15,-(SP)  
 MOV #PHR1,-(SP)  
 MOV R1,-(SP)  
 MOV #FMT6,-(SP)  
 MOV #5,-(SP)  
 MOV SP,R0 ; SP,\*  
 TRAP 14  
 ADD #14,SP  
 ADD #12,SP ;  
 MOV R2,R0 ; VALUE,\*  
 RTS PC  
 CLR R0 ;  
 RTS PC

; Routine Size: 59 words  
 ; Maximum stack depth per invocation: 14 words

```

5431 :MLX4
5432 :
5433 :
5434 : 3802 ROUTINE CHOOSE =
5435 : 3803 BEGIN
5436 : 3804
5437 : 3805 !++
5438 : 3806 ! ROUTINE: CHOOSE
5439 : 3807 !
5440 : 3808 ! PURPOSE: TO DECIDE WHETHER TO DO A 'WRITE CHECK' OR A 'READ'.
5441 : 3809 ! THE 'READ' WILL BE SELECTED APPROXIMATELY 25% OF THE
5442 : 3810 ! TIME. THIS IS ACCOMPLISHED BY EXAMINING THE MOST AND
5443 : 3811 ! THE LEAST SIGNIFICANT BITS OF A RANDOM NUMBER. IF
5444 : 3812 ! BOTH ARE SET, THEN 'READ' IS CHOSEN.
5445 : 3813 !
5446 : 3814 ! RESULTS: THE VALUE RETURNED FOR THIS ROUTINE IS THE ADDRESS OF
5447 : 3815 ! THE CHOSEN COMMAND (EITHER 'READ' OR 'CHECK').
5448 : 3816 !--
5449 : 3817
5450 : 3818 LOCAL
5451 : 3819 VALUE;
5452 : 3820
5453 : 3821 RN();
5454 : 3822
5455 : 3823 IF ((.RANDOM) AND (.RANDOM LSS 0))
5456 : 3824 THEN VALUE = RFAD !CHOOSE THE READ COMMAND
5457 : 3825 ELSE VALUE = CHECK; !CHOOSE THE WRITE CHECK COMMAND
5458 : 3826
5459 : 3827 RETURN .VALUE;
5460 : 3828 END;
5464
5465
5469 043010 004737 005256 CHOOSE: JSR .SBTTL CHOOSE ML11 TRANSFER COMMANDS : 3821
5470 043014 032737 000001 005352 BIT #1,RANDOM : 3823
5471 043022 001406 BEQ 1$ :
5472 043024 005737 005352 TST RANDOM
5473 043030 002003 BGE 1$
5474 043032 012700 042434 MOV #READ,R0 ; *.VALUE 3824
5475 043036 000207 RTS PC ; 3823
5476 043040 012700 042622 1$: MOV #CHECK,R0 ; *.VALUE 3825
5477 043044 000207 RTS PC ; 3802
5478
5479 ; Routine Size: 15 words
5480 ; Maximum stack depth per invocation: 0 words
    
```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (35)

```

5486 ;MLX4
5487 :
5488 :
5489 : 3829 ROUTINE RETRY(TIMES,COMMAND,LUN,WRDCNT,BUFFER,SECTOR) =
5490 : 3830 BEGIN !* 1 *
5491 : 3831
5492 : 3832
5493 : 3833 !+
5494 : 3834 ROUTINE: RETRY(TIMES,COMMAND,LUN,WRDCNT,BUFFER,SECTOR)
5495 : 3835
5496 : 3836 PURPOSE: (1) IF THE OPERATOR HAS ALLOWED ERROR PRINTOUTS, AND
5497 : 3837 IF THE RETRY IS NOT TO CATEGORIZE ERRORS, THEN SAY
5498 : 3838 THAT THE RETRY IS BEGINNING.
5499 : 3839
5500 : 3840 (2) REISSUE THE WRITE, WRITE CHECK OR READ COMMAND
5501 : 3841 UNTIL THE COMMAND SUCCEEDS OR UNTIL ALL PERMITTED
5502 : 3842 RETRIES HAVE FAILED.
5503 : 3843
5504 : 3844 (3) IF A 'BEGIN RETRY' MESSAGE WAS PRINTED, THEN A FINAL
5505 : 3845 MESSAGE ABOUT THE SUCCESS OR FAILURE OF THE RETRIES
5506 : 3846 SHOULD ALSO BE PRINTED.
5507 : 3847
5508 : 3848 (4) INCREMENT THE RETRY COUNTER FOR EVERY RETRY DONE
5509 : 3849 WHICH WAS NOT FOR ERROR CLASSIFICATION.
5510 : 3850
5511 : 3851 ARGUMENTS: (1) TIMES - THE NUMBER OF RETRIES PERMITTED BEFORE
5512 : 3852 IT IS CALLED A FAILURE.
5513 : 3853
5514 : 3854 (2) OTHER ARGUMENTS ARE THE SAME AS FOR 'START_IT'
5515 : 3855 ROUTINE ABOVE.
5516 : 3856
5517 : 3857 RESULTS: THE VALUES RETURNED FOR THIS SUBROUTINE ARE THE SAME AS
5518 : 3858 FOR THE SYSERR ROUTINE ABOVE.
5519 : 3859
5520 : 3860 !--
5521 : 3861
5522 : 3862 LABEL
5523 : 3863 LOOP:
5524 : 3864
5525 : 3865 LOCAL
5526 : 3866 TEMP, VALUE, COUNT, TBOARD, TBANK;
5527 : 3867
5528 : 3868 !+
5529 : 3869 ! THIS IS THE CODE FOR PURPOSE 1:
5530 : 3870 !-
5531 : 3871
5532 : 3872
5533 : 3873 RETRYING = ACTIVE;
5534 : 3874 IF ((.TIMES NEQ 1) AND (.ERROUT))
5535 : 3875 THEN
5536 : 3876 BEGIN !* 2 *
5537 : 3877 IF .COMMAND EQL WRITE
5538 : 3878 THEN PRINTB(SAY3,WRD2,WRD16,WRD18);
5539 : 3879 !BEGIN WRITE RETRY
5540 : 3880 IF .COMMAND EQL CHECK
  
```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (36)

5542 :MLX4

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (36)

ML11 TRANSFER COMMANDS

```
5545 : 3881 THEN PRINTB(SAY3,WRD2,PHR1,WRD18);
5546 : 3882 !BEGIN WRITE CHECK RETRY
5547 : 3883 IF .COMMAND EQL READ
5548 : 3884 THEN PRINTB(SAY3,WRD2,WRD17,WRD18);
5549 : 3885 !BEGIN READ RETRY
5550 : 3886 END; !* 2 *
5551 : 3887
5552 : 3888 !*
5553 : 3889 ! THIS IS THE CODE FOR PURPOSE 2:
5554 : 3890 !-
5555 : 3891
5556 : 3892 LOOP:
5557 : 3893 BEGIN !* 3 *
5558 : 3894 INCR COUNT FROM 1 TO .TIMES DO
5559 : 3895 BEGIN !* 4 *
5560 : 3896 IF .COMMAND EQL WRITE
5561 : 3897 THEN
5562 : 3898 VALUE = WRITE(.LUN,.WRDCNT,.BUFFER,.SECTOR);
5563 : 3899 IF .COMMAND EQL CHECK
5564 : 3900 THEN
5565 : 3901 BEGIN
5566 : 3902 WRITE(.LUN,.WRDCNT,.BUFFER,.SECTOR);
5567 : 3903 VALUE = CHECK(.LUN,.WRDCNT,.BUFFER,.SECTOR);
5568 : 3904 END;
5569 : 3905 IF .COMMAND EQL READ
5570 : 3906 THEN
5571 : 3907 BEGIN
5572 : 3908 WRITE(.LUN,.WRDCNT,(.BUFFER-BUFSIZ*2),.SECTOR);
5573 : 3909 VALUE = READ(.LUN,.WRDCNT,.BUFFER,.SECTOR);
5574 : 3910 END;
5575 : 3911
5576 : 3912 !*
5577 : 3913 ! THIS IS THE CODE FOR PURPOSE 3:
5578 : 3914 !-
5579 : 3915
5580 : 3916 IF .VALUE EQL 0
5581 : 3917 THEN !THE RETRY WAS SUCCESSFUL
5582 : 3918 BEGIN !* 5 *
5583 : 3919 IF ((.TIMES NEQ 1) AND (.ERROUT))
5584 : 3920 THEN
5585 : 3921 BEGIN
5586 : 3922 PRINTB(SAY2,WRD'3,WRD19);
5587 : 3923 PRINTB(CRLF);
5588 : 3924 !RETRY SUCCEEDED
5589 : 3925 END;
5590 : 3926 COUNT = .KOUNT;
5591 : 3927 LEAVE LOOP;
5592 : 3928 END; !* 5 *
5593 : 3929 END; !* 4 *
5594 : 3930
5595 : 3931 !*
5596 : 3932 ! FALLS THROUGH HERE IF ALL RETRIES FAILED:
```

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (36)

```

5598 ;MLX4
5599 ;
5600 ;
5601 ; 3933 !-
5602 ; 3934
5603 ; 3935 IF ((.TIMES NEQ 1) AND (.ERROUT))
5604 ; 3936 THEN
5605 ; 3937 BEGIN
5606 ; 3938 PRINTB(SAY2,WRD18,WRD20);
5607 ; 3939 PRINTB(CRLF);
5608 ; 3940 !RETRY FAILED
5609 ; 3941 COUNT = .TIMES;
5610 ; 3942 END;
5611 ; 3943 END; !* 3 *
5612 ; 3944
5613 ; 3945 !+
5614 ; 3946 ! THIS IS THE CODE FOR PURPOSE 4:
5615 ; 3947 !-
5616 ; 3948
5617 ; 3949 IF .TIMES NEQ 1
5618 ; 3950 THEN
5619 ; 3951 BEGIN
5620 ; 3952 TBOARD = .BOARD;
5621 ; 3953 TBANK = .BANK;
5622 ; 3954 DECODE(.SECTOR);
5623 ; 3955 TRIES[.LUN,.BOARD,0,16,0] = .TRIES[.LUN,.BOARD,0,16,0] + .COUNT;
5624 ; 3956 BOARD = .TBOARD;
5625 ; 3957 BANK = .TBANK;
5626 ; 3958 END;
5627 ; 3959
5628 ; 3960 RETRYING = INACTIVE;
5629 ; 3961 RETURN .VALUE;
5630 ; 3962
5631 ; 3963 END; !* 1 *
5635
5636 ;

```

```

5640 043046 004137 005222 RETRY: .SBTTL RETRY ML11 TRANSFER COMMANDS
5641 043052 012737 000001 032356 JSR R1,$SAVE5
5642 043060 016601 000030 MOV #1,RETRYING
5643 043064 005046 CLR -(SP)
5644 043066 020127 000001 CMP R1,#1
5645 043072 001472 BEQ 3$
5646 043074 005216 INC (SP)
5647 043076 032737 000001 002260 BIT #1,ERROUT
5648 043104 001465 BEQ 3$
5649 043106 016602 000030 MOV 30(SP),R2
5650 043112 020227 042246 CMP R2,#WRITE
5651 043116 001016 BNE 1$
5652 043120 012746 006714 MOV #WRD18,-(SP)

```

3829  
 3873  
 3874  
 3877  
 3878

Address	Hex	Hex	Hex	Label	Command	Comments	Line No.
5654				:MLX4			
5655				:			
5656					ML11 TRANSFER COMMANDS		
5657	043124	012746	006700		MOV #WRD16,-(SP)		
5658	043130	012746	006622		MOV #WRD2,-(SP)		
5659	043134	012746	006534		MOV #SAY3,-(SP)		
5660	043140	012746	000004		MOV #4,-(SP)		
5661	043144	010600			MOV SP,R0	; SP,*	
5662	043146	104414			TRAP 14		
5663	043150	062706	000012		ADD #12,SP		
5664	043154	020227	042622	1\$:	CMP R2,#CHECK	:	3880
5665	043160	001016			BNE 2\$	:	
5666	043162	012746	006714		MOV #WRD18,-(SP)	:	3881
5667	043166	012746	007336		MOV #PHR1,-(SP)		
5668	043172	012746	006622		MOV #WRD2,-(SP)		
5669	043176	012746	006534		MOV #SAY3,-(SP)		
5670	043202	012746	000004		MOV #4,-(SP)		
5671	043206	010600			MOV SP,R0	; SP,*	
5672	043210	104414			TRAP 14		
5673	043212	062706	000012		ADD #12,SP		
5674	043216	020227	042434	2\$:	CMP R2,#READ	:	3883
5675	043222	001016			BNE 3\$	:	
5676	043224	012746	006714		MOV #WRD18,-(SP)	:	3884
5677	043230	012746	006706		MOV #WRD17,-(SP)		
5678	043234	012746	006622		MOV #WRD2,-(SP)		
5679	043240	012746	006534		MOV #SAY3,-(SP)		
5680	043244	012746	000004		MOV #4,-(SP)		
5681	043250	010600			MOV SP,R0	; SP,*	
5682	043252	104414			TRAP 14		
5683	043254	062706	000012		ADD #12,SP		
5684	043260	016602	000030	3\$:	MOV 30(SP),R2	; COMMAND,*	3896
5685	043264	005004			CLR R4	; KOUNT	3894
5686	043266	000543			BR 9\$		
5687	043270	020227	042246	4\$:	CMP R2,#WRITE	:	3896
5688	043274	001015			BNE 5\$		
5689	043276	016646	000026		MOV 26(SP),-(SP)	; LUN,*	3898
5690	043302	016646	000026		MOV 26(SP),-(SP)	; WRDCNT,*	
5691	043306	016646	000026		MOV 26(SP),-(SP)	; BUFFER,*	
5692	043312	016646	000026		MOV 26(SP),-(SP)	; SECTOR,*	
5693	043316	004737	042246		JSR PC,WRITE		
5694	043322	010003			MOV R0,R3	; *,VALUE	
5695	043324	062706	000010		ADD #10,SP		
5696	043330	020227	042622	5\$:	CMP R2,#CHECK	:	3899
5697	043334	001027			BNE 6\$		
5698	043336	016646	000026		MOV 26(SP),-(SP)	; LUN,*	3902
5699	043342	016646	000026		MOV 26(SP),-(SP)	; WRDCNT,*	
5700	043346	016646	000026		MOV 26(SP),-(SP)	; BUFFER,*	
5701	043352	016646	000026		MOV 26(SP),-(SP)	; SECTOR,*	
5702	043356	004737	042246		JSR PC,WRITE		
5703	043362	016616	000036		MOV 36(SP),(SP)	; LUN,*	3903
5704	043366	016646	000034		MOV 34(SP),-(SP)	; WRDCNT,*	
5705	043372	016646	000034		MOV 34(SP),-(SP)	; BUFFER,*	
5706	043376	016646	000034		MOV 34(SP),-(SP)	; SECTOR,*	
5707	043402	004737	042622		JSR PC,CHECK		
5708	043406	010003			MOV R0,R3	; *,VALUE	



Address	Hex	Hex	Hex	Label	Code	Comment	Time	Page
5710				:MLX4			23-Oct-1980 15:01:43	TOPS
5711				:			23-Oct-1980 14:57:05	PA:<
5712					ML11 TRANSFER COMMANDS			
5713	043410	062706	000016		ADD #16,SP	:		3901
5714	043414	020227	042434	6\$:	CMP R2,#READ	:		3905
5715	043420	001031			BNE 7\$			
5716	043422	016646	000026		MOV 26(SP),-(SP)	: LUN,*		3908
5717	043426	016646	000026		MOV 26(SP),-(SP)	: WRDCNT,*		
5718	043432	016646	000026		MOV 26(SP),-(SP)	: BUFFER,*		
5719	043436	162716	010000		SUB #10000,(SP)			
5720	043442	016646	000026		MOV 26(SP),-(SP)	: SECTOR,*		
5721	043446	004737	042246		JSR PC,WRITE			
5722	043452	016616	000036		MOV 36(SP),(SP)	: LUN,*		3909
5723	043456	016646	000034		MOV 34(SP),-(SP)	: WRDCNT,*		
5724	043462	016646	000034		MOV 34(SP),-(SP)	: BUFFER,*		
5725	043466	016646	000034		MOV 34(SP),-(SP)	: SECTOR,*		
5726	043472	004737	042434		JSR PC,READ			
5727	043476	010003			MOV R0,R3	: *,VALUE		
5728	043500	062706	000016		ADD #16,SP	:		3907
5729	043504	005703		7\$:	TST R3	: VALUE		3916
5730	043506	001033			BNE 9\$			
5731	043510	032716	000001		BIT #1,(SP)	:		3919
5732	043514	001426			BEQ 8\$			
5733	043516	032737	000001	002260	BIT #1,ERRROUT			
5734	043524	001422			BEQ 8\$			
5735	043526	012746	006722		MOV #WRD19,-(SP)	:		3922
5736	043532	012746	006714		MOV #WRD18,-(SP)			
5737	043536	012746	006522		MOV #SAY2,-(SP)			
5738	043542	012746	000003		MOV #3,-(SP)			
5739	043546	010600			MOV SP,R0	: SP,*		
5740	043550	104414			TRAP 14			
5741	043552	012716	006510		MOV #CRLF,(SP)	:		3923
5742	043556	012746	000001		MOV #1,-(SP)			
5743	043562	010600			MOV SP,R0	: SP,*		
5744	043564	104414			TRAP 14			
5745	043566	062706	000012		ADD #12,SP	:		3921
5746	043572	010405		8\$:	MOV R4,R5	: KOUNT,COUNT		3926
5747	043574	000435			BR 10\$	:		3927
5748	043576	005204		9\$:	INC R4	: KOUNT		3894
5749	043600	020401			CMP R4,R1	: KOUNT,*		
5750	043602	003632			BLE 4\$			
5751	043604	032716	000001		BIT #1,(SP)	:		3935
5752	043610	001462			BEQ 11\$			
5753	043612	032737	000001	002260	BIT #1,ERRROUT			
5754	043620	001423			BEQ 10\$			
5755	043622	012746	006734		MOV #WRD20,-(SP)	:		3938
5756	043626	012746	006714		MOV #WRD18,-(SP)			
5757	043632	012746	006522		MOV #SAY2,-(SP)			
5758	043636	012746	000003		MOV #3,-(SP)			
5759	043642	010600			MOV SP,R0	: SP,*		
5760	043644	104414			TRAP 14			
5761	043646	012716	006510		MOV #CRLF,(SP)	:		3939
5762	043652	012746	000001		MOV #1,-(SP)			
5763	043656	010600			MOV SP,R0	: SP,*		
5764	043660	104414			TRAP 14			

23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<

5766									
5767									
5768									
5769	043662	010105			MOV	R1,R5		: *.COUNT	3941
5770	043664	062706	000012		ADD	#12,SP		:	3937
5771	043670	032716	000001	10\$:	BIT	#1,(SP)		:	3949
5772	043674	001430			BEQ	11\$		:	
5773	043676	013701	032360		MOV	BOARD,R1		: *.TBOARD	3952
5774	043702	013702	032362		MOV	BANK,R2		: *.TBANK	3953
5775	043706	016646	000020		MOV	20(SP),-(SP)		: SECTOR,*	3954
5776	043712	004737	035106		JSR	PC,DECODE			
5777	043716	016600	000030		MOV	30(SP),R0		: LUN,*	3955
5778	043722	006300			ASL	R0			
5779	043724	006300			ASL	R0			
5780	043726	006300			ASL	R0			
5781	043730	006300			ASL	R0			
5782	043732	063700	032360		ADD	BOARD,R0			
5783	043736	006300			ASL	R0			
5784	043740	060560	031732		ADD	R5,TRIES(R0)		: COUNT,*	
5785	043744	010137	032360		MOV	R1,BOARD		: TBOARD,*	3956
5786	043750	010237	032362		MOV	R2,BANK		: TBANK,*	3957
5787	043754	005726			TST	(SP)+		:	3951
5788	043756	005037	032356	11\$:	CLR	RETRYING		:	3960
5789	043762	010300			MOV	R3,R0		: VALUE,*	3830
5790	043764	005726			TST	(SP)+		:	3829
5791	043766	000207			RTS	PC		:	

: Routine Size: 233 words  
: Maximum stack depth per invocation: 14 words

5792  
5793  
5794  
5799  
5800

```

5802 ;MLX4
5803 :
5804 :
5805 : 3964 %SBTTL 'TO SET UP BUFFER POINTERS BEFORE A TRANSFER'
5806 : 3965
5807 : 3966 ROUTINE SET_PTRS(WRDCNT): NOVALUE =
5808 : 3967 BEGIN
5809 : 3968
5810 : 3969 !++
5811 : 3970 ROUTINE: SET_PTRS(WRDCNT)
5812 : 3971
5813 : 3972 PURPOSE: TO MAKE SURE THAT THE BUFFER POINTERS ARE SUITABLY PLACED
5814 : 3973 BEFORE A TRANSFER, SO THAT THEY WILL SUPPORT THE CHOSEN WORD
5815 : 3974 COUNT WITHIN THE BUFFER SPACE.
5816 : 3975
5817 : 3976 ARGUMENT: WRDCNT = THE NUMBER OF WORDS IN A TRANSFER.
5818 : 3977
5819 : 3978 RESULTS: 'WPTR' AND 'RPTR' ARE SET BACK TO THE START OF THE BUFFERS
5820 : 3979 ONLY IF THE WORD COUNT WON'T FIT. OTHERWISE, THEY ARE LEFT
5821 : 3980 UNCHANGED.
5822 : 3981
5823 : 3982 NOTE: THE WRITE BUFFER IS LOCATED BEFORE THE READ BUFFER.
5824 : 3983 !--
5825 : 3984
5826 : 3985 IF (.WPTR + .WRDCNT * 2) GEQA END_WBUFF
5827 : 3986 THEN WPTR = WBUFF;
5828 : 3987 RPTR = .WPTR + (BUFSIZ * 2);
5829 : 3988 RETURN;
5830 : 3989
5831 : 3990 END;
5835 :
5836 : .SBTTL SET_PTRS TO SET UP BUFFER POINTERS BEFORE A TRANSFER
5840 043770 SET_PTRS:
5841 043770 016600 000002 MOV 2(SP),RO ; WRDCNT,* 3985
5842 043774 006300 ASL RO
5843 043776 063700 030706 ADD WPTR,RO
5844 044002 020027 020706 CMP RO,#END.WBUFF
5845 044006 103403 BLO 1$
5846 044010 012737 010706 030706 MOV #WBUFF,WPTR ; 3986
5847 044016 013700 030706 1$: MOV WPTR,RO ; 3987
5848 044022 062700 010000 ADD #10000,RO
5849 044026 010037 030710 MOV RO,RPTR
5850 044032 000207 RTS PC ; 3966
5851 :
5852 : ; Routine Size: 18 words
5853 : ; Maximum stack depth per invocation: 0 words

```

MLX4 MACRO M1113 24-OCT-80 09:59 PAGE 207  
SET.PTRS TO SET UP BUFFER POINTERS BEFORE A TRANSFER

SEQ 0162

5862  
5863

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 v2(20c)  
23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (39)

```

5865 ;MLX4
5866 :
5867 :
5868 : 3991 %SBTTL 'CHOOSING A WORD COUNT'
5869 : 3992
5870 : 3993 ROUTINE GET_WRCNT(SECTOR, LAST) =
5871 : 3994 BEGIN
5872 : 3995
5873 : 3996 LOCAL
5874 : 3997     TEMP, WRCNT;
5875 : 3998
5876 : 3999 TEMP = .LAST - .SECTOR + 1;      !NUMBER OF SECTORS LEFT TO TEST
5877 : 4000
5878 : 4001 IF (BUFSIZ/256) GTR .TEMP          !LESS THAN A FULL BUFFER LEFT?
5879 : 4002 THEN WRCNT = .TEMP * 256         !YES -- USE AS LARGE A COUNT AS FITS
5880 : 4003 ELSE WRCNT = BUFSIZ;             !NO -- USE THE ENTIRE BUFFER SIZE
5881 : 4004
5882 : 4005 RETURN .WRCNT;
5883 : 4006
5884 : 4007 END;

```

```

5888
5889         .SBTTL GET.WRCNT CHOOSING A WORD COUNT
5893 044034 GET.WRCNT:
5894 044034 016600 000002      MOV     2(SP),R0      ; LAST,*      3999
5895 044040 166600 000004      SUB     4(SP),R0      ; SECTOR,*
5896 044044 005200              INC     R0
5897 044046 020027 000010      CMP     R0,#10       ; TEMP,*      4001
5898 044052 002003              BGE    1$
5899 044054 000300              SWAB   R0             ;              4002
5900 044056 105000              CLRB  R0
5901 044060 000207              RTS    PC            ;              4001
5902 044062 012700 004000      1$:   MOV     #4000,R0 ; *,WRCNT     4003
5903 044066 000207              RTS    PC            ;              3993

```

; Routine Size: 14 words  
; Maximum stack depth per invocation: 0 words

5904  
5905  
5906  
5911  
5912

```

5914 :MLX4
5915 :
5916 :
5917 : 4008 %SBTTL 'COMMAND INTEGRITY ROUTINE'
5918 : 4009
5919 : 4010 ROUTINE INTEGRITY: NOVALUE =
5920 : 4011 BEGIN !* 1 * START OF ROUTINE
5921 : 4012
5922 : 4013 !++
5923 : 4014 ! ROUTINE: INTEGRITY
5924 : 4015 !
5925 : 4016 ! PURPOSE: TO MAKE SURE THAT THE BASIC ML11 TRANSFER COMMANDS
5926 : 4017 ! WHICH WILL BE USED BY THE EXERCISER (WRITE,READ,
5927 : 4018 ! WRITE CHECK) WORK PROPERLY.
5928 : 4019 !
5929 : 4020 ! THE CODE FOR 'INTEGRITY' IN BRIEF:
5930 : 4021 !
5931 : 4022 ! BEGIN 1 (START OF ROUTINE)
5932 : 4023 ! SAY ROUTINE IS RUNNING
5933 : 4024 ! INCR COMPLEMENT FLAG FROM 0 TO 1
5934 : 4025 ! : BEGIN 2 (START OF COMPLEMENT FLAG SELECTION LOOP)
5935 : 4026 ! : GENERATE THE PATTERN
5936 : 4027 ! : INCR LOGICAL UNIT NUMBER FROM 0 TO LAST
5937 : 4028 ! : : BEGIN 3 (START OF LOGICAL UNIT SELECTION LOOP)
5938 : 4029 ! : : TESTLOOP:
5939 : 4030 ! : : : BEGIN 4 (START OF LOOP THAT COMPLETELY TESTS 1 UNIT)
5940 : 4031 ! : : : IF UNIT IS ACTIVE
5941 : 4032 ! : : : THEN
5942 : 4033 ! : : : : BEGIN 5 (START OF TEST FOR AN ACTIVE UNIT)
5943 : 4034 ! : : : : SECTOR = LOWEST
5944 : 4035 ! : : : : GET WRDCNT
5945 : 4036 ! : : : : WRITE
5946 : 4037 ! : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
5947 : 4038 ! : : : : READ
5948 : 4039 ! : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
5949 : 4040 ! : : : : WRITE CHECK
5950 : 4041 ! : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
5951 : 4042 ! : : : : END 5 (END OF TEST FOR AN ACTIVE UNIT)
5952 : 4043 ! : : : : END 4 (END OF LOOP THAT COMPLETELY TESTS 1 UNIT)
5953 : 4044 ! : : : : END 3 (END OF LOGICAL UNIT SELECTION LOOP)
5954 : 4045 ! : : : : END 2 (END OF COMPLEMENT FLAG SELECTION LOOP)
5955 : 4046 ! RETURN
5956 : 4047 ! END 1 (END OF ROUTINE)
5957 : 4048 !--
5958 : 4049
5959 : 4050 LABEL
5960 : 4051 LOOP;
5961 : 4052
5962 : 4053 LOCAL
5963 : 4054 VALUE, WRDCNT, OLDSEC, OLDCHN, SECTOR, DBL_VALUE;
5964 : 4055
5965 : 4056 PRINTB(SAY2,WRD34,RTN0);
5966 : 4057 'RUNNING COMMAND INTEGRITY ROUTINE'
5967 : 4058
5968 : 4059 INCR COMP_FLAG FROM 0 TO 1 DO

```

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (39)

5970 :MLX4  
 5971 :  
 5972 :  
 5973 :  
 5974 :  
 5975 :  
 5976 :  
 5977 :  
 5978 :  
 5979 :  
 5980 :  
 5981 :  
 5982 :  
 5983 :  
 5984 :  
 5985 :  
 5986 :  
 5987 :  
 5988 :  
 5989 :  
 5990 :  
 5991 :  
 5992 :  
 5993 :  
 5994 :  
 5995 :  
 5996 :  
 5997 :  
 5998 :  
 5999 :  
 6000 :  
 6001 :  
 6002 :  
 6003 :  
 6004 :  
 6005 :  
 6006 :  
 6007 :  
 6008 :  
 6009 :  
 6010 :  
 6011 :  
 6012 :  
 6013 :  
 6014 :  
 6015 :  
 6016 :  
 6017 :  
 6018 :  
 6019 :  
 6020 :  
 6021 :  
 6022 :  
 6023 :  
 6024 :

COMMAND INTEGRITY ROUTINE

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (39)

```

BEGIN !* 2 * START OF COMPLEMENT FLAG SELECTION LOOP
GEN3(.COMP FLAG);
INCR LUN FROM 0 TO (.LSUNIT - 1) DO
  BEGIN !* 3 * START OF LOGICAL UNIT SELECTION LOOP
  LOOP:
  BEGIN !* 4 * START OF LOOP THAT COMPLETELY TESTS 1 UNIT
  IF .DRIVE_STATUS[.LUN] EQL ACTIVE
  THEN
  BEGIN !* 5 * START OF TEST FOR AN ACTIVE UNIT
  L$LUN = .LUN;
  SECTOR = LOWEST;
  WRDCNT = GET_WRDCNT(.SECTOR,HIGHEST);

  VALUE = WRITE(.LUN,.WRDCNT,WBUFF,LOWEST);
  !+
  !- SEE HOW SUCCESSFUL THE WRITE WAS:
  !-
  SELECTONE .VALUE OF          !SEE 'SYSERR' FOR DEFINITION
  SET                          !OF ERROR # CONTAINED IN 'VALUE'
  [1] :
  BEGIN !* 5A * RETRY ALLOWED
  IF RETRY(SIX,WRITE,.LUN,.WRDCNT,WBUFF,LOWEST) NEQ 0
  THEN !THE RETRY FAILED -- SYSTEM FATAL ERROR
  BEGIN
  WHY DROPT[.LUN] = CODE_4;
  ERRDF(1,MSG1,0); !*** INTEGRITY ROUTINE ERROR 01 ***
  DODU(.LUN);
  LEAVE LOOP;          !JUMP JUST BEYOND END OF BLOCK * 4 *
  END;
  !* 5A *
  [2] :
  BEGIN !* 5B * FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
  WHY DROPT[.LUN] = CODE_5;
  ERRDF(2,MSG1,0); !*** INTEGRITY ROUTINE ERROR 02 ***
  DODU(.LUN);
  LEAVE LOOP;          !JUMP JUST BEYOND END OF BLOCK * 4 *
  END;
  !* 5B *
  [3] :
  BEGIN !* 5C * FATAL DRIVE ERROR -- NO RETRY ALLOWED
  ERRDF(3,MSG1,0); !*** INTEGRITY ROUTINE ERROR 03 ***
  WHY DROPT[.LUN] = CODE_6;
  DODU(.LUN);
  LEAVE LOOP;          !JUMP JUST BEYOND END OF BLOCK * 4 *
  END;
  !* 5C *

  TES;

  VALUE = READ(.LUN,.WRDCNT,RBUFF,LOWEST);
  !+
  !- SEE HOW SUCCESSFUL THE READ WAS:
  !-
  SELECTONE .VALUE OF          !SEE 'SYSERR' FOR DEFINITION
  
```

6026 :MLX4  
 6027 :  
 6028 :  
 6029 :  
 6030 :  
 6031 :  
 6032 :  
 6033 :  
 6034 :  
 6035 :  
 6036 :  
 6037 :  
 6038 :  
 6039 :  
 6040 :  
 6041 :  
 6042 :  
 6043 :  
 6044 :  
 6045 :  
 6046 :  
 6047 :  
 6048 :  
 6049 :  
 6050 :  
 6051 :  
 6052 :  
 6053 :  
 6054 :  
 6055 :  
 6056 :  
 6057 :  
 6058 :  
 6059 :  
 6060 :  
 6061 :  
 6062 :  
 6063 :  
 6064 :  
 6065 :  
 6066 :  
 6067 :  
 6068 :  
 6069 :  
 6070 :  
 6071 :  
 6072 :  
 6073 :  
 6074 :  
 6075 :  
 6076 :  
 6077 :  
 6078 :  
 6079 :  
 6080 :

COMMAND INTEGRITY ROUTINE

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (39)

```

SET                                !OF ERROR # CONTAINED IN 'VALUE'
[0] :
IF (DBL_VALUE = DOUBLE_CHECK(WBUFF,RBUFF,.WRDCNT)) NEQ 0
THEN
BEGIN
  SAYWHO(.LUN);
  PRINTB(SAY1,MSG5);
  !'ECC LOGIC FAILED TO DETECT DATA ERROR'
  PRINTB(FMT12A,..DBL_VALUE,..DBL_VALUE);
  !'GOOD DATA: XXXXX AT LOCATION YYYYYY'
  DBL_VALUE = .DBL_VALUE + BUFSIZ * 2;
  PRINTB(FMT12B,..DBL_VALUE,..DBL_VALUE);
  !'BAD DATA: PPPPP AT LOCATION QQQQQQ'
  WHY DROPT(.LUN) = CODE_8;
  ERRDF(4,MSG1,0);  !**** INTEGRITY ROUTINE ERROR 04 ****
  DODU(.LUN);
  LEAVE LOOP;      !JUMP JUST BEYOND END OF BLOCK * 4 *
END;

[1] :
BEGIN !* 5D *  RETRY ALLOWED
IF RETRY(SIX,READ,.LUN,.WRDCNT,RBUFF,LOWEST) NEQ 0
THEN !THE RETRY FAILED -- SYSTEM FATAL ERROR
BEGIN
  WHY DROPT(.LUN) = CODE_4;
  ERRDF(5,MSG1,0);  !**** INTEGRITY ROUTINE ERROR 05 ****
  DODU(.LUN);
  LEAVE LOOP;      !JUMP JUST BEYOND END OF BLOCK * 4 *
END;
END;  !* 5D *

[2] :
BEGIN !* 5E *  FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
WHY DROPT(.LUN) = CODE_5;
ERRDF(6,MSG1,0);  !**** INTEGRITY ROUTINE ERROR 06 ****
DODU(.LUN);
LEAVE LOOP;      !JUMP JUST BEYOND END OF BLOCK * 4 *
END;  !* 5E *

[3] :
BEGIN !* 5F *  FATAL DRIVE ERROR -- NO RETRY ALLOWED
ERRDF(7,MSG1,0);  !**** INTEGRITY ROUTINE ERROR 07 ****
WHY DROPT(.LUN) = CODE_6;
DODU(.LUN);
LEAVE LOOP;      !JUMP JUST BEYOND END OF BLOCK * 4 *
END;  !* 5F *

[4] :
BEGIN !* 5G *  UNRECOVERABLE DATA ERROR
ISOLATE();
ERRDF(8,MSG2,0);  !**** INTEGRITY ROUTINE ERROR 08 ****
WHY DROPT(.LUN) = CODE_7;
DODU(.LUN);
LEAVE LOOP;      !JUMP JUST BEYOND END OF BLOCK * 4 *
END;  !* 5G *

[5] :

```



```

6082 :MLX4
6083 :
6084 :
6085 : 4164 BEGIN !* 5H * RECOVERABLE DATA ERROR
6086 : 4165 ISOLATE();
6087 : 4166 IF .ERROUT
6088 : 4167 THEN PRINTB(FMT10B,.CHAN);
6089 : 4168 !' BIT QQ'
6090 : 4169 OLDSEC = .MLEL;
6091 : 4170 OLDCHN = .CHAN;
6092 : 4171 IF RETRY(ONE,READ,.LUN,.WRDCNT,RBUFF,LOWEST) EQL 5
6093 : 4172 THEN
6094 : 4173 IF ((.MLEL EQL .OLDSEC) AND (.CHAN EQL .OLDCHN))
6095 : 4174 THEN
6096 : 4175 BEGIN
6097 : 4176 IF .ERROUT
6098 : 4177 THEN
6099 : 4178 ERRHRD(9,MSG4,0); !**** INTEGRITY ROUTINE ERROR 09 ****
6100 : 4179 UP_HARD_COUNT(.LUN,.BOARD);
6101 : 4180 END
6102 : 4181 ELSE
6103 : 4182 BEGIN
6104 : 4183 IF .ERROUT
6105 : 4184 THEN
6106 : 4185 ERRSOFT(10,MSG3,0); !**** INTEGRITY ROUTINE ERROR 10 ****
6107 : 4186 UP_SOFT_COUNT(.LUN,.BOARD);
6108 : 4187 END
6109 : 4188 ELSE
6110 : 4189 BEGIN
6111 : 4190 IF .ERROUT
6112 : 4191 THEN
6113 : 4192 ERRSOFT(11,MSG3,0); !**** INTEGRITY ROUTINE ERROR 11 ****
6114 : 4193 UP_SOFT_COUNT(.LUN,.BOARD);
6115 : 4194 END;
6116 : 4195 END; !* 5H *
6117 : 4196
6118 : 4197 TES;
6119 : 4198
6120 : 4199 VALUE = CHECK(.LUN,.WRDCNT,WBUFF,LOWEST);
6121 : 4200 !+
6122 : 4201 !SEE HOW SUCCESSFUL THE WRITE CHECK WAS:
6123 : 4202 !-
6124 : 4203 SELECTONE .VALUE OF !SEE 'SYSERR' FOR DEFINITION
6125 : 4204 SET !OF ERROR # CONTAINED IN 'VALUE'
6126 : 4205 [1] :
6127 : 4206 BEGIN !* 5I * RETRY ALLOWED
6128 : 4207 IF RETRY(SIX,CHECK,.LUN,.WRDCNT,WBUFF,LOWEST) NEQ 0
6129 : 4208 THEN !THE RETRY FAILED -- SYSTEM FATAL ERROR
6130 : 4209 BEGIN
6131 : 4210 WHY DROPT[.LUN] = CODE_4;
6132 : 4211 ERRDF(12,MSG1,0); !**** INTEGRITY ROUTINE ERROR 12 ****
6133 : 4212 DODU(.LUN);
6134 : 4213 LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
6135 : 4214 END;
6136 : 4215 END; !* 5I *

```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (39)

6138 :MLX4

COMMAND INTEGRITY ROUTINE

23-Oct-1980 15:01:43

TOPS-20 Bliss-16 V2(206)

23-Oct-1980 14:57:05

PA:<ROSEN>MLX4.BLI.1 (39)

6139 :  
6140 :  
6141 :  
6142 :  
6143 :  
6144 :  
6145 :  
6146 :  
6147 :  
6148 :  
6149 :  
6150 :  
6151 :  
6152 :  
6153 :  
6154 :  
6155 :  
6156 :  
6157 :  
6158 :  
6159 :  
6160 :  
6161 :  
6162 :  
6163 :  
6164 :  
6165 :  
6166 :  
6167 :  
6168 :  
6169 :  
6170 :  
6171 :  
6172 :  
6173 :  
6174 :  
6175 :  
6176 :  
6177 :  
6178 :  
6179 :  
6180 :  
6181 :  
6182 :  
6183 :  
6184 :  
6185 :  
6186 :  
6187 :  
6188 :  
6189 :  
6190 :  
6191 :  
6192 :

4216  
4217  
4218  
4219  
4220  
4221  
4222  
4223  
4224  
4225  
4226  
4227  
4228  
4229  
4230  
4231  
4232  
4233  
4234  
4235  
4236  
4237  
4238  
4239  
4240  
4241  
4242  
4243  
4244  
4245  
4246  
4247  
4248  
4249  
4250  
4251  
4252  
4253  
4254  
4255  
4256  
4257  
4258  
4259  
4260  
4261  
4262  
4263  
4264  
4265  
4266  
4267

```
[2] :  
BEGIN !* 5J * FATAL CONTROLLER ERROR -- NO RETRY ALLOWED  
WHY DROPT[.LUN] = CODE_5;  
ERRDF(13,MSG1,0); !**** INTEGRITY ROUTINE ERROR 13 ****  
DODU(.LUN);  
LEAVE LOOP; !JUMP JUST BEYO'D END OF BLOCK * 4 *  
END; !* 5J *  
[3] :  
BEGIN !* 5K * FATAL DRIVE ERROR -- NO RETRY ALLOWED  
ERRDF(14,MSG1,0); !**** INTEGRITY ROUTINE ERROR 14 ****  
WHY DROPT[.LUN] = CODE_6;  
DODU(.LUN);  
LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *  
END; !* 5K *  
[4] :  
BEGIN !* 5L * UNRECOVERABLE DATA ERROR  
ISOLATE();  
ERRDF(15,MSG2,0); !**** INTEGRITY ROUTINE ERROR 15 ****  
WHY DROPT[.LUN] = CODE_7;  
DODU(.LUN);  
LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *  
END; !* 5L *  
[5] :  
BEGIN !* 5M * RECOVERABLE DATA ERROR  
ISOLATE();  
IF .ERROUT  
THEN PRINTB(FMT10B,.CHAN);  
!' BIT 00'  
OLDSEC = .MLEL;  
OLDCHN = .CHAN;  
IF RETRY(ONE,CHECK,.LUN,.WRDCNT,WBUFF,LOWEST) EQL 5  
THEN  
IF ((.MLEL EQL .OLDSEC) AND (.CHAN EQL .OLDCHN))  
THEN  
BEGIN  
IF .ERROUT  
THEN  
ERRHRD(16,MSG4,0); !**** INTEGRITY ROUTINE ERROR 16 ****  
UP_HARD_COUNT(.LUN,.BOARD);  
END  
ELSE  
BEGIN  
IF .ERROUT  
THEN  
ERRSOFT(17,MSG3,0); !**** INTEGRITY ROUTINE ERROR 17 ****  
UP_SOFT_COUNT(.LUN,.BOARD);  
END  
ELSE  
BEGIN  
IF .ERROUT  
THEN  
ERRSOFT(18,MSG3,0); !**** INTEGRITY ROUTINE ERROR 18 ****
```

```

6194 :MLX4
6195 :
6196 :
6197 : 4268 UP_SOFT_COUNT(.LUN,.BOARD);
6198 : 4269 END;
6199 : 4270 END; !* 5M *
6200 : 4271
6201 : 4272 TES;
6202 : 4273
6203 : 4274 END; !* 5 * END OF TEST FOR AN ACTIVE DRIVE
6204 : 4275 END; !* 4 * END OF LOOP THAT COMPLETELY TESTS 1 UNIT
6205 : 4276 END; !* 3 * END OF LOGICAL UNIT SELECTION LOOP
6206 : 4277 END; !* 2 * END OF COMPLEMENT FLAG SELECTION LOOP
6207 : 4278 RETURN;
6208 : 4279 END; !* 1 * END OF ROUTINE
  
```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (39)

```

6212 :
6213 :
6217 044070 .SBTTL INTEGRITY COMMAND INTEGRITY ROUTINE
6218 044070 004137 005222 INTEGRITY:
6219 044074 162706 000012 JSR R1,$SAVE5 ; 4010
6220 044100 012746 007216 SUB #12,SP ;
6221 044104 012746 006766 MOV #RTNO,-(SP) ; 4056
6222 044110 012746 006522 MOV #WRD34,-(SP)
6223 044114 012746 000003 MOV #SAY2,-(SP)
6224 044120 010600 MOV #3,-(SP)
6225 044122 104414 MOV SP,R0 ; SP,*
6226 044124 005001 TRAP 14
6227 044126 010146 CLR R1 ; COMP.FLAG 4059
6228 044130 004737 036556 1$: MOV R1,-(SP) ; COMP.FLAG,* 4061
6229 044134 013766 002012 000020 JSR PC,GEN3
6230 044142 005004 MOV L$UNIT,20(SP) ; 4062
6231 044144 000137 046260 CLR R4 ; LUN
6232 044150 010400 2$: JMP 38$ ; LUN,* 4066
6233 044152 006200 MOV R4,R0
6234 044154 006200 ASR R0
6235 044156 006200 ASR R0
6236 044160 062700 032460 ASR R0
6237 044164 010046 ADD #DRIVE.STATUS,R0
6238 044166 010446 MOV R0,-(SP)
6239 044170 042716 177770 MOV R4,-(SP) ; LUN,*
6240 044174 012746 000001 BIC #177770,(SP)
6241 044200 005046 CLR -(SP)
6242 044202 004737 004244 JSR PC,BL$GT2
6243 044206 062706 000010 ADD #10,SP
6244 044212 005300 DEC R0
6245 044214 001117 BNE 6$
6246 044216 010437 002074 MOV R4,L$LUN ; LUN,* 4069
6247 044222 010400 MOV R4,R0 ; LUN,* 4070
6248 044224 006300 ASL R0
  
```

```

6250 ;MLX4
6251 ;
6252
6253 044226 012702 032476 MOV #LOW.SECT,R2
6254 044232 060002 ADD R0,R2
6255 044234 011266 MOV (R2),22(SP) ; *,SECTOR
6256 044240 016646 MOV 22(SP),-(SP) ; SECTOR,* 4071
6257 044244 016046 MOV TOP.SECT(R0),-(SP)
6258 044250 004737 JSR PC,GET.WRDCNT
6259 044254 010003 MOV R0,R3 ; *,WRDCNT
6260 044256 010416 MOV R4,(SP) ; LUN,* 4073
6261 044260 010346 MOV R3,-(SP) ; WRDCNT,*
6262 044262 012746 010706 MOV #WBUFF,-(SP)
6263 044266 011246 MOV (R2),-(SP)
6264 044270 004737 042246 JSR PC,WRITE
6265 044274 010005 MOV R0,R5 ; *,VALUE
6266 044276 020527 000001 CMP R5,#1 ; VALUE,* 4077
6267 044302 001031 BNE 3$
6268 044304 012746 000006 MOV #6,-(SP) ; 4081
6269 044310 012746 042246 MOV #WRITE,-(SP)
6270 044314 010446 MOV R4,-(SP) ; LUN,*
6271 044316 010346 MOV R3,-(SP) ; WRDCNT,*
6272 044320 012746 010706 MOV #WBUFF,-(SP)
6273 044324 011246 MOV (R2),-(SP)
6274 044326 004737 043046 JSR PC,RETRY
6275 044332 062706 000014 ADD #14,SP
6276 044336 005700 TST R0
6277 044340 001446 BEQ 7$
6278 044342 112764 000004 032464 MOVB #4,WHY.DROPT(R4) ; *,*(LUN) 4084
6279 044350 104455 TRAP 55 ; 4085
6280 044352 000001 .WORD 1
6281 044354 010512 .WORD MSG1
6282 044356 000000 .WORD 0
6283 044360 010400 MOV R4,R0 ; LUN,* 4086
6284 044362 104451 TRAP 51
6285 044364 000431 BR 5$ ; 4087
6286 044366 020527 000002 3$: CMP R5,#2 ; VALUE,* 4077
6287 044372 001012 BNE 4$
6288 044374 112764 000005 032464 MOVB #5,WHY.DROPT(R4) ; *,*(LUN) 4092
6289 044402 104455 TRAP 55 ; 4093
6290 044404 000002 .WORD 2
6291 044406 010512 .WORD MSG1
6292 044410 000000 .WORD 0
6293 044412 010400 MOV R4,R0 ; LUN,* 4094
6294 044414 104451 TRAP 51
6295 044416 000414 BR 5$ ; 4095
6296 044420 020527 000003 4$: CMP R5,#3 ; VALUE,* 4077
6297 044424 001014 BNE 7$
6298 044426 104455 TRAP 55 ; 4099
6299 044430 000003 .WORD 3
6300 044432 010512 .WORD MSG1
6301 044434 000000 .WORD 0
6302 044436 112764 000006 032464 MOVB #6,WHY.DROPT(R4) ; *,*(LUN) 4100
6303 044444 010400 MOV R4,R0 ; LUN,* 4101
6304 044446 104451 TRAP 51
  
```

Address	Hex	Hex	Hex	Label	Instruction	Comments	Address
6306				:MLX4			23-Oct-1980 15:01:43 TOPS
6307				:	COMMAND INTEGRITY ROUTINE		23-Oct-1980 14:57:05 PA:<
6308							
6309	044450	062706	000012	5\$:	ADD #12,SP		4102
6310	044454	000502		6\$:	BR 8\$		
6311	044456	010416		7\$:	MOV R4,(SP)	: LUN,*	4107
6312	044460	010346			MOV R3,-(SP)	: WRDCNT,*	
6313	044462	012746	020706		MOV #RBUFF,-(SP)		
6314	044466	011246			MOV (R2),-(SP)		
6315	044470	004737	042434		JSR PC,READ		
6316	044474	010005			MOV R0,R5	: *,VALUE	
6317	044476	001072			BNE 9\$		4111
6318	044500	012746	010706		MOV #WBUFF,-(SP)		4114
6319	044504	012746	020706		MOV #RBUFF,-(SP)		
6320	044510	010346			MOV R3,-(SP)	: WRDCNT,*	
6321	044512	004737	041226		JSR PC,DOUBLE.CHECK		
6322	044516	062706	000006		ADD #6,SP		
6323	044522	010066	000032		MOV R0,32(SP)	: *,DBL.VALUE	
6324	044526	001477			BEQ 10\$		
6325	044530	010446			MOV R4,-(SP)	: LUN,*	4117
6326	044532	004737	033436		JSR PC,SAYWHO		
6327	044536	012716	010640		MOV #MSG5,(SP)		4118
6328	044542	012746	006514		MOV #SAY1,-(SP)		
6329	044546	012746	000002		MOV #2,-(SP)		
6330	044552	010600			MOV SP,R0	: SP,*	
6331	044554	104414			TRAP 14		
6332	044556	016616	000040		MOV 40(SP),(SP)	: DBL.VALUE,*	4120
6333	044562	017646	000040		MOV @40(SP),-(SP)	: DBL.VALUE,*	
6334	044566	012746	006332		MOV #FMT12A,-(SP)		
6335	044572	012746	000003		MOV #3,-(SP)		
6336	044576	010600			MOV SP,R0	: SP,*	
6337	044600	104414			TRAP 14		
6338	044602	062766	010000 000046		ADD #10000,46(SP)	: *,DBL.VALUE	4122
6339	044610	016616	000046		MOV 46(SP),(SP)	: DBL.VALUE,*	4123
6340	044614	017646	000046		MOV @46(SP),-(SP)	: DBL.VALUE,*	
6341	044620	012746	006400		MOV #FMT12B,-(SP)		
6342	044624	012746	000003		MOV #3,-(SP)		
6343	044630	010600			MOV SP,R0	: SP,*	
6344	044632	104414			TRAP 14		
6345	044634	112764	000010 032464		MOVB #10,WHY.DROPT(R4)	: *,*(LUN)	4125
6346	044642	104455			TRAP 55		4126
6347	044644	000004			.WORD 4		
6348	044646	010512			.WORD MSG1		
6349	044650	000000			.WORD 0		
6350	044652	010400			MOV R4,R0	: LUN,*	4127
6351	044654	104451			TRAP 51		
6352	044656	062706	000042		ADD #42,SP		4128
6353	044662	000510		8\$:	BR 16\$		
6354	044664	020527	000001	9\$:	CMP R5,#1	: VALUE,*	4111
6355	044670	001033			BNE 12\$		
6356	044672	012746	000006		MOV #6,-(SP)		4132
6357	044676	012746	042434		MOV #READ,-(SP)		
6358	044702	010446			MOV R4,-(SP)	: LUN,*	
6359	044704	010346			MOV R3,-(SP)	: WRDCNT,*	
6360	044706	012746	020706		MOV #RBUFF,-(SP)		

Address	Hex	Hex	Hex	Label	Instruction	Comments	Time	Page
6362				:MLX4			23-Oct-1980 15:01:43	TOPS
6363				:	COMMAND INTEGRITY ROUTINE		23-Oct-1980 14:57:05	PA:<
6364								
6365	044712	011246			MOV (R2),-(SP)			
6366	044714	004737	043046		JSR PC,RETRY			
6367	044720	062706	000014		ADD #14,SP			
6368	044724	005700			TST R0			
6369	044726	001002		10\$:	BNE 11\$			
6370	044730	000137	045454		JMP 24\$			
6371	044734	112764	000004	032464	11\$:	MOVB #4,WHY.DROPT(R4)		4135
6372	044742	104455			TRAP 55			4136
6373	044744	000005			.WORD 5			
6374	044746	010512			.WORD MSG1			
6375	044750	000000			.WORD 0			
6376	044752	010400			MOV R4,R0			4137
6377	044754	104451			TRAP 51			
6378	044756	000450			BR 15\$			4138
6379	044760	020527	000002	12\$:	CMP R5,#2			4111
6380	044764	001012			BNE 13\$			
6381	044766	112764	000005	032464	MOVB #5,WHY.DROPT(R4)			4143
6382	044774	104455			TRAP 55			4144
6383	044776	000006			.WORD 6			
6384	045000	010512			.WORD MSG1			
6385	045002	000000			.WORD 0			
6386	045004	010400			MOV R4,R0			4145
6387	045006	104451			TRAP 51			
6388	045010	000433			BR 15\$			4146
6389	045012	020527	000003	15\$:	CMP R5,#3			4111
6390	045016	001012			BNE 14\$			
6391	045020	104455			TRAP 55			4150
6392	045022	000007			.WORD 7			
6393	045024	010512			.WORD MSG1			
6394	045026	000000			.WORD 0			
6395	045030	112764	000006	032464	MOVB #6,WHY.DROPT(R4)			4151
6396	045036	010400			MOV R4,R0			4152
6397	045040	104451			TRAP 51			
6398	045042	000416			BR 15\$			4153
6399	045044	020527	000004	14\$:	CMP R5,#4			4111
6400	045050	001017			BNE 17\$			
6401	045052	004737	035212		JSR PC,ISOLATE			4157
6402	045056	104455			TRAP 55			4158
6403	045060	000010			.WORD 10			
6404	045062	010542			.WORD MSG2			
6405	045064	000000			.WORD 0			
6406	045066	112764	000007	032464	MOVB #7,WHY.DROPT(R4)			4159
6407	045074	010400			MOV R4,R0			4160
6408	045076	104451			TRAP 51			
6409	045100	062706	000020	15\$:	ADD #20,SP			4161
6410	045104	000137	046256	16\$:	JMP 37\$			
6411	045110	020527	000005	17\$:	CMP R5,#5			4111
6412	045114	001157			BNE 24\$			
6413	045116	004737	035212		JSR PC,ISOLATE			4165
6414	045122	032737	000001	002260	BIT #1,ERROUT			4166
6415	045130	001423			BEQ 18\$			
6416	045132	017700	165270		MOV @ML.REG+42,R0			4167

23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<

```

6418 ;MLX4
6419 ;
6420
6421 045136 006200 ASR R0
6422 045140 006200 ASR R0
6423 045142 006200 ASR R0
6424 045144 006200 ASR R0
6425 045146 006200 ASR R0
6426 045150 006200 ASR R0
6427 045152 042700 177700 BIC #177700,R0
6428 045156 010046 MOV R0,-(SP)
6429 045160 012746 006262 MOV #FMT10B,-(SP)
6430 045164 012746 000002 MOV #2,-(SP)
6431 045170 010600 MOV SP,R0 ; SP,*
6432 045172 104414 TRAP 14
6433 045174 062706 000006 ADD #6,SP
6434 045200 017766 165224 000036 18$: MOV @ML.REG+44,36(SP) ; *,OLDSEC 4169
6435 045206 017700 165214 MOV @ML.REG+42,R0 ; 4170
6436 045212 006200 ASR R0
6437 045214 006200 ASR R0
6438 045216 006200 ASR R0
6439 045220 006200 ASR R0
6440 045222 006200 ASR R0
6441 045224 006200 ASR R0
6442 045226 042700 177700 BIC #177700,R0
6443 045232 010066 000034 MOV R0,34(SP) ; *,OLDCHN
6444 045236 012746 000001 MOV #1,-(SP) ; 4171
6445 045242 012746 042434 MOV #READ,-(SP)
6446 045246 010446 MOV R4,-(SP) ; LUN,*
6447 045250 010346 MOV R3,-(SP) ; WRDCNT,*
6448 045252 012746 020706 MOV #RBUFF,-(SP)
6449 045256 011246 MOV (R2),-(SP)
6450 045260 004737 043046 JSR PC,RETRY
6451 045264 062706 000014 ADD #14,SP
6452 045270 020027 000005 CMP R0,#5
6453 045274 001051 BNE 21$
6454 045276 027766 165126 000036 CMP @ML.REG+44,36(SP) ; *,OLDSEC 4173
6455 045304 001034 BNE 20$
6456 045306 016646 000034 MOV 34(SP),-(SP) ; OLDCHN,*
6457 045312 017700 165110 MOV @ML.REG+42,R0
6458 045316 006200 ASR R0
6459 045320 006200 ASR R0
6460 045322 006200 ASR R0
6461 045324 006200 ASR R0
6462 045326 006200 ASR R0
6463 045330 006200 ASR R0
6464 045332 042700 177700 BIC #177700,R0
6465 045336 020026 CMP R0,(SP)+
6466 045340 001016 BNE 20$
6467 045342 032737 000001 002260 BIT #1,ERROUT ; 4176
6468 045350 001404 BEQ 19$ ; 4178
6469 045352 104456 TRAP 56
6470 045354 000011 .WORD 11
6471 045356 010624 .WORD MSG4
6472 045360 000000 .WORD 0

```

Address	OpCode	Operand 1	Operand 2	Operand 3	Label	Instruction	Comments	Timestamp	Page
6474					;MLX4			23-Oct-1980 15:01:43	TOPS
6475					:	COMMAND INTEGRITY ROUTINE		23-Oct-1980 14:57:05	PA:<
6476									
6477	045362	010446			19\$:	MOV R4,-(SP)	; LUN,*		4179
6478	045364	013746	032360			MOV BOARD,-(SP)			
6479	045370	004737	035302			JSR PC,UP.HARD.COUNT			
6480	045374	000426				BR 23\$	:		4173
6481	045376	032737	000001	002260	20\$:	BIT #1,ERROUT	:		4183
6482	045404	001415				BEQ 22\$			
6483	045406	104457				TRAP 57	:		4185
6484	045410	000012				.WORD 12			
6485	045412	010610				.WORD MSG3			
6486	045414	000000				.WORD 0			
6487	045416	000410				BR 22\$	:		4186
6488	045420	032737	000001	002260	21\$:	BIT #1,ERROUT	:		4190
6489	045426	001404				BEQ 22\$			
6490	045430	104457				TRAP 57	:		4192
6491	045432	000013				.WORD 13			
6492	045434	010610				.WORD MSG3			
6493	045436	000000				.WORD 0			
6494	045440	010446			22\$:	MOV R4,-(SP)	; LUN,*		4193
6495	045442	013746	032360			MOV BOARD,-(SP)			
6496	045446	004737	035430			JSR PC,UP.SOFT.COUNT			
6497	045452	022626			23\$:	CMP (SP)+,(SP)+	:		4164
6498	045454	010416			24\$:	MOV R4,(SP)	; LUN,*		4199
6499	045456	010346				MOV R3,-(SP)	; WRDCNT,*		
6500	045460	012746	010706			MOV #WBUFF,-(SP)			
6501	045464	011246				MOV (R2)-,(SP)			
6502	045466	004737	042622			JSR PC,CHECK			
6503	045472	010005				MOV R0,R5	; *,VALUE		
6504	045474	020527	000001			CMP R5,#1	; VALUE,*		4203
6505	045500	001031				BNE 25\$			
6506	045502	012746	000006			MOV #6,-(SP)	:		4207
6507	045506	012746	042622			MOV #CHECK,-(SP)			
6508	045512	010446				MOV R4,-(SP)	; LUN,*		
6509	045514	010346				MOV R3,-(SP)	; WRDCNT,*		
6510	045516	012746	010706			MOV #WBUFF,-(SP)			
6511	045522	011246				MOV (R2)-,(SP)			
6512	045524	004737	043046			JSR PC,RETRY			
6513	045530	062706	000014			ADD #14,SP			
6514	045534	005700				TST R0			
6515	045536	001462				BEQ 28\$			
6516	045540	112764	000004	032464		MOVB #4,WHY.DROPT(R4)	; *,*(LUN)		4210
6517	045546	104455				TRAP 55	:		4211
6518	045550	000014				.WORD 14			
6519	045552	010512				.WORD MSG1			
6520	045554	000000				.WORD 0			
6521	045556	010400				MOV R4,R0	; LUN,*		4212
6522	045560	104451				TRAP 51			
6523	045562	000450				BR 28\$	:		4213
6524	045564	020527	000002		25\$:	CMP R5,#2	; VALUE,*		4203
6525	045570	001012				BNE 26\$			
6526	045572	112764	000005	032464		MOVB #5,WHY.DROPT(R4)	; *,*(LUN)		4218
6527	045600	104455				TRAP 55	:		4219
6528	045602	000015				.WORD 15			



Address	Hex	Dec	Label	Op	Opnd	Comments	Seq		
6530			:MLX4						
6531			:						
6532									
6533	045604	010512		.WORD	MSG1				
6534	045606	000000		.WORD	0				
6535	045610	010400		MOV	R4,R0	: LUN,*	4220		
6536	045612	104451		TRAP	51				
6537	045614	000433		BR	28\$	:	4221		
6538	045616	020527	000003	26\$:	CMP	R5,#3	: VALUE,*	4203	
6539	045622	001012		BNE	27\$	:			
6540	045624	104455		TRAP	55	:	4225		
6541	045626	000016		.WORD	16				
6542	045630	010512		.WORD	MSG1				
6543	045632	000000		.WORD	0				
6544	045634	112764	000006	032464	MOVB	#6,WHY.DROPT(R4)	: *,*(LUN)	4226	
6545	045642	010400		MOV	R4,R0	: LUN,*	4227		
6546	045644	104451		TRAP	51				
6547	045646	000416		BR	28\$	:	4228		
6548	045650	020527	000004	27\$:	CMP	R5,#4	: VALUE,*	4203	
6549	045654	001014		BNE	29\$	:			
6550	045656	004737	035212		JSR	PC,ISOLATE	:	4232	
6551	045662	104455		TRAP	55	:	4233		
6552	045664	000017		.WORD	17				
6553	045666	010542		.WORD	MSG2				
6554	045670	000000		.WORD	0				
6555	045672	112764	000007	032464	MOVB	#7,WHY.DROPT(R4)	: *,*(LUN)	4234	
6556	045700	010400		MOV	R4,R0	: LUN,*	4235		
6557	045702	104451		TRAP	51				
6558	045704	000562		BR	36\$	:	4236		
6559	045706	020527	000005	28\$: 29\$:	CMP	R5,#5	: VALUE,*	4203	
6560	045712	001157		BNE	36\$	:			
6561	045714	004737	035212		JSR	PC,ISOLATE	:	4240	
6562	045720	032737	000001	002260	BIT	#1,ERROUT	:	4241	
6563	045726	001423		BEQ	30\$	:			
6564	045730	017700	164472		MOV	@ML.REG+42,R0	:	4242	
6565	045734	006200		ASR	R0				
6566	045736	006200		ASR	R0				
6567	045740	006200		ASR	R0				
6568	045742	006200		ASR	R0				
6569	045744	006200		ASR	R0				
6570	045746	006200		ASR	R0				
6571	045750	042700	177700		BIC	#177700,R0			
6572	045754	010046		MOV	R0,-(SP)				
6573	045756	012746	006262		MOV	#FMT10B,-(SP)			
6574	045762	012746	000002		MOV	#2,-(SP)			
6575	045766	010600		MOV	SP,R0	: SP,*			
6576	045770	104414		TRAP	14				
6577	045772	062706	000006		ADD	#6,SP			
6578	045776	017766	164426	000044	30\$:	MOV	@ML.REG+44,44(SP)	: *,OLDSEC	4244
6579	046004	017700	164416		MOV	@ML.REG+42,R0	:	4245	
6580	046010	006200		ASR	R0				
6581	046012	006200		ASR	R0				
6582	046014	006200		ASR	R0				
6583	046016	006200		ASR	R0				
6584	046020	006200		ASR	R0				

Address	Instruction	Operand 1	Operand 2	Operand 3	Operand 4	Label	Comment	Time	Page
6586							COMMAND INTEGRITY ROUTINE	23-Oct-1980 15:01:43	TOPS
6587								23-Oct-1980 14:57:05	PA:<
6588									
6589	046022	006200				ASR	R0		
6590	046024	042700	177700			BIC	#177700,R0		
6591	046030	010066	000042			MOV	R0,42(SP)	; *.OLDCHN	
6592	046034	012746	000001			MOV	#1,-(SP)	:	4246
6593	046040	012746	042622			MOV	#CHECK,-(SP)		
6594	046044	010446				MOV	R4,-(SP)	; LUN,*	
6595	046046	010346				MOV	R3,-(SP)	; WRDCNT,*	
6596	046050	012746	010706			MOV	#WBUFF,-(SP)		
6597	046054	011246				MOV	(R2),-(SP)		
6598	046056	004737	043046			JSR	PC,RETRY		
6599	046062	062706	000014			ADD	#14,SP		
6600	046066	020027	000005			CMP	R0,#5		
6601	046072	001051				BNE	33\$		
6602	046074	027766	164330	000044		CMP	@ML.REG+44,44(SP)	; *.OLDSEC	4248
6603	046102	001034				BNE	32\$		
6604	046104	016600	000042			MOV	42(SP),R0	; OLDCHN,*	
6605	046110	017702	164312			MOV	@ML.REG+42,R2		
6606	046114	006202				ASR	R2		
6607	046116	006202				ASR	R2		
6608	046120	006202				ASR	R2		
6609	046122	006202				ASR	R2		
6610	046124	006202				ASR	R2		
6611	046126	006202				ASR	R2		
6612	046130	042702	177700			BIC	#177700,R2		
6613	046134	020200				CMP	R2,R0		
6614	046136	001016				BNE	32\$		
6615	046140	032737	000001	002260		BIT	#1,ERRROUT	:	4251
6616	046146	001404				BEQ	31\$		
6617	046150	104456				TRAP	56	:	4253
6618	046152	000020				.WORD	20		
6619	046154	010624				.WORD	MSG4		
6620	046156	000000				.WORD	0		
6621	046160	010446			31\$:	MOV	R4,-(SP)	; LUN,*	4254
6622	046162	013746	032360			MOV	BOARD,-(SP)		
6623	046166	004737	035302			JSR	PC,UP.HARD.COUNT		
6624	046172	000426				BR	35\$	:	4248
6625	046174	032737	000001	002260	32\$:	BIT	#1,ERRROUT	:	4258
6626	046202	001415				BEQ	34\$		
6627	046204	104457				TRAP	57	:	4260
6628	046206	000021				.WORD	21		
6629	046210	010610				.WORD	MSG3		
6630	046212	000000				.WORD	0		
6631	046214	000410				BR	34\$	:	4261
6632	046216	032737	000001	002260	33\$:	BIT	#1,ERRROUT	:	4265
6633	046224	001404				BEQ	34\$		
6634	046226	104457				TRAP	57	:	4267
6635	046230	000022				.WORD	22		
6636	046232	010610				.WORD	MSG3		
6637	046234	000000				.WORD	0		
6638	046236	010446			34\$:	MOV	R4,-(SP)	; LUN,*	4268
6639	046240	013746	032360			MOV	BOARD,-(SP)		
6640	046244	004737	035430			JSR	PC,UP.SOFT.COUNT		

6642				;MLX4			23-Oct-1980 15:01:43	TOPS
6643				:	COMMAND INTEGRITY ROUTINE		23-Oct-1980 14:57:05	PA:<
6644								
6645	046250	022626		35\$:	CMP	(SP)+,(SP)+		4239
6646	046252	062706	000026	36\$:	ADD	#26,SP		4068
6647	046256	005204		37\$:	INC	R4	: LUN	4062
6648	046260	020466	000020	38\$:	CMP	R4,20(SP)	: LUN,*	
6649	046264	002002			BGE	39\$		
6650	046266	000137	044150		JMP	2\$		
6651	046272	005726		39\$:	TST	(SP)+		4060
6652	046274	005201			INC	R1	: COMP.FLAG	4059
6653	046276	020127	000001		CMP	R1,#1	: COMP.FLAG,*	
6654	046302	003002			BGT	40\$		
6655	046304	000137	044126		JMP	1\$		
6656	046310	062706	000022	40\$:	ADD	#22,SP		4010
6657	046314	000207			RTS	PC		
6658								
6659								
6660								
6665								
6666								

; Routine Size: 587 words  
 ; Maximum stack depth per invocation: 33 words

```

6668 :MLX4
6669 :
6670 :
6671 : 4280 %SBTTL 'DEFINITION OF OPTION 1'
6672 : 4281
6673 : 4282 ROUTINE OPT1: NOVALUE =
6674 : 4283 BEGIN !* 1 * START OF ROUTINE
6675 : 4284
6676 : 4285 !**
6677 : 4286 ROUTINE: OPT1
6678 : 4287
6679 : 4288 PURPOSE: TO CHECK ADDRESSES USING DATA = SECTOR NUMBER.
6680 : 4289 TRANSFERS ARE 4K WORDS IN LENGTH, AND ALL SECTORS
6681 : 4290 ARE TESTED.
6682 : 4291
6683 : 4292 THE CODE FOR 'OPT1' IN BRIEF:
6684 : 4293
6685 : 4294 BEGIN 1 (START OF ROUTINE)
6686 : 4295 SAY ROUTINE IS RUNNING
6687 : 4296 INCR COMPLEMENT FLAG FROM 0 TO 1
6688 : 4297 : BEGIN 2 (START OF COMPLEMENT FLAG SELECTION LOOP)
6689 : 4298 : INCR LOGICAL UNIT FROM 0 TO LAST
6690 : 4299 : : BEGIN 3 (START OF LOGICAL UNIT SELECTION LOOP)
6691 : 4300 : : TESTLOOP:
6692 : 4301 : : : BEGIN 4 (START OF LOOP THAT COMPLETELY TESTS ONE UNIT)
6693 : 4302 : : : IF UNIT IS ACTIVE
6694 : 4303 : : : THEN
6695 : 4304 : : : : BEGIN 5 (START OF TEST FOR AN ACTIVE UNIT)
6696 : 4305 : : : : INITIALIZE WRITE AND READ BUFFER POINTERS
6697 : 4306 : : : : SECTOR = LOWEST
6698 : 4307 : : : : WHILE SECTOR LEQ HIGHEST DO
6699 : 4308 : : : : : BEGIN 6 (START OF SECTOR SELECTION LOOP)
6700 : 4309 : : : : : GET WRDCNT
6701 : 4310 : : : : : GENERATE THE PATTERN
6702 : 4311 : : : : : SET UP BUFFER POINTERS BEFORE TRANSFER
6703 : 4312 : : : : : WRITE
6704 : 4313 : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
6705 : 4314 : : : : : CHOOSE WHETHER TO WRITE CHECK OR READ
6706 : 4315 : : : : : DO THE WRITE CHECK OR READ
6707 : 4316 : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
6708 : 4317 : : : : : CHANGE BUFFER POINTERS AFTER TRANSFER
6709 : 4318 : : : : : UPDATE SECTOR NUMBER BY NUMBER OF SECTORS IN PREVIOUS TRANSFER
6710 : 4319 : : : : : END 6 (END OF SECTOR SELECTION LOOP)
6711 : 4320 : : : : : END 5 (END OF TEST FOR AN ACTIVE UNIT)
6712 : 4321 : : : : : END 4 (END OF TESTLOOP)
6713 : 4322 : : : : : END 3 (END OF LOGICAL UNIT SELECTION LOOP)
6714 : 4323 : : : : : END 2 (END OF COMPLEMENT FLAG SELECTION LOOP)
6715 : 4324 : : : : : RETURN
6716 : 4325 : : : : : END 1 (END OF ROUTINE)
6717 : 4326 : : : : : --
6718 : 4327 : : : : :
6719 : 4328 LABEL
6720 : 4329 LOOP:
6721 : 4330
6722 : 4331 LOCAL
    
```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (40)

```

6724 :MLX4
6725 :
6726 :
6727 : 4332 VALUE, WRDCNT, COMMAND, PTR, OLDSEC, OLDCHN, SECTOR, DBL_VALUE;
6728 : 4333
6729 : 4334 PRINTB(SAY2,WRD34,RTN1);
6730 : 4335 !'RUNNING OPT1'
6731 : 4336
6732 : 4337 INCR COMP_FLAG FROM 0 TO 1 DO
6733 : 4338 BEGIN !* 2 * START OF COMPLEMENT FLAG SELECTION LOOP
6734 : 4339 INCR LUN FROM 0 TO (.L$UNIT - 1) DO
6735 : 4340 BEGIN !* 3 * START OF LOGICAL UNIT SELECTION LOOP
6736 : 4341 LOOP:
6737 : 4342 BEGIN !* 4 * START OF THE LOOP THAT COMPLETELY TESTS ONE UNIT
6738 : 4343 IF .DRIVE_STATUS[LUN] EQL ACTIVE
6739 : 4344 THEN
6740 : 4345 BEGIN !* 5 * START OF TEST FOR AN ACTIVE UNIT
6741 : 4346 L$LUN = .LUN;
6742 : 4347 WPTR = WBUFF;
6743 : 4348 RPTR = RBUFF;
6744 : 4349 SECTOR = LOWEST;
6745 : 4350 WHILE .SECTOR LEQ HIGHEST DO
6746 : 4351 BEGIN !* 6 * START OF SECTOR SELECTION LOOP
6747 : 4352 GEN1(.SECTOR,.COMP_FLAG);
6748 : 4353 WRDCNT = GET WRDCNT(.SECTOR,HIGHEST);
6749 : 4354 GET_PTRS(.WRDCNT);
6750 : 4355
6751 : 4356 VALUE = WRITE(.LUN,.WRDCNT,.WPTR,.SECTOR);
6752 : 4357 !+
6753 : 4358 ! SEE HOW SUCCESSFUL THE WRITE WAS:
6754 : 4359 !-
6755 : 4360 SELECTONE .VALUE OF !SEE 'SYSERR' FOR DEFINITION
6756 : 4361 SET !OF ERROR # CONTAINED IN 'VALUE'
6757 : 4362 [1] :
6758 : 4363 BEGIN !* 6A * RETRY ALLOWED
6759 : 4364 IF RETRY(SIX,WRITE,.LUN,.WRDCNT,.WPTR,.SECTOR) NEQ 0
6760 : 4365 THEN !THE RETRY FAILED -- SYSTEM FATAL ERROR
6761 : 4366 BEGIN
6762 : 4367 WHY DROPT[LUN] = CODE_4;
6763 : 4368 ERRDF(101,MSG1,0); !**** OPTION 1 ERROR 01 ****
6764 : 4369 DODU(.LUN);
6765 : 4370 LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
6766 : 4371 END;
6767 : 4372 END; !* 6A *
6768 : 4373 [2] :
6769 : 4374 BEGIN !* 6B * FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
6770 : 4375 WHY DROPT[LUN] = CODE_5;
6771 : 4376 ERRDF(102,MSG1,0); !**** OPTION 1 ERROR 02 ****
6772 : 4377 DODU(.LUN);
6773 : 4378 LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
6774 : 4379 END; !* 6B *
6775 : 4380 [3] :
6776 : 4381 BEGIN !* 6C * FATAL DRIVE ERROR -- NO RETRY ALLOWED
6777 : 4382 ERRDF(103,MSG1,0); !**** OPTION 1 ERROR 03 ****
6778 : 4383 WHY_DROPT[LUN] = CODE_6;

```

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (40)

```

6780 :MLX4
6781 :
6782 :
6783 : 4384
6784 : 4385
6785 : 4386
6786 : 4387
6787 : 4388
6788 : 4389
6789 : 4390
6790 : 4391
6791 : 4392
6792 : 4393
6793 : 4394
6794 : 4395
6795 : 4396
6796 : 4397
6797 : 4398
6798 : 4399
6799 : 4400
6800 : 4401
6801 : 4402
6802 : 4403
6803 : 4404
6804 : 4405
6805 : 4406
6806 : 4407
6807 : 4408
6808 : 4409
6809 : 4410
6810 : 4411
6811 : 4412
6812 : 4413
6813 : 4414
6814 : 4415
6815 : 4416
6816 : 4417
6817 : 4418
6818 : 4419
6819 : 4420
6820 : 4421
6821 : 4422
6822 : 4423
6823 : 4424
6824 : 4425
6825 : 4426
6826 : 4427
6827 : 4428
6828 : 4429
6829 : 4430
6830 : 4431
6831 : 4432
6832 : 4433
6833 : 4434
6834 : 4435

DEFINITION OF OPTION 1

DODU(.LUN);
LEAVE LOOP;          !JUMP JUST BEYOND END OF BLOCK * 4 *
END;  !* 6C *

TES;

COMMAND = CHOOSE();
IF .COMMAND EQL READ
THEN
BEGIN
PTR = .RPTR;
VALUE = READ(.LUN,.WRDCNT,.RPTR,.SECTOR);
END
ELSE
BEGIN
PTR = .WPTR;
VALUE = CHECK(.LUN,.WRDCNT,.WPTR,.SECTOR);
END;

!+
!-
SEE HOW SUCCESSFUL THE OPERATION WAS:
SFLECTONE .VALUE OF      !SEE 'SYSERR' FOR DEFINITION
SET                      OF ERROR # CONTAINED IN 'VALUE'
[0] :
IF .COMMAND EQL READ
THEN
BEGIN
IF (DBL_VALUE = DOUBLE_CHECK(.WPTR,.RPTR,.WRDCNT)) NEQ 0
THEN
BEGIN
SAYWHO(.LUN);
PRINTB(SAY1,MSG5);
!'ECC LOGIC FAILED TO DETECT DATA ERROR'
PRINTB(FMT12A,..DBL_VALUE,..DBL_VALUE);
!'GOOD DATA: XXXXXX AT LOCATION YYYYYY'
DBL_VALUE = .DBL_VALUE + BUFSIZ * 2;
PRINTB(FMT12B,..DBL_VALUE,..DBL_VALUE);
!'BAD DATA: PPPPPP AT LOCATION QQQQQQ'
WHY DROPT[.LUN] = CODE 8;
ERRDF(104,MSG1,0);  !**** OPTION 1 ERROR 04 ****
DODU(.LUN);
LEAVE LOOP;          !JUMP JUST BEYOND END OF BLOCK * 4 *
END;
END;
[1] :
BEGIN !* 6D *          RETRY ALLOWED
IF RETRY(SIX,.COMMAND,.LUN,.WRDCNT,.PTR,.SECTOR) NEQ 0
THEN !THE RETRY FAILED -- SYSTEM FATAL ERROR
BEGIN
WHY DROPT[.LUN] = CODE 4;
ERRDF(105,MSG1,0);  !**** OPTION 1 ERROR 05 ****
DODU(.LUN);

```

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (40)

```

6836 :MLX4
6837 :
6838 :
6839 : 4436 LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
6840 : 4437 END;
6841 : 4438 END; !* 6D *
6842 : 4439 [2] :
6843 : 4440 BEGIN !* 6E * FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
6844 : 4441 WHY DROPT[.LUN] = CODE_5;
6845 : 4442 ERRDF(106,MSG1,0); !**** OPTION 1 ERROR 06 ****
6846 : 4443 DODU(.LUN);
6847 : 4444 LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
6848 : 4445 END; !* 6E *
6849 : 4446 [3] :
6850 : 4447 BEGIN !* 6F * FATAL DRIVE ERROR -- NO RETRY ALLOWED
6851 : 4448 WHY DROPT[.LUN] = CODE_6;
6852 : 4449 ERRDF(107,MSG1,0); !**** OPTION 1 ERROR 07 ****
6853 : 4450 DODU(.LUN);
6854 : 4451 LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
6855 : 4452 END; !* 6F *
6856 : 4453 [4] :
6857 : 4454 BEGIN !* 6G * UNRECOVERABLE DATA ERROR
6858 : 4455 ISOLATE();
6859 : 4456 ERRDF(108,MSG2,0); !**** OPTION 1 ERROR 08 ****
6860 : 4457 WHY DROPT[.LUN] = CODE_7;
6861 : 4458 DODU(.LUN);
6862 : 4459 LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
6863 : 4460 END; !* 6G *
6864 : 4461 [5] :
6865 : 4462 BEGIN !* 6H * RECOVERABLE DATA ERROR
6866 : 4463 ISOLATE();
6867 : 4464 IF .ERROUT
6868 : 4465 THEN PRINTB(FMT10B,.CHAN);
6869 : 4466 !' BIT 00'
6870 : 4467 OLDSEC = .MLEL;
6871 : 4468 OLDCHN = .CHAN;
6872 : 4469 IF RETRY(ONE,.COMMAND,.LUN,.WRDCNT,.PTR,.SECTOR) EQL 5
6873 : 4470 THEN
6874 : 4471 IF ((.MLEL EQL .OLDSEC) AND (.CHAN EQL .OLDCHN))
6875 : 4472 THEN
6876 : 4473 BEGIN
6877 : 4474 IF .ERROUT
6878 : 4475 THEN
6879 : 4476 ERRHRD(109,MSG4,0); !**** OPTION 1 ERROR 09 ****
6880 : 4477 UP_HARD_COUNT(.LUN,.BOARD);
6881 : 4478 END
6882 : 4479 ELSE
6883 : 4480 BEGIN
6884 : 4481 IF .ERROUT
6885 : 4482 THEN
6886 : 4483 ERRSOFT(110,MSG3,0); !**** OPTION 1 ERROR 10 ****
6887 : 4484 UP_SOFT_COUNT(.LUN,.BOARD);
6888 : 4485 END
6889 : 4486 ELSE
6890 : 4487 BEGIN

```

```

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)
23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (40)

```

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (40)

```

6892 ;MLX4
6893 ;      DEFINITION OF OPTION 1
6894
6895 :      4488          IF .ERROUT
6896 :      4489          THEN
6897 :      4490          ERRSOFT(111,MSG3,0); !*** OPTION 1 ERROR 11 ***
6898 :      4491          UP SOFT_COUNT(.LUN,.BOARD);
6899 :      4492          END;
6900 :      4493          END; !* 6H *
6901 :      4494
6902 :      4495          TES;
6903 :      4496
6904 :      4497          WPTR = .WPTR + (.WRDCNT * 2);
6905 :      4498          SECTOR = .SECTOR + (.WRDCNT/256);
6906 :      4499          END; !* 6 * END OF SECTOR SELECTION LOOP
6907 :      4500          END; !* 5 * END OF TEST FOR AN ACTIVE UNIT
6908 :      4501          END; !* 4 * END OF LOOP THAT COMPLETELY TESTS ONE UNIT
6909 :      4502          END; !* 3 * END OF LOGICAL UNIT SELECTION LOOP
6910 :      4503          END; !* 2 * END OF COMPLEMENT FLAG SELECTION LOOP
6911 :      4504          RETURN;
6912 :      4505          END; !* 1 * END OF ROUTINE
6916
6917

```

```

6921 046316 004137 005222      OPT1: .SBTTL OPT1 DEFINITION OF OPTION 1
6922 046322 162706 000020      JSR      R1,$SAVE5 ; 4282
6923 046326 012746 007250      SUB      #20,SP ;
6924 046332 012746 006766      MOV      #RTN1,-(SP) ; 4334
6925 046336 012746 006522      MOV      #WRD34,-(SP)
6926 046342 012746 000003      MOV      #SAY2,-(SP)
6927 046346 010600      MOV      #3,-(SP)
6928 046350 104414      MOV      SP,RO ; SP,*
6929 046352 005066 000026      TRAP    14
6930 046356 013766 002012 000012 1$: CLR      26(SP) ; COMP.FLAG 4337
6931 046364 005066 000010      MOV      L$UNIT,12(SP) ; 4339
6932 046370 000137 050170      CLR      10(SP) ; LUN
6933 046374 016600 000010 2$: JMP      29$
6934 046400 006200      MOV      10(SP),RO ; LUN,* 4343
6935 046402 006200      ASR      RO
6936 046404 006200      ASR      RO
6937 046406 062700 032460      ASR      RO
6938 046412 010046      ADD      #DRIVE.STATUS,RO
6939 046414 016646 000012      MOV      RO,-(SP)
6940 0 5420 042716 177770      MOV      12(SP),-(SP) ; LUN,*
6941 046424 012746 000001      BIC      #177770,(SP)
6942 046430 005046      MOV      #1,-(SP)
6943 046432 004737 004244      CLR      -(SP)
6944 046436 062706 000010      JSR      PC,BL$GT2
6945 046442 005300      ADD      #10,SP
6946 046444 001150      DEC      RO
        BNE      7$

```





Address	OpCode	Operand1	Operand2	Operand3	Label	Instruction	Comments	Line No.
7004					;MLX4			
7005					;			
7006						DEFINITION OF OPTION 1		
7007	046716	016600	000024			MOV 24(SP),R0	; LUN,*	4377
7008	046722	104451				TRAP 51		
7009	046724	000416				BR 6\$		4378
7010	046726	020427	000003	5\$:		CMP R4,#3	; VALUE,*	4360
7011	046732	001016				BNE 8\$		
7012	046734	104455				TRAP 55		4382
7013	046736	000147				.WORD 147		
7014	046740	010512				.WORD MSG1		
7015	046742	000000				.WORD 0		
7016	046744	016602	000024			MOV 24(SP),R2	; LUN,*	4383
7017	046750	112762	000006	032464		MOVB #6,WHY.DROPT(R2)		
7018	046756	010200				MOV R2,R0	; LUN,*	4384
7019	046760	104451				TRAP 51		
7020	046762	062706	000014	6\$:		ADD #14,SP		4385
7021	046766	000543		7\$:		BR 13\$		
7022	046770	004737	043010	8\$:		JSR PC,CHOOSE		4390
7023	046774	010066	000034			MOV R0,34(SP)	; *,COMMAND	
7024	047000	005002				CLR R2		4391
7025	047002	020027	042434			CMP R0,#READ	; COMMAND,*	
7026	047006	001015				BNE 9\$		
7027	047010	005202				INC R2		
7028	047012	013766	030710	000036		MOV RPTR,36(SP)	; *,PTR	4394
7029	047020	016646	000024			MOV 24(SP),-(SP)	; LUN,*	4395
7030	047024	010346				MOV R3,-(SP)	; WRDCNT,*	
7031	047026	013746	030710			MOV RPTR,-(SP)		
7032	047032	010546				MOV R5,-(SP)	; SECTOR,*	
7033	047034	004737	042434			JSR PC,READ		
7034	047040	000413				BR 10\$		
7035	047042	013766	030706	000036	9\$:	MOV WPTR,36(SP)	; *,PTR	4399
7036	047050	016646	000024			MOV 24(SP),-(SP)	; LUN,*	4400
7037	047054	010346				MOV R3,-(SP)	; WRDCNT,*	
7038	047056	013746	030706			MOV WPTR,-(SP)		
7039	047062	010546				MOV R5,-(SP)	; SECTOR,*	
7040	047064	004737	042622			JSR PC,CHECK		
7041	047070	010004		10\$:		MOV R0,4	; *,VALUE	
7042	047072	001102				BNE 14\$		4405
7043	047074	006002				ROR R2		4408
7044	047076	103402				BLO 12\$		
7045	047100	000137	050122	11\$:		JMP 27\$		
7046	047104	013746	030706	12\$:		MOV WPTR,-(SP)		4411
7047	047110	013746	030710			MOV RPTR,-(SP)		
7048	047114	010346				MOV R3,-(SP)	; WRDCNT,*	
7049	047116	004737	041226			JSR PC,DOUBLE.CHECK		
7050	047122	062706	000006			ADD #6,SP		
7051	047126	010066	000050			MOV R0,50(SP)	; *,DBL.VALUE	
7052	047132	001742				BEQ 11\$		
7053	047134	016646	000034			MOV 34(SP),-(SP)	; LUN,*	4414
7054	047140	004737	033436			JSR PC,SAYWHO		
7055	047144	012716	010640			MOV #MSG5,(SP)		4415
7056	047150	012746	006514			MOV #SAY1,-(SP)		
7057	047154	012746	000002			MOV #2,-(SP)		
7058	047160	010600				MOV SP,R0	; SP,*	

Address	Offset	Value	Label	Instruction	Comments	Line #	
7060			:MLX4				
7061			:				
7062							
7063	047162	104414		TRAP 14			
7064	047164	016616	000056	MOV 56(SP),(SP)	: DBL.VALUE,*	4417	
7065	047170	017646	000056	MOV @56(SP),-(SP)	: DBL.VALUE,*		
7066	047174	012746	006332	MOV #FMT12A,-(SP)			
7067	047200	012746	000003	MOV #3,-(SP)			
7068	047204	010600		MOV SP,R0	: SP,*		
7069	047206	104414		TRAP 14			
7070	047210	062766	010000 000064	ADD #10000,64(SP)	: *,DBL.VALUE	4419	
7071	047216	016616	000064	MOV 64(SP),(SP)	: DBL.VALUE,*	4420	
7072	047222	017646	000064	MOV @64(SP),-(SP)	: DBL.VALUE,*		
7073	047226	012746	006400	MOV #FMT12B,-(SP)			
7074	047232	012746	000003	MOV #3,-(SP)			
7075	047236	010600		MOV SP,R0	: SP,*		
7076	047240	104414		TRAP 14			
7077	047242	016602	000056	MOV 56(SP),R2	: LUN,*	4422	
7078	047246	112762	000010 032464	MOVB #10,WHY.DROPT(R2)			
7079	047254	104455		TRAP 55	:	4423	
7080	047256	000150		.WORD 150			
7081	047260	010512		.WORD MSG1			
7082	047262	000000		.WORD 0			
7083	047264	016600	000056	MOV 56(SP),R0	: LUN,*	4424	
7084	047270	104451		TRAP 51			
7085	047272	062706	000046	ADD #46,SP	:	4425	
7086	047276	000522	13\$:	BR 19\$			
7087	047300	020427	000001	14\$:	CMP R4,#1	: VALUE,*	4405
7088	047304	001035		BNE 15\$			
7089	047306	012746	000006	MOV #6,-(SP)	:	4430	
7090	047312	016646	000046	MOV 46(SP),-(SP)	: COMMAND,*		
7091	047316	016646	000040	MOV 40(SP),-(SP)	: LUN,*		
7092	047322	010346		MOV R3,-(SP)	: WRDCNT,*		
7093	047324	016646	000056	MOV 56(SP),-(SP)	: PTR,*		
7094	047330	010546		MOV R5,-(SP)	: SECTOR,*		
7095	047332	004737	043046	JSR PC,RETRY			
7096	047336	062706	000014	ADD #14,SP			
7097	047342	005700		TST R0			
7098	047344	001655		BEQ 11\$			
7099	047346	016602	000034	MOV 34(SP),R2	: LUN,*	4433	
7100	047352	112762	000004 032464	MOVB #4,WHY.DROPT(R2)			
7101	047360	104455		TRAP 55	:	4434	
7102	047362	000151		.WORD 151			
7103	047364	010512		.WORD MSG1			
7104	047366	000000		.WORD 0			
7105	047370	016600	000034	MOV 34(SP),R0	: LUN,*	4435	
7106	047374	104451		TRAP 51			
7107	047376	000460		BR 18\$	:	4436	
7108	047400	020427	000002	15\$:	CMP R4,#2	: VALUE,*	4405
7109	047404	001015		BNE 16\$			
7110	047406	016602	000034	MOV 34(SP),R2	: LUN,*	4441	
7111	047412	112762	000005 032464	MOVB #5,WHY.DROPT(R2)			
7112	047420	104455		TRAP 55	:	4442	
7113	047422	000152		.WORD 152			
7114	047424	010512		.WORD MSG1			

						23-Oct-1980 15:01:43	TOPS
						23-Oct-1980 14:57:05	PA:<
7116				:MLX4			
7117				:	DEFINITION OF OPTION 1		
7118							
7119	047426	000000			.WCRD 0		
7120	047430	016600	000034		MOV 34(SP),R0	: LUN,*	4443
7121	047434	104451			TRAP 51		
7122	047436	000440			BR 18\$	: VALUE,*	4444
7123	047440	020427	000003	16\$:	CMP R4,#3		4405
7124	047444	001015			BNE 17\$		
7125	047446	016602	000034		MOV 34(SP),R2	: LUN,*	4448
7126	047452	112762	000006	032464	MOVB #6,WHY.DROPT(R2)		
7127	047460	104455			TRAP 55		4449
7128	047462	000153			.WORD 153		
7129	047464	010512			.WORD MSG1		
7130	047466	000000			.WORD 0		
7131	047470	016600	000034		MOV 34(SP),R0	: LUN,*	4450
7132	047474	104451			TRAP 51		
7133	047476	000420			BR 18\$	: VALUE,*	4451
7134	047500	020427	000004	17\$:	CMP R4,#4		4405
7135	047504	001021			BNE 20\$		
7136	047506	004737	035212		JSR PC,ISOLATE		4455
7137	047512	104455			TRAP 55		4456
7138	047514	000154			.WORD 154		
7139	047516	010542			.WORD MSG2		
7140	047520	000000			.WORD 0		
7141	047522	016602	000034		MOV 34(SP),R2	: LUN,*	4457
7142	047526	112762	000007	032464	MOVB #7,WHY.DROPT(R2)		
7143	047534	010200			MOV R2,R0	: LUN,*	4458
7144	047536	104451			TRAP 51		
7145	047540	062706	000024	18\$:	ADD #24,SP		4459
7146	047544	000137	050164	19\$:	JMP 28\$		
7147	047550	020427	000005	20\$:	CMP R4,#5	: VALUE,*	4405
7148	047554	001162			BNE 27\$		
7149	047556	004737	035212		JSR PC,ISOLATE		4463
7150	047562	032737	000001	002260	BIT #1,ERROUT		4464
7151	047570	001423			BEQ 21\$		
7152	047572	017702	162630		MOV @ML.REG+42,R2		4465
7153	047576	006202			ASR R2		
7154	047600	006202			ASR R2		
7155	047602	006202			ASR R2		
7156	047604	006202			ASR R2		
7157	047606	006202			ASR R2		
7158	047610	006202			ASR R2		
7159	047612	042702	177700		BIC #177700,R2		
7160	047616	010246			MOV R2,-(SP)		
7161	047620	012746	006262		MOV #FMT10B,-(SP)		
7162	047624	012746	000002		MOV #2,-(SP)		
7163	047630	010600			MOV SP,R0	: SP,*	
7164	047632	104414			TRAP 14		
7165	047634	062706	000006		ADD #6,SP		
7166	047640	017766	162564	000040	MOV @ML.REG+44,40(SP)	: *,OLDSEC	4467
7167	047646	017702	162554		MOV @ML.REG+42,R2		4468
7168	047652	006202			ASR R2		
7169	047654	006202			ASR R2		
7170	047656	006202			ASR R2		

```

7172          :MLX4
7173          :
7174          :
7175 047660 006202          ASR      R2
7176 047662 006202          ASR      R2
7177 047654 006202          ASR      R2
7178 047666 042702 177700  B!C      #177700,R2
7179 047672 010266 000042  MOV      R2,42(SP)           : *,OLDCHN
7180 047676 012746 000001  MOV      #1,-(SP)           :
7181 047702 016646 000046  MOV      46(SP),-(SP)      : COMMAND,*
7182 047706 016646 000040  MOV      40(SP),-(SP)      : LUN,*
7183 047712 010346          MOV      R3,-(SP)           : WRDCNT,*
7184 047714 016646 000056  MOV      56(SP),-(SP)      : PTR,*
7185 047720 010546          MOV      R5,-(SP)           : SECTOR,*
7186 047722 004737 043046  JSR      PC,RETRY
7187 047726 062706 000014  ADD      #14,SP
7188 047732 020027 000005  CMP      R0,#5
7189 047736 001052          BNE      24$
7190 047740 027766 162464 000040  CMP      @ML.REG+44,40(SP)   : *,OLDSEC
7191 047746 001035          BNE      23$
7192 047750 016600 000042  MOV      42(SP),R0          : OLDCHN,*
7193 047754 017702 162446  MOV      @ML.REG+42,R2
7194 047760 006202          ASR      R2
7195 047762 006202          ASR      R2
7196 047764 006202          ASR      R2
7197 047766 006202          ASR      R2
7198 047770 006202          ASR      R2
7199 047772 006202          ASR      R2
7200 047774 042702 177700  B!C      #177700,R2
7201 050000 020200          CMP      R2,R0
7202 050002 001017          BNE      23$
7203 050004 032737 000001 002260  BIT      #1,ERROUT          :
7204 050012 001404          BEQ      22$
7205 050014 104456          TRAP     56
7206 050016 000155          .WORD   155
7207 050020 010624          .WORD   MSG4
7208 050022 000000          .WORD   0
7209 050024 016646 000034 22$:   MOV      34(SP),-(SP)      : LUN,*
7210 050030 013746 032360          MOV      BOARD,-(SP)
7211 050034 004737 035302          JSR      PC,UP.HARD.COUNT
7212 050040 000427          BR      26$
7213 050042 032737 000001 002260 23$:   BIT      #1,ERROUT          :
7214 050050 001415          BEQ      25$
7215 050052 104457          TRAP     57
7216 050054 000156          .WORD   156
7217 050056 010610          .WORD   MSG3
7218 050060 000000          .WORD   0
7219 050062 000410          BR      25$
7220 050064 032737 000001 002260 24$:   BIT      #1,ERROUT          :
7221 050072 001404          BEQ      25$
7222 050074 104457          TRAP     57
7223 050076 000157          .WORD   157
7224 050100 010610          .WORD   MSG3
7225 050102 000000          .WORD   0
7226 050104 016646 000034 25$:   MOV      34(SP),-(SP)      : LUN,*

```

23-Oct-1980 15:01:43 TOPS  
 23-Oct-1980 14:57:05 PA:<

```

7228 ;MLX4
7229 ;
7230 ;
7231 050110 013746 032360 MOV BOARD,-(SP)
7232 050114 004737 035430 JSR PC,UP.SOFT.COUNT
7233 050120 022626 26$: CMP (SP)+,(SP)+ ;
7234 050122 010300 27$: MOV R3,R0 ; WRDCNT,*
7235 050124 006300 ASL R0
7236 050126 063700 030706 ADD WPTR,R0
7237 050132 010037 030706 MOV RO,WPTR
7238 050136 010316 MOV R3,(SP) ; WRDCNT,*
7239 050140 012746 000400 MOV #400,-(SP)
7240 050144 004737 005056 JSR PC,BL$DIV
7241 050150 060500 ADD R5,R0 ; SECTOR,*
7242 050152 010005 MOV RO,R5 ; *,SECTOR
7243 050154 062706 000026 ADD #26,SP ;
7244 050160 000137 046502 JMP 3$ ;
7245 050164 005266 000010 28$: INC 10(SP) ; LUN
7246 050170 026666 000010 000012 29$: CMP 10(SP),12(SP) ; LUN,*
7247 050176 002002 BGE 30$
7248 050200 000137 046374 JMP 2$
7249 050204 005266 000026 30$: INC 26(SP) ; COMP.FLAG
7250 050210 026627 000026 000001 CMP 26(SP),#1 ; COMP.FLAG,*
7251 050216 003002 BGT 31$
7252 050220 000137 046356 JMP 1$
7253 050224 062706 000030 31$: ADD #30,SP ;
7254 050230 000207 RTS PC ;
7255
7256 ; Routine Size: 486 words
7257 ; Maximum stack depth per invocation: 37 words
7262
7263
  
```

```

7265 ;MLX4
7266 :
7267 :
7268 : 4506 %SBTTL 'DEFINITION OF OPTION 2'
7269 : 4507
7270 : 4508 ROUTINE OPT2: NOVALUE =
7271 : 4509 BEGIN !* 1 * START OF ROUTINE
7272 : 4510
7273 : 4511 !**
7274 : 4512 : ROUTINE: OPT2
7275 : 4513
7276 : 4514 : PURPOSE: TO CHECK ON DATA RELIABILITY USING THE PATTERNS FROM
7277 : 4515 : THE PATTERN TABLE.
7278 : 4516
7279 : 4517 : THE CODE FOR 'OPT2' IN BRIEF:
7280 : 4518
7281 : 4519 : BEGIN 1 (START OF ROUTINE)
7282 : 4520 : SAY THE ROUTINE IS RUNNING
7283 : 4521 : CHOOSE A MAXIMUM PATTERN NUMBER
7284 : 4522 : INCR COUNT FROM 1 TO (2*MAX)
7285 : 4523 : : BEGIN 2 (START OF PATTERN SELECTION LOOP)
7286 : 4524 : : GENERATE THE PATTERN
7287 : 4525 : : INCR LUN FROM 0 TO LAST
7288 : 4526 : : : BEGIN 3 (START OF LOGICAL UNIT SELECTION LOOP)
7289 : 4527 : : : TESTLOOP:
7290 : 4528 : : : : BEGIN 4 (START OF LOOP THAT COMPLETELY TESTS 1 UNIT)
7291 : 4529 : : : : IF UNIT IS ACTIVE
7292 : 4530 : : : : THEN
7293 : 4531 : : : : : BEGIN 5 (START OF TEST FOR AN ACTIVE UNIT)
7294 : 4532 : : : : : INITIALIZE THE WRITE AND READ BUFFER POINTERS
7295 : 4533 : : : : : SECTOR = LOWEST
7296 : 4534 : : : : : WHILE SECTOR LEQ HIGHEST DO
7297 : 4535 : : : : : : BEGIN 6 (START OF SECTOR SELECTION LOOP)
7298 : 4536 : : : : : : GET WRDCNT
7299 : 4537 : : : : : : SET UP BUFFER POINTERS BEFORE TRANSFER
7300 : 4538 : : : : : : WRITE
7301 : 4539 : : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
7302 : 4540 : : : : : : CHOOSE WHETHER TO WRITE CHECK OR READ
7303 : 4541 : : : : : : DO THE WRITE CHECK OR READ
7304 : 4542 : : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
7305 : 4543 : : : : : : CHANGE BUFFER POINTERS AFTER TRANSFER
7306 : 4544 : : : : : : UPDATE SECTOR NUMBER BY NUMBER OF SECTORS IN PREVIOUS TRANSFER
7307 : 4545 : : : : : : END 6 (END OF SECTOR SELECTION LOOP)
7308 : 4546 : : : : : : END 5 (END OF TEST FOR AN ACTIVE UNIT)
7309 : 4547 : : : : : : END 4 (END OF TESTLOOP)
7310 : 4548 : : : : : : END 3 (END OF LOGICAL UNIT SELECTION LOOP)
7311 : 4549 : : : : : : IF NOT THE QUICK VERIFY PASS THEN 'LOOP READ' (DESCRIBED BELOW)
7312 : 4550 : : : : : : END 2 (END OF PATTERN SELECTION LOOP)
7313 : 4551 : : : : : : RETURN
7314 : 4552 : : : : : : END 1 (END OF ROUTINE)
7315 : 4553 : : : : : : --
7316 : 4554
7317 : 4555 LABEL
7318 : 4556 LOOP, LOOP2;
7319 : 4557
  
```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (41)

```

7321 :MLX4
7322 :
7323 :
7324 : 4558 LOCAL
7325 : 4559 WRDCNT, VALUE, TEMP, MAXPAT, OLDSEC, OLDCHN, SECTOR,
7326 : 4560 PTR, COMMAND, DBL_VALUE;
7327 : 4561
7328 : 4562 PRINTB(SAY2,WRD34,RTN2);
7329 : 4563 !'RUNNING OPT2'
7330 : 4564
7331 : 4565 PATTERN = 0;
7332 : 4566
7333 : 4567 IF .QUICK NEQ 0
7334 : 4568 THEN MAXPAT = 1
7335 : 4569 ELSE MAXPAT = NUM_PATS;
7336 : 4570
7337 : 4571 INCR COUNT FROM 1 TO (.MAXPAT*2) DO
7338 : 4572 BEGIN !* 2 * START OF PATTERN SELECTION LOOP
7339 : 4573 SELPAT();
7340 : 4574 IF .PATTERN GTR 0
7341 : 4575 THEN
7342 : 4576 PRINTB(FMT1A,PHR9,.PATTERN)
7343 : 4577 !'PATTERN NUMBER XX'
7344 : 4578 ELSE
7345 : 4579 BEGIN
7346 : 4580 TEMP = -(.PATTERN);
7347 : 4581 PRINTB(FMT1B,PHR9,.TEMP);
7348 : 4582 !' PATTERN NUMBER - XX'
7349 : 4583 END;
7350 : 4584 GEN2(.PATTERN);
7351 : 4585 INCR LUN FROM 0 TO (.LSUNIT - 1) DO
7352 : 4586 BEGIN !* 3 * START OF LOGICAL UNIT SELECTION LOOP
7353 : 4587 LOOP:
7354 : 4588 BEGIN !* 4 * START OF THE LOOP THAT COMPLETELY TESTS 1 UNIT
7355 : 4589 IF .DRIVE_STATUS[.LUN] EQL ACTIVE
7356 : 4590 THEN
7357 : 4591 BEGIN !* 5 * START OF TEST FOR AN ACTIVE UNIT
7358 : 4592 L$LUN = .LUN;
7359 : 4593 WPTR = WBUFF;
7360 : 4594 RPTR = RBUFF;
7361 : 4595 SECTOR = LOWEST;
7362 : 4596 WHILE .SECTOR LEQ HIGHEST DO
7363 : 4597 BEGIN !* 6 * START OF SECTOR SELECTION LOOP
7364 : 4598 WRDCNT = GET WRDCNT(.SECTOR,HIGHEST);
7365 : 4599 SET_PTRS(.WRDCNT);
7366 : 4600
7367 : 4601 VALUE = WRITE(.LUN,.WRDCNT,.WPTR,.SECTOR);
7368 : 4602 !+
7369 : 4603 ! SEE HOW SUCCESSFUL THE WRITE WAS:
7370 : 4604 !-
7371 : 4605 SELECTONE .VALUE OF !SEE 'SYSERR' FOR DEFINITION
7372 : 4606 SET !OF ERROR # CONTAINED IN 'VALUE'
7373 : 4607 [1] :
7374 : 4608 BEGIN !* 6A * RETRY ALLOWED
7375 : 4609 IF RETRY(SIX,WRITE,.LUN,.WRDCNT,.WPTR,.SECTOR) NEQ 0

```

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (41)



```

7377 :MLX4
7378 :
7379 :
7380 : 4610 THEN !THE RETRY FAILED -- SYSTEM FATAL ERROR
7381 : 4611 BEGIN
7382 : 4612 WHY DROPT[.LUN] = CODE_4;
7383 : 4613 ERRDF(201,MSG1,0); !**** OPTION 2 ERROR 01 ****
7384 : 4614 DODU(.LUN);
7385 : 4615 LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
7386 : 4616 END;
7387 : 4617 END; !* 6A *
7388 : 4618 [2] :
7389 : 4619 BEGIN !* 6B * FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
7390 : 4620 WHY DROPT[.LUN] = CODE_5;
7391 : 4621 ERRDF(202,MSG1,0); !**** OPTION 2 ERROR 02 ****
7392 : 4622 DODU(.LUN);
7393 : 4623 LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
7394 : 4624 END; !* 6B *
7395 : 4625 [3] :
7396 : 4626 BEGIN !* 6C * FATAL DRIVE ERROR -- NO RETRY ALLOWED
7397 : 4627 ERRDF(203,MSG1,0); !**** OPTION 2 ERROR 03 ****
7398 : 4628 WHY DROPT[.LUN] = CODE_6;
7399 : 4629 DODU(.LUN);
7400 : 4630 LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
7401 : 4631 END; !* 6C *
7402 : 4632
7403 : 4633 TES;
7404 : 4634
7405 : 4635 COMMAND = CHOOSE();
7406 : 4636 IF .COMMAND EQL READ
7407 : 4637 THEN
7408 : 4638 BEGIN
7409 : 4639 PTR = .RPTR;
7410 : 4640 VALUE = READ(.LUN,.WRDCNT,.RPTR,.SECTOR);
7411 : 4641 END
7412 : 4642 ELSE
7413 : 4643 BEGIN
7414 : 4644 PTR = .WPTR;
7415 : 4645 VALUE = CHECK(.LUN,.WRDCNT,.WPTR,.SECTOR);
7416 : 4646 END;
7417 : 4647 !+
7418 : 4648 ! SEE HOW SUCCESSFUL THE OPERATION WAS:
7419 : 4649 !-
7420 : 4650 SELECTONE .VALUE OF !SEE 'SYSERR' FOR DEFINITION
7421 : 4651 SET !OF ERROR # CONTAINED IN 'VALUE'
7422 : 4652 [0] :
7423 : 4653 IF .COMMAND EQL READ
7424 : 4654 THEN
7425 : 4655 BEGIN
7426 : 4656 IF (DBL_VALUE = DOUBLE_CHECK(.WPTR,.RPTR,.WRDCNT)) NEQ 0
7427 : 4657 THEN
7428 : 4658 BEGIN
7429 : 4659 SAYWHO(.LUN);
7430 : 4660 PRINTB(SAY1,MSG5);
7431 : 4661 !'ECC LOGIC FAILED TO DETECT DATA ERROR'

```

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (41)

7433 :MLX4

DEFINITION OF OPTION 2

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (41)

7434 :  
 7435 :  
 7436 : 4662  
 7437 : 4663  
 7438 : 4664  
 7439 : 4665  
 7440 : 4666  
 7441 : 4667  
 7442 : 4668  
 7443 : 4669  
 7444 : 4670  
 7445 : 4671  
 7446 : 4672  
 7447 : 4673  
 7448 : 4674  
 7449 : 4675  
 7450 : 4676  
 7451 : 4677  
 7452 : 4678  
 7453 : 4679  
 7454 : 4680  
 7455 : 4681  
 7456 : 4682  
 7457 : 4683  
 7458 : 4684  
 7459 : 4685  
 7460 : 4686  
 7461 : 4687  
 7462 : 4688  
 7463 : 4689  
 7464 : 4690  
 7465 : 4691  
 7466 : 4692  
 7467 : 4693  
 7468 : 4694  
 7469 : 4695  
 7470 : 4696  
 7471 : 4697  
 7472 : 4698  
 7473 : 4699  
 7474 : 4700  
 7475 : 4701  
 7476 : 4702  
 7477 : 4703  
 7478 : 4704  
 7479 : 4705  
 7480 : 4706  
 7481 : 4707  
 7482 : 4708  
 7483 : 4709  
 7484 : 4710  
 7485 : 4711  
 7486 : 4712  
 7487 : 4713

```

PRINTB(FMT12A,..DBL VALUE,..DBL VALUE);
!'GOOD DATA: XXXXXX AT LOCATION YYYYYY'
DBL VALUE = .DBL VALUE + BUFSIZ * 2;
PRINTB(FMT12B,..DBL VALUE,..DBL VALUE);
!'BAD DATA: PPPPPP AT LOCATION QQQQQQ'
WHY DROPT[.LUN] = CODE 8;
ERRDF(204,MSG1,0); !**** OPTION 2 ERROR 04 ****
DODU(.LUN);
LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
END;
END;
[1] :
BEGIN !* 6D * RETRY ALLOWED
IF RETRY(SIX,..COMMAND,..LUN,..WRDCNT,..PTR,..SECTOR) NEQ 0
THEN !THE RETRY FAILED -- SYSTEM FATAL ERROR
BEGIN
WHY DROPT[.LUN] = CODE 4;
ERRDF(205,MSG1,0); !**** OPTION 2 ERROR 05 ****
DODU(.LUN);
LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
END;
END; !* 6D *
[2] :
BEGIN !* 6E * FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
WHY DROPT[.LUN] = CODE 5;
ERRDF(206,MSG1,0); !**** OPTION 2 ERROR 06 ****
DODU(.LUN);
LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
END; !* 6E *
[3] :
BEGIN !* 6F * FATAL DRIVE ERROR -- NO RETRY ALLOWED
ERRDF(207,MSG1,0); !**** OPTION 2 ERROR 07 ****
WHY DROPT[.LUN] = CODE 6;
DODU(.LUN);
LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
END; !* 6F *
[4] :
BEGIN !* 6G * UNRECOVERABLE DATA ERROR
ISOLATE();
ERRDF(208,MSG2,0); !**** OPTION 2 ERROR 08 ****
WHY DROPT[.LUN] = CODE 7;
DODU(.LUN);
LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
END; !* 6G *
[5] :
BEGIN !* 6H * RECOVERABLE DATA ERROR
ISOLATE();
IF .ERROUT
THEN PRINTB(FMT10B,..CHAN);
!' BIT 00'
OLDSEC = .MLEL;
OLDCHN = .CHAN;

```

7489 :MLX4

23-Oct-1980 15:01:43

TOPS-20 Bliss-16 V2(206)

7490 :

DEFINITION OF OPTION 2

23-Oct-1980 14:57:05

PA:<ROSEN>MLX4.BLI.1 (41)

7491 :

7492 :

4714

IF RETRY(ONE,.COMMAND,.LUN,.WRDCNT,.PTR,.SECTOR) EQL 5

7493 :

4715

THEN

7494 :

4716

IF ((.MLEL EQL .OLDSEC) AND (.CHAN EQL .OLDCHN))

7495 :

4717

THEN

7496 :

4718

BEGIN

7497 :

4719

IF .ERROUT

7498 :

4720

THEN

7499 :

4721

ERRHRD(209,MSG4,0); !\*\*\*\* OPTION 2 ERROR 09 \*\*\*\*

7500 :

4722

UP\_HARD\_COUNT(.LUN,.BOARD);

7501 :

4723

END

7502 :

4724

ELSE

7503 :

4725

BEGIN

7504 :

4726

IF .ERROUT

7505 :

4727

THEN

7506 :

4728

ERRSOFT(210,MSG3,0); !\*\*\*\* OPTION 2 ERROR 10 \*\*\*\*

7507 :

4729

UP\_SOFT\_COUNT(.LUN,.BOARD);

7508 :

4730

END

7509 :

4731

ELSE

7510 :

4732

BEGIN

7511 :

4733

IF .ERROUT

7512 :

4734

THEN

7513 :

4735

ERRSOFT(211,MSG3,0); !\*\*\*\* OPTION 2 ERROR 11 \*\*\*\*

7514 :

4736

UP\_SOFT\_COUNT(.LUN,.BOARD);

7515 :

4737

END;

7516 :

4738

END; !\* 6H \*

7517 :

4739

TES;

7518 :

4740

7519 :

4741

7520 :

4742

WPTR = .WPTR + (.WRDCNT \* 2);

7521 :

4743

SECTOR = .SECTOR + (.WRDCNT/256);

7522 :

4744

7523 :

4745

END; !\* 6 \* END OF SECTOR SELECTION LOOP

7524 :

4746

END; !\* 5 \* END OF TEST FOR AN ACTIVE UNIT

7525 :

4747

END; !\* 4 \* END OF TESTLOOP

7526 :

4748

END; !\* 3 \* END OF LOGICAL UNIT SELECTION LOOP

```

7528 :MLX4
7529 :
7530 :
7531 : 4749 IF .QUICK EQL 0
7532 : 4750 THEN
7533 : 4751 BEGIN !* 11 * START OF LOOP READING SECTION
7534 : 4752
7535 : 4753
7536 : 4754
7537 : 4755
7538 : 4756
7539 : 4757
7540 : 4758
7541 : 4759
7542 : 4760
7543 : 4761
7544 : 4762
7545 : 4763
7546 : 4764
7547 : 4765
7548 : 4766
7549 : 4767
7550 : 4768
7551 : 4769
7552 : 4770
7553 : 4771
7554 : 4772
7555 : 4773
7556 : 4774
7557 : 4775
7558 : 4776
7559 : 4777
7560 : 4778
7561 : 4779
7562 : 4780
7563 : 4781
7564 : 4782
7565 : 4783
7566 : 4784
7567 : 4785
7568 : 4786
7569 : 4787
7570 : 4788
7571 : 4789
7572 : 4790
7573 : 4791
7574 : 4792
7575 : 4793
7576 : 4794
7577 : 4795
7578 : 4796
7579 : 4797
7580 : 4798
7581 : 4799
7582 : 4800
  
```

DEFINITION OF OPTION 2

```

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)
23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (42)
  
```

```

  **
  THIS IS THE 'LOOP READ' SECTION WHICH WAS MENTIONED IN
  THE DOCUMENTATION AT THE BEGINNING OF THIS ROUTINE. IT
  IS NOT EXECUTED DURING THE QUICK VERIFY PASS, BUT IT IS
  FOR EVERY OTHER PASS THROUGH OPT2.

  THE CODE IN BRIEF:

  BEGIN 11 (START OF LOOP READING SECTION)
  INCR LUN FROM 0 TO LAST
  : BEGIN 12 (START OF LOGICAL UNIT SELECTION LOOP)
  : TESTLOOP2:
  : : BEGIN 13 (START OF LOOP THAT COMPLETELY TESTS 1 UNIT)
  : : IF UNIT IS ACTIVE
  : : THEN
  : : : BEGIN 14 (START OF TEST FOR AN ACTIVE UNIT)
  : : : INITIALIZE THE WRITE AND READ BUFFER POINTERS
  : : : SFCTOR = LOWEST
  : : : WHILE SECTOR LEQ HIGHEST DO
  : : : : BEGIN 15 (START OF SECTOR SELECTION LOOP)
  : : : : GET_WRCNT
  : : : : SET_UP BUFFER POINTERS BEFORE TRANSFER
  : : : : CHOOSE WHETHER TO WRITE CHECK OR READ
  : : : : INCR KOUNT FROM 1 TO TIMES
  : : : : : BEGIN 16 (START OF COUNTING LOOP FOR LOOP READING)
  : : : : : DO THE WRITE CHECK OR READ
  : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP2)
  : : : : : END 16 (END OF COUNTING LOOP FOR LOOP READING)
  : : : : CHANGE BUFFER POINTERS AFTER TRANSFER
  : : : : UPDATE SECTOR NUMBER BY # SECTORS IN PREVIOUS TRANSFER
  : : : : END 15 (END OF SECTOR SELECTION LOOP)
  : : : : END 14 (END OF TEST FOR AN ACTIVE UNIT)
  : : : : END 13 (END OF TESTLOOP2)
  : : : END 12 (END OF LOGICAL UNIT SELECTION LOOP)
  : END 11 (END OF LOOP READING SECTION)
  --
  INCR LUN FROM 0 TO (.LSUNIT - 1) DO
  BEGIN !* 12 * START OF LOGICAL UNIT SELECTION LOOP
  LOOP2:
  BEGIN !* 13 * START OF THE 2ND LOOP THAT COMPLETELY TESTS 1 UNIT
  IF .DRIVE_STATUS[LUN] EQL ACTIVE
  THEN
  BEGIN !* 14 * START OF LOOP THAT TESTS AN ACTIVE UNIT
  L$LUN = .LUN;
  WPTR = WBUFF;
  RPTR = RBUFF;
  SECTOR = LOWEST;
  
```

```

7584 ;MLX4
7585 :
7586 :
7587 : 4801
7588 : 4802
7589 : 4803
7590 : 4804
7591 : 4805
7592 : 4806
7593 : 4807
7594 : 4808
7595 : 4809
7596 : 4810
7597 : 4811
7598 : 4812
7599 : 4813
7600 : 4814
7601 : 4815
7602 : 4816
7603 : 4817
7604 : 4818
7605 : 4819
7606 : 4820
7607 : 4821
7608 : 4822
7609 : 4823
7610 : 4824
7611 : 4825
7612 : 4826
7613 : 4827
7614 : 4828
7615 : 4829
7616 : 4830
7617 : 4831
7618 : 4832
7619 : 4833
7620 : 4834
7621 : 4835
7622 : 4836
7623 : 4837
7624 : 4838
7625 : 4839
7626 : 4840
7627 : 4841
7628 : 4842
7629 : 4843
7630 : 4844
7631 : 4845
7632 : 4846
7633 : 4847
7634 : 4848
7635 : 4849
7636 : 4850
7637 : 4851
7638 : 4852

DEFINITION OF OPTION 2

      WHILE .SECTOR LEQ HIGHEST DO
      BEGIN !* 15 * START OF SECTOR SELECTION LOOP
      WRDCNT = GET WRDCNT(.SECTOR,HIGHEST);
      SET_PTRS(.WRDCNT);

      COMMAND = CHOOSE();
      INCR COUNT FROM 1 TO TIMES TO LOOP DO
      BEGIN !* 16 * START OF COUNTING LOOP FOR LOOP READING
      IF .COMMAND EQL READ
      THEN
      BEGIN
      PTR = .RPTR;
      VALUE = READ(.LUN,.WRDCNT,.RPTR,.SECTOR);
      END
      ELSE
      BEGIN
      PTR = .WPTR;
      VALUE = CHECK(.LUN,.WRDCNT,.WPTR,.SECTOR);
      END;
      !+
      !- SEE HOW SUCCESSFUL THE OPERATION WAS:
      !-
      SELECTONE .VALUE OF          !SEE 'SYSERR' FOR DEFINITION
      SET                          !OF ERROR # CONTAINED IN 'VALUE'
      [0] :
      IF .COMMAND EQL READ
      THEN
      BEGIN
      IF (DBL_VALUE = DOUBLE_CHECK(.WPTR,.RPTR,.WRDCNT)) NEQ 0
      THEN
      BEGIN
      SAYWHO(.LUN);
      PRINTB(SAY1,MSG5);
      !'ECC LOGIC FAILED TO DETECT DATA ERROR'
      PRINTB(FMT12A,..DBL_VALUE,..DBL_VALUE);
      !'GOOD DATA: XXXXXX AT LOCATION YYYYYY'
      DBL_VALUE = .DBL_VALUE + BUFSIZ * 2;
      PRINTB(FMT12B,..DBL_VALUE,..DBL_VALUE);
      !'BAD DATA: P P P P P AT LOCATION Q Q Q Q Q'
      WHY DROPT[.LUN] = CODE 8;
      ERRDF(212,MSG1,0);      !**** OPTION 2 ERROR 12 ****
      DODU(.LUN);
      LEAVE LOOP2;          !JUMP JUST BEYOND END OF BLOCK * 4 *
      END;
      END;
      [1] :
      BEGIN !* 16A * RETRY ALLOWED
      IF RETRY(SIX,.COMMAND,.LUN,.WRDCNT,.PTR,.SECTOR) NEQ 0
      THEN !THE RETRY FAILED -- SYSTEM FATAL ERROR
      BEGIN
      WHY DROPT[.LUN] = CODE 4;
      ERRDF(213,MSG1,0);      !**** OPTION 2 ERROR 13 ****

```

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (42)

23-OCT-1980 15:01:43 TOPS-20 Bligg-16 v2(204)  
 23-OCT-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (42)

7640 ;MLX4

DEFINITION OF OPTION 2

```

7641 :
7642 :
7643 :      4853      DODU(.LUN);
7644 :      4854      LEAVE LOOP2; !JUMP JUST BEYOND END OF BLOCK * 13 *
7645 :      4855      END;
7646 :      4856      END; !* 16A *
7647 :      4857      [2] :
7648 :      4858      BEGIN !* 16B * FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
7649 :      4859      WHY DROPT(.LUN) = CODE_5;
7650 :      4860      ERRDF(214,MSG1,0); !**** OPTION 2 ERROR 14 ****
7651 :      4861      DODU(.LUN);
7652 :      4862      LEAVE LOOP2; !JUMP JUST BEYOND END OF BLOCK * 13 *
7653 :      4863      END; !* 16B *
7654 :      4864      [3] :
7655 :      4865      BEGIN !* 16C * FATAL DRIVE ERROR -- NO RETRY ALLOWED
7656 :      4866      ERRDF(215,MSG1,0); !**** OPT,ON 2 ERROR 15 ****
7657 :      4867      WHY DROPT(.LUN) = CODE_6;
7658 :      4868      DODU(.LUN);
7659 :      4869      LEAVE LOOP2; !JUMP JUST BEYOND END OF BLOCK * 13 *
7660 :      4870      END; !* 16C *
7661 :      4871      [4] :
7662 :      4872      BEGIN !* 16D * UNRECOVERABLE DATA ERROR
7663 :      4873      ISOLATE();
7664 :      4874      ERRDF(216,MSG2,0); !**** OPTION 2 ERROR 16 ****
7665 :      4875      WHY DROPT(.LUN) = CODE_7;
7666 :      4876      DODU(.LUN);
7667 :      4877      LEAVE LOOP2; !JUMP JUST BEYOND END OF BLOCK * 13 *
7668 :      4878      END; !* 16D *
7669 :      4879      [5] :
7670 :      4880      BEGIN !* 16E * RECOVERABLE DATA ERROR
7671 :      4881      ISOLATE();
7672 :      4882      IF .ERROUT
7673 :      4883      THEN PRINTB(FMT10B,.CHAN);
7674 :      4884      !' BIT 00'
7675 :      4885      OLDSEC = .MLEL;
7676 :      4886      OLDCHN = .CHAN;
7677 :      4887      IF RETRY(ONE,.COMMAND,.LUN,.WRDCNT,.PTR,.SECTOR) EQL 5
7678 :      4888      THEN
7679 :      4889      IF ((.MLEL EQL .OLDSEC) AND (.CHAN EQL .OLDCHN))
7680 :      4890      THEN
7681 :      4891      BEGIN
7682 :      4892      IF .ERROUT
7683 :      4893      THEN
7684 :      4894      ERRHRD(217,MSG4,0); !**** OPTION 2 ERROR 17 ****
7685 :      4895      UP_HARD_COUNT(.LUN,.BOARD);
7686 :      4896      END
7687 :      4897      ELSE
7688 :      4898      BEGIN
7689 :      4899      IF .ERROUT
7690 :      4900      THEN
7691 :      4901      ERRSOF(218,MSG3,0); !**** OPTION 2 ERROR 18 ****
7692 :      4902      UP_SOFT_COUNT(.LUN,.BOARD);
7693 :      4903      END
7694 :      4904      ELSE
  
```

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (42)

```

7696 :MLX4
7697 :
7698 :
7699 : 4905 BEGIN
7700 : 4906 IF .ERROUT
7701 : 4907 THEN
7702 : 4908 ERRSOFT(219,MSG3,0); !**** OPTION 2 ERROR 19 ****
7703 : 4909 UP SOFT_COUNT(.LJN,.BOARD);
7704 : 4910 END;
7705 : 4911 END; !* 16E *
7706 : 4912
7707 : 4913 TES;
7708 : 4914
7709 : 4915 END; !* 16 * END OF COUNTING LOOP FOR LOOP READING
7710 : 4916 WPTR = .WPTR + (.WRDCNT * 2);
7711 : 4917 SECTOR = .SECTOR + (.WRDCNT/256);
7712 : 4918 END; !* 15 * END OF SECTOR SELECTION LOOP
7713 : 4919 END; !* 14 * END OF TEST FOR AN ACTIVE UNIT
7714 : 4920 END; !* 13 * END OF TESTLOOP2
7715 : 4921 END; !* 12 * END OF LOGICAL UNIT SELECTION LOOP
7716 : 4922 END; !* 11 * END OF LOOP READING SECTION
7717 : 4923 END; !* 2 * END OF PATTERN SELECTION LOOP
7718 : 4924 RETURN;
7719 : 4925 END; !* 1 * END OF ROUTINE
  
```

```

7723 :
7724 :
7728 050232 004137 005222 OPT2: .SBTTL OPT2 DEFINITION OF OPTION 2 : 4508
7729 050236 162706 000026 JSR R1,$SAVE5 :
7730 050242 012746 007256 SUB #26,SP : 4562
7731 050246 012746 006766 MOV #RTN2,-(SP) :
7732 050252 012746 006522 MOV #WRD34,(SP) :
7733 050256 012746 000003 MOV #SAY2,-(SP) :
7734 050262 010600 MOV #3,-(SP) :
7735 050264 104414 MOV SP,R0 : SP,*
7736 050266 005037 030714 TRAP 14 :
7737 050272 005737 030712 CLR PATTERN : 4565
7738 050276 001403 TST QUICK : 4567
7739 050300 012705 000001 BEQ 1$ :
7740 050304 000402 MOV #1,R5 : *,MAXPAT 4568
7741 050306 012705 000012 BR 2$ : 4567
7742 050312 010566 000032 1$: MOV #12,R5 : *,MAXPAT 4569
7743 050316 006366 000032 2$: MOV R5,32(SP) : MAXPAT,* 4571
7744 050322 005066 000030 ASL 32(SP) :
7745 050326 000137 053532 CLR 30(SP) : COUNT
7746 050332 004737 036156 3$: JSR PC,SELPAT : 4573
7747 050336 005737 030714 TST PATTERN : 4574
7748 050342 003413 BLE 4$ :
7749 050344 013746 030714 MOV PATTERN,-(SP) : 4576
7750 050350 012746 007522 MOV #PHR9,-(SP) :
  
```

Address	Label	Source	Destination	Operation	Comments	Date/Time	Page
7752						23-Oct-1980 15:01:43	TOPS
7753						23-Oct-1980 14:57:05	PA:<
7754							
7755	050354	012746	005550	MOV	#FMT1A,-(SP)		
7756	050360	012746	000003	MOV	#3,-(SP)		
7757	050364	010600		MOV	SP,R0	: SP,*	
7758	050366	104414		TRAP	14		
7759	050370	000417		BR	5\$		
7760	050372	013766	030714 000034	MOV	PATTERN,34(SP)	: *,TEMP	4574
7761	050400	005466	000034	NEG	34(SP)	: TEMP	4580
7762	050404	016646	000034	MOV	34(SP),-(SP)	: TEMP,*	4581
7763	050410	012746	007522	MOV	#PHR9,-(SP)		
7764	050414	012746	005564	MOV	#FMT1B,-(SP)		
7765	050420	012746	000003	MOV	#3,-(SP)		
7766	050424	010600		MOV	SP,R0	: SP,*	
7767	050426	104414		TRAP	14		
7768	050430	013716	030714	MOV	PATTERN,(SP)	:	4584
7769	050434	004737	036522	JSR	PC,GEN2		
7770	050440	013766	002012 000030	MOV	L\$UNIT,30(SP)	:	4585
7771	050446	005005		CLR	R5	: LUN	
7772	050450	000137	052136	JMP	33\$		
7773	050454	010500		MOV	R5,R0	: LUN,*	4589
7774	050456	006200		ASR	R0		
7775	050460	006200		ASR	R0		
7776	050462	006200		ASR	R0		
7777	050464	062700	032460	ADD	#DRIVE.STATUS,R0		
7778	050470	010046		MOV	R0,-(SP)		
7779	050472	010546		MOV	R5,-(SP)	: LUN,*	
7780	050474	042716	177770	BIC	#177770,(SP)		
7781	050500	012746	000001	MOV	#1,-(SP)		
7782	050504	005046		CLR	-(SP)		
7783	050506	004737	004244	JSR	PC,BL\$GT2		
7784	050512	062706	000010	ADD	#10,SP		
7785	050516	005300		DEC	R0		
7786	050520	001127		BNE	11\$		
7787	050522	010537	002074	MOV	R5,L\$LUN	: LUN,*	4592
7788	050526	012737	010706 030706	MOV	#WBUF,WPTR	:	4593
7789	050534	012737	020706 030710	MOV	#RBUF,RPTR	:	4594
7790	050542	010501		MOV	R5,R1	: LUN,*	4595
7791	050544	006301		ASL	R1		
7792	050546	016104	032476	MOV	LOW.SECT(R1),R4	: *,SECTOR	
7793	050552	020461	032516	7\$: CMP	R4, TOP.SECT(R1)	: SECTOR,*	4596
7794	050556	003110		BGT	11\$		
7795	050560	010446		MOV	R4,-(SP)	: SECTOR,*	4598
7796	050562	016146	032516	MOV	TOP.SECT(R1),-(SP)		
7797	050566	004737	044034	JSR	PC,GET.WRDCNT		
7798	050572	010002		MOV	R0,R2	: *,WRDCNT	
7799	050574	010216		MOV	R2,(SP)	: WRDCNT,*	4599
7800	050576	004737	043770	JSR	PC,SET.PTRS		
7801	050602	010516		MOV	R5,(SP)	: LUN,*	4601
7802	050604	010246		MOV	R2,-(SP)	: WRDCNT,*	
7803	050606	013746	030706	MOV	WPTR,-(SP)		
7804	050612	010446		MOV	R4,-(SP)	: SECTOR,*	
7805	050614	004737	042246	JSR	PC,WRITE		
7806	050620	010003		MOV	R0,R3	: *,VALUE	



Address	Label	OpCode	Operand 1	Operand 2	Instruction	Comments	Line No.
7808							
7809	:MLX4						
7810	:				DEFINITION OF OPTION 2		
7811	050622	020327	000001		CMP R3,#1	: VALUE,*	4605
7812	050626	001031			BNE 8\$		
7813	050630	012746	000006		MOV #6,-(SP)	:	4609
7814	050634	012746	042246		MOV #WRITE,-(SP)		
7815	050640	010546			MOV R5,-(SP)	: LUN,*	
7816	050642	010246			MOV R2,-(SP)	: WRDCNT,*	
7817	050644	013746	030706		MOV WPTR,-(SP)		
7818	050650	010446			MOV R4,-(SP)	: SECTOR,*	
7819	050652	004737	043046		JSR PC,RETRY		
7820	050656	062706	000014		ADD #14,SP		
7821	050662	005700			TST R0		
7822	050664	001446			BEQ 12\$		
7823	050666	112765	000004	032464	MOVB #4,WHY.DROPT(R5)	: *,*(LUN)	4612
7824	050674	104455			TRAP 55	:	4613
7825	050676	000311			.WORD 311		
7826	050700	010512			.WORD MSG1		
7827	050702	000000			.WORD 0		
7828	050704	010500			MOV R5,R0	: LUN,*	4614
7829	050706	104451			TRAP 51		
7830	050710	000431			BR 10\$		4615
7831	050712	020327	000002	8\$:	CMP R3,#2	: VALUE,*	4605
7832	050716	001012			BNE 9\$		
7833	050720	112765	000005	032464	MOVB #5,WHY.DROPT(R5)	: *,*(LUN)	4620
7834	050726	104455			TRAP 55	:	4621
7835	050730	000312			.WORD 312		
7836	050732	010512			.WORD MSG1		
7837	050734	000000			.WORD 0		
7838	050736	010500			MOV R5,R0	: LUN,*	4622
7839	050740	104451			TRAP 51		
7840	050742	000414			BR 10\$		4623
7841	050744	020327	000003	9\$:	CMP R3,#3	: VALUE,*	4605
7842	050750	001014			BNE 12\$		
7843	050752	104455			TRAP 55	:	4627
7844	050754	000313			.WORD 313		
7845	050756	010512			.WORD MSG1		
7846	050760	000000			.WORD 0		
7847	050762	112765	000006	032464	MOVB #6,WHY.DROPT(R5)	: *,*(LUN)	4628
7848	050770	010500			MOV R5,R0	: LUN,*	4629
7849	050772	104451			TRAP 51		
7850	050774	062706	000012	10\$:	ADD #12,SP	:	4630
7851	051000	000537			BR 15\$		
7852	051002	004737	043010	12\$:	JSR PC,CHOOSE		4635
7853	051006	010066	000036		MOV R0,36(SP)	: *,COMMAND	
7854	051012	005066	000034		CLR 34(SP)	:	4636
7855	051016	020027	042434		CMP R0,#READ	: COMMAND,*	
7856	051022	001015			BNE 13\$		
7857	051024	005266	000034		INC 34(SP)		
7858	051030	013766	030710	000040	MOV RPTR,40(SP)	: *,PTR	4639
7859	051036	010546			MOV R5,-(SP)	: LUN,*	4640
7860	051040	010246			MOV R2,-(SP)	: WRDCNT,*	
7861	051042	013746	030710		MOV RPTR,-(SP)		
7862	051046	010446			MOV R4,-(SP)	: SECTOR,*	

23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<

					23-Oct-1980 15:01:43	TOPS
					23-Oct-1980 14:57:05	PA:<
7864				:MLX4		
7865				:		
7866					DEFINITION OF OPTION 2	
7867	051050	004737	042434	JSR	PC,READ	
7868	051054	000412		BR	14\$	
7869	051056	013766	030706 000040	13\$:	MOV WPTR,40(SP)	: *,PTR 4644
7870	051064	010546		MOV	R5,-(SP)	: LUN,* 4645
7871	051066	010246		MOV	R2,-(SP)	: WRDCNT,*
7872	051070	013746	030706	MOV	WPTR,-(SP)	
7873	051074	010446		MOV	R4,-(SP)	: SECTOR,*
7874	051076	004737	042622	JSR	PC,CHECK	
7875	051102	010003		14\$:	MOV R0,R3	: *,VALUE
7876	051104	001076		BNE	16\$	: 4650
7877	051106	032766	000001 000044	BIT	#1,44(SP)	: 4653
7878	051114	001513		BEQ	17\$	:
7879	051116	013746	030706	MOV	WPTR,-(SP)	: 4656
7880	051122	013746	030710	MOV	RPTR,-(SP)	
7881	051126	010246		MOV	R2,-(SP)	: WRDCNT,*
7882	051130	004737	041226	JSR	PC,DOUBLE.CHECK	
7883	051134	062706	000006	ADD	#6,SP	
7884	051140	010066	000054	MOV	R0,54(SP)	: *,DBL.VALUE
7885	051144	001477		BEQ	17\$	
7886	051146	010546		MOV	R5,-(SP)	: LUN,* 4659
7887	051150	004737	033436	JSR	PC,SAYWHO	
7888	051154	012716	010640	MOV	#MSG5,(SP)	: 4660
7889	051160	012746	006514	MOV	#SAY1,-(SP)	
7890	051164	012746	000002	MOV	#2,-(SP)	
7891	051170	010600		MOV	SP,R0	: SP,*
7892	051172	104414		TRAP	14	
7893	051174	016616	000062	MOV	62(SP),(SP)	: DBL.VALUE,* 4662
7894	051200	017646	000062	MOV	@62(SP),-(SP)	: DBL.VALUE,*
7895	051204	012746	006332	MOV	#FMT12A,-(SP)	
7896	051210	012746	000003	MOV	#3,-(SP)	
7897	051214	010600		MOV	SP,R0	: SP,*
7898	051216	104414		TRAP	14	
7899	051220	062766	010000 000070	ADD	#10000,70(SP)	: *,DBL.VALUE 4664
7900	051226	016616	000070	MOV	70(SP),(SP)	: DBL.VALUE,* 4665
7901	051232	017646	000070	MOV	@70(SP),-(SP)	: DBL.VALUE,*
7902	051236	012746	006400	MOV	#FMT12B,-(SP)	
7903	051242	012746	000003	MOV	#3,-(SP)	
7904	051246	010600		MOV	SP,R0	: SP,*
7905	051250	104414		TRAP	14	
7906	051252	112765	000010 032464	MOVB	#10,WHY.DROPT(R5)	: *,*(LUN) 4667
7907	051260	104455		TRAP	55	: 4668
7908	051262	000314		.WORD	314	
7909	051264	010512		.WORD	MSG1	
7910	051266	000000		.WORD	0	
7911	051270	010500		MOV	R5,R0	: LUN,* 4669
7912	051272	104451		TRAP	51	
7913	051274	062706	000044	ADD	#44,SP	: 4670
7914	051300	000510		15\$:	BR 23\$	
7915	051302	020327	000001	16\$:	CMP R3,#1	: VALUE,* 4650
7916	051306	001033		BNE	19\$	
7917	051310	012746	000006	MOV	#6,-(SP)	: 4675
7918	051314	016646	000050	MOV	50(SP),-(SP)	: COMMAND,*

Address	Offset	OpCode	OpData	Label	Comment	Address	Time	Page
7920					:MLX4		23-Oct-1980 15:01:43	TOPS
7921					:		23-Oct-1980 14:57:05	PA:<
7922					DEFINITION OF OPTION 2			
7923	051320	010546			MOV R5,-(SP)	: LUN,*		
7924	051322	010246			MOV R2,-(SP)	: WRDCNT,*		
7925	051324	016646	000060		MOV 60(SP),-(SP)	: PTR,*		
7926	051330	010446			MOV R4,-(SP)	: SECTOR,*		
7927	051332	004737	043046		JSR PC,RETRY			
7928	051336	062706	000014		ADD #14,SP			
7929	051342	005700			TST R0			
7930	051344	001002		17\$:	BNE 18\$			
7931	051346	000137	052072		JMP 31\$			
7932	051352	112765	000004	032464	MOVB #4,WHY.DROPT(R5)	: *,*(LUN)		4678
7933	051360	104455			TRAP 55	:		4679
7934	051362	000315			.WORD 315			
7935	051364	010512			.WORD MSG1			
7936	051366	000000			.WORD 0			
7937	051370	010500			MOV R5,R0	: LUN,*		4680
7938	051372	104451			TRAP 51			
7939	051374	000450			BR 22\$	:		4681
7940	051376	020327	000002		CMP R3,#2	: VALUE,*		4650
7941	051402	001012			BNE 20\$			
7942	051404	112765	000005	032464	MOVB #5,WHY.DROPT(R5)	: *,*(LUN)		4686
7943	051412	104455			TRAP 55	:		4687
7944	051414	000316			.WORD 316			
7945	051416	010512			.WORD MSG1			
7946	051420	000000			.WORD 0			
7947	051422	010500			MOV R5,R0	: LUN,*		4688
7948	051424	104451			TRAP 51			
7949	051426	000433			BR 22\$	:		4689
7950	051430	020327	000003		CMP R3,#3	: VALUE,*		4650
7951	051434	001012			BNE 21\$			
7952	051436	104455			TRAP 55	:		4693
7953	051440	000317			.WORD 317			
7954	051442	010512			.WORD MSG1			
7955	051444	000000			.WORD 0			
7956	051446	112765	000006	032464	MOVB #6,WHY.DROPT(R5)	: *,*(LUN)		4694
7957	051454	010500			MOV R5,R0	: LUN,*		4695
7958	051456	104451			TRAP 51			
7959	051460	000416			BR 22\$	:		4696
7960	051462	020327	000004		CMP R3,#4	: VALUE,*		4650
7961	051466	001017			BNE 24\$			
7962	051470	004737	035212		JSR PC,ISOLATE	:		4700
7963	051474	104455			TRAP 55	:		4701
7964	051476	000320			.WORD 320			
7965	051500	010542			.WORD MSG2			
7966	051502	000000			.WORD 0			
7967	051504	112765	000007	032464	MOVB #7,WHY.DROPT(R5)	: *,*(LUN)		4702
7968	051512	010500			MOV R5,R0	: LUN,*		4703
7969	051514	104451			TRAP 51			
7970	051516	062706	000022		ADD #22,SP	:		4704
7971	051522	000137	052134		JMP 32\$			
7972	051526	020327	000005		CMP R3,#5	: VALUE,*		4650
7973	051532	001157			BNE 31\$			
7974	051534	004737	035212		JSR PC,ISOLATE	:		4703

```

7976          ;MLX4
7977          ;
7978
7979 051540 032737 000001 002260          BIT    #1,ERROUT          ;
7980 051546 001423              BEQ    25$              :
7981 051550 017700 160652              MOV    @ML.REG+42,R0        ;
7982 051554 006200              ASR    R0              :
7983 051556 006200              ASR    R0              :
7984 051560 006200              ASR    R0              :
7985 051562 006200              ASR    R0              :
7986 051564 006200              ASR    R0              :
7987 051566 006200              ASR    R0              :
7988 051570 042700 177700              BIC    #177700,R0
7989 051574 010046              MOV    R0,-(SP)
7990 051576 012746 006262              MOV    #FMT10B,-(SP)
7991 051602 012746 000002              MOV    #2,-(SP)
7992 051606 010600              MOV    SP,R0          ; SP,*
7993 051610 104414              TRAP   14
7994 051612 062706 000006              ADD    #6,SP
7995 051616 017766 160606 000060 25$:  MOV    @ML.REG+44,60(SP)  ; *,OLDSEC      4712
7996 051624 017700 160576              MOV    @ML.REG+42,R0        ;
7997 051630 006200              ASR    R0
7998 051632 006200              ASR    R0
7999 051634 006200              ASR    R0
8000 051636 006200              ASR    R0
8001 051640 006200              ASR    R0
8002 051642 006200              ASR    R0
8003 051644 042700 177700              BIC    #177700,R0
8004 051650 010066 000056              MOV    R0,56(SP)          ; *,OLDCHN
8005 051654 012746 000001              MOV    #1,-(SP)          ;
8006 051660 016646 000050              MOV    50(SP),-(SP)      ; COMMAND,*
8007 051664 010546              MOV    R5,-(SP)          ; LUN,*
8008 051666 010246              MOV    R2,-(SP)          ; WRDCNT,*
8009 051670 016646 000060              MOV    60(SP),-(SP)      ; PTR,*
8010 051674 010446              MOV    R4,-(SP)          ; SECTOR,*
8011 051676 004737 043046              JSR    PC,RETRY
8012 051702 062706 000014              ADD    #14,SP
8013 051706 020027 000005              CMP    R0,#5
8014 051712 001051              BNE    28$
8015 051714 027766 160510 000060              CMP    @ML.REG+44,60(SP)  ; *,OLDSEC      4716
8016 051722 001034              BNE    27$
8017 051724 016646 000056              MOV    56(SP),-(SP)      ; OLDCHN,*
8018 051730 017700 160472              MOV    @ML.REG+42,R0
8019 051734 006200              ASR    R0
8020 051736 006200              ASR    R0
8021 051740 006200              ASR    R0
8022 051742 006200              ASR    R0
8023 051744 006200              ASR    R0
8024 051746 006200              ASR    R0
8025 051750 042700 177700              BIC    #177700,R0
8026 051754 020026              CMP    R0,(SP)+
8027 051756 001016              BNE    27$
8028 051760 032737 000001 002260          BIT    #1,ERROUT          ;
8029 051766 001404              BEQ    26$              :
8030 051770 104456              TRAP   56              :

```

Address	Label	Operation	Operand 1	Operand 2	Comment	Seq
8032						
8033	:MLX4					
8034	:				DEFINITION OF OPTION 2	
8035		.WORD	321			
8036		.WORD	MSG4			
8037		.WORD	0			
8038	26\$:	MOV	R5,-(SP)		: LUN,*	4722
8039		MOV	BOARD,-(SP)			
8040		JSR	PC,UP.HARD.COUNT			
8041		BR	30\$		:	4716
8042	27\$:	BIT	#1,ERROUT		:	4726
8043		BEQ	29\$		:	
8044		TRAP	57		:	4728
8045		.WORD	322			
8046		.WORD	MSG3			
8047		.WORD	0			
8048		BR	29\$		:	4729
8049	28\$:	BIT	#1,ERROUT		:	4733
8050		BEQ	29\$		:	
8051		TRAP	57		:	4735
8052		.WORD	323			
8053		.WORD	MSG3			
8054		.WORD	0			
8055	29\$:	MOV	R5,-(SP)		: LUN,*	4736
8056		MOV	BOARD,-(SP)			
8057		JSR	PC,UP.SOFT.COUNT			
8058	30\$:	CMP	(SP)+,(SP)+		:	4707
8059	31\$:	MOV	R2,R0		: WRDCNT,*	4742
8060		ASL	R0			
8061		ADD	WPTR,R0			
8062		MOV	R0,WPTR			
8063		MOV	R2,(SP)		: WRDCNT,*	4743
8064		MOV	#400,-(SP)			
8065		JSR	PC,BL\$DIV			
8066		ADD	R4,R0		: SECTOR,*	
8067		MOV	R0,R4		: *,SECTOR	
8068		ADD	#24,SP		:	4597
8069		JMP	7\$		:	4596
8070	32\$:	INC	R5		: LUN	4585
8071	33\$:	CMP	R5,30(SP)		: LUN,*	
8072		BGE	34\$		:	
8073		JMP	6\$		:	
8074	34\$:	TST	QUICK		:	4749
8075		BEQ	35\$		:	
8076		JMP	61\$		:	
8077	35\$:	MOV	L\$UNIT,30(SP)		:	4790
8078		CLR	R5		: LUN	
8079		JMP	60\$		:	
8080	36\$:	MOV	R5,R0		: LUN,*	4794
8081		ASR	R0			
8082		ASR	R0			
8083		ASR	R0			
8084		ADD	#DRIVE.STATUS,R0			
8085		MOV	R0,-(SP)			
8086		MOV	R5,-(SP)		: LUN,*	



23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<

Address	Hex	Hex	Hex	Label	Text	Comments	Address
8144							
8145				;MLX4			
8146				:	DEFINITION OF OPTION 2		
8147	052500	010546		MOV	R5,-(SP)	; LUN,*	4832
8148	052502	004737	033436	JSR	PC,SAYWHO		
8149	052506	012716	010640	MOV	#MSG5,(SP)		4833
8150	052512	012746	006514	MOV	#SAY1,-(SP)		
8151	052516	012746	000002	MOV	#2,-(SP)		
8152	052522	010600		MOV	SP,R0	; SP,*	
8153	052524	104414		TRAP	14		
8154	052526	016616	000054	MOV	54(SP),(SP)	; DBL.VALUE,*	4835
8155	052532	017646	000054	MOV	54(SP),-(SP)	; DBL.VALUE,*	
8156	052536	012746	006332	MOV	#FMT12A,-(SP)		
8157	052542	012746	000003	MOV	#3,-(SP)		
8158	052546	010600		MOV	SP,R0	; SP,*	
8159	052550	104414		TRAP	14		
8160	052552	062766	010000 000062	ADD	#10000,62(SP)	; *,DBL.VALUE	4837
8161	052560	016616	000062	MOV	62(SP),(SP)	; DBL.VALUE,*	4838
8162	052564	017646	000062	MOV	62(SP),-(SP)	; DBL.VALUE,*	
8163	052570	012746	006400	MOV	#FMT12B,-(SP)		
8164	052574	012746	000003	MOV	#3,-(SP)		
8165	052600	010600		MOV	SP,R0	; SP,*	
8166	052602	104414		TRAP	14		
8167	052604	112765	000010 032464	MOVB	#10,WHY.DROPT(R5)	; *,*(LUN)	4840
8168	052612	104455		TRAP	55		4841
8169	052614	000324		.WORD	324		
8170	052616	010512		.WORD	MSG1		
8171	052620	000000		.WORD	0		
8172	052622	010500		MOV	R5,R0	; LUN,*	4842
8173	052624	104451		TRAP	51		
8174	052626	062706	000036	ADD	#36,SP		4843
8175	052632	000510		BR	49\$		
8176	052634	020327	000001	CMP	R3,#1	; VALUE,*	4823
8177	052640	001033		BNE	45\$		
8178	052642	012746	000006	MOV	#6,-(SP)		4848
8179	052646	016646	000042	MOV	42(SP),-(SP)	; COMMAND,*	
8180	052652	010546		MOV	R5,-(SP)	; LUN,*	
8181	052654	010246		MOV	R2,-(SP)	; WRDCNT,*	
8182	052656	016646	000052	MOV	52(SP),-(SP)	; PTR,*	
8183	052662	010446		MOV	R4,-(SP)	; SECTOR,*	
8184	052664	004737	043046	JSR	PC,RETRY		
8185	052670	062706	000014	ADD	#14,SP		
8186	052674	005700		TST	R0		
8187	052676	001002		BNE	44\$		
8188	052700	000137	053424	JMP	57\$		
8189	052704	112765	000004 032464	MOVB	#4,WHY.DROPT(R5)	; *,*(LUN)	4851
8190	052712	104455		TRAP	55		4852
8191	052714	000325		.WORD	325		
8192	052716	010512		.WORD	MSG1		
8193	052720	000000		.WORD	0		
8194	052722	010500		MOV	R5,R0	; LUN,*	4853
8195	052724	104451		TRAP	51		
8196	052726	000450		BR	48\$		4854
8197	052730	020327	000002	CMP	R3,#2	; VALUE,*	4823
8198	052734	001012		BNE	46\$		

Address	Offset	Label	Value	Symbol	Operation	Comments	Date	Time	Page
8200							23-Oct-1980	15:01:43	TOPS
8201							23-Oct-1980	14:57:05	PA:<
8202									
8203	052736	112765	000005	032464	MOVB	#5,WHY.DROPT(R5)			4859
8204	052744	104455			TRAP	55			4860
8205	052746	000326			.WORD	326			
8206	052750	010512			.WORD	MSG1			
8207	052752	000000			.WORD	0			
8208	052754	010500			MOV	R5,R0			4861
8209	052756	104451			TRAP	51			
8210	052760	000433			BR	48\$			4862
8211	052762	020327	000003	46\$:	CMP	R3,#3			4823
8212	052766	001012			BNE	47\$			
8213	052770	104455			TRAP	55			4866
8214	052772	000327			.WORD	327			
8215	052774	010512			.WORD	MSG1			
8216	052776	000000			.WORD	0			
8217	053000	112765	000006	032464	MOVB	#6,WHY.DROPT(R5)			4867
8218	053006	010500			MOV	R5,R0			4868
8219	053010	104451			TRAP	51			
8220	053012	000416			BR	48\$			4869
8221	053014	020327	000004	47\$:	CMP	R3,#4			4823
8222	053020	001017			BNE	50\$			
8223	053022	004737	035212		JSR	PC,ISOLAIE			4873
8224	053026	104455			TRAP	55			4874
8225	053030	000330			.WORD	330			
8226	053032	010542			.WORD	MSG2			
8227	053034	000000			.WORD	0			
8228	053036	112765	000007	032464	MOVB	#7,WHY.DROPT(R5)			4875
8229	053044	010500			MOV	R5,R0			4876
8230	053046	104451			TRAP	51			
8231	053050	062706	000014	48\$:	ADD	#14,SP			4877
8232	053054	000137	053512	49\$:	JMP	59\$			
8233	053060	020327	000005	50\$:	CMP	R3,#5			4823
8234	053064	001157			BNE	57\$			
8235	053066	004737	035212		JSR	PC,ISOLATE			4881
8236	053072	032737	000001	002260	BIT	#1,ERROUT			4882
8237	053100	001423			BEQ	51\$			
8238	053102	017700	157320		MOV	@ML.REG+42,R0			4883
8239	053106	006200			ASR	R0			
8240	053110	006200			ASR	R0			
8241	053112	006200			ASR	R0			
8242	053114	006200			ASR	R0			
8243	053116	006200			ASR	R0			
8244	053120	006200			ASR	R0			
8245	053122	042700	177700		BIC	#177700,R0			
8246	053126	010046			MOV	R0,-(SP)			
8247	053130	012746	006262		MOV	#FMT10B,-(SP)			
8248	053134	012746	000002		MOV	#2,-(SP)			
8249	053140	010600			MOV	SP,R0			
8250	053142	104414			TRAP	14			
8251	053144	062706	000006		ADD	#6,SP			
8252	053150	017766	157254	000052	MOV	@ML.REG+44,52(SP)			4885
8253	053156	017700	157244		MOV	@ML.REG+42,R0			4886
8254	053162	006200			ASR	R0			



```

8256          ;MLX4
8257          ;
8258
8259 053164 006200 ASR R0
8260 053166 006200 ASR R0
8261 053170 006200 ASR R0
8262 053172 006200 ASR R0
8263 053174 006200 ASR R0
8264 053176 042700 177700 BIC #177700,R0
8265 053202 010066 000050 MOV R0,50(SP) ; *,OLDCHN
8266 053206 012746 000001 MOV #1,-(SP) ;
8267 053212 016646 000042 MOV 42(SP),-(SP) ; COMMAND,*
8268 053216 010546 MOV R5,-(SP) ; LUN,*
8269 053220 010246 MOV R2,-(SP) ; WRDCNT,*
8270 053222 016646 000052 MOV 52(SP),-(SP) ; PTR,*
8271 053226 010446 MOV R4,-(SP) ; SECTOR,*
8272 053230 004737 043046 JSR PC,RETRY
8273 053234 062706 000014 ADD #14,SP
8274 053240 020027 000005 CMP R0,#5
8275 053244 001051 BNE 54$
8276 053246 027766 157156 000052 CMP @ML.REG+44,52(SP) ; *,OLDSEC
8277 053254 001034 BNE 53$
8278 053256 016646 000050 MOV 50(SP),-(SP) ; OLDCHN,*
8279 053262 017700 157140 MOV @ML.REG+42,R0
8280 053266 006200 ASR R0
8281 053270 006200 ASR R0
8282 053272 006200 ASR R0
8283 053274 006200 ASR R0
8284 053276 006200 ASR R0
8285 053300 006200 ASR R0
8286 053302 042700 177700 BIC #177700,R0
8287 053306 020026 CMP R0,(SP)+
8288 053310 001016 BNE 53$
8289 053312 032737 000001 002260 BIT #1,ERROUT ;
8290 053320 001404 BEQ 52$ ;
8291 053322 104456 TRAP 56 ;
8292 053324 000331 .WORD 331
8293 053326 010624 .WORD MSG4
8294 053330 000000 .WORD 0
8295 053332 010546 52$: MOV R5,-(SP) ; LUN,*
8296 053334 013746 032360 MOV BOARD,-(SP)
8297 053340 004737 035302 JSR PC,UP.HARD.COUNT
8298 053344 000426 BR 56$ ;
8299 053346 032737 000001 002260 53$: BIT #1,ERROUT ;
8300 053354 001415 BEQ 55$ ;
8301 053356 104457 TRAP 57 ;
8302 053360 000332 .WORD 332
8303 053362 010610 .WORD MSG3
8304 053364 000000 .WORD 0
8305 053366 000410 BR 55$ ;
8306 053370 032737 000001 002260 54$: BIT #1,ERROUT ;
8307 053376 001404 BEQ 55$ ;
8308 053400 104457 TRAP 57 ;
8309 053402 000333 .WORD 333
8310 053404 010610 .WORD MSG3

```

23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<

```

8312      :MLX4
8313      :
8314
8315 053406 000000      .WORD 0
8316 053410 010546      55$: MOV R5,-(SP)      ; LUN,*      4909
8317 053412 013746 032360      MOV BOARD,-(SP)
8318 053416 004737 035430      JSR PC,UP.SOFT.COUNT
8319 053422 022626      56$: CMP (SP)+,(SP)+      ;
8320 053424 062706 000010      57$: ADD #10,SP      ;
8321 053430 005266 000026      INC 26(SP)      ; KOUNT      4808
8322 053434 026627 000026 000002      CMP 26(SP),#2      ; KOUNT,*      4807
8323 053442 003002      BGT 58$
8324 053444 000137 052342      JMP 38$
8325 053450 010200      58$: MOV R2,R0      ; WRDCNT,*      4916
8326 053452 006300      ASL R0
8327 053454 063700 030706      ADD WPTR,R0
8328 053460 010037 030706      MOV R0,WPTR
8329 053464 010216      MOV R2,(SP)      ; WRDCNT,*      4917
8330 053466 012746 000400      MOV #400,-(SP)
8331 053472 004737 005056      JSR PC,BLSDIV
8332 053476 060400      ADD R4,R0      ; SECTOR,*
8333 053500 010004      MOV R0,R4      ; *,SECTOR
8334 053502 062706 000006      ADD #6,SP      ;
8335 053506 000137 052274      JMP 37$      ;
8336 053512 005205      59$: INC R5      ; LUN
8337 053514 020566 000030      60$: CMP R5,30(SP)      ; LUN,*
8338 053520 002002      BGE 61$
8339 053522 000137 052176      JMP 36$
8340 053526 062706 000010      61$: ADD #10,SP      ;
8341 053532 005266 000030      62$: INC 30(SP)      ; COUNT      4572
8342 053536 026666 000030 000032      CMP 30(SP),32(SP)      ; COUNT,*      4571
8343 053544 003002      BGT 63$
8344 053546 000137 050332      JMP 3$
8345 053552 062706 000036      63$: ADD #36,SP      ;
8346 053556 000207      RTS PC      ;

```

: Routine Size: 875 words  
: Maximum stack depth per invocation: 43 words

8347  
8348  
8349  
8354  
8355

```

8357 :MLX4
8358 :
8359 :
8360 : 4926 %SBTTL 'DEFINITION OF OPTION 3'
8361 : 4927
8362 : 4928 ROUTINE OPT3: NOVALUE =
8363 : 4929 BEGIN !* 1 * START OF ROUTINE
8364 : 4930
8365 : 4931 !++
8366 : 4932 ROUTINE: OPT3
8367 : 4933
8368 : 4934 PURPOSE: TO DO A UNIQUE DATA CHECK ON ALL AVAILABLE UNITS.
8369 : 4935
8370 : 4936 THE CODE FOR 'OPT3' IN BRIEF:
8371 : 4937
8372 : 4938 BEGIN 1 (START OF ROUTINE)
8373 : 4939 SAY ROUTINE IS RUNNING
8374 : 4940 INCR COMPLEMENT FLAG FROM 0 TO 1
8375 : 4941 : BEGIN 2 (START OF COMPLEMENT FLAG SELECTION LOOP)
8376 : 4942 : GENERATE THE PATTERN
8377 : 4943 : INCR LUN FROM 0 TO LAST
8378 : 4944 : : BEGIN 3 (START OF LOGICAL UNIT SELECTION LOOP)
8379 : 4945 : : TESTLOOP:
8380 : 4946 : : : BEGIN 4 (START OF LOOP THAT COMPLETELY TESTS 1 UNIT)
8381 : 4947 : : : IF UNIT IS ACTIVE
8382 : 4948 : : : THEN
8383 : 4949 : : : : BEGIN 5 (START OF TEST FOR AN ACTIVE UNIT)
8384 : 4950 : : : : INITIALIZE THE WRITE AND READ BUFFER POINTERS
8385 : 4951 : : : : SECTOR = LOWEST
8386 : 4952 : : : : WHILE SECTOR LEQ HIGHEST DO
8387 : 4953 : : : : : BEGIN 6 (START OF SECTOR SELECTION LOOP)
8388 : 4954 : : : : : GET WRDCNT
8389 : 4955 : : : : : SET UP BUFFER POINTERS BEFORE TRANSFER
8390 : 4956 : : : : : WRITE
8391 : 4957 : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
8392 : 4958 : : : : : CHOOSE WHETHER TO WRITE CHECK OR READ
8393 : 4959 : : : : : DO THE WRITE CHECK OR READ
8394 : 4960 : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
8395 : 4961 : : : : : CHANGE BUFFER POINTERS AFTER TRANSFER
8396 : 4962 : : : : : UPDATE SECTOR NUMBER BY NUMBER OF SECTORS IN PREVIOUS TRANSFER
8397 : 4963 : : : : : END 6 (END OF SECTOR SELECTION LOOP)
8398 : 4964 : : : : : END 5 (END OF TEST FOR AN ACTIVE UNIT)
8399 : 4965 : : : : : END 4 (END OF TESTLOOP)
8400 : 4966 : : : : : END 3 (END OF LOGICAL UNIT SELECTION LOOP)
8401 : 4967 : : : : : END 2 (END OF COMPLEMENT FLAG SELECTION LOOP)
8402 : 4968 : : : : : RETURN
8403 : 4969 : : : : : END 1 (END OF ROUTINE)
8404 : 4970 : : : : : --
8405 : 4971
8406 : 4972 LABEL
8407 : 4973 LOOP;
8408 : 4974
8409 : 4975 LOCAL
8410 : 4976 WRDCNT, VALUE, OLDSEC, OLDCHN, SECTOR, PTR, COMMAND, DBL_VALUE;
8411 : 4977
  
```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (43)

```

8413 :MLX4
8414 :
8415 :
8416 : 4978 PRINTB(SAY2,WRD34,RTN3);
8417 : 4979 !'RUNNING OPT3'
8418 : 4980
8419 : 4981 INCR COMP_FLAG FROM 0 TO 1 DO
8420 : 4982 BEGIN !* 2 * START OF COMPLEMENT FLAG SELECTION LOOP
8421 : 4983 GEN3(.COMP_FLAG);
8422 : 4984 INCR LUN FROM 0 TO (.L$UNIT - 1) DO
8423 : 4985 BEGIN !* 3 * START OF LOGICAL UNIT SELECTION LOOP
8424 : 4986 LOOP:
8425 : 4987 BEGIN !* 4 * START OF THE LOOP THAT COMPLETELY TESTS 1 UNIT
8426 : 4988 IF .DRIVE_STATUS[.LUN] EQL ACTIVE
8427 : 4989 THEN
8428 : 4990 BEGIN !* 5 * START OF TEST FOR AN ACTIVE UNIT
8429 : 4991 L$LUN = .LUN;
8430 : 4992 WPTR = WBUFF;
8431 : 4993 RPTR = RBUFF;
8432 : 4994 SECTOR = LOWEST;
8433 : 4995 WHILE .SECTOR LEQ HIGHEST DO
8434 : 4996 BEGIN !* 6 * START OF SECTOR SELECTION LOOP
8435 : 4997 WRDCNT = GET WRDCNT(.SECTOR,HIGHEST);
8436 : 4998 SET PTRS(.WRDCNT);
8437 : 4999 VALUE = WRITE(.LUN,.WRDCNT,.WPTR,.SECTOR);
8438 : 5000
8439 : 5001 ! SEE HOW SUCCESSFUL THE WRITE WAS:
8440 : 5002 !-
8441 : 5003 SELECTONE .VALUE OF !SEE 'SYSERR' FOR DEFINITION
8442 : 5004 SET !OF ERROR # CONTAINED IN 'VALUE'
8443 : 5005 [1] :
8444 : 5006 BEGIN !* 6A * REIRY ALLOWED
8445 : 5007 IF RETRY(SIX,WRITE,.LUN,.WRDCNT,.WPTR,.SECTOR) NEQ 0
8446 : 5008 THEN !THE RETRY FAILED -- SYSTEM FATAL ERROR
8447 : 5009 BEGIN
8448 : 5010 WHY DROPT[.LUN] = CODE_4;
8449 : 5011 ERRDF(301,MSG1,0); !**** OPTION 3 ERROR 01 ****
8450 : 5012 DODU(.LUN);
8451 : 5013 LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
8452 : 5014 END;
8453 : 5015 END; !* 6A *
8454 : 5016 [2] :
8455 : 5017 BEGIN !* 6B * FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
8456 : 5018 WHY DROPT[.LUN] = CODE_5;
8457 : 5019 ERRDF(302,MSG1,0); !**** OPTION 3 ERROR 02 ****
8458 : 5020 DODU(.LUN);
8459 : 5021 LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
8460 : 5022 END; !* 6B *
8461 : 5023 [3] :
8462 : 5024 BEGIN !* 6C * FATAL DRIVE ERROR -- NO RETRY ALLOWED
8463 : 5025 ERRDF(303,MSG1,0); !**** OPTION 3 ERROR 03 ****
8464 : 5026 WHY DROPT[.LUN] = CODE_6;
8465 : 5027 DODU(.LUN);
8466 : 5028 LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
8467 : 5029 END; !* 6C *
  
```

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (43)

8469 :MLX4  
 8470 :  
 8471 :  
 8472 : 5030  
 8473 : 5031  
 8474 : 5032  
 8475 : 5033  
 8476 : 5034  
 8477 : 5035  
 8478 : 5036  
 8479 : 5037  
 8480 : 5038  
 8481 : 5039  
 8482 : 5040  
 8483 : 5041  
 8484 : 5042  
 8485 : 5043  
 8486 : 5044  
 8487 : 5045  
 8488 : 5046  
 8489 : 5047  
 8490 : 5048  
 8491 : 5049  
 8492 : 5050  
 8493 : 5051  
 8494 : 5052  
 8495 : 5053  
 8496 : 5054  
 8497 : 5055  
 8498 : 5056  
 8499 : 5057  
 8500 : 5058  
 8501 : 5059  
 8502 : 5060  
 8503 : 5061  
 8504 : 5062  
 8505 : 5063  
 8506 : 5064  
 8507 : 5065  
 8508 : 5066  
 8509 : 5067  
 8510 : 5068  
 8511 : 5069  
 8512 : 5070  
 8513 : 5071  
 8514 : 5072  
 8515 : 5073  
 8516 : 5074  
 8517 : 5075  
 8518 : 5076  
 8519 : 5077  
 8520 : 5078  
 8521 : 5079  
 8522 : 5080  
 8523 : 5081

DEFINITION OF OPTION 3

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (43)

```

TES:
COMMAND = CHOOSE();
IF .COMMAND EQL READ
THEN
  BEGIN
    PTR = .RPTR;
    VALUE = READ(.LUN,.WRDCNT,.RPTR,.SECTOR);
  END
ELSE
  BEGIN
    PTR = .WPTR;
    VALUE = CHECK(.LUN,.WRDCNT,.WPTR,.SECTOR);
  END;
+
! SEE HOW SUCCESSFUL THE OPERATION WAS:
!-
SELECTONE .VALUE OF      !SEE 'SYSERR' FOR DEFINITION
SET                      !OF ERROR # CONTAINED IN 'VALUE'
[0] :
  IF .COMMAND EQL READ
  THEN
  BEGIN
    IF (DBL_VALUE = DOUBLE_CHECK(.WPTR,.RPTR,.WRDCNT)) NEQ 0
    THEN
    BEGIN
      SAYWHO(.LUN);
      PRINTB(SAY1,MSG5);
      !'ECC LOGIC FAILED TO DETECT DATA ERROR'
      PRINTB(FMT12A,..DBL_VALUE,..DBL_VALUE);
      !'GOOD DATA: XXXXXX AT LOCATION YYYYYY'
      DBL_VALUE = .DBL_VALUE + BUFSIZ * 2;
      PRINTB(FMT12B,..DBL_VALUE,..DBL_VALUE);
      !'BAD DATA: PPPPPP AT LOCATION QQQQQQ'
      WHY DROPT[.LUN] = CODE 8;
      ERRDF(304,MSG1,0);  !*** OPTION 3 ERROR 04 ***
      DODU(.LUN);
      LEAVE LOOP;      !JUMP JUST BEYOND END OF BLOCK * 4 *
    END;
  END;
[1] :
  BEGIN !* 6D *      RETRY ALLOWED
  IF RETRY(SIX,.COMMAND,.LUN,.WRDCNT,.PTR,.SECTOR) NEQ 0
  THEN !THE RETRY FAILED -- SYSTEM FATAL ERROR
  BEGIN
    WHY DROPT[.LUN] = CODE 4;
    ERRDF(305,MSG1,0);  !*** OPTION 3 ERROR 05 ***
    DODU(.LUN);
    LEAVE LOOP;      !JUMP JUST BEYOND END OF BLOCK * 4 *
  END;
END;  !* 6D *
  
```

```

8525 :MLX4
8526 :
8527 :
8528 : 5082
8529 : 5083
8530 : 5084
8531 : 5085
8532 : 5086
8533 : 5087
8534 : 5088
8535 : 5089
8536 : 5090
8537 : 5091
8538 : 5092
8539 : 5093
8540 : 5094
8541 : 5095
8542 : 5096
8543 : 5097
8544 : 5098
8545 : 5099
8546 : 5100
8547 : 5101
8548 : 5102
8549 : 5103
8550 : 5104
8551 : 5105
8552 : 5106
8553 : 5107
8554 : 5108
8555 : 5109
8556 : 5110
8557 : 5111
8558 : 5112
8559 : 5113
8560 : 5114
8561 : 5115
8562 : 5116
8563 : 5117
8564 : 5118
8565 : 5119
8566 : 5120
8567 : 5121
8568 : 5122
8569 : 5123
8570 : 5124
8571 : 5125
8572 : 5126
8573 : 5127
8574 : 5128
8575 : 5129
8576 : 5130
8577 : 5131
8578 : 5132
8579 : 5133

      DEFINITION OF OPTION 3

      [2] :
      BEGIN !* 6E *          FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
      WHY DROPT[.LUN] = CODE_5;
      ERRDF(306,MSG1,0); !**** OPTION 3 ERROR 06 ****
      DODU(.LUN);
      LEAVE LOOP;          !JUMP JUST BEYOND END OF BLOCK * 4 *
      END; !* 6E *

      [3] :
      BEGIN !* 6F *          FATAL DRIVE ERROR -- NO RETRY ALLOWED
      ERRDF(307,MSG1,0); !**** OPTION 3 ERROR 07 ****
      WHY DROPT[.LUN] = CODE_6;
      DODU(.LUN);
      LEAVE LOOP;          !JUMP JUST BEYOND END OF BLOCK * 4 *
      END; !* 6F *

      [4] :
      BEGIN !* 6G *          UNRECOVERABLE DATA ERROR
      ISOLATE();
      FRRDF(308,MSG2,0); !**** OPTION 3 ERROR 08 ****
      WHY DROPT[.LUN] = CODE_7;
      DODU(.LUN);
      LEAVE LOOP;          !JUMP JUST BEYOND END OF BLOCK * 4 *
      END; !* 6G *

      [5] :
      BEGIN !* 6H *          RECOVERABLE DATA FRORR
      ISOLATE();
      IF .ERROUT
      THEN PRINTB(FMT10B,.CHAN);
      !' BIT QQ'
      OLDSEC = .MLEL;
      OLDCHN = .CHAN;
      IF RETRY(ONE,.COMMAND,.LUN,.WRDCNT,.PTR,.SECTOR) EQL 5
      THEN
      IF ((.MLEL EQL .OLDSEC) AND (.CHAN EQL .OLDCHN))
      THEN
      BEGIN
      IF .ERROUT
      THEN
      ERRHRD(309,MSG4,0); !**** OPTION 3 ERROR 09 ****
      UP_HARD_COUNT(.LUN,.BOARD);
      END
      ELSE
      BEGIN
      IF .ERROUT
      THEN
      ERRSOFT(310,MSG3,0); !**** OPTION 3 ERROR 10 ****
      UP_SOFT_COUNT(.LUN,.BOARD);
      END
      ELSE
      BEGIN
      IF .ERROUT
      THEN
      ERRSOFT(311,MSG3,0); !**** OPTION 3 ERROR 11 ****
  
```

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (43)

```

8581 ;MLX4
8582 :
8583 :
8584 : 5134 UP_SOFT_COUNT(.LUN,.BOARD);
8585 : 5135 END;
8586 : 5136 END; !* 6H *
8587 : 5137
8588 : 5138 TES;
8589 : 5139
8590 : 5140 WPTR = .WPTR + (.WRDCNT * 2);
8591 : 5141 SECTOR = .SECTOR + (.WRDCNT/256);
8592 : 5142
8593 : 5143 END; !* 6 * END OF SECTOR SELECTION LOOP
8594 : 5144 END; !* 5 * END OF TEST FOR AN ACTIVE UNIT
8595 : 5145 END; !* 4 * END OF TESTLOOP
8596 : 5146 END; !* 3 * END OF LOGICAL UNIT SELECTION LOOP
8597 : 5147 END; !* 2 * END OF COMPLEMENT FLAG SELECTION LOOP
8598 : 5148 RETURN;
8599 : 5149 END; !* 1 * END OF ROUTINE
  
```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (43)

```

8603
8604
8608 053560 004137 005222 OPT3: .SBITL OPT3 DEFINITION OF OPTION 3 ; 4928
8609 053564 162706 000020 JSR R1,$SAVE5 ;
8610 053570 012746 007264 SUB #20,SP ; 4978
8611 053574 012746 006766 MOV #RTN3,-(SP) ;
8612 053600 012746 006522 MOV #WRD34,-(SP) ;
8613 053604 012746 000003 MOV #SAY2,-(SP) ;
8614 053610 010600 MOV #3,-(SP) ;
8615 053612 104414 MOV SP,R0 ; SP,*
8616 053614 005066 000026 TRAP 14 ;
8617 053620 016646 000026 CLR 26(SP) ; COMP.FLAG 4981
8618 053624 004737 036556 1$: MOV 26(SP),-(SP) ; COMP.FLAG,* 4983
8619 053630 013766 002012 000014 JSR PC,GEN3 ;
8620 053636 005066 000012 MOV L$UNIT,14(SP) ; 4984
8621 053642 000137 055426 CLR 12(SP) ; LUN
8622 053646 016600 000012 2$: JMP 29$ ; LUN,* 4988
8623 053652 006200 ASR R0 ;
8624 053654 006200 ASR R0 ;
8625 053656 006200 ASR R0 ;
8626 053660 062700 032460 ADD #DRIVE.STATUS,R0 ;
8627 053664 010046 MOV R0,-(SP) ;
8628 053666 016646 000014 MOV 14(SP),-(SP) ; LUN,*
8629 053672 042716 177770 BIC #177770,(SP) ;
8630 053676 012746 000001 MOV #1,-(SP) ;
8631 053702 005046 CLR -(SP) ;
8632 053704 004737 004244 JSR PC,BL$GT2 ;
8633 053710 062706 000010 ADD #10,SP ;
8634 053714 005300 DEC R0 ;
8635 053716 001143 BNE 7$ ;
  
```

Address	Op1	Op2	Op3	Op4	Op5	Instruction	Comments	Seq
8637								
8638								
8639								
8640	053720	016637	000012	002074		MOV 12(SP),L\$LUN	: LUN,*	4991
8641	053726	012737	010706	030706		MOV #WBUF,WPTR	:	4992
8642	053734	012737	020706	030710		MOV #RBUF,RPTR	:	4993
8643	053742	016601	000012			MOV 12(SP),R1	: LUN,*	4994
8644	053746	006301				ASL R1		
8645	053750	016104	032476			MOV LOW.SECT(R1),R4	: *,SECTOR	
8646	053754	020461	032516		3\$:	CMP R4,TOP.SECT(R1)	: SECTOR,*	4995
8647	053760	003122				BGT 7\$		
8648	053762	010446				MOV R4,-(SP)	: SECTOR,*	4997
8649	053764	016146	032516			MOV TOP.SECT(R1),-(SP)		
8650	053770	004737	044034			JSR PC,GET.WRDCNT		
8651	053774	010003				MOV R0,R3	: *,WRDCNT	
8652	053776	010316				MOV R3,(SP)	: WRDCNT,*	4998
8653	054000	004737	043770			JSR PC,SET.PTRS		
8654	054004	016616	000016			MOV 16(SP),(SP)	: LUN,*	4999
8655	054010	010346				MOV R3,-(SP)	: WRDCNT,*	
8656	054012	013746	030706			MOV WPTR,-(SP)		
8657	054016	010446				MOV R4,-(SP)	: SECTOR,*	
8658	054020	004737	042246			JSR PC,WRITE		
8659	054024	010005				MOV R0,R5	: *,VALUE	
8660	054026	020527	000001			CMP R5,#1	: VALUE,*	5003
8661	054032	001035				BNE 4\$		
8662	054034	012746	000006			MOV #6,-(SP)	:	5007
8663	054040	012746	042246			MOV #WRITE,-(SP)		
8664	054044	016646	000030			MOV 30(SP),-(SP)	: LUN,*	
8665	054050	010346				MOV R3,-(SP)	: WRDCNT,*	
8666	054052	013746	030706			MOV WPTR,-(SP)		
8667	054056	010446				MOV R4,-(SP)	: SECTOR,*	
8668	054060	004737	043046			JSR PC,RETRY		
8669	054064	062706	000014			ADD #14,SP		
8670	054070	005700				TST R0		
8671	054072	001456				BEQ 8\$		
8672	054074	016602	000024			MOV 24(SP),R2	: LUN,*	5010
8673	054100	112762	000004	032464		MOVB #4,WHY.DROPT(R2)	:	5011
8674	054106	104455				TRAP 55		
8675	054110	000455				.WORD 455		
8676	054112	010512				.WORD MSG1		
8677	054114	000000				.WORD 0		
8678	054116	016600	000024			MOV 24(SP),R0	: LUN,*	5012
8679	054122	104451				TRAP 51		
8680	054124	000436				BR 6\$		5013
8681	054126	020527	000002		4\$:	CMP R5,#2	: VALUE,*	5003
8682	054132	001015				BNE 5\$		
8683	054134	016602	000024			MOV 24(SP),R2	: LUN,*	5018
8684	054140	112762	000005	032464		MOVB #5,WHY.DROPT(R2)	:	5019
8685	054146	104455				TRAP 55		
8686	054150	000456				.WORD 456		
8687	054152	010512				.WORD MSG1		
8688	054154	000000				.WORD 0		
8689	054156	016600	000024			MOV 24(SP),R0	: LUN,*	5020
8690	054162	104451				TRAP 51		
8691	054164	000416				BR 6\$	:	5021



Address	Label	OpCode	OpData	OpLabel	OpComment	SeqNo
8693				;MLX4		
8694				:	DEFINITION OF OPTION 3	
8695						
8696	054166	020527	000003	5\$:	CMP R5,#3	; VALUE,* 5003
8697	054172	001016			BNE 8\$	
8698	054174	104455			TRAP 55	; 5025
8699	054176	000457			.WORD 457	
8700	054200	010512			.WORD MSG1	
8701	054202	000000			.WORD 0	
8702	054204	016602	000024		MOV 24(SP),R2	; LUN,* 5026
8703	054210	112762	000006 032464		MOVB #6,WHY.DROPT(R2)	
8704	054216	010200			MOV R2,RO	; LUN,* 5027
8705	054220	104451			TRAP 51	
8706	054222	062706	000012	6\$:	ADD #12,SP	; 5028
8707	054226	000543		7\$:	BR 13\$	
8708	054230	004737	043010	8\$:	JSR PC,CHOOSE	; 5033
8709	054234	010066	000036		MOV RO,36(SP)	; *,COMMAND
8710	054240	005002			CLR R2	; 5034
8711	054242	020027	042434		CMP RO,#READ	; COMMAND,*
8712	054246	001015			BNE 9\$	
8713	054250	005202			INC R2	
8714	054252	013766	030710 000034		MOV RPTR,34(SP)	; *,PTR 5037
8715	054260	016646	000024		MOV 24(SP),-(SP)	; LUN,* 5038
8716	054264	010346			MOV R3,-(SP)	; WRDCNT,*
8717	054266	013746	030710		MOV RPTR,-(SP)	
8718	054272	010446			MOV R4,-(SP)	; SECTOR,*
8719	054274	004737	042434		JSR PC,READ	
8720	054300	000413			BR 10\$	
8721	054302	013766	030706 000034	9\$:	MOV WPTR,34(SP)	; *,PTR 5042
8722	054310	016646	000024		MOV 24(SP),-(SP)	; LUN,* 5043
8723	054314	010346			MOV R3,-(SP)	; WRDCNT,*
8724	054316	013746	030706		MOV WPTR,-(SP)	
8725	054322	010446			MOV R4,-(SP)	; SECTOR,*
8726	054324	004737	042622		JSR PC,CHECK	
8727	054330	010005		10\$:	MOV RO,R5	; *,VALUE
8728	054332	001102			BNE 14\$	; 5048
8729	054334	006002			ROR R2	; 5051
8730	054336	103402			BLO 12\$	
8731	054340	000137	055360	11\$:	JMP 27\$	
8732	054344	013746	030706	12\$:	MOV WPTR,-(SP)	; 5054
8733	054350	013746	030710		MOV RPTR,-(SP)	
8734	054354	010346			MOV R3,-(SP)	; WRDCNT,*
8735	054356	004737	041226		JSR PC,DOUBLE.CHECK	
8736	054362	062706	000006		ADD #6,SP	
8737	054366	010066	000050		MOV RO,50(SP)	; *,DBL.VALUE
8738	054372	001762			BEQ 11\$	
8739	054374	016646	000034		MOV 34(SP),-(SP)	; LUN,* 5057
8740	054400	004737	033436		JSR PC,SAYWHO	
8741	054404	012716	010640		MOV #MSG5,(SP)	; 5058
8742	054410	012746	006514		MOV #SAY1,-(SP)	
8743	054414	012746	000002		MOV #2,-(SP)	
8744	054420	010600			MOV SP,RO	; SP,*
8745	054422	104414			TRAP 14	
8746	054424	016616	000056		MOV 56(SP),(SP)	; DBL.VALUE,* 5060
8747	054430	017646	000056		MOV @56(SP),-(SP)	; DBL.VALUE,*

```

8749           ;MLX4
8750           ;
8751
8752 054434 012746 006332      MOV   #FMT12A,-(SP)
8753 054440 012746 000003      MOV   #3,-(SP)
8754 054444 010600             MOV   SP,R0                ; SP,*
8755 054446 104414             TRAP  14
8756 054450 062766 010000 000064  ADD   #10000,64(SP)        ; *,DBL.VALUE 5062
8757 054456 016616 000064             MOV   64(SP),(SP)         ; DBL.VALUE,* 5063
8758 054462 017646 000064             MOV   @64(SP),-(SP)       ; DBL.VALUE,*
8759 054466 012746 006400             MOV   #FMT12B,-(SP)
8760 054472 012746 000003             MOV   #3,-(SP)
8761 054476 010600             MOV   SP,R0                ; SP,*
8762 054500 104414             TRAP  14
8763 054502 016602 000056             MOV   56(SP),R2          ; LUN,* 5065
8764 054506 112762 000010 032464  MOVB  #10,WHY.DROPT(R2)
8765 054514 104455             TRAP  55                  ; 5066
8766 054516 000460             .WORD 460
8767 054520 010512             .WORD MSG1
8768 054522 000000             .WORD 0
8769 054524 016600 000056             MOV   56(SP),R0          ; LUN,* 5067
8770 054530 104451             TRAP  51
8771 054532 062706 000044             ADD   #44,SP              ; 5068
8772 054536 000521             BR    19$                 ;
8773 054540 020527 000001             CMP   R5,#1              ; /ALUE,* 5048
8774 054544 001035             BNE   15$                 ; 5073
8775 054546 012746 000006             MOV   #6,-(SP)           ;
8776 054552 016646 000050             MOV   50(SP),-(SP)       ; COMMAND,*
8777 054556 016646 000040             MOV   40(SP),-(SP)       ; LUN,*
8778 054562 010346             MOV   R3,-(SP)           ; WRDCNT,*
8779 054564 016646 000054             MOV   54(SP),-(SP)       ; PTR,*
8780 054570 010446             MOV   R4,-(SP)           ; SECTOR,*
8781 054572 004737 043046             JSR   PC,RETRY
8782 054576 062706 000014             ADD   #14,SP
8783 054602 005700             TST   R0
8784 054604 001655             BEQ   11$
8785 054606 016602 000034             MOV   34(SP),R2          ; LUN,* 5076
8786 054612 112762 000004 032464  MOVB  #4,WHY.DROPT(R2)
8787 054620 104455             TRAP  55                  ; 5077
8788 054622 000461             .WORD 461
8789 054624 010512             .WORD MSG1
8790 054626 000000             .WORD 0
8791 054630 016600 000034             MOV   34(SP),R0          ; LUN,* 5078
8792 054634 104451             TRAP  51
8793 054636 000457             BR    18$                 ; 5079
8794 054640 020527 000002             CMP   R5,#2              ; VALUE,* 5048
8795 054644 001015             BNE   16$                 ;
8796 054646 016602 000034             MOV   34(SP),R2          ; LUN,* 5084
8797 054652 112762 000005 032464  MOVB  #5,WHY.DROPT(R2)
8798 054660 104455             TRAP  55                  ; 5085
8799 054662 000462             .WORD 462
8800 054664 010512             .WORD MSG1
8801 054666 000000             .WORD 0
8802 054670 016600 000034             MOV   34(SP),R0          ; LUN,* 5086
8803 054674 104451             TRAP  51

```

Address	Label	OpCode	OpData	OpLabel	OpText	OpComment	Seq
8805							
8806							
8807							
8808	054676	000437			BR	18\$	5087
8809	054700	020527	000003	16\$:	CMP	R5,#3	5048
8810	054704	001014			BNE	17\$	
8811	054706	104455			TRAP	55	5091
8812	054710	000463			.WORD	463	
8813	054712	010512			.WORD	MSG1	
8814	054714	000000			.WORD	0	
8815	054716	016602	000034		MOV	34(SP),R2	5092
8816	054722	112762	000006	032464	MOVB	#6,WHY.DROPT(R2)	
8817	054730	010200			MOV	R2,R0	5093
8818	054732	104451			TRAP	51	
8819	054734	000420			BR	18\$	5094
8820	054736	020527	000004	17\$:	CMP	R5,#4	5048
8821	054742	001021			BNE	20\$	
8822	054744	004737	035212		JSR	PC,ISOLATE	5098
8823	054750	104455			TRAP	55	5099
8824	054752	000464			.WORD	464	
8825	054754	010542			.WORD	MSG2	
8826	054756	000000			.WORD	0	
8827	054760	016602	000034		MOV	34(SP),R2	5100
8828	054764	112762	000007	032464	MOVB	#7,WHY.DROPT(R2)	
8829	054772	010200			MOV	R2,R0	5101
8830	054774	104451			TRAP	51	
8831	054776	062706	000022	18\$:	ADD	#22,SP	5102
8832	055002	000137	055422	19\$:	JMP	28\$	
8833	055006	020527	000005	20\$:	CMP	R5,#5	5048
8834	055012	001162			BNE	27\$	
8835	055014	004737	035212		JSR	PC,ISOLATE	5106
8836	055020	032737	000001	002260	BIT	#1,ERROUT	5107
8837	055026	001423			BEQ	21\$	
8838	055030	017702	155372		MOV	@ML.REG+42,R2	5108
8839	055034	006202			ASR	R2	
8840	055036	006202			ASR	R2	
8841	055040	006202			ASR	R2	
8842	055042	006202			ASR	R2	
8843	055044	006202			ASR	R2	
8844	055046	006202			ASR	R2	
8845	055050	042702	177700		BIC	#177700,R2	
8846	055054	010246			MOV	R2,-(SP)	
8847	055056	012746	006262		MOV	#FMT10B,-(SP)	
8848	055062	012746	000002		MOV	#2,-(SP)	
8849	055066	010600			MOV	SP,R0	
8850	055070	104414			TRAP	14	
8851	055072	062706	000006		ADD	#6,SP	
8852	055076	017766	155326	000040	MOV	@ML.REG+44,40(SP)	5110
8853	055104	017702	155316		MOV	@ML.REG+42,R2	5111
8854	055110	006202			ASR	R2	
8855	055112	006202			ASR	R2	
8856	055114	006202			ASR	R2	
8857	055116	006202			ASR	R2	
8858	055120	006202			ASR	R2	
8859	055122	006202			ASR	R2	

23-Oct-1980 15:01:43 TOPS  
 23-Oct-1980 14:57:05 PA:<

8861					:MLX4	DEFINITION OF OPTION 3	23-Oct-1980 15:01:43	TOPS
8862					:		23-Oct-1980 14:57:05	PA:<
8863								
8864	055124	042702	177700			BIC #177700,R2		
8865	055130	010266	000042			MOV R2,42(SP)	:*,OLDCHN	
8866	055134	012746	000001			MOV #1,-(SP)	:	5112
8867	055140	016646	000050			MOV 50(SP),-(SP)	:COMMAND,*	
8868	055144	016646	000040			MOV 40(SP),-(SP)	:LUN,*	
8869	055150	010346				MOV R3,-(SP)	:WRDCNT,*	
8870	055152	016646	000054			MOV 54(SP),-(SP)	:PTR,*	
8871	055156	010446				MOV R4,-(SP)	:SECTOR,*	
8872	055160	004737	043046			JSR PC,RETRY		
8873	055164	062706	000014			ADD #14,SP		
8874	055170	020027	000005			CMP R0,#5		
8875	055174	001052				BNE 24\$		
8876	055176	027766	155226	000040		CMP @ML.REG+44,40(SP)	:*,OLDSEC	5114
8877	055204	001035				BNF 23\$		
8878	055206	016600	000042			MOV 42(SP),R0	:OLDCHN,*	
8879	055212	017702	155210			MOV @ML.REG+42,R2		
8880	055216	006202				ASR R2		
8881	055220	006202				ASR R2		
8882	055222	006202				ASR R2		
8883	055224	006202				ASR R2		
8884	055226	006202				ASR R2		
8885	055230	006202				ASR R2		
8886	055232	042702	177700			BIC #177700,R2		
8887	055236	020200				CMP R2,R0		
8888	055240	001017				BNE 23\$		
8889	055242	032737	000001	002260		BIT #1,ERROUT	:	5117
8890	055250	001404				BEQ 22\$	:	
8891	055252	104456				TRAP 56	:	5119
8892	055254	000465				.WORD 465		
8893	055256	010624				.WORD MSG4		
8894	055260	000000				.WORD 0		
8895	055262	016646	000034		22\$:	MOV 34(SP),-(SP)	:LUN,*	5120
8896	055266	013746	032360			MOV BOARD,-(SP)		
8897	055272	004737	035302			JSR PC,UP.HARD.COUNT		
8898	055276	000427				BR 26\$	:	5114
8899	055300	032737	000001	002260	23\$:	BIT #1,ERROUT	:	5124
8900	055306	001415				BEQ 25\$	:	
8901	055310	104457				TRAP 57	:	5126
8902	055312	000466				.WORD 466		
8903	055314	010610				.WORD MSG3		
8904	055316	000000				.WORD 0		
8905	055320	000410				BR 25\$	:	5127
8906	055322	032737	000001	002260	24\$:	BIT #1,ERROUT	:	5131
8907	055330	001404				BEQ 25\$	:	
8908	055332	104457				TRAP 57	:	5133
8909	055334	000467				.WORD 467		
8910	055336	010610				.WORD MSG3		
8911	055340	000000				.WORD 0		
8912	055342	016646	000034		25\$:	MOV 34(SP),-(SP)	:LUN,*	5134
8913	055346	013746	032360			MOV BOARD,-(SP)		
8914	055352	004737	035430			JSR PC,UP.SOFT.COUNT		
8915	055356	022626			26\$:	CMP (SP)+,(SP)+	:	5105

```

8917 ;MLX4
8918 ;
8919 ;
8920 055360 010300 27$: MOV R3,R0 ; WRDCNT,* 5140
8921 055362 006300 ASL R0
8922 055364 063700 030706 ADD WPTR,R0
8923 055370 010037 030706 MOV R0,WPTR
8924 055374 010316 MOV R3,(SP) ; WRDCNT,* 5141
8925 055376 012746 000400 MOV #400,-(SP)
8926 055402 004737 005056 JSR PC,BL$DIV
8927 055406 060400 ADD R4,R0 ; SECTOR,*
8928 055410 010004 MOV R0,R4 ; *,SECTOR
8929 055412 062706 000024 ADD #24,SP ; 4996
8930 055416 000137 053754 JMP 3$ ; 4995
8931 055422 005266 000012 28$: INC 12(SP) ; LUN 4984
8932 055426 026666 000012 000014 29$: CMP 12(SP),14(SP) ; LUN,*
8933 055434 002002 BGE 30$
8934 055436 000137 053646 JMP 2$
8935 055442 005726 30$: TST (SP)+ ; 4982
8936 055444 005266 000026 INC 26(SP) ; COMP.FLAG 4981
8937 055450 026627 000026 000001 CMP 26(SP),#1 ; COMP.FLAG,*
8938 055456 003002 BGT 31$
8939 055460 000137 053620 JMP 1$
8940 055464 062706 000030 31$: ADD #30,SP ; 4928
8941 055470 000207 RTS PC
8942
8943 ; Routine Size: 485 words
8944 ; Maximum stack depth per invocation: 37 words
8949
8950
  
```

23-Oct-1980 15:01:43 TOPS  
 23-Oct-1980 14:57:05 PA:<

8952 :MLX4

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (44)

```

8953 :      DEFINITION OF OPTION 4
8954 :
8955 :      5150 %SBTTL 'DEFINITION OF OPTION 4'
8956 :      5151
8957 :      5152 ROUTINE OPT4: NOVALUE =
8958 :      5153 BEGIN !* 1 * START OF ROUTINE
8959 :      5154
8960 :      5155 !++
8961 :      5156 ROUTINE:      OPT4
8962 :      5157
8963 :      5158 PURPOSE:      TO LOOK FOR INTERACTIONS BETWEEN SECTORS
8964 :      5159 USING A MARCH TEST.
8965 :      5160
8966 :      5161 THE CODE IN BRIEF:
8967 :      5162
8968 :      5163 BEGIN 1 (START OF ROUTINE)
8969 :      5164 SAY ROUTINE IS RUNNING
8970 :      5165 WORD COUNT = 256
8971 :      5166 GENERATE A BUFFER OF DATA
8972 :      5167 GENERATE A BUFFER OF COMP
8973 :      5168 INITIALIZE POINTERS TO 4 BUFFERS FOR WRITE/READ DATA/COMP
8974 :      5169 INCR LUN FROM 0 TO LAST
8975 :      5170 : BEGIN 2 (START OF LOGICAL UNIT SELECTION LOOP)
8976 :      5171 : TESTLOOP:
8977 :      5172 : : BEGIN 3 (START OF LOOP THAT COMPLETELY TESTS 1 UNIT)
8978 :      5173 : : IF UNIT IS ACTIVE
8979 :      5174 : : THEN
8980 :      5175 : : : BEGIN 4 (START OF TEST FOR AN ACTIVE UNIT)
8981 :      5176 : : : INCR SECTOR FROM LOWEST TO HIGHEST
8982 :      5177 : : : : BEGIN 4A (START OF 1ST SECTOR SELECTION LOOP)
8983 :      5178 : : : : WRITE 'DATA'
8984 :      5179 : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
8985 :      5180 : : : : END 4A (END OF 1ST SECTOR SELECTION LOOP)
8986 :      5181 : : : : CHOOSE WHETHER TO WRITE CHECK OR READ 'DATA'
8987 :      5182 : : : : INCR SECTOR FROM LOWEST TO HIGHEST
8988 :      5183 : : : : BEGIN 4B (START OF 2ND SECTOR SELECTION LOOP)
8989 :      5184 : : : : DO THE WRITE CHECK OR READ OF 'DATA'
8990 :      5185 : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
8991 :      5186 : : : : WRITE 'COMP'
8992 :      5187 : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
8993 :      5188 : : : : END 4B (END OF 2ND SECTOR SELECTION LOOP)
8994 :      5189 : : : : CHOOSE WHETHER TO WRITE CHECK OR READ 'COMP'
8995 :      5190 : : : : DECR SECTOR FROM HIGHEST TO LOWEST
8996 :      5191 : : : : BEGIN 4C (START OF 3RD SECTOR SELECTION LOOP)
8997 :      5192 : : : : DO THE WRITE CHECK OR READ OF 'COMP'
8998 :      5193 : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
8999 :      5194 : : : : WRITE DATA
9000 :      5195 : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
9001 :      5196 : : : : END 4C (END OF 3RD SECTOR SELECTION LOOP)
9002 :      5197 : : : : CHOOSE WHETHER TO WRITE CHECK OR READ 'DATA'
9003 :      5198 : : : : INCR SECTOR FROM LOWEST TO HIGHEST
9004 :      5199 : : : : BEGIN 4D (START OF 4TH SECTOR SELECTION LOOP)
9005 :      5200 : : : : DO THE WRITE CHECK OR READ OF 'DATA'
9006 :      5201 : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
  
```

9008 :MLX4

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (44)

9009 :  
 9010 :  
 9011 :  
 9012 :  
 9013 :  
 9014 :  
 9015 :  
 9016 :  
 9017 :  
 9018 :  
 9019 :  
 9020 :  
 9021 :  
 9022 :  
 9023 :  
 9024 :  
 9025 :  
 9026 :  
 9027 :  
 9028 :  
 9029 :  
 9030 :  
 9031 :  
 9032 :  
 9033 :  
 9034 :  
 9035 :  
 9036 :  
 9037 :  
 9038 :  
 9039 :  
 9040 :  
 9041 :  
 9042 :  
 9043 :  
 9044 :  
 9045 :  
 9046 :  
 9047 :  
 9048 :  
 9049 :  
 9050 :  
 9051 :  
 9052 :  
 9053 :  
 9054 :  
 9055 :  
 9056 :  
 9057 :  
 9058 :  
 9059 :  
 9060 :  
 9061 :  
 9062 :

```

DEFINITION OF OPTION 4
5202 : : : : END 4D (END OF 4TH SECTOR SELECTION LOOP)
5203 : : : : END 4 (END OF TEST FOR AN ACTIVE UNIT)
5204 : : : : END 3 (END OF TESTLOOP)
5205 : : : : END 2 (END OF LOGICAL UNIT SELECTION LOOP)
5206 : : : : RETURN
5207 : : : : END 1 (END OF ROUTINE)
5208 : : : : --
5209 : : : :
5210 LABEL
5211 LOOP:
5212
5213 LOCAL
5214 VALUE, OLDSEC, OLDCHN, PTR, COMMAND, DBL_VALUE,
5215 WDPTR, RDPTR, WCPTR, RCPTR;
5216
5217 PRINTB(SAY2,WRD34,RTN4);
5218 !:RUNNING OPT4;
5219
5220 GEN4(.MARPAT,WDBUFF);
5221 GEN4((-MARPAT),WCBUFF);
5222 WDPTR = WDBUFF; !POINTER TO 256-WORD WRITE DATA BUFFER
5223 WCPTR = WCBUFF; !POINTER TO 256-WORD WRITE COMP BUFFER
5224 RDPTR = RDBUFF; !POINTER TO 256-WORD READ DATA BUFFER
5225 RCPTR = RCBUFF; !POINTER TO 256-WORD READ COMP BUFFER
5226
5227 INCR LUN FROM 0 TO (.L$UNIT - 1) DO
5228 BEGIN !* 2 * START OF LOGICAL UNIT SELECTION LOOP
5229 LOOP:
5230 BEGIN !* 3 * START OF THE LOOP THAT COMPLETELY TESTS 1 UNIT
5231 IF .DRIVE_STATUS[LUN] EQL ACTIVE
5232 THEN
5233 BEGIN !* 4 * START OF TEST FOR AN ACTIVE UNIT
5234 L$LUN = .LUN;
5235 PRINTB(SAY1,WRD35);
5236 !'MARCHING'
5237 PRINTB(FMT2,WRD24);
5238 !' UP'
5239 INCR SECTOR FROM LOWEST TO HIGHEST DO
5240 BEGIN !* 4A * START OF 1ST SECTOR SELECTION LOOP
5241 VALUE = WRITE(.LUN,256,.WDPTR,.SECTOR);
5242 !+
5243 !SEE HOW SUCCESSFUL THE WRITE WAS:
5244 !-
5245 SELECTONE .VALUE OF !SEE 'SYSERR' FOR DEFINITION
5246 SET !OF ERROR # CONTAINED IN 'VALUE'
5247 [1] :
5248 BEGIN !* 4A1 * RETRY ALLOWED
5249 IF RETRY(SIX,WRITE,.LUN,256,.WDPTR,.SECTOR) NEQ 0
5250 THEN !THE RETRY FAILED -- SYSTEM FATAL ERROR
5251 BEGIN
5252 WHY DROPT[LUN] = CODE 4;
5253 ERRDF(401,MSG1,0); !*** OPTION 4 ERROR 01 ****

```

```

9064 :MLX4
9065 :
9066 :
9067 : 5254          DODU(.LUN);
9068 : 5255          LEAVE LOOP;          .JUMP JUST BEYOND END OF BLOCK * 3 *
9069 : 5256          END;
9070 : 5257          END;      !* 4A1 *
9071 : 5258          [2] :
9072 : 5259          BEGIN !* 4A2 * FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
9073 : 5260          WHY DROPT(.LUN) = CODE_5;
9074 : 5261          ERRDF(402,MSG1,0);    !**** OPTION 4 ERROR 02 ****
9075 : 5262          DODU(.LUN);
9076 : 5263          LEAVE LOOP;          !JUMP JUST BEYOND END OF BLOCK * 3 *
9077 : 5264          END;      !* 4A2 *
9078 : 5265          [3] :
9079 : 5266          BEGIN !* 4A3 * FATAL DRIVE ERROR -- NO RETRY ALLOWED
9080 : 5267          ERRDF(403,MSG1,0);    !**** OPTION 4 ERROR 03 ****
9081 : 5268          WHY DROPT(.LUN) = CODE_6;
9082 : 5269          DODU(.LUN);
9083 : 5270          LEAVE LOOP;          .JUMP JUST BEYOND END OF BLOCK * 3 *
9084 : 5271          END;      !* 4A3 *
9085 : 5272          TES;
9086 : 5273
9087 : 5274
9088 : 5275          END;      !* 4A * END OF 1ST SECTOR SELECTION LOOP
9089 : 5276
9090 : 5277          PRINTB(FMT2,WRD24);
9091 : 5278          ' UP'
9092 : 5279          COMMAND = CHOOSE();
9093 : 5280          INCR SECTOR FROM LOWEST TO HIGHEST DO
9094 : 5281          BEGIN !* 4B * START OF 2ND SECTOR SELECTION LOOP
9095 : 5282          IF .COMMAND EQL READ
9096 : 5283          THEN
9097 : 5284          BEGIN
9098 : 5285          PTR = .RDPTR;
9099 : 5286          VALUE = READ(.LUN,256,.PTR,.SECTOR);
9100 : 5287          END
9101 : 5288          ELSE
9102 : 5289          BEGIN
9103 : 5290          PTR = .WDPTR;
9104 : 5291          VALUE = CHECK(.LUN,256,.PTR,.SECTOR);
9105 : 5292          END;
9106 : 5293          !+
9107 : 5294          ! SEE HOW SUCCESSFUL THE OPERATION WAS:
9108 : 5295          !-
9109 : 5296          SELECTONE .VALUE OF          !SEE 'SYSERR' FOR DEFINITION
9110 : 5297          SET                          !OF ERROR # CONTAINED IN 'VALUE'
9111 : 5298          [0] :
9112 : 5299          IF .COMMAND EQL READ
9113 : 5300          THEN
9114 : 5301          BEGIN
9115 : 5302          IF (DBL_VALUE = DOUBLE_CHECK(.WDPTR,.RDPTR,256)) NEQ 0
9116 : 5303          THEN
9117 : 5304          BEGIN
9118 : 5305          SAYWHO(.LUN);

```

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (44)



```

9120 ;MLX4
9121 :
9122 :
9123 : 5306 PRINTB(SAY1,MSG5);
9124 : 5307 !'ECC LOGIC FAILED TO DETECT DATA ERROR'
9125 : 5308 PRINTB(FMT12A,..DBL VALUE,..DBL VALUE);
9126 : 5309 !'GOOD DATA: XXXXXX AT LOCATION YYYYYY'
9127 : 5310 DBL VALUE = .DBL VALUE + BUFSIZ * 2;
9128 : 5311 PRINTB(FMT12B,..DBL VALUE,..DBL VALUE);
9129 : 5312 !'BAD DATA: PPPPP AT LOCATION QQQQQQ'
9130 : 5313 WHY DROPT[.LUN] = CODE 8;
9131 : 5314 ERRDF(404,MSG1,0); !**** OPTION 4 ERROR 04 ****
9132 : 5315 DODU(.LUN);
9133 : 5316 LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
9134 : 5317 END;
9135 : 5318 END;
9136 : 5319 [1] :
9137 : 5320 BEGIN !* 4B1 * RETRY ALLOWED
9138 : 5321 IF RETRY(SIX,.COMMAND,.LUN,256,.PTR,.SECTOR) NEQ 0
9139 : 5322 THEN !THE RETRY FAILED -- SYSTEM FATAL ERROR
9140 : 5323 BEGIN
9141 : 5324 WHY DROPT[.LUN] = CODE 4;
9142 : 5325 ERRDF(405,MSG1,0); !**** OPTION 4 ERROR 05 ****
9143 : 5326 DODU(.LUN);
9144 : 5327 LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 3 *
9145 : 5328 END;
9146 : 5329 END; !* 4B1 *
9147 : 5330 [2] :
9148 : 5331 BEGIN !* 4B2 * FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
9149 : 5332 WHY DROPT[.LUN] = CODE 5;
9150 : 5333 ERRDF(406,MSG1,0); !**** OPTION 4 ERROR 06 ****
9151 : 5334 DODU(.LUN);
9152 : 5335 LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 3 *
9153 : 5336 END; !* 4B2 *
9154 : 5337 [3] :
9155 : 5338 BEGIN !* 4B3 * FATAL DRIVE ERROR -- NO RETRY ALLOWED
9156 : 5339 ERRDF(407,MSG1,0); !**** OPTION 4 ERROR 07 ****
9157 : 5340 WHY DROPT[.LUN] = CODE_6;
9158 : 5341 DODU(.LUN);
9159 : 5342 LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 3 *
9160 : 5343 END; !* 4B3 *
9161 : 5344 [4] :
9162 : 5345 BEGIN !* 4B4 * UNRECOVERABLE DATA ERROR
9163 : 5346 ISOLATE();
9164 : 5347 ERRDF(408,MSG2,0); !**** OPTION 4 ERROR 08 ****
9165 : 5348 WHY DROPT[.LUN] = CODE_7;
9166 : 5349 DODU(.LUN);
9167 : 5350 LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 3 *
9168 : 5351 END; !* 4B4 *
9169 : 5352 [5] :
9170 : 5353 BEGIN !* 4B5 * RECOVERABLE DATA ERROR
9171 : 5354 ISOLATE();
9172 : 5355 IF .ERROUT
9173 : 5356 THEN PRINTB(FMT10B,.CHAN);
9174 : 5357 !' BIT 00'
```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (44)

```

9176 :MLX4
9177 :
9178 :
9179 : 5358 OLDSEC = .MLEL;
9180 : 5359 OLDCHN = .CHAN;
9181 : 5360 IF RETRY(ONE, .COMMAND, .LUN, 256, .PTR, .SECTOR) EQL 5
9182 : 5361 THEN
9183 : 5362 IF ((.MLEL EQL .OLDSEC) AND (.CHAN EQL .OLDCHN))
9184 : 5363 THEN
9185 : 5364 BEGIN
9186 : 5365 IF .ERROUT
9187 : 5366 THEN
9188 : 5367 ERRHRD(409,MSG4,0); !**** OPTION 4 ERROR 09 ****
9189 : 5368 UP_HARD_COUNT(.LUN,.BOARD);
9190 : 5369 END
9191 : 5370 ELSE
9192 : 5371 BEGIN
9193 : 5372 IF .ERROUT
9194 : 5373 THEN
9195 : 5374 ERRSOFT(410,MSG3,0); !**** OPTION 4 ERROR 10 ****
9196 : 5375 UP_SOFT_COUNT(.LUN,.BOARD);
9197 : 5376 END
9198 : 5377 ELSE
9199 : 5378 BEGIN
9200 : 5379 IF .ERROUT
9201 : 5380 THEN
9202 : 5381 ERRSOFT(411,MSG3,0); !**** OPTION 4 ERROR 11 ****
9203 : 5382 UP_SOFT_COUNT(.LUN,.BOARD);
9204 : 5383 END;
9205 : 5384 END; !* 4B5 *
9206 : 5385
9207 : 5386 TES;
9208 : 5387
9209 : 5388 VALUE = WRITE(.LUN,256,.WCPTR,.SECTOR);
9210 : 5389 !+
9211 : 5390 ! SEE HOW SUCCESSFUL THE WRITE WAS:
9212 : 5391 !-
9213 : 5392 SELECTONE .VALUE OF !SEE 'SYSERR' FOR DEFINITION
9214 : 5393 SET !OF ERROR # CONTAINED IN 'VALUE'
9215 : 5394 [1] :
9216 : 5395 BEGIN !* 4B6 * RETRY ALLOWED
9217 : 5396 IF RETRY(SIX,WRITE,.LUN,256,.WCPTR,.SECTOR) NEQ 0
9218 : 5397 THEN !THE RETRY FAILED -- SYSTEM FATAL ERROR
9219 : 5398 BEGIN
9220 : 5399 WHY DROPT[.LUN] = CODE_4;
9221 : 5400 ERRDF(412,MSG1,0); !**** OPTION 4 ERROR 12 ****
9222 : 5401 DODU(.LUN);
9223 : 5402 LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 3 *
9224 : 5403 END;
9225 : 5404 END; !* 4B6 *
9226 : 5405 [2] :
9227 : 5406 BEGIN !* 4B7 * FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
9228 : 5407 WHY DROPT[.LUN] = CODE_5;
9229 : 5408 ERRDF(413,MSG1,0); !**** OPTION 4 ERROR 13 ****
9230 : 5409 DODU(.LUN);
  
```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (44)

9232 :MLX4  
9233 :  
9234 :  
9235 :  
9236 :  
9237 :  
9238 :  
9239 :  
9240 :  
9241 :  
9242 :  
9243 :  
9244 :  
9245 :  
9246 :  
9247 :  
9248 :  
9249 :  
9250 :  
9251 :  
9252 :  
9253 :  
9254 :  
9255 :  
9256 :  
9257 :  
9258 :  
9259 :  
9260 :  
9261 :  
9262 :  
9263 :  
9264 :  
9265 :  
9266 :  
9267 :  
9268 :  
9269 :  
9270 :  
9271 :  
9272 :  
9273 :  
9274 :  
9275 :  
9276 :  
9277 :  
9278 :  
9279 :  
9280 :  
9281 :  
9282 :  
9283 :  
9284 :  
9285 :  
9286 :

DEFINITION OF OPTION 4

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (44)

```
LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 3 *
END; !* 4B7 *
[3] :
BEGIN !* 4B8 * FATAL DRIVE ERROR -- NO RETRY ALLOWED
ERRDF(414,MSG1,0); !**** OPTION 4 ERROR 14 ****
WHY DROPT(.LUN) = CODE_6;
DODU(.LUN);
LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 3 *
END; !* 4B8 *

TES;

END; !* 4B * END OF 2ND SECTOR SELECTION LOOP

PRINTB(FMT2,WRD25);
!' DOWN'
COMMAND = CHOOSE();
DECR SECTOR FROM HIGHEST TO LOWEST DO
BEGIN !* 4C * START OF 3RD SECTOR SELECTION LOOP
IF .COMMAND EQL READ
THEN
BEGIN
PTR = .RCPTR;
VALUE = READ(.LUN,256,.PTR,.SECTOR);
END
ELSE
BEGIN
PTR = .WCPTR;
VALUE = CHECK(.LUN,256,.PTR,.SECTOR);
END;

!+
!- SEE HOW SUCCESSFUL THE OPERATION WAS:
!-
SELECTONE .VALUE OF !SEE 'SYSERR' FOR DEFINITION
SET !OF ERROR # CONTAINED IN 'VALUE'
[0] :
IF .COMMAND EQL READ
THEN
BEGIN
IF (DBL_VALUE = DOUBLE_CHECK(.WCPTR,.RCPTR,256)) NEQ 0
THEN
BEGIN
SAYWHO(.LUN);
PRINTB(SAY1,MSG5);
!'ECC LOGIC FAILED TO DETECT DATA ERROR'
PRINTB(FMT12A,..DBL_VALUE,..DBL_VALUE);
!'GOOD DATA: XXXXX AT LOCATION YYYYYY'
DBL_VALUE = .DBL_VALUE + BUFSIZ * 2;
PRINTB(FMT12B,..DBL_VALUE,..DBL_VALUE);
!'BAD DATA: PPPPP AT LOCATION QQQQQQ'
WHY DROPT(.LUN) = CODE_8;
ERRDF(415,MSG1,0); !**** OPTION 4 ERROR 15 ****
```

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (44)

```

9288 ;MLX4
9289 :
9290 :
9291 : 5462          DODU(.LUN);
9292 : 5463          LEAVE LOOP;          !JUMP JUST BEYOND END OF BLOCK * 4 *
9293 : 5464          END;
9294 : 5465          END;
9295 : 5466          [1] :
9296 : 5467          BEGIN !* 4C1 * RETRY ALLOWED
9297 : 5468          IF RETRY(SIX,.COMMAND,.LUN,256,.PTR,.SECTOR) NEQ 0
9298 : 5469          THEN !THE RETRY FAILED -- SYSTEM FATAL ERROR
9299 : 5470          BEGIN
9300 : 5471          WHY DROPT(.LUN) = CODE_4;
9301 : 5472          ERRDF(416,MSG1,0);      !**** OPTION 4 ERROR 16 ****
9302 : 5473          DODU(.LUN);
9303 : 5474          LEAVE LOOP;          !JUMP JUST BEYOND END OF BLOCK * 3 *
9304 : 5475          END;
9305 : 5476          END; !* 4C1 *
9306 : 5477          [2] :
9307 : 5478          BEGIN !* 4C2 * FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
9308 : 5479          WHY DROPT(.LUN) = CODE_5;
9309 : 5480          ERRDF(417,MSG1,0);      !**** OPTION 4 ERROR 17 ****
9310 : 5481          DODU(.LUN);
9311 : 5482          LEAVE LOOP;          !JUMP JUST BEYOND END OF BLOCK * 3 *
9312 : 5483          END; !* 4C2 *
9313 : 5484          [3] :
9314 : 5485          BEGIN !* 4C3 * FATAL DRIVE ERROR -- NO RETRY ALLOWED
9315 : 5486          ERRDF(418,MSG1,0);      !**** OPTION 4 ERROR 18 ****
9316 : 5487          WHY DROPT(.LUN) = CODE_6;
9317 : 5488          DODU(.LUN);
9318 : 5489          LEAVE LOOP;          !JUMP JUST BEYOND END OF BLOCK * 3 *
9319 : 5490          END; !* 4C3 *
9320 : 5491          [4] :
9321 : 5492          BEGIN !* 4C4 * UNRECOVERABLE DATA ERROR
9322 : 5493          ISOLATE();
9323 : 5494          ERRDF(419,MSG2,0);      !**** OPTION 4 ERROR 19 ****
9324 : 5495          WHY DROPT(.LUN) = CODE_7;
9325 : 5496          DODU(.LUN);
9326 : 5497          LEAVE LOOP;          !JUMP JUST BEYOND END OF BLOCK * 3 *
9327 : 5498          END; !* 4C4 *
9328 : 5499          [5] :
9329 : 5500          BEGIN !* 4C5 * RECOVERABLE DATA ERROR
9330 : 5501          ISOLATE();
9331 : 5502          IF .ERROUT
9332 : 5503          THEN PRINTB(FMT10B,.CHAN);
9333 : 5504          !' BIT 00'
9334 : 5505          OLDSEC = .MLEL;
9335 : 5506          OLDCHN = .CHAN;
9336 : 5507          IF RETRY(ONE,.COMMAND,.LUN,256,.PTR,.SECTOR) EQL 5
9337 : 5508          THEN
9338 : 5509          IF ((.MLEL EQL .OLDSEC) AND (.CHAN EQL .OLDCHN))
9339 : 5510          THEN
9340 : 5511          BEGIN
9341 : 5512          IF .ERROUT
9342 : 5513          THEN
  
```

```

9344 :MLX4
9345 :
9346 :
9347 : 5514 ERRHRD(420,MSG4,0); ***** OPTION 4 ERROR 20 *****
9348 : 5515 UP_HARD_COUNT(.LUN,.BOARD);
9349 : 5516 END
9350 : 5517 ELSE
9351 : 5518 BEGIN
9352 : 5519 IF .ERROUT
9353 : 5520 THEN
9354 : 5521 ERRSOFT(421,MSG3,0); !***** OPTION 4 ERROR 21 *****
9355 : 5522 UP_SOFT_COUNT(.LUN,.BOARD);
9356 : 5523 END
9357 : 5524 ELSE
9358 : 5525 BEGIN
9359 : 5526 IF .ERROUT
9360 : 5527 THEN
9361 : 5528 ERRSOFT(422,MSG3,0); !***** OPTION 4 ERROR 22 *****
9362 : 5529 UP_SOFT_COUNT(.LUN,.BOARD);
9363 : 5530 END;
9364 : 5531 END: !* 4C5 *
9365 : 5532
9366 : 5533 TES;
9367 : 5534
9368 : 5535 VALUF = WRITE(.LUN,256,.WDPTR,.SECTOR);
9369 : 5536 !+
9370 : 5537 ! SEE HOW SUCCESSFUL THE WRITE WAS:
9371 : 5538 !-
9372 : 5539 SELECTONE .VALUE OF !SEE 'SYSERR' FOR DEFINITION
9373 : 5540 SET !OF ERROR # CONTAINED IN 'VALUE'
9374 : 5541 [1] :
9375 : 5542 BEGIN !* 4C6 * RETRY ALLOWED
9376 : 5543 IF RETRY(SIX,WRITE,.LUN,256,.WDPTR,.SECTOR) NEQ 0
9377 : 5544 THEN !THE RETRY FAILED -- SYSTEM FATAL ERROR
9378 : 5545 BEGIN
9379 : 5546 WHY DROPT[.LUN] = CODE_4;
9380 : 5547 ERRDF(423,MSG1,0); !***** OPTION 4 ERROR 23 *****
9381 : 5548 DODU(.LUN);
9382 : 5549 LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 3 *
9383 : 5550 END;
9384 : 5551 END: !* 4C6 *
9385 : 5552 [2] :
9386 : 5553 BEGIN !* 4C7 * FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
9387 : 5554 WHY DROPT[.LUN] = CODE_5;
9388 : 5555 ERRDF(424,MSG1,0); !***** OPTION 4 ERROR 24 *****
9389 : 5556 DODU(.LUN);
9390 : 5557 LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 3 *
9391 : 5558 END: !* 4C7 *
9392 : 5559 [3] :
9393 : 5560 BEGIN !* 4C8 * FATAL DRIVE ERROR -- NO RETRY ALLOWED
9394 : 5561 ERRDF(425,MSG1,0); !***** OPTION 4 ERROR 25 *****
9395 : 5562 WHY DROPT[.LUN] = CODE_6;
9396 : 5563 DODU(.LUN);
9397 : 5564 LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 3 *
9398 : 5565 END: !* 4C8 *
    
```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<POSEN>MLX4.BLI.1 (44)

9400 :MLX4  
 9401 :  
 9402 :  
 9403 :  
 9404 :  
 9405 :  
 9406 :  
 9407 :  
 9408 :  
 9409 :  
 9410 :  
 9411 :  
 9412 :  
 9413 :  
 9414 :  
 9415 :  
 9416 :  
 9417 :  
 9418 :  
 9419 :  
 9420 :  
 9421 :  
 9422 :  
 9423 :  
 9424 :  
 9425 :  
 9426 :  
 9427 :  
 9428 :  
 9429 :  
 9430 :  
 9431 :  
 9432 :  
 9433 :  
 9434 :  
 9435 :  
 9436 :  
 9437 :  
 9438 :  
 9439 :  
 9440 :  
 9441 :  
 9442 :  
 9443 :  
 9444 :  
 9445 :  
 9446 :  
 9447 :  
 9448 :  
 9449 :  
 9450 :  
 9451 :  
 9452 :  
 9453 :  
 9454 :

DEFINITION OF OPTION 4

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (44)

```

      TES;
    END;  !* 4C * END OF 3RD SECTOR SELECTION LOOP
PRINTB(FMT2,WRD24);
! ' UP'
COMMAND = CHOOSE();
INCR SECTOR FROM LOWEST TO HIGHEST DO
  BEGIN !* 4D * START OF 4TH SECTOR SELECTION LOOP
    IF .COMMAND EQL READ
    THEN
      BEGIN
        PTR = .RDPTR;
        VALUE = READ(.LUN,256,.PTR,.SECTOR);
      END
    ELSE
      BEGIN
        PTR = .WDPTR;
        VALUE = CHECK(.LUN,256,.PTR,.SECTOR);
      END;
    !+
    !- SEE HOW SUCCESSFUL THE OPERATION WAS:
    !-
    SELECTONE .VALUE OF          !SEE 'SYSERR' FOR DEFINITION
    SET                          !OF ERROR # CONTAINED IN 'VALUE'
  [0] :
    IF .COMMAND EQL READ
    THEN
      BEGIN
        IF (DBL_VALUE = DOUBLE_CHECK(.WDPTR,.RDPTR,256)) NEQ 0
        THEN
          BEGIN
            SAYWHO(.LUN);
            PRINTB(SAY1,MSG5);
            !'ECC LOGIC FAILED TO DETECT DATA ERROR'
            PRINTB(FMT12A,..DBL_VALUE,..DBL_VALUE);
            !'GOOD DATA: XXXXXX AT LOCATION YYYYYY'
            DBL_VALUE = .DBL_VALUE + BUFSIZ * 2;
            PRINTB(FMT12B,..DBL_VALUE,..DBL_VALUE);
            !'BAD DATA: PPPPPP AT LOCATION QQQQQQ'
            WHY DROPT[LUN] = CODE 8;
            ERRDF(426,MSG1,0);  !**** OPTION 4 ERROR 26 ****
            DODU(.LUN);
            LEAVE LOOP;          !JUMP JUST BEYOND END OF BLOCK * 4 *
          END;
        END;
      END;
    [1] :
      BEGIN !* 4D1 * RETRY ALLOWED
        IF RETRY(SIX,.COMMAND,.LUN,256,.PTR,.SECTOR) NEQ 0
        THEN !THE RETRY FAILED -- SYSTEM FATAL ERROR
          BEGIN

```

9456 :MLX4  
 9457 :  
 9458 :  
 9459 :  
 9460 :  
 9461 :  
 9462 :  
 9463 :  
 9464 :  
 9465 :  
 9466 :  
 9467 :  
 9468 :  
 9469 :  
 9470 :  
 9471 :  
 9472 :  
 9473 :  
 9474 :  
 9475 :  
 9476 :  
 9477 :  
 9478 :  
 9479 :  
 9480 :  
 9481 :  
 9482 :  
 9483 :  
 9484 :  
 9485 :  
 9486 :  
 9487 :  
 9488 :  
 9489 :  
 9490 :  
 9491 :  
 9492 :  
 9493 :  
 9494 :  
 9495 :  
 9496 :  
 9497 :  
 9498 :  
 9499 :  
 9500 :  
 9501 :  
 9502 :  
 9503 :  
 9504 :  
 9505 :  
 9506 :  
 9507 :  
 9508 :  
 9509 :  
 9510 :

DEFINITION OF OPTION 4

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (44)

```

    WHY DROPT[.LUN] = CODE_4;
    ERRDF(427,MSG1,0); !**** OPTION 4 ERROR 27 ****
    DODU(.LUN);
    LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 3 *
    END;
  [2] :
    BEGIN !* 4D1 *
    WHY DROPT[.LUN] = CODE_5;
    ERRDF(428,MSG1,0); !**** OPTION 4 ERROR 28 ****
    DODU(.LUN);
    LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 3 *
    END; !* 4D2 *
  [3] :
    BEGIN !* 4D3 * FATAL DRIVE ERROR -- NO RETRY ALLOWED
    ERRDF(429,MSG1,0); !**** OPTION 4 ERROR 29 ****
    WHY DROPT[.LUN] = CODE_6;
    DODU(.LUN);
    LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 3 *
    END; !* 4D3 *
  [4] :
    BEGIN !* 4D4 * UNRECOVERABLE DATA ERROR
    ISOLATE();
    ERRDF(430,MSG2,0); !**** OPTION 4 ERROR 30 ****
    WHY DROPT[.LUN] = CODE_7;
    DODU(.LUN);
    LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 3 *
    END; !* 4D4 *
  [5] :
    BEGIN !* 4D5 * RECOVERABLE DATA ERROR
    ISOLATE();
    IF .ERROUT
    THEN PRINTB(FMT10B,.CHAN);
    !' BIT 00'
    OLDSEC = .MLEL;
    OLDCHN = .CHAN;
    IF RETRY(ONE,.COMMAND,.LUN,256,.PTR,.SECTOR) EQL 5
    THEN
    IF ((.MLEL EQL .OLDSEC) AND (.CHAN EQL .OLDCHN))
    THEN
    BEGIN
    IF .ERROUT
    THEN
    ERRHRD(431,MSG4,0); !**** OPTION 4 ERROR 31 ****
    UP_HARD_COUNT(.LUN,.BOARD);
    END
    ELSE
    BEGIN
    IF .ERROUT
    THEN
    ERRSOFT(432,MSG3,0); !**** OPTION 4 ERROR 32 ****
    UP_SOFT_COUNT(.LUN,.BOARD);
  
```

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (44)

```

9512 :MLX4
9513 :
9514 :
9515 :      5670      END
9516 :      5671      ELSE
9517 :      5672      BEGIN
9518 :      5673      IF .ERROUT
9519 :      5674      THEN
9520 :      5675      ERRSOFT(433,MSG3,0); !**** OPTION 4 ERROR 33 ****
9521 :      5676      UP_SOFT_COUNT(.LUN,.BOARD);
9522 :      5677      END;
9523 :      5678      END; !* 4D *
9524 :      5679
9525 :      5680      TES;
9526 :      5681
9527 :      5682      END; !* 4D * END OF 4TH SECTOR SELECTION LOOP
9528 :      5683      END; !* 4 * END OF TEST FOR AN ACTIVE UNIT
9529 :      5684      END; !* 3 * END OF LOOP THAT COMPLETELY TESTS 1 UNIT
9530 :      5685      END; !* 2 * END OF LOGICAL UNIT SELECTION LOOP
9531 :      5686      RETURN;
9532 :      5687      END; !* 1 * END OF ROUTINE
9533 :
9534 :
9535 :
9536 :
9537 :

```

```

9541 055472 004137 005222      OPT4: .SBTTL OPT4 DEFINITION OF OPTION 4
9542 055476 162706 000030      JSR    R1,$SAVE5
9543 055502 012746 007272      SUB    #30,SP
9544 055506 012746 006766      MOV    #RTN4,-(SP)
9545 055512 012746 006522      MOV    #WRD34,-(SP)
9546 055516 012746 000003      MOV    #SAY2,-(SP)
9547 055522 010600      MOV    #3,-(SP)
9548 055524 104414      MOV    SP,R0
9549 055526 013716 002246      TRAP  14
9550 055532 012746 010706      MOV    MARPAT,(SP)
9551 055536 004737 036654      MOV    #WDBUFF,-(SP)
9552 055542 013716 002246      JSR    PC,GEN4
9553 055546 005416      MOV    MARPAT,(SP)
9554 055550 012746 011706      NEG   (SP)
9555 055554 004737 036654      MOV    #WCBUFF,-(SP)
9556 055560 012766 010706 000016      JSR    PC,GEN4
9557 055566 012766 011706 000030      MOV    #WDBUFF,16(SP)
9558 055574 012766 020706 000026      MOV    #WCBUFF,30(SP)
9559 055602 012766 021706 000034      MOV    #RDBUFF,26(SP)
9560 055610 013766 002012 000042      MOV    #RCBUFF,34(SP)
9561 055616 005066 000014      MOV    LSUNIT,42(SP)
9562 055622 000137 062536      CLR   14(SP)
9563 055626 016600 000014      JMP   83$
9564 055632 006200      1$:  MOV    14(SP),R0
9565 055634 006200      ASR   R0
9566 055636 006200      ASR   R0

```

5152  
5217  
5220  
5221  
5222  
5223  
5224  
5225  
5227  
5231



23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<

Address	Hex	Hex	Hex	Label	Code	Comment	Line No.
9568							
9569							
9570							
9571	055640	062700	032460		ADD	#DRIVE.STATUS,RO	
9572	055644	010046			MOV	RO,-(SP)	
9573	055646	016646	000016		MOV	16(SP),-(SP) ; LUN,*	
9574	055652	042716	177770		BIC	#177770,(SP)	
9575	055656	012746	000001		MOV	#1,-(SP)	
9576	055662	005046			CLR	-(SP)	
9577	055664	004737	004244		JSR	PC,BL\$GT2	
9578	055670	062706	000010		ADD	#10,SP	
9579	055674	005300			DEC	RO	
9580	055676	001160			BNE	6\$	
9581	055700	016637	000014	002074	MOV	14(SP),L\$LUN ; LUN,*	5234
9582	055706	012746	006776		MOV	#WRD35,-(SP) ;	5235
9583	055712	012746	006514		MOV	#SAY1,-(SP)	
9584	055716	012746	000002		MOV	#2,-(SP)	
9585	055722	010600			MOV	SP,RO ; SP,*	
9586	055724	104414			TRAP	14	
9587	055726	012716	006754		MOV	#WRD24,(SP) ;	5237
9588	055732	012746	005602		MOV	#FMT2,-(SP)	
9589	055736	012746	000002		MOV	#2,-(SP)	
9590	055742	010600			MOV	SP,RO ; SP,*	
9591	055744	104414			TRAP	14	
9592	055746	016600	000026		MOV	26(SP),RO ; LUN,*	5239
9593	055752	006300			ASL	RO	
9594	055754	012766	032476	000052	MOV	#LOW.SECT,52(SP)	
9595	055762	060066	000052		ADD	RO,52(SP)	
9596	055766	012766	032516	000050	MOV	#TOP.SECT,50(SP)	
9597	055774	060066	000050		ADD	RO,50(SP)	
9598	056000	017602	000050		MOV	@50(SP),R2	
9599	056004	017604	000052		MOV	@52(SP),R4 ; *,SECTOR	
9600	056010	005304			DEC	R4 ; SECTOR	
9601	056012	000515			BR	8\$	
9602	056014	016646	000026	2\$:	MOV	26(SP),-(SP) ; LUN,*	5241
9603	056020	012746	000400		MOV	#400,-(SP)	
9604	056024	016646	000034		MOV	34(SP),-(SP) ; WDPTR,*	
9605	056030	010446			MOV	R4,-(SP) ; SECTOR,*	
9606	056032	004737	042246		JSR	PC,WRITE	
9607	056036	010005			MOV	RO,R5 ; *,VALUE	
9608	056040	020527	000001		CMP	R5,#1 ; VALUE,*	5245
9609	056044	001036			BNE	3\$	
9610	056046	012746	000006		MOV	#6,-(SP) ;	5249
9611	056052	012746	042246		MOV	#WRITE,-(SP)	
9612	056056	016646	000042		MOV	42(SP),-(SP) ; LUN,*	
9613	056062	012746	000400		MOV	#400,-(SP)	
9614	056066	016646	000050		MOV	50(SP),-(SP) ; WDPTR,*	
9615	056072	010446			MOV	R4,-(SP) ; SECTOR,*	
9616	056074	004737	043046		JSR	PC,RETRY	
9617	056100	062706	000014		ADD	#14,SP	
9618	056104	005700			TST	RO	
9619	056106	001455			BEQ	7\$	
9620	056110	016600	000036		MOV	36(SP),RO ; LUN,*	5252
9621	056114	112760	000004	032464	MOVB	#4,WHY.DROPT(RO)	
9622	056122	104455			TRAP	55 ;	5253

23-Oct-1980 15:01:43 TOPS  
 23-Oct-1980 14:57:05 PA:<

Address	Offset	Value	Label	Instruction	Operand	Comment	Line No.
9624			:MLX4				
9625			:			DEFINITION OF OPTION 4	
9626							
9627	056124	000621		.WORD	621		
9628	056126	010512		.WORD	MSG1		
9629	056130	000000		.WORD	0		
9630	056132	016600	000036	MOV	36(SP),R0	; LUN,*	5254
9631	056136	104451		TRAP	51		
9632	056140	000435		BR	5\$		5255
9633	056142	020527	000002	3\$: CMP	R5,#2	; VALUE,*	5245
9634	056146	001015		BNE	4\$		
9635	056150	016600	000036	MOV	36(SP),R0	; LUN,*	5260
9636	056154	112760	000005	032464	MOVB	#5,WHY.DROPT(R0)	
9637	056162	104455		TRAP	55		5261
9638	056164	000622		.WORD	622		
9639	056166	010512		.WORD	MSG1		
9640	056170	000000		.WORD	0		
9641	056172	016600	000036	MOV	36(SP),R0	; LUN,*	5262
9642	056176	104451		TRAP	51		
9643	056200	000415		BR	5\$		5263
9644	056202	020527	000003	4\$: CMP	R5,#3	; VALUE,*	5245
9645	056206	001015		BNE	7\$		
9646	056210	104455		TRAP	55		5267
9647	056212	000623		.WORD	623		
9648	056214	010512		.WORD	MSG1		
9649	056216	000000		.WORD	0		
9650	056220	016600	000036	MOV	36(SP),R0	; LUN,*	5268
9651	056224	112760	000006	032464	MOVB	#6,WHY.DROPT(R0)	
9652	056232	104451		TRAP	51		5269
9653	056234	062706	000022	5\$: ADD	#22,SP		5270
9654	056240	000566		6\$: BR	14\$		
9655	056242	062706	000010	7\$: ADD	#10,SP		5240
9656	056246	005204		8\$: INC	R4	; SECTOR	5239
9657	056250	020402		CMP	R4,R2	; SECTOR,*	
9658	056252	003660		BLE	2\$		
9659	056254	012716	006754	MOV	#WRD24,(SP)		5277
9660	056260	012746	005602	MOV	#FMT2,-(SP)		
9661	056264	012746	000002	MOV	#2,-(SP)		
9662	056270	010600		MOV	SP,R0	; SP,*	
9663	056272	104414		TRAP	14		
9664	056274	004737	043010	JSR	PC,CHOOSE		5279
9665	056300	010001		MOV	R0,R1	; *,COMMAND	
9666	056302	017666	000054	000036	MCV	@54(SP),36(SP)	5280
9667	056310	017604	000056	MOV	@56(SP),R4	; *,SECTOR	
9668	056314	005304		DEC	R4	; SECTOR	
9669	056316	000137	057666	JMP	32\$		
9670	056322	005002		9\$: CLR	R2		5282
9671	056324	020127	042434	CMP	R1,#READ	; COMMAND,*	
9672	056330	001014		BNE	10\$		
9673	056332	005202		INC	R2		
9674	056334	016603	000044	MOV	44(SP),R3	; RDPTR,PTR	5285
9675	056340	016646	000032	MOV	32(SP),-(SP)	; LUN,*	5286
9676	056344	012746	000400	MOV	#400,-(SP)		
9677	056350	010346		MOV	R3,-(SP)	; PTR,*	
9678	056352	010446		MOV	R4,-(SP)	; SECTOR,*	

23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<

Address	OpCode	Op1	Op2	Op3	Label	Instruction	Comments	Line No.
9680					:MLX4			
9681					:			
9682						DEFINITION OF OPTION 4		
9683	056354	004737	042434			JSR PC,READ		
9684	056360	000412				BR 11\$		
9685	056362	016603	000034	10\$:		MOV 34(SP),R3	: WDPTR, PTR	5290
9686	056366	016646	000032			MOV 32(SP),-(SP)	: LUN,*	5291
9687	056372	012746	000400			MOV #400, -(SP)		
9688	056376	010346				MOV R3, -(SP)	: PTR,*	
9689	056400	010446				MOV R4, -(SP)	: SECTOR,*	
9690	056402	004737	042622			JSR PC,CHECK		
9691	056406	010005		11\$:		MOV R0,R5	: *,VALUE	
9692	056410	001104				BNE 15\$		5296
9693	056412	006002				ROR R2		5299
9694	056414	103402				BLO 13\$		
9695	056416	000137	057432	12\$:		JMP 27\$		
9696	056422	016646	000044	13\$:		MOV 44(SP), -(SP)	: WDPTR,*	5302
9697	056426	016646	000056			MOV 56(SP), -(SP)	: RDPTR,*	
9698	056432	012746	000400			MOV #400, -(SP)		
9699	056436	004737	041226			JSR PC,DOUBLE.CHECK		
9700	056442	062706	000006			ADD #6, SP		
9701	056446	010066	000060			MOV R0,60(SP)	: *,DBL.VALUE	
9702	056452	001761				BEQ 12\$		
9703	056454	016646	000042			MOV 42(SP), -(SP)	: LUN,*	5305
9704	056460	004737	033436			JSR PC,SAYWHO		
9705	056464	012716	010640			MOV #MSG5, (SP)		5306
9706	056470	012746	006514			MOV #SAY1, -(SP)		
9707	0 6474	012746	000002			MOV #2, -(SP)		
9708	056500	010600				MOV SP,R0	: SP,*	
9709	056502	104414				TRAP 14		
9710	056504	016616	000066			MOV 66(SP), (SP)	: DBL.VALUE,*	5308
9711	056510	017646	000066			MOV @66(SP), -(SP)	: DBL.VALUE,*	
9712	056514	012746	006332			MOV #FMT12A, -(SP)		
9713	056520	012746	000003			MOV #3, -(SP)		
9714	056524	010600				MOV SP,R0	: SP,*	
9715	056526	104414				TRAP 14		
9716	056530	062766	010000 000074			ADD #10000,74(SP)	: *,DBL.VALUE	5310
9717	056536	016616	000074			MOV 74(SP), (SP)	: DBL.VALUE,*	5311
9718	056542	017646	000074			MOV @74(SP), -(SP)	: DBL.VALUE,*	
9719	056546	012746	006400			MOV #FMT12B, -(SP)		
9720	056552	012746	000003			MOV #3, -(SP)		
9721	056556	010600				MOV SP,R0	: SP,*	
9722	056560	104414				TRAP 14		
9723	056562	016602	000064			MOV 64(SP),R2	: LUN,*	5313
9724	056566	112762	000010 032464			MOVB #10,WHY.DROPT(R2)		
9725	056574	104455				TRAP 55		5314
9726	056576	000624				.WORD 624		
9727	056600	010512				.WORD MSG1		
9728	056602	000000				.WORD 0		
9729	056604	016600	000064			MOV 64(SP),R0	: LUN,*	5315
9730	056610	104451				TRAP 51		
9731	056612	062706	000050			ADD #50, SP		5316
9732	056616	000137	062532	14\$:		JMP 82\$		
9733	056622	020527	000001	15\$:		CMP R5,#1	: VALUE,*	5296
9734	056624	001034				BNE 16\$		

23-Oct-1980 15:01:43 TOPS  
 23-Oct-1980 14:57:05 PA:<

Address	Offset	Value	Label	Instruction	Comments	Seq
9736						
9737						
9738						
9739	056630	012746	000006	MOV #6,-(SP)		5321
9740	056634	010146		MOV R1,-(SP)	: COMMAND,*	
9741	056636	016646	000046	MOV 46(SP),-(SP)	: LUN,*	
9742	056642	012746	000400	MOV #400,-(SP)		
9743	056646	010346		MOV R3,-(SP)	: PTR,*	
9744	056650	010446		MOV R4,-(SP)	: SECTOR,*	
9745	056652	004737	043046	JSR PC,RETRY		
9746	056656	062706	000014	ADD #14,SP		
9747	056662	005700		TST R0		
9748	056664	001654		BEQ 12\$		
9749	056666	016602	000042	MOV 42(SP),R2	: LUN,*	5324
9750	056672	112762	000004 032464	MOVB #4,WHY.DROPT(R2)		
9751	056700	104455		TRAP 55		5325
9752	056702	000625		.WORD 625		
9753	056704	010512		.WORD MSG1		
9754	056706	000000		.WORD 0		
9755	056710	016600	000042	MOV 42(SP),R0	: LUN,*	5326
9756	056714	104451		TRAP 51		
9757	056716	000457		BR 19\$		5327
9758	056720	020527	000002 16\$:	CMP R5,#2	: VALUE,*	5296
9759	056724	001015		BNE 17\$		
9760	056726	016602	000042	MOV 42(SP),R2	: LUN,*	5332
9761	056732	112762	000005 032464	MOVB #5,WHY.DROPT(R2)		
9762	056740	104455		TRAP 55		5333
9763	056742	000626		.WORD 626		
9764	056744	010512		.WORD MSG1		
9765	056746	000000		.WORD 0		
9766	056750	016600	000042	MOV 42(SP),R0	: LUN,*	5334
9767	056754	104451		TRAP 51		
9768	056756	000437		BR 19\$		5335
9769	056760	020527	000003 17\$:	CMP R5,#3	: VALUE,*	5296
9770	056764	001014		BNE 18\$		
9771	056766	104455		TRAP 55		5339
9772	056770	000627		.WORD 627		
9773	056772	010512		.WORD MSG1		
9774	056774	000000		.WORD 0		
9775	056776	016602	000042	MOV 42(SP),R2	: LUN,*	5340
9776	057002	112762	000006 032464	MOVB #6,WHY.DROPT(R2)		
9777	057010	010200		MOV R2,R0	: LUN,*	5341
9778	057012	104451		TRAP 51		
9779	057014	000420		BR 19\$		5342
9780	057016	020527	000004 18\$:	CMP R5,#4	: VALUE,*	5296
9781	057022	001017		BNE 20\$		
9782	057024	004737	035212	JSR PC,ISOLATE		5346
9783	057030	104455		TRAP 55		5347
9784	057032	000630		.WORD 630		
9785	057034	010542		.WORD MSG2		
9786	057036	000000		.WORD 0		
9787	057040	016602	000042	MOV 42(SP),R2	: LUN,*	5348
9788	057044	112762	000007 032464	MOVB #7,WHY.DROPT(R2)		
9789	057052	010200		MOV R2,R0	: LUN,*	5349
9790	057054	104451		TRAP 51		

9792 ;MLX4 23-Oct-1980 15:01:43 TOPS  
9793 ; 23-Oct-1980 14:57:05 PA:<  
9794 ;  
9795 057056 000137 062526 19\$: JMP 81\$ : VALUE,\* 5350  
9796 057062 020527 000005 20\$: CMP R5,#5 : 5296  
9797 057066 001161 BNE 27\$  
9798 057070 004737 035212 JSR PC,ISCLATE : 5354  
9799 057074 032737 000001 002260 BIT #1,ERROUT : 5355  
9800 057102 001423 BEQ 21\$  
9801 057104 017702 153316 MOV @ML.REG+42,R2 : 5356  
9802 057110 006202 ASR R2  
9803 057112 006202 ASR R2  
9804 057114 006202 ASR R2  
9805 057116 006202 ASR R2  
9806 057120 006202 ASR R2  
9807 057122 006202 ASR R2  
9808 057124 042702 177700 BIC #177700,R2  
9809 057130 010246 MOV R2,-(SP)  
9810 057132 012746 006262 MOV #FMT10B,-(SP)  
9811 057136 012746 000002 MOV #2,-(SP)  
9812 057142 010600 MOV SP,R0 : SP,\*  
9813 057144 044414 TRAP 14  
9814 057146 062706 000006 ADD #6,SP  
9815 057152 017766 153252 000052 21\$: MOV @ML.REG+44,52(SP) : \*,OLDSEC 5358  
9816 057160 017702 153242 MOV @ML.REG+42,R2 : 5359  
9817 057164 006202 ASR R2  
9818 057166 006202 ASR R2  
9819 057170 006202 ASR R2  
9820 057172 006202 ASR R2  
9821 057174 006202 ASR R2  
9822 057176 006202 ASR R2  
9823 057200 042702 177700 BIC #177700,R2  
9824 057204 010266 000050 MOV R2,50(SP) : \*,OLDCHN  
9825 057210 012746 000001 MOV #1,-(SP) : 5360  
9826 057214 010146 MOV R1,-(SP) : COMMAND,\*  
9827 057216 016646 000046 MOV 46(SP),-(SP) : LUN,\*  
9828 057222 012746 000400 MOV #400,-(SP)  
9829 057226 010346 MOV R3,-(SP) : PTR,\*  
9830 057230 010446 MOV R4,-(SP) : SECTOR,\*  
9831 057232 004737 043046 JSR PC,RETRY  
9832 057236 062706 000014 ADD #14,SP  
9833 057242 020027 000005 CMP R0,#5  
9834 057246 001052 BNE 24\$  
9835 057250 027766 153154 000052 CMP @ML.REG+44,52(SP) : \*,OLDSEC 5362  
9836 057256 001035 BNE 23\$  
9837 057260 016600 000050 MOV 50(SP),R0 : OLDCHN,\*  
9838 057264 017702 153136 MOV @ML.REG+42,R2  
9839 057270 006202 ASR R2  
9840 057272 006202 ASR R2  
9841 057274 006202 ASR R2  
9842 057276 006202 ASR R2  
9843 057300 006202 ASR R2  
9844 057302 006202 ASR R2  
9845 057304 042702 177700 BIC #177700,R2  
9846 057310 020200 CMP R2,R0

```

9848 ;MLX4
9849 :
9850 :
9851 057312 001017 BNE 23$
9852 057314 032737 000001 002260 BIT #1,ERROUT ; 5365
9853 057322 001404 BEQ 22$ ;
9854 057324 104456 TRAP 56 ; 5367
9855 057326 000631 .WORD 631
9856 057330 010624 .WORD MSG4
9857 057332 000000 .WORD 0
9858 057334 016646 000042 22$: MOV 42(SP),-(SP) ; LUN,* 5368
9859 057340 013746 032360 MOV BOARD,-(SP)
9860 057344 004737 035302 JSR PC,UP.HARD.COUNT
9861 057350 000427 BR 26$ ; 5362
9862 057352 032737 000001 002260 23$: BIT #1,ERROUT ; 5372
9863 057360 001415 BEQ 25$ ;
9864 057362 104457 TRAP 57 ; 5374
9865 057364 000632 .WORD 632
9866 057366 010610 .WORD MSG3
9867 057370 000000 .WORD 0
9868 057372 000410 BR 25$ ; 5375
9869 057374 032737 000001 002260 24$: BIT #1,ERROUT ; 5379
9870 057402 001404 BEQ 25$ ;
9871 057404 104457 TRAP 57 ; 5381
9872 057406 000633 .WORD 633
9873 057410 010610 .WORD MSG3
9874 057412 000000 .WORD 0
9875 057414 016646 000042 25$: MOV 42(SP),-(SP) ; LUN,* 5382
9876 057420 013746 032360 MOV BOARD,-(SP)
9877 057424 004737 035430 JSR PC,UP.SOFT.COUNT
9878 057430 022626 26$: CMP (SP)+,(SP)+ ; 5353
9879 057432 016616 000042 27$: MOV 42(SP),(SP) ; LUN,* 5388
9880 057436 012746 000400 MOV #400,-(SP)
9881 057442 016646 000060 MOV 60(SP),-(SP) ; WCPTR,*
9882 057446 010446 MOV R4,-(SP) ; SECTOR,*
9883 057450 004737 042246 JSR PC,WRITE
9884 057454 010005 MOV R0,R5 ; *,VALUE
9885 057456 020527 000001 CMP R5,#1 ; VALUE,* 5392
9886 057462 001036 BNE 28$ ;
9887 057464 012746 000006 MOV #6,-(SP) ; 5396
9888 057470 012746 042246 MOV #WRITE,-(SP)
9889 057474 016646 000054 MOV 54(SP),-(SP) ; LUN,*
9890 057500 012746 000400 MOV #400,-(SP)
9891 057504 016646 000074 MOV 74(SP),-(SP) ; WCPTR,*
9892 057510 010446 MOV R4,-(SP) ; SECTOR,*
9893 057512 004737 043046 JSR PC,RETRY
9894 057516 062706 000014 ADD #14,SP
9895 057522 005700 TST R0
9896 057524 001456 BEQ 31$ ;
9897 057526 016602 000050 MOV 50(SP),R2 ; LUN,* 5399
9898 057532 112762 000004 032464 MOVB #4,WHY.DROPT(R2)
9899 057540 104455 TRAP 55 ; 5400
9900 057542 000634 .WORD 634
9901 057544 010512 .WORD MSG1
9902 057546 000000 .WORD 0

```

23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<

23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<

Address	OpCode	Operand1	Operand2	Label	Instruction	Comments	Line No.
9904				:MLX4			
9905				:			
9906					DEFINITION OF OPTION 4		
9907	057550	016600	000050		MOV 50(SP),R0	: LUN,*	5401
9908	057554	104451			TRAP 51		
9909	057556	000436			BR 30\$		5402
9910	057560	020527	000002	28\$:	CMP R5,#2	: VALUE,*	5392
9911	057564	001015			BNE 29\$		
9912	057566	016602	000050		MOV 50(SP),R2	: LUN,*	5407
9913	057572	112762	000005	032464	MOVB #5,WHY.DROPT(R2)		
9914	057600	104455			TRAP 55	:	5408
9915	057602	000635			.WORD 635		
9916	057604	010512			.WORD MSG1		
9917	057606	000000			.WORD 0		
9918	057610	016600	000050		MOV 50(SP),R0	: LUN,*	5409
9919	057614	104451			TRAP 51		
9920	057616	000416			BR 30\$		5410
9921	057620	020527	000003	29\$:	CMP R5,#3	: VALUE,*	5392
9922	057624	001016			BNE 31\$		
9923	057626	104455			TRAP 55	:	5414
9924	057630	000636			.WORD 636		
9925	057632	010512			.WORD MSG1		
9926	057634	000000			.WORD 0		
9927	057636	016602	000050		MOV 50(SP),R2	: LUN,*	5415
9928	057642	112762	000006	032464	MOVB #6,WHY.DROPT(R2)		
9929	057650	010200			MOV R2,R0	: LUN,*	5416
9930	057652	104451			TRAP 51		
9931	057654	062706	000034	30\$:	ADD #34,SP	:	5417
9932	057660	000570			BR 39\$		
9933	057662	062706	000016	31\$:	ADD #16,SP	:	5281
9934	057666	005204		32\$:	INC R4	: SECTOR	5280
9935	057670	020466	000036		CMP R4,36(SP)	: SECTOR,*	
9936	057674	003002			BGT 33\$		
9937	057676	000137	056322		JMP 9\$		
9938	057702	012716	006760	33\$:	MOV #WORD25,(SP)	:	5424
9939	057706	012746	005602		MOV #FMT2,-(SP)		
9940	057712	012746	000002		MOV #2,-(SP)		
9941	057716	010600			MOV SP,R0	: SP,*	
9942	057720	104414			TRAP 14		
9943	057722	004737	043010		JSR PC,CHOOSE	:	5426
9944	057726	010001			MOV R0,R1	: *,COMMAND	
9945	057730	017666	000062	000042	MOV @62(SP),42(SP)	:	5427
9946	057736	017604	000060		MOV @60(SP),R4	: *,SECTOR	
9947	057742	000137	061316		JMP 58\$		
9948	057746	005002		34\$:	CLR R2	:	5429
9949	057750	020127	042434		CMP R1,#READ	: COMMAND,*	
9950	057754	001014			BNE 35\$		
9951	057756	005202			INC R2		
9952	057760	016603	000056		MOV 56(SP),R3	: RCPTR,PTR	5432
9953	057764	016646	000036		MOV 36(SP),-(SP)	: LUN,*	5433
9954	057770	012746	000400		MOV #400,-(SP)		
9955	057774	010346			MOV R3,-(SP)	: PTR,*	
9956	057776	010446			MOV R4,-(SP)	: SECTOR,*	
9957	060000	004737	042434		JSR PC,READ		
9958	060004	000412			BR 36\$		

Address	OpCode	Operand1	Operand2	Operand3	Label	Instruction	Comments	Date/Time	Page
9960								23-Oct-1980 15:01:43	TOPS
9961								23-Oct-1980 14:57:05	PA:<
9962									
9963	MOV	060006	016603	000052	35\$:	52(SP),R3	; WCPTR,*		5437
9964	MOV	060012	016646	000036		36(SP),-(SP)	; LUN,*		5438
9965	MOV	060016	012746	000400		#400, -(SP)			
9966	MOV	060022	010346			R3, -(SP)	; PTR,*		
9967	MOV	060024	010446			R4, -(SP)	; SECTOR,*		
9968	JSR	060026	004737	042622		PC,CHECK			
9969	MOV	060032	010005		36\$:	R0,R5	; *,VALUE		
9970	BNE	060034	001103			40\$			5443
9971	ROR	060036	006002			R2			5446
9972	BLO	060040	103402			38\$			
9973	JMP	060042	000137	061060	37\$:	53\$			
9974	MOV	060046	016646	000062	38\$:	62(SP), -(SP)	; WCPTR,*		5449
9975	MOV	060052	016646	000070		70(SP), -(SP)	; RCPTR,*		
9976	MOV	060056	012746	000400		#400, -(SP)			
9977	JSR	060062	004737	041226		PC,DOUBLF.CHECK			
9978	ADD	060066	062706	000006		#6,SP			
9979	MOV	060072	010066	000064		R0,64(SP)	; *,DBL.VALUE		
9980	BEQ	060076	001761			37\$			
9981	MOV	060100	016646	000046		46(SP), -(SP)	; LUN,*		5452
9982	JSR	060104	004737	033436		PC,SAYWHO			
9983	MOV	060110	012716	010640		#MSG5,(SP)			5453
9984	MOV	060114	012746	006514		#SAY1, -(SP)			
9985	MOV	060120	012746	000002		#2, -(SP)			
9986	MOV	060124	010600			SP,R0	; SP,*		
9987	TRAP	060126	104414			14			
9988	MOV	060130	016616	000072		72(SP), (SP)	; DBL.VALUE,*		5455
9989	MOV	060134	017646	000072		@72(SP), -(SP)	; DBL.VALUE,*		
9990	MOV	060140	012746	006332		#FMT12A, -(SP)			
9991	MOV	060144	012746	000003		#3, -(SP)			
9992	MOV	060150	010600			SP,R0	; SP,*		
9993	TRAP	060152	104414			14			
9994	ADD	060154	062766	010000 000100		#10000,100(SP)	; *,DBL.VALUE		5457
9995	MOV	060162	016616	000100		100(SP), (SP)	; DBL.VALUE,*		5458
9996	MOV	060166	017646	000100		@100(SP), -(SP)	; DBL.VALUE,*		
9997	MOV	060172	012746	006400		#FMT12B, -(SP)			
9998	MOV	060176	012746	000003		#3, -(SP)			
9999	MOV	060202	010600			SP,R0	; SP,*		
10000	TRAP	060204	104414			14			
10001	MOV	060206	016602	000070		70(SP),R2	; LUN,*		5460
10002	MOVB	060212	112762	000010 032464		#10,WHY.DROPT(R2)			
10003	TRAP	060220	104455			55			5461
10004	.WORD	060222	000637			637			
10005	.WORD	060224	010512			MSG1			
10006	.WORD	060226	000000			0			
10007	MOV	060230	016600	000070		70(SP),R0	; LUN,*		5462
10008	TRAP	060234	104451			51			
10009	ADD	060236	062706	000054		#54,SP			5463
10010	BR	060242	000520		39\$:	45\$			
10011	CMPL	060244	020527	000001	40\$:	R5,#1	; VALUE,*		5443
10012	BNE	060250	001034			41\$			
10013	MOV	060252	012746	000006		#6, -(SP)			5468
10014	MOV	060256	010146			R1, -(SP)	; COMMAND,*		



Address	OpCode	Operand 1	Operand 2	Operand 3	Instruction	Comments	Time	Page
10016						:MLX4	23-Oct-1980 15:01:43	TOPS
10017						:	23-Oct-1980 14:57:05	PA:<
10018						DEFINITION OF OPTION 4		
10019	060260	016646	000052		MOV 52(SP),-(SP)	: LUN,*		
10020	060264	012746	000400		MOV #400,-(SP)			
10021	060270	010346			MOV R3,-(SP)	: PTR,*		
10022	060272	010446			MOV R4,-(SP)	: SECTOR,*		
10023	060274	004737	043046		JSR PC,RETRY			
10024	060300	062706	000014		ADD #14,SP			
10025	060304	005700			TST R0			
10026	060306	001655			BEQ 37\$			
10027	060310	016602	000046		MOV 46(SP),R2	: LUN,*		5471
10028	060314	112762	000004	032464	MOVB #4,WHY.DROPT(R2)			
10029	060322	104455			TRAP 55	:		5472
10030	060324	000640			.WORD 640			
10031	060326	010512			.WORD MSG1			
10032	060330	000000			.WORD 0			
10033	060332	016600	000046		MOV 46(SP),R0	: LUN,*		5473
10034	060336	104451			TRAP 51			
10035	060340	000457			BR 44\$	:		5474
10036	060342	020527	000002	41\$:	CMP R5,#2	: VALUE,*		5443
10037	060346	001015			BNE 42\$			
10038	060350	016602	000046		MOV 46(SP),R2	: LUN,*		5479
10039	060354	112762	000005	032464	MOVB #5,WHY.DROPT(R2)			
10040	060362	104455			TRAP 55	:		5480
10041	060364	000641			.WORD 641			
10042	060366	010512			.WORD MSG1			
10043	060370	000000			.WORD 0			
10044	060372	016600	000046		MOV 46(SP),R0	: LUN,*		5481
10045	060376	104451			TRAP 51			
10046	060400	000437			BR 44\$	:		5482
10047	060402	020527	000003	42\$:	CMP R5,#3	: VALUE,*		5443
10048	060406	001014			BNE 43\$			
10049	060410	104455			TRAP 55	:		5486
10050	060412	000642			.WORD 642			
10051	060414	010512			.WORD MSG1			
10052	060416	000000			.WORD 0			
10053	060420	016602	000046		MOV 46(SP),R2	: LUN,*		5487
10054	060424	112762	000006	032464	MOVB #6,WHY.DROPT(R2)			
10055	060432	010200			MOV R2,R0	: LUN,*		5488
10056	060434	104451			TRAP 51			
10057	060436	000420			BR 44\$	:		5489
10058	060440	020527	000004	43\$:	CMP R5,#4	: VALUE,*		5443
10059	060444	001021			BNE 46\$			
10060	060446	004737	035212		JSR PC,ISOLATE	:		5493
10061	060452	104455			TRAP 55	:		5494
10062	060454	000643			.WORD 643			
10063	060456	010542			.WORD MSG2			
10064	060460	000000			.WORD 0			
10065	060462	016602	000046		MOV 46(SP),R2	: LUN,*		5495
10066	060466	112762	000007	032464	MOVB #7,WHY.DROPT(R2)			
10067	060474	010200			MOV R2,R0	: LUN,*		5496
10068	060476	104451			TRAP 51			
10069	060500	062706	000032	44\$:	ADD #32,SP	:		5497
10070	060504	000137	062532	45\$:	JMP 82\$			

10072									23-Oct-1980 15:01:43	TOPS
10073				:MLX4					23-Oct-1980 14:57:05	PA:<
10074				:	DEFINITION OF OPTION 4					
10075	060510	020527	000005	46\$:	CMP	R5,#5		: VALUE,*		5443
10076	060514	001161			BNE	53\$				
10077	060516	004737	035212		JSR	PC,ISOLATE		:		5501
10078	060522	032737	000001	002260	BIT	#1,ERROUT		:		5502
10079	060530	001423			BEQ	47\$				
10080	060532	017702	151670		MOV	@ML.REG+42,R2		:		5503
10081	060536	006202			ASR	R2				
10082	060540	006202			ASR	R2				
10083	060542	006202			ASR	R2				
10084	060544	006202			ASR	R2				
10085	060546	006202			ASR	R2				
10086	060550	006202			ASR	R2				
10087	060552	042702	177700		BIC	#177700,R2				
10088	060556	010246			MOV	R2,-(SP)				
10089	060560	012746	006262		MOV	#FMT10B,-(SP)				
10090	060564	012746	000002		MOV	#2,-(SP)				
10091	060570	010600			MOV	SP,R0		: SP,*		
10092	060572	104414			TRAP	14				
10093	060574	062706	000006		ADD	#6,SP				
10094	060600	017766	151624	000056	47\$:	MOV	@ML.REG+44,56(SP)	: *,OLDSEC		5505
10095	060606	017702	151614		MOV	@ML.REG+42,R2		:		5506
10096	060612	006202			ASR	R2				
10097	060614	006202			ASR	R2				
10098	060616	006202			ASR	R2				
10099	060620	006202			ASR	R2				
10100	060622	006202			ASR	R2				
10101	060624	006202			ASR	R2				
10102	060626	042702	177700		BIC	#177700,R2				
10103	060632	010266	000054		MOV	R2,54(SP)		: *,OLDCHN		
10104	060636	012746	000001		MOV	#1,-(SP)		:		5507
10105	060642	010146			MOV	R1,-(SP)		: COMMAND,*		
10106	060644	016646	000052		MOV	52(SP),-(SP)		: LUN,*		
10107	060650	012746	000400		MOV	#400,-(SP)				
10108	060654	010346			MOV	R3,-(SP)		: PTR,*		
10109	060656	010446			MOV	R4,-(SP)		: SECTOR,*		
10110	060660	004737	043046		JSR	PC,RETRY				
10111	060664	062706	000014		ADD	#14,SP				
10112	060670	020027	000005		CMP	R0,#5				
10113	060674	001052			BNE	50\$				
10114	060676	027766	151526	000056	CMP	@ML.REG+44,56(SP)		: *,OLDSEC		5509
10115	060704	001035			BNE	49\$				
10116	060706	016600	000054		MOV	54(SP),R0		: OLDCHN,*		
10117	060712	017702	151510		MOV	@ML.REG+42,R2				
10118	060716	006202			ASR	R2				
10119	060720	006202			ASR	R2				
10120	060722	006202			ASR	R2				
10121	060724	006202			ASR	R2				
10122	060726	006202			ASR	R2				
10123	060730	006202			ASR	R2				
10124	060732	042702	177700		BIC	#177700,R2				
10125	060736	020200			CMP	R2,R0				
10126	060740	001017			BNE	49\$				

Address	Offset	Label	Operation	Comment	Time	Page
10128		:MLX4			23-Oct-1980 15:01:43	TOPS
10129		:		DEFINITION OF OPTION 4	23-Oct-1980 14:57:05	PA:<
10130						
10131	060742	032737 000001 C02260	BIT #1,ERROUT	:		5512
10132	060750	001404	BEQ 48\$	:		
10133	060752	104456	TRAP 56	:		5514
10134	060754	000644	.WORD 644			
10135	060756	010624	.WORD MSG4			
10136	060760	000000	.WORD 0			
10137	060762	016646 000046 48\$:	MOV 46(SP),-(SP)	: LUN,*		5515
10138	060766	013746 032360	MOV BOARD,-(SP)			
10139	060772	004737 035302	JSR PC,UP.HARD.COUNT			
10140	060776	000427	BR 52\$	:		5509
10141	061000	032737 000001 C02260 49\$:	BIT #1,ERROUT	:		5519
10142	061006	001415	BEQ 51\$	:		
10143	061010	104457	TRAP 57	:		5521
10144	061012	000645	.WORD 645			
10145	061014	010610	.WORD MSG3			
10146	061016	000000	.WORD 0			
10147	061020	000410	BR 51\$	:		5522
10148	061022	032737 000001 002260 50\$:	BIT #1,ERROUT	:		5526
10149	061030	001404	BEQ 51\$	:		
10150	061032	104457	TRAP 57	:		5528
10151	061034	000646	.WORD 646			
10152	061036	010610	.WORD MSG3			
10153	061040	000000	.WORD 0			
10154	061042	016646 000046 51\$:	MOV 46(SP),-(SP)	: LUN,*		5529
10155	061046	013746 032360	MOV BOARD,-(SP)			
10156	061052	004737 035430	JSR PC,UP.SOFT.COUNT			
10157	061056	022626	CMP (SP)+,(SP)+	:		5500
10158	061060	016616 000046 53\$:	MOV 46(SP),(SP)	: LUN,*		5535
10159	061064	012746 000400	MOV #400,-(SP)			
10160	061070	016646 000052	MOV 52(SP),-(SP)	: WDPTR,*		
10161	061074	010446	MOV R4,-(SP)	: SECTOR,*		
10162	061076	004737 042246	JSR PC,WRITE			
10163	061102	010005	MOV R0,R5	: *,VALUE		
10164	061104	020527 000001	CMP R5,#1	: VALUF,*		5539
10165	061110	001036	BNE 54\$			
10166	061112	012746 000006	MOV #6,-(SP)	:		5543
10167	061116	012746 042246	MOV #WRITE,-(SP)			
10168	061122	016646 000060	MOV 60(SP),-(SP)	: LUN,*		
10169	061126	012746 000400	MOV #400,-(SP)			
10170	061132	016646 000066	MOV 66(SP),-(SP)	: WDPTR,*		
10171	061136	010446	MOV R4,-(SP)	: SECTOR,*		
10172	061140	004737 043046	JSR PC,RETRY			
10173	061144	062706 000014	ADD #14,SP			
10174	061150	005700	TST R0			
10175	061152	001456	BEQ 57\$			
10176	061154	016602 000054	MOV 54(SP),R2	: LUN,*		5546
10177	061160	112762 000004 032464	MOVB #4,WHY.DROPT(R2)			
10178	061166	104455	TRAP 55	:		5547
10179	061170	000647	.WORD 647			
10180	061172	010512	.WORD MSG1			
10181	061174	000000	.WORD 0			
10182	061176	016600 000054	MOV 54(SP),R0	: LUN,*		5543

23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<

Address	OpCode	Operand1	Operand2	Label	Instruction	Comments	Line No.
10184				:MLX4			
10185				:			
10186					DEFINITION OF OPTION 4		
10187	061202	104451			TRAP 51		
10188	061204	000436			BR 56\$		5549
10189	061206	020527	000002	54\$:	CMP R5,#2	: VALUE,*	5539
10190	061212	001015			BNE 55\$		
10191	061214	016602	000054		MOV 54(SP),R2	: LUN,*	5554
10192	061220	112762	000005	032464	MOVB #5,WHY.DROPT(R2)		
10193	061226	104455			TRAP 55	:	5555
10194	061230	000650			.WORD 650		
10195	061232	010512			.WORD MSG1		
10196	061234	000000			.WORD 0		
10197	061236	016600	000054		MOV 54(SP),R0	: LUN,*	5556
10198	061242	104451			TRAP 51		
10199	061244	000416			BR 56\$	:	5557
10200	061246	020527	000003	55\$:	CMP R5,#3	: VALUE,*	5539
10201	061252	001016			BNE 57\$		
10202	061254	104455			TRAP 55	:	5561
10203	061256	000651			.WORD 651		
10204	061260	010512			.WORD MSG1		
10205	061262	000000			.WORD 0		
10206	061264	016602	000054		MOV 54(SP),R2	: LUN,*	5562
10207	061270	112762	000006	032464	MOVB #6,WHY.DROPT(R2)		
10208	061276	010200			MOV R2,R0	: LUN,*	5563
10209	061300	104451			TRAP 51		
10210	061302	062706	000040	56\$:	ADD #40,SP	:	5564
10211	061306	000571			BR 65\$		
10212	061310	062706	000016	57\$:	ADD #16,SP	:	5428
10213	061314	005304			DEC R4	: SECTOR	5427
10214	061316	020466	000042	58\$:	CMP R4,42(SP)	: SECTOR,*	
10215	061322	002402			BLT 59\$		
10216	061324	000137	057746		JMP 34\$		
10217	061330	012716	006754	59\$:	MOV #WRD24,(SP)	:	5571
10218	061334	012746	005602		MOV #FMT2,-(SP)		
10219	061340	012746	000002		MOV #2,-(SP)		
10220	061344	010600			MOV SP,R0	: SP,*	
10221	061346	104414			TRAP 14		
10222	061350	004737	043010		JSR PC,CHOOSE	:	5573
10223	061354	010001			MOV R0,R1	: *,COMMAND	
10224	061356	017666	000064	000064	MOV @64(SP),64(SP)	:	5574
10225	061364	017604	000066		MOV @66(SP),R4	: *,SECTOR	
10226	061370	005304			DEC R4	: SECTOR	
10227	061372	000137	062512		JMP 80\$		
10228	061376	005002		60\$:	CLR R2	:	5576
10229	061400	020127	042434		CMP R1,#READ	: COMMAND,*	
10230	061404	001014			BNE 61\$		
10231	061406	005202			INC R2		
10232	061410	016603	000054		MOV 54(SP),R3	: RDPTR,PTR	5579
10233	061414	016646	000042		MOV 42(SP),-(SP)	: LUN,*	5580
10234	061420	012746	000400		MOV #400,-(SP)		
10235	061424	010346			MOV R3,-(SP)	: PTR,*	
10236	061426	010446			MOV R4,-(SP)	: SECTUR,*	
10237	061430	004737	042434		JSR PC,READ		
10238	061434	000412			BR 62\$		

				23-Oct-1980 15:01:43	TOPS
				23-Oct-1980 14:57:05	PA:<
10240					
10241					
10242					
10243	061436	016603	000044	61\$:	MOV 44(SP),R3 ; WDPTR, PTR 5584
10244	061442	016646	000042		MOV 42(SP),-(SP) ; LUN,* 5585
10245	061446	012746	000400		MOV #400, -(SP)
10246	061452	010346			MOV R3, -(SP) ; PTR,*
10247	061454	010446			MOV R4, -(SP) ; SECTOR,*
10248	061456	004737	042622		JSR PC,CHECK
10249	061462	010005		62\$:	MOV R0,R5 ; *,VALUE
10250	061464	001103			BNE 66\$
10251	061466	006002			ROR R2
10252	061470	103402			BLO 64\$
10253	061472	000137	062506	63\$:	JMP 79\$
10254	061476	016646	000054	64\$:	MOV 54(SP), -(SP) ; WDPTR,* 5596
10255	061502	016646	000066		MOV 66(SP), -(SP) ; RDPTR,*
10256	061506	012746	000400		MOV #400, -(SP)
10257	061512	004737	041226		JSR PC,DOUBLE.CHECK
10258	061516	062706	000006		ADD #6,SP
10259	061522	010066	000070		MOV R0,70(SP) ; *,DBL.VALUE
10260	061526	001761			BEQ 63\$
10261	061530	016646	000052		MOV 52(SP), -(SP) ; LUN,* 5599
10262	061534	004737	033436		JSR PC,SAYWHO
10263	061540	012716	010640		MOV #MSG5,(SP) ;
10264	061544	012746	006514		MOV #SAY1, -(SP)
10265	061550	012746	000002		MOV #2, -(SP)
10266	061554	010600			MOV SP,R0 ; SP,*
10267	061556	104414			TRAP 14
10268	061560	016616	000076		MOV 76(SP),(SP) ; DBL.VALUE,* 5602
10269	061564	017646	000076		MOV @76(SP), -(SP) ; DBL.VALUE,*
10270	061570	012746	006332		MOV #FMT12A, -(SP)
10271	061574	012746	000003		MOV #3, -(SP)
10272	061600	010600			MOV SP,R0 ; SP,*
10273	061602	104414			TRAP 14
10274	061604	062766	010000	000104	ADD #10000,104(SP) ; *,DBL.VALUE 5604
10275	061612	016616	000104		MOV 104(SP),(SP) ; DBL.VALUE,* 5605
10276	061616	017646	000104		MOV @104(SP), -(SP) ; DBL.VALUE,*
10277	061622	012746	006400		MOV #FMT12B, -(SP)
10278	061626	012746	000003		MOV #3, -(SP)
10279	061632	010600			MOV SP,R0 ; SP,*
10280	061634	104414			TRAP 14
10281	061636	016602	000074		MOV 74(SP),R2 ; LUN,* 5607
10282	061642	112762	000010	032464	MOVB #10,WHY.DROPT(R2)
10283	061650	104455			TRAP 55 ;
10284	061652	000652			.WORD 652
10285	061654	010512			.WORD MSG1
10286	061656	000000			.WORD 0
10287	061660	016600	000074		MOV 74(SP),R0 ; LUN,* 5609
10288	061664	104451			TRAP 51
10289	061665	062706	000060		ADD #60,SP ;
10290	061672	000520		65\$:	BR 71\$
10291	061674	020527	000001	66\$:	CMP R5,#1 ; VALUE,* 5590
10292	061700	001034			BNE 67\$
10293	061702	012746	000006		MOV #6, -(SP) ;
10294	061706	010146			MOV R1, -(SP) ; COMMAND,* 5615

23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<

Address	Offset	Value	Label	Instruction	Comment	Page
10296			:MLX4			
10297			:		DEFINITION OF OPTION 4	
10298						
10299	061710	016646		MOV	56(SP),-(SP)	: LUN,*
10300	061714	012746		MOV	#400,-(SP)	
10301	061720	010346		MOV	R3,-(SP)	: PTR,*
10302	061722	010446		MOV	R4,-(SP)	: SECTOR,*
10303	061724	004737	043046	JSR	PC,RETRY	
10304	061730	062706	000014	ADD	#14,SP	
10305	061734	005700		TST	R0	
10306	061736	001655		BEQ	63\$	
10307	061740	016602	000052	MOV	52(SP),R2	: LUN,*
10308	061744	112762	000004 032464	MOVB	#4,WHY.DROPT(R2)	
10309	061752	104455		TRAP	55	: 5619
10310	061754	000653		.WORD	653	
10311	061756	010512		.WORD	MSG1	
10312	061760	000000		.WORD	0	
10313	061762	016600	000052	MOV	52(SP),R0	: LUN,*
10314	061766	104451		TRAP	51	
10315	061770	000457		BR	70\$	: 5621
10316	061772	020527	000002 67\$:	CMP	R5,#2	: VALUE,*
10317	061776	001015		BNE	68\$	
10318	062000	016602	000052	MOV	52(SP),R2	: LUN,*
10319	062004	112762	000005 032464	MOVB	#5,WHY.DROPT(R2)	
10320	062012	104455		TRAP	55	: 5627
10321	062014	000654		.WORD	654	
10322	062016	010512		.WORD	MSG1	
10323	062020	000000		.WORD	0	
10324	062022	016600	000052	MOV	52(SP),R0	: LUN,*
10325	062026	104451		TRAP	51	
10326	062030	000437		BR	70\$	: 5629
10327	062032	020527	000003 68\$:	CMP	R5,#3	: VALUE,*
10328	062036	001014		BNE	69\$	
10329	062040	104455		TRAP	55	: 5633
10330	062042	000655		.WORD	655	
10331	062044	010512		.WORD	MSG1	
10332	062046	000000		.WORD	0	
10333	062050	016602	000052	MOV	52(SP),R2	: LUN,*
10334	062054	112762	000006 032464	MOVB	#6,WHY.DROPT(R2)	
10335	062062	010200		MOV	R2,R0	: LUN,*
10336	062064	104451		TRAP	51	
10337	062066	000420		BR	70\$	: 5636
10338	062070	020527	000004 69\$:	CMP	R5,#4	: VALUE,*
10339	062074	001020		BNE	72\$	
10340	062076	004737	035212	JSR	PC,ISOLATE	: 5640
10341	062102	104455		TRAP	55	: 5641
10342	062104	000656		.WORD	656	
10343	062106	010542		.WORD	MSG2	
10344	062110	000000		.WORD	0	
10345	062112	016602	000052	MOV	52(SP),R2	: LUN,*
10346	062116	112762	000007 032464	MOVB	#7,WHY.DROPT(R2)	
10347	062124	010200		MOV	R2,R0	: LUN,*
10348	062126	104451		TRAP	51	
10349	062130	062706	000036 70\$:	ADD	#36,SP	: 5644
10350	062134	000576	71\$:	BR	82\$	

Address	Op1	Op2	Op3	Op4	Label	Op	Op1	Op2	Op3	Op4	Comments	Date/Time	Page
10352						CMP	R5	#5			: VALUE, *	23-Oct-1980 15:01:43	TOPS
10353						BNE	79\$					23-Oct-1980 14:57:05	PA:<
10354						JSR	PC	ISOLATE					
10355	062136	020527	000005		72\$:	JSR	PC	ISOLATE					5590
10356	062142	001161				BIT	#1	ERROUT					5648
10357	062144	004737	035212			BEQ	73\$						5649
10358	062150	032737	000001	002260		MOV	@ML	REG+42,R2					5650
10359	062156	001423				ASR	R2						
10360	062160	017702	150242			ASR	R2						
10361	062164	006202				ASR	R2						
10362	062166	006202				ASR	R2						
10363	062170	006202				ASR	R2						
10364	062172	006202				ASR	R2						
10365	062174	006202				ASR	R2						
10366	062176	006202				ASR	R2						
10367	062200	042702	177700			BIC	#177700	R2					
10368	062204	010246				MOV	R2	-(SP)					
10369	062206	012746	006262			MOV	#FMT10B	-(SP)					
10370	062212	012746	000002			MOV	#2	-(SP)					
10371	062216	010600				MOV	SP	RO			: SP, *		
10372	062220	104414				TRAP	14						
10373	062222	062706	000006			ADD	#6	SP					
10374	062226	017766	150176	000062	73\$:	MOV	@ML	REG+44,62(SP)			: *, OLDSEC		5652
10375	062234	017702	150166			MOV	@ML	REG+42,R2					5653
10376	062240	006202				ASR	R2						
10377	062242	006202				ASR	R2						
10378	062244	006202				ASR	R2						
10379	062246	006202				ASR	R2						
10380	062250	006202				ASR	R2						
10381	062252	006202				ASR	R2						
10382	062254	042702	177700			BIC	#177700	R2					
10383	062260	010266	000060			MOV	R2	60(SP)			: *, OLDCHN		
10384	062264	012746	000001			MOV	#1	-(SP)					5654
10385	062270	010146				MOV	R1	-(SP)			: COMMAND, *		
10386	062272	016646	000056			MOV	56	(SP), -(SP)			: LUN, *		
10387	062276	012746	000400			MOV	#400	-(SP)					
10388	062302	010346				MOV	R3	-(SP)			: PTR, *		
10389	062304	010446				MOV	R4	-(SP)			: SECTOR, *		
10390	062306	004737	043046			JSR	PC	RETRY					
10391	062312	062706	000014			ADD	#14	SP					
10392	062316	020027	000005			CMP	RO	#5					
10393	062322	001052				BNE	76\$						
10394	062324	027766	150100	000062		CMP	@ML	REG+44,62(SP)			: *, OLDSEC		5656
10395	062332	001035				BNE	75\$						
10396	062334	016600	000060			MOV	60	(SP), RO			: OLDCHN, *		
10397	062340	017702	150062			MOV	@ML	REG+42,R2					
10398	062344	006202				ASR	R2						
10399	062346	006202				ASR	R2						
10400	062350	006202				ASR	R2						
10401	062352	006202				ASR	R2						
10402	062354	006202				ASR	R2						
10403	062356	006202				ASR	R2						
10404	062360	042702	177700			BIC	#177700	R2					
10405	062364	020200				CMP	R2	RO					
10406	062366	001017				BNE	75\$						

```

10408      ;MLX4
10409      ;
10410      ;
10411 062370 032737 000001 002260      BIT      #1,ERROUT      ;
10412 062376 001404                    BEQ      74$
10413 062400 104456                    TRAP    56      ;
10414 062402 000657                    .WORD  657
10415 062404 010624                    .WORD  MSG4
10416 062406 000000                    .WORD  0
10417 062410 016646 000052      74$:   MOV      52(SP),-(SP)      ; LUN,*
10418 062414 013746 032360      MOV      BOARD,-(SP)
10419 062420 004737 035302      JSR     PC,UP.HARD.COUNT
10420 062424 000427                    BR      78$      ;
10421 062426 032737 000001 002260 75$:   BIT      #1,ERROUT      ;
10422 062434 001415                    BEQ      77$
10423 062436 104457                    TRAP    57      ;
10424 062440 000660                    .WORD  660
10425 062442 010610                    .WORD  MSG3
10426 062444 000000                    .WORD  0
10427 062446 000410                    BR      77$      ;
10428 062450 032737 000001 002260 76$:   BIT      #1,ERROUT      ;
10429 062456 001404                    BEQ      77$
10430 062460 104457                    TRAP    57      ;
10431 062462 000661                    .WORD  661
10432 062464 010610                    .WORD  MSG3
10433 062466 000000                    .WORD  0
10434 062470 016646 000052      77$:   MOV      52(SP),-(SP)      ; LUN,*
10435 062474 013746 032360      MOV      BOARD,-(SP)
10436 062500 004737 035430      JSR     PC,UP.SOFT.COUNT
10437 062504 022626                    78$:   CMP      (SP)+,(SP)+      ;
10438 062506 062706 000010      79$:   ADD      #10,SP      ;
10439 062512 005204                    80$:   INC      R4      ; SECTOR
10440 062514 020466 000064      CMP      R4,64(SP)      ; SECTOR,*
10441 062520 003002                    BGT     81$
10442 062522 000137 061376      JMP     60$
10443 062526 062706 000026      81$:   ADD      #26,SP      ;
10444 062532 005266 000014      82$:   INC      14(SP)      ; LUN
10445 062536 026666 000014 000042 83$:   CMP      14(SP),42(SP)  ; LUN,*
10446 062544 002002                    BGE     84$
10447 062546 000137 055626      JMP     1$
10448 062552 062706 000044      84$:   ADD      #44,SP      ;
10449 062556 000207                    RTS     PC
10450
10451      ; Routine Size: 1307 words
10452      ; Maximum stack depth per invocation: 48 words
10453
10454
10455
10456
10457
10458

```

23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<

5659  
5661  
5662  
5656  
5666  
5668  
5669  
5673  
5675  
5676  
5647  
5575  
5574  
5233  
5227  
5152



```

10460 ;MLX4
10461 :
10462 :
10463 : 5688 %SBTTL 'SELECTING A RANDOM WORD COUNT'
10464 : 5689
10465 : 5690 ROUTINE RNDWC =
10466 : 5691 BEGIN
10467 : 5692
10468 : 5693 !++
10469 : 5694 ROUTINE: RNDWC
10470 : 5695
10471 : 5696 PURPOSE: TO SELECT A RANDOM WORD COUNT WITHIN THE RANGE
10472 : 5697 1 TO BUFSIZ.
10473 : 5698
10474 : 5699 RESULT: THE VALUE RETURNED WILL BE USED BY THE CALLER
10475 : 5700 AS ITS 'WRDCNT'
10476 : 5701 !--
10477 : 5702
10478 : 5703 LOCAL
10479 : 5704 WRDCNT;
10480 : 5705
10481 : 5706 RN();
10482 : 5707
10483 : 5708 RANDOM = .RANDOM AND %0'077777'; !IGNORE SIGN BIT
10484 : 5709
10485 : 5710 WRDCNT = ((.RANDOM MOD BUFSIZ) + 1); .FORCE THE RANGE
10486 : 5711
10487 : 5712 RETURN .WRDCNT;
10488 : 5713
10489 : 5714 END;

```

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (45)

```

10493 :
10494 :
10498 062560 004737 005256 RNDWC: .SBTTL RNDWC SELECTING A RANDOM WORD COUNT 5706
10499 062564 042737 100000 005352 JSR PC,RN 5708
10500 062572 013746 005352 BIC #100000,RANDOM 5710
10501 062576 012746 004000 MOV RANDOM,-(SP)
10502 062602 004737 005070 MOV #4000,-(SP)
10503 062606 005200 JSR PC,BLSMOD
10504 062610 022626 INC R0 5690
10505 062612 000207 CMP (SP)+,(SP)+
10506 :
10507 : RTS PC
10508 :
10513 :

```

; Routine Size: 14 words  
; Maximum stack depth per invocation. 2 words

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA: <ROSEN>MLX4.BLI.1 (46)

```

10515 :MLX4
10516 :
10517 :
10518 : 5715 %SBTTL 'SELECTING A RANDOM SECTOR'
10519 : 5716
10520 : 5717 ROUTINE RNDSEC(LUN) =
10521 : 5718 BEGIN
10522 : 5719
10523 : 5720 !++
10524 : 5721 ROUTINE: RNDSEC(LUN)
10525 : 5722
10526 : 5723 PURPOSE: TO SELECT A RANDOM SECTOR NUMBER WITHIN THE RANGE
10527 : 5724 OF TESTABLE SECTORS (LOWEST TO HIGHEST)
10528 : 5725
10529 : 5726 ARGUMENT: LUN = THE CURRENT LOGICAL UNIT
10530 : 5727 NOTE: 'LUN' IS REQUIRED TO CALCULATE LOWEST/HIGHEST
10531 : 5728 FOR THE PARTICULAR LOGICAL UNIT.
10532 : 5729
10533 : 5730 RESULT: THE VALUE RETURNED WILL BE USED BY THE CALLER
10534 : 5731 AS ITS 'SECTOR'.
10535 : 5732 !--
10536 : 5733
10537 : 5734 LOCAL
10538 : 5735 SECTOR, SIZE;
10539 : 5736
10540 : 5737 IF LOWEST EQL HIGHEST
10541 : 5738 THEN
10542 : 5739 SECTOR = LOWEST
10543 : 5740 ELSE
10544 : 5741 BEGIN
10545 : 5742 RN();
10546 : 5743 RANDOM = .RANDOM AND %0'077777'; !IGNORE THE SIGN BIT
10547 : 5744
10548 : 5745 SIZE = HIGHEST - LOWEST + 1;
10549 : 5746 SECTOR = (LOWEST + (.RANDOM MOD .SIZE)); !FORCE RANGE
10550 : 5747 END;
10551 : 5748
10552 : 5749 RETURN .SECTOR;
10553 : 5750
10554 : 5751 END;

```

```

10559 : .SBTTL RNDSEC SELECTING A RANDOM SECTOR
10563 062614 004137 005164 RNDSEC: JSR R1,$SAVE3 :
10564 062620 016601 000012 MOV 12(SP),R1 : LUN,*
10565 062624 006301 ASL R1
10566 062626 012703 032476 MOV #LOW.SECT,R3
10567 062632 060103 ADD R1,R3
10568 062634 021361 032516 CMP (R3),TOP.SECT(R1)
10569 062640 001002 BNE 1$

```

5717  
 5737

```

10571          ;MLX4
10572          ;
10573          ;
10574 062642 011302          MOV      (R3),R2          ; *,SECTOR
10575 062644 000421          BR       2$              ;
10576 062646 004737 005256 1$: JSR      PC,RN          ;
10577 062652 042737 100000 005352 BIC     #100000,RANDOM ;
10578 062660 016101 032516          MOV     TOP,SECT(R1),R1 ;
10579 062664 161301          SUB     (R3),R1          ;
10580 062666 005201          INC     R1              ;
10581 062670 013746 005352          MOV     RANDOM,-(SP)    ;
10582 062674 010146          MOV     R1,-(SP)      ; SIZE,*
10583 062676 004737 005070          JSR     PC,BL$MOD     ;
10584 062702 061300          ADD     (R3),R0       ;
10585 062704 010002          MOV     R0,R2        ; *,SECTOR
10586 062706 022626          CMP     (SP)+,(SP)+  ;
10587 062710 010200          2$: MOV   R2,R0        ; SECTOR,*
10588 062712 000207          RTS     PC           ;
10589
10590          ; Routine Size: 32 words
10591          ; Maximum stack depth per invocation: 6 words
10596
10597
  
```

23-Oct-1980 15:01:43 TOPS  
 23-Oct-1980 14:57:05 PA:<

5739  
 5737  
 5742  
 5743  
 5745  
  
 5746  
  
  
 5741  
 5718  
 5717

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (47)

```
10599 :MLX4
10600 :
10601 :
10602 : 5752 %SBTTL 'SELECTING A RANDOM UNIT'
10603 : 5753
10604 : 5754 ROUTINE RNDU =
10605 : 5755 BEGIN !* 1 *
10606 : 5756
10607 : 5757 !++
10608 : 5758 ROUTINE: RNDU
10609 : 5759
10610 : 5760 PURPOSE: TO SELECT A RANDOM LOGICAL UNIT NUMBER WITHIN
10611 : 5761 THE RANGE OF TESTABLE UNITS (0 TO .LSUNIT-1)
10612 : 5762 !--
10613 : 5763
10614 : 5764 LOCAL
10615 : 5765 LUN;
10616 : 5766
10617 : 5767 IF .LSUNIT EQL 1
10618 : 5768 THEN
10619 : 5769 LUN = 0
10620 : 5770 ELSE
10621 : 5771 BEGIN !* 2 *
10622 : 5772 RN();
10623 : 5773 RANDOM = .RANDOM AND %0'077777'; !IGNORE THE SIGN BIT
10624 : 5774 RANDOM = (.RANDOM MOD .LSUNIT); !FORCE THE RANGE
10625 : 5775
10626 : 5776 !+
10627 : 5777 ! MAKE SURE THE DRIVE IS ACTIVE. IF IT ISN'T,
10628 : 5778 ! THEN FIND THE NEXT AVAILABLE ACTIVE DRIVE:
10629 : 5779 !-
10630 : 5780
10631 : 5781 INCR COUNT FROM 0 TO (.LSUNIT - 1) DO
10632 : 5782 IF .DRIVE_STATUS[.RANDOM] EQL ACTIVE
10633 : 5783 THEN
10634 : 5784 BEGIN
10635 : 5785 LUN = .RANDOM;
10636 : 5786 EXITLOOP;
10637 : 5787 END
10638 : 5788 ELSE
10639 : 5789 RANDOM = ((.RANDOM + 1) MOD .LSUNIT);
10640 : 5790
10641 : 5791 END; !* 2 *
10642 : 5792
10643 : 5793 RETURN .LUN;
10644 : 5794
10645 : 5795 END; !* 1 *
10649 :
10650 : .SBTTL RNDU SELECTING A RANDOM UNIT
```

10655					:MLX4					23-Oct-1980 15:01:43	TOPS
10656					:		SELECTING A RANDOM UNIT			23-Oct-1980 14:57:05	PA:<
10657											
10658	062714	004137	005202		RNDU:	JSR	R1,\$SAVE4	:			5754
10659	062720	023727	002012	000001		CMP	LSUNIT,#1	:			5767
10660	062726	001002				BNE	1\$	:			
10661	062730	005004				CLR	R4	:	LUN		5769
10662	062732	000467				BR	6\$	:			5767
10663	062734	004737	005256		1\$:	JSR	PC,RN	:			5772
10664	062740	042737	100000	005352		BIC	#100000,RANDOM	:			5773
10665	062746	013746	005352			MOV	RANDOM,-(SP)	:			5774
10666	062752	013746	002012			MOV	LSUNIT,-(SP)	:			
10667	062756	004737	005070			JSR	PC,BL\$MOD	:			
10668	062762	010037	005352			MOV	R0,RANDOM	:			
10669	062766	013703	002012			MOV	LSUNIT,R3	:			5781
10670	062772	005001				CLR	R1	:	COUNT		
10671	062774	000443				BR	4\$	:			
10672	062776	013702	005352		2\$:	MOV	RANDOM,R2	:			5782
10673	063002	006202				ASR	R2	:			
10674	063004	006202				ASR	R2	:			
10675	063006	006202				ASR	R2	:			
10676	063010	062702	032460			ADD	#DRIVE.STATUS,R2	:			
10677	063014	010246				MOV	R2,-(SP)	:			
10678	063016	013746	005352			MOV	RANDOM,-(SP)	:			
10679	063022	042716	177770			BIC	#177770,(SP)	:			
10680	063026	012746	000001			MOV	#1,-(SP)	:			
10681	063032	005046				CLR	-(SP)	:			
10682	063034	004737	004244			JSR	PC,BL\$GT2	:			
10683	063040	062706	000010			ADD	#10,SP	:			
10684	063044	005300				DEC	R0	:			
10685	063046	001003				BNE	3\$	:			
10686	063050	013704	005352			MOV	RANDOM,R4	:	*.LUN		5785
10687	063054	000415				BR	5\$	:			5786
10688	063056	013746	005352		3\$:	MOV	RANDOM,-(SP)	:			5789
10689	063062	005216				INC	(SP)	:			
10690	063064	013746	002012			MOV	LSUNIT,-(SP)	:			
10691	063070	004737	005070			JSR	PC,BL\$MOD	:			
10692	063074	010037	005352			MOV	R0,RANDOM	:			
10693	063100	022626				CMP	(SP)+,(SP)+	:			
10694	063102	005201				INC	R1	:	COUNT		5781
10695	063104	020103			4\$:	CMP	R1,R3	:	COUNT,*		
10696	063106	002733				BLT	2\$	:			
10697	063110	022626			5\$:	CMP	(SP)+,(SP)+	:			5771
10698	063112	010400			6\$:	MOV	R4,R0	:	LUN,*		5755
10699	063114	000207				RTS	PC	:			5754

10700  
10701  
10702  
10707  
10708

: Routine Size: 65 words  
: Maximum stack depth per invocation: 11 words

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (48)

```

10710 :MLX4
10711 :
10712 : TESTING RANDOM DATA
10713 : 5796 %SBTTL 'TESTING RANDOM DATA'
10714 : 5797
10715 : 5798 ROUTINE RAND1(REPEAT): NOVALUE =
10716 : 5799 BEGIN !* 1 * START OF ROUTINE
10717 : 5800
10718 : 5801 !++
10719 : 5802 ROUTINE: RAND1(REPEAT)
10720 : 5803
10721 : 5804 PURPOSE: TO TEST USING RANDOM DATA
10722 : 5805
10723 : 5806 ARGUMENT: REPEAT = NUMBER OF TIMES TO EXECUTE THIS ROUTINE
10724 : 5807 BEFORE RETURNING TO THE CALLER (OPT5).
10725 : 5808
10726 : 5809 NOTE: THIS TEST CODE FOLLOWS THE SAME FLOW AS OPT3,
10727 : 5810 BUT USES A RANDOM DATA PATTERN.
10728 : 5811
10729 : 5812 THE CODE FOR 'RAND1' IN BRIEF:
10730 : 5813
10731 : 5814 BEGIN 1 (START OF ROUTINE)
10732 : 5815 SAY ROUTINE IS RUNNING
10733 : 5816 INCR COUNT FROM 1 TO REPEAT
10734 : 5817 : BEGIN 2 (START OF REPEAT LOOP FOR THE ROUTINE)
10735 : 5818 : GENERATE THE RANDOM PATTERN
10736 : 5819 : INCR LUN FROM 0 TO LAST
10737 : 5820 : : BEGIN 3 (START OF LOGICAL UNIT SELECTION LOOP)
10738 : 5821 : : TESTLOOP:
10739 : 5822 : : : BEGIN 4 (START OF LOOP THAT COMPLETELY TESTS 1 UNIT)
10740 : 5823 : : : IF UNIT IS ACTIVE
10741 : 5824 : : : THEN
10742 : 5825 : : : : BEGIN 5 (START OF TEST FOR AN ACTIVE UNIT)
10743 : 5826 : : : : INITIALIZE THE WRITE AND READ BUFFER POINTERS
10744 : 5827 : : : : SECTOR = LOWEST
10745 : 5828 : : : : WHILE SECTOR LEQ HIGHEST DO
10746 : 5829 : : : : : BEGIN 6 (START OF SECTOR SELECTION LOOP)
10747 : 5830 : : : : : GET_WRCNT
10748 : 5831 : : : : : SET_UP BUFFER POINTERS BEFORE TRANSFER
10749 : 5832 : : : : : WRITE
10750 : 5833 : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
10751 : 5834 : : : : : CHOOSE WHETHER TO WRITE CHECK OR READ
10752 : 5835 : : : : : DO THE WRITE CHECK OR READ
10753 : 5836 : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
10754 : 5837 : : : : : CHANGE BUFFER POINTERS AFTER TRANSFER
10755 : 5838 : : : : : UPDATE SECTOR NUMBER BY NUMBER OF SECTORS IN PREVIOUS TRANSFER
10756 : 5839 : : : : : END 6 (END OF SECTOR SELECTION LOOP)
10757 : 5840 : : : : : END 5 (END OF TEST FOR AN ACTIVE UNIT)
10758 : 5841 : : : : : END 4 (END OF TESTLOOP)
10759 : 5842 : : : : : END 3 (END OF LOGICAL UNIT SELECTION LOOP)
10760 : 5843 : : : : : END 2 (END OF REPEAT LOOP FOR THIS ROUTINE)
10761 : 5844 RETURN
10762 : 5845 END 1 (END OF ROUTINE)
10763 : 5846
10764 : 5847
  
```

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (48)

```

10766 :MLX4
10767 :
10768 :
10769 : 5848
10770 : 5849 LABEL
10771 : 5850 LOOP;
10772 : 5851
10773 : 5852 LOCAL
10774 : 5853 WRDCNT, VALUE, OLDSEC, OLDCHN, SECTOR, PTR, COMMAND, DBL_VALUE;
10775 : 5854
10776 : 5855
10777 : 5856 PRINTB(SAY1,RTN5A);
10778 : 5857 !'RAND1'
10779 : 5858
10780 : 5859 INCR COUNT FROM 1 TO .REPEAT DO
10781 : 5860 BEGIN !* 2 * START OF REPEAT LOOP FOR THIS ROUTINE
10782 : 5861 GEN5(); !FIRST 3 WORDS OF WBUFF ARE THE SEEDS
10783 : 5862 INCR LUN FROM 0 TO (.LSUNIT - 1) DO
10784 : 5863 BEGIN !* 3 * START OF LOGICAL UNIT SELECTION LOOP
10785 : 5864 LOOP:
10786 : 5865 BEGIN !* 4 * START OF THE LOOP THAT COMPLETELY TESTS 1 UNIT
10787 : 5866 IF .DRIVE_STATUS[.LUN] EQL ACTIVE
10788 : 5867 THEN
10789 : 5868 BEGIN !* 5 * START OF TEST FOR AN ACTIVE UNIT
10790 : 5869 LSLUN = .LUN;
10791 : 5870 WPTR = WBUFF;
10792 : 5871 RPTR = RBUFF;
10793 : 5872 SECTOR = LOWEST;
10794 : 5873 WHILE .SECTOR LEQ HIGHEST DO
10795 : 5874 BEGIN !* 6 * START OF SECTOR SELECTION LOOP
10796 : 5875 WRDCNT = GET WRDCNT(.SECTOR,HIGHEST);
10797 : 5876 SET PTRS(.WRDCNT);
10798 : 5877 VALDE = WRITE(.LUN,.WRDCNT,.WPTR,.SECTOR);
10799 : 5878 !*
10800 : 5879 ! SEE HOW SUCCESSFUL THE WRITE WAS:
10801 : 5880 !-
10802 : 5881 SELECTONE .VALUE OF !SEE 'SYSERR' FOR DEFINITION
10803 : 5882 SET !OF ERROR # CONTAINED IN 'VALUE'
10804 : 5883 [1] :
10805 : 5884 BEGIN !* 6A * RETRY ALLOWED
10806 : 5885 IF RETRY(SIX,WRITE,.LUN,.WRDCNT,.WPTR,.SECTOR) NEQ 0
10807 : 5886 THEN !THE RETRY FAILED -- SYSTEM FATAL ERROR
10808 : 5887 BEGIN
10809 : 5888 WHY DROPT[.LUN] = CODE 4;
10810 : 5889 ERRDF(5101,MSG1 0); !**** OPTION 5, RAND1 ERROR 01 ****
10811 : 5890 DODU(.LUN);
10812 : 5891 LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
10813 : 5892 END;
10814 : 5893 END; !* 6A *
10815 : 5894 [2] :
10816 : 5895 BEGIN !* 6B * FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
10817 : 5896 WHY DROPT[.LUN] = CODE 5;
10818 : 5897 ERRDF(5102,MSG1,0); !**** OPTION 5, RAND1 ERROR 02 ****
10819 : 5898 DODU(.LUN);
10820 : 5899 LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *

```

10822 :MLX4  
 10823 :  
 10824 :  
 10825 : 5900  
 10826 : 5901  
 10827 : 5902  
 10828 : 5903  
 10829 : 5904  
 10830 : 5905  
 10831 : 5906  
 10832 : 5907  
 10833 : 5908  
 10834 : 5909  
 10835 : 5910  
 10836 : 5911  
 10837 : 5912  
 10838 : 5913  
 10839 : 5914  
 10840 : 5915  
 10841 : 5916  
 10842 : 5917  
 10843 : 5918  
 10844 : 5919  
 10845 : 5920  
 10846 : 5921  
 10847 : 5922  
 10848 : 5923  
 10849 : 5924  
 10850 : 5925  
 10851 : 5926  
 10852 : 5927  
 10853 : 5928  
 10854 : 5929  
 10855 : 5930  
 10856 : 5931  
 10857 : 5932  
 10858 : 5933  
 10859 : 5934  
 10860 : 5935  
 10861 : 5936  
 10862 : 5937  
 10863 : 5938  
 10864 : 5939  
 10865 : 5940  
 10866 : 5941  
 10867 : 5942  
 10868 : 5943  
 10869 : 5944  
 10870 : 5945  
 10871 : 5946  
 10872 : 5947  
 10873 : 5948  
 10874 : 5949  
 10875 : 5950  
 10876 : 5951

TESTING RANDOM DATA

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (48)

```

END;      !* 6B *
[3] :
BEGIN !* 6C *      FATAL DRIVE ERROR -- NO RETRY ALLOWED
ERRDF(5103,MSG1,0); !*** OPTION 5, RAND1 ERROR 03 ***
WHY DROPT[.LUN] = CODE_6;
DODU(.LUN);
LEAVE LOOP;      !JUMP JUST BEYOND END OF BLOCK * 4 *
END;      !* 6C *

TES;

COMMAND = CHOOSE();
IF .COMMAND EQL READ
THEN
  BEGIN
  PTR = .RPTR;
  VALUE = READ(.LUN,.WRDCNT,.RPTR,.SECTOR);
  END
ELSE
  BEGIN
  PTR = .WPTR;
  VALUE = CHECK(.LUN,.WRDCNT,.WPTR,.SECTOR);
  END;
+
SEE HOW SUCCESSFUL THE OPERATION WAS:
SELECTONE .VALUE OF      !SEE 'SYSERR' FOR DEFINITION
SET      !OF ERROR # CONTAINED IN 'VALUE'
[0] :
IF .COMMAND EQL READ
THEN
  BEGIN
  IF (DBL_VALUE = DOUBLE_CHECK(.WPTR,.RPTR,.WRDCNT)) NEQ 0
  THEN
    BEGIN
    SAYWHO(.LUN);
    PRINTB(SAY1,MSG5);
    !'ECC LOGIC FAILED TO DETECT DATA ERROR'
    PRINTB(FMT12A,..DBL_VALUE,..DBL_VALUE);
    !'GOOD DATA: XXXXXX AT LOCATION YYYYYY'
    DBL_VALUE = .DBL_VALUE + BUFSIZ * 2;
    PRINTB(FMT12B,..DBL_VALUE,..DBL_VALUE);
    !'BAD DATA: PPPPPP AT LOCATION QQQQQQ'
    WHY DROPT[.LUN] = CODE_8;
    ERRDF(5104,MSG1,0); !*** OPTION 5, RAND1 ERROR 04 ***
    DODU(.LUN);
    LEAVE LOOP;      !JUMP JUST BEYOND END OF BLOCK * 4 *
    END;
  END;
[1] :
BEGIN !* 6D *      RETRY ALLOWED
IF RETRY(SIX,.COMMAND,.LUN,.WRDCNT,.PTR,.SECTOR) NEQ 0

```



10878 :MLX4  
 10879 :  
 10880 :  
 10881 :  
 10882 :  
 10883 :  
 10884 :  
 10885 :  
 10886 :  
 10887 :  
 10888 :  
 10889 :  
 10890 :  
 10891 :  
 10892 :  
 10893 :  
 10894 :  
 10895 :  
 10896 :  
 10897 :  
 10898 :  
 10899 :  
 10900 :  
 10901 :  
 10902 :  
 10903 :  
 10904 :  
 10905 :  
 10906 :  
 10907 :  
 10908 :  
 10909 :  
 10910 :  
 10911 :  
 10912 :  
 10913 :  
 10914 :  
 10915 :  
 10916 :  
 10917 :  
 10918 :  
 10919 :  
 10920 :  
 10921 :  
 10922 :  
 10923 :  
 10924 :  
 10925 :  
 10926 :  
 10927 :  
 10928 :  
 10929 :  
 10930 :  
 10931 :  
 10932 :

TESTING RANDOM DATA

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (48)

```

THEN !THE RETRY FAILED -- SYSTEM FATAL ERROR
BEGIN
  WHY DROPT[.LUN] = CODE_4;
  ERRDF(5105,MSG1,0); !**** OPTION 5, RAND1 ERROR 05 ****
  DODU(.LUN);
  LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
END;
END; !* 6D *
[2] :
BEGIN !* 6E * FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
  WHY DROPT[.LUN] = CODE_5;
  ERRDF(5106,MSG1,0); !**** OPTION 5, RAND1 ERROR 06 ****
  DODU(.LUN);
  LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
END; !* 6E *
[3] :
BEGIN !* 6F * FATAL DRIVE ERROR -- NO RETRY ALLOWED
  ERRDF(5107,MSG1,0); !**** OPTION 5, RAND1 ERROR 07 ****
  WHY DROPT[.LUN] = CODE_6;
  DODU(.LUN);
  LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
END; !* 6F *
[4] :
BEGIN !* 6G * UNRECOVERABLE DATA ERROR
  ISOLATE();
  ERRDF(5108,MSG2,0); !**** OPTION 5, RAND1 ERROR 08 ****
  WHY DROPT[.LUN] = CODE_7;
  DODU(.LUN);
  LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
END; !* 6G *
[5] :
BEGIN !* 6H * RECOVERABLE DATA ERROR
  ISOLATE();
  IF .ERROUT
  THEN PRINTB(FMT10B,.CHAN);
  !' BIT QQ'
  OLDSEC = .MLEL;
  OLDCHN = .CHAN;
  IF RETRY(ONE,.COMMAND,.LUN,.WRDCNT,.PTR,.SECTOR) EQL 5
  THEN
  IF ((.MLEL EQL .OLDSEC) AND (.CHAN EQL .OLDCHN))
  THEN
  BEGIN
  IF .ERROUT
  THEN
  ERRHRD(5109,MSG4,0); !**** OPTION 5, RAND1 ERROR 09 ****
  UP_HARD_COUNT(.LUN,.BOARD);
  END
  ELSE
  BEGIN
  IF .ERROUT
  THEN
  
```

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (48)

```

10934 :MLX4
10935 :
10936 :
10937 : 6004 ERRSOF1(5110,MSG3,0); !**** OPTION 5, RAND1 ERROR 10 ****
10938 : 6005 UP SOF1_COUNT(.LUN,.BOARD);
10939 : 6006 END
10940 : 6007 ELSE
10941 : 6008 BEGIN
10942 : 6009 IF .ERROUT
10943 : 6010 THEN
10944 : 6011 ERRSOF1(5111,MSG3,0); !**** OPTION 5, RAND1 ERROR 11 ****
10945 : 6012 UP SOF1_COUNT(.LUN,.BOARD);
10946 : 6013 END;
10947 : 6014 END; !* 6H *
10948 : 6015
10949 : 6016 TES;
10950 : 6017
10951 : 6018 WPTR = .WPTR + (.WRDCNT * 2);
10952 : 6019 SECTOR = .SECTOR + (.WRDCNT/256);
10953 : 6020
10954 : 6021 END; !* 6 * END OF SECTOR SELECTION LOOP
10955 : 6022 END; !* 5 * END OF TEST FOR AN ACTIVE UNIT
10956 : 6023 END; !* 4 * END OF LOOP THAT COMPLETELY TESTS 1 UNIT
10957 : 6024 END; !* 3 * END OF LOGICAL UNIT SELECTION LOOP
10958 : 6025 END; !* 2 * END OF REPEAT LOOP FOR THIS ROUTINE
10959 : 6026 RETURN;
10960 : 6027 END; !* 1 * END OF ROUTINE
  
```

Address	Hex	Hex	Hex	Label	Instruction	Comment	Address
10965				.SBTTL	RAND1 TESTING RANDOM DATA		
10969	063116	004137	005222	RAND1:	JSR R1,\$SAVE5		5798
10970	063122	162706	000020		SUB #20,SP		
10971	063126	012746	007306		MOV #RTN5A,-(SP)		5856
10972	063132	012746	006514		MOV #SAY1,-(SP)		
10973	063136	012746	000002		MOV #2,-(SP)		
10974	063142	010600			MOV SP,RO	: SP,*	
10975	063144	104414			TRAP 14		
10976	063146	005066	000024		CLR 24(SP)	: COUNT	5859
10977	063152	000137	064774		JMP 30\$		
10978	063156	004737	036710	1\$:	JSR PC,GEN5		5861
10979	063162	013766	002012	000010	MOV L\$UNIT,10(SP)		5862
10980	063170	005066	000006		CLR 6(SP)	: LUN	
10981	063174	000137	064760		JMP 29\$		
10982	063200	016500	000006	2\$:	MOV 6(SP),RO	: LUN,*	5866
10983	063204	006200			ASR RO		
10984	063206	006200			ASR RO		
10985	063210	006200			ASR RO		
10986	063212	062700	032460		ADD #DRIVE.STATUS,RO		
10987	063216	010046			MOV RO,-(SP)		
10988	063220	016646	000010		MOV 10(SP),-(SP)	: LUN,*	

Line	Address	Hex	Hex	Hex	Label	Code	Comment	Time	Page
10990								23-Oct-1980 15:01:43	TOPS
10991								23-Oct-1980 14:57:05	PA:<
10992									
10993	063224	042716	177770		BIC	#177770,(SP)			
10994	063230	012746	000001		MOV	#1,-(SP)			
10995	063234	005046			CLR	-(SP)			
10996	063236	004737	004244		JSR	PC,BL\$GT2			
10997	063242	062706	000010		ADD	#10,SP			
10998	063246	005300			DEC	R0			
10999	063250	C01143			BNE	7\$			
11000	063252	016637	000006	002074	MOV	6(SP),L\$LUN	: LUN,*		5869
11001	063260	012737	010706	030706	MOV	#WBUF,RPTR	:		5870
11002	063266	012737	020706	030710	MOV	#RBUF,RPTR	:		5871
11003	063274	016601	000006		MOV	6(SP),R1	: LUN,*		5872
11004	063300	006301			ASL	R1			
11005	063302	016104	032476		MOV	LOW.SECT(R1),R4	: *,SECTOR		
11006	063306	020461	032516	3\$:	CMP	R4,TOP.SECT(R1)	: SECTOR,*		5873
11007	063312	003122			BGT	7\$			
11008	063314	010446			MOV	R4,-(SP)	: SECTOR,*		5875
11009	063316	016146	032516		MOV	TOP.SECT(R1),-(SP)			
11010	063322	004737	044034		JSR	PC,GET.WRDCNT			
11011	063326	010003			MOV	R0,R3	: *,WRDCNT		
11012	063330	010316			MOV	R3,(SP)	: WRDCNT,*		5876
11013	063332	004737	043770		JSR	PC,SET.PTRS			
11014	063336	016616	000012		MOV	12(SP),(SP)	: LUN,*		5877
11015	063342	010346			MOV	R3,-(SP)	: WRDCNT,*		
11016	063344	013746	030706		MOV	WPTR,-(SP)			
11017	063350	010446			MOV	R4,-(SP)	: SECTOR,*		
11018	063352	004737	042246		JSR	PC,WRITE			
11019	063356	010005			MOV	R0,R5	: *,VALUE		
11020	063360	020527	000001		CMP	R5,#1	: VALUE,*		5881
11021	063364	001035			BNE	4\$			
11022	063366	012746	000006		MOV	#6,-(SP)	:		5885
11023	063372	012746	042246		MOV	#WRITE,-(SP)			
11024	063376	016646	000024		MOV	24(SP),-(SP)	: LUN,*		
11025	063402	010346			MOV	R3,-(SP)	: WRDCNT,*		
11026	063404	013746	030706		MOV	WPTR,-(SP)			
11027	063410	010446			MOV	R4,-(SP)	: SECTOR,*		
11028	063412	004737	043046		JSR	PC,RETRY			
11029	063416	062706	000014		ADD	#14,SP			
11030	063422	005700			TST	R0			
11031	063424	001456			BEQ	8\$			
11032	063426	016602	000020		MOV	20(SP),R2	: LUN,*		5888
11033	063432	112762	000004	032464	MOVB	#4,WHY.DROPT(R2)			
11034	063440	104455			TRAP	55	:		5889
11035	063442	011755			.WORD	11755			
11036	063444	010512			.WORD	MSG1			
11037	063446	000000			.WORD	0			
11038	063450	016600	000020		MOV	20(SP),R0	: LUN,*		5890
11039	063454	104451			TRAP	51			
11040	063456	000436			BR	6\$	:		5891
11041	063460	020527	000002	4\$:	CMP	R5,#2	: VALUE,*		5881
11042	063464	001015			BNE	5\$			
11043	063466	016602	000020		MOV	20(SP),R2	: LUN,*		5896
11044	063472	112762	000005	032464	MOVB	#5,WHY.DROPT(R2)			

Address	Hex	Hex	Hex	Label	Instruction	Comments	Seq
11046				:MLX4			
11047				:			
11048					TESTING RANDOM DATA		
11049	063500	104455			TRAP 55		5897
11050	063502	011756			.WORD 11756		
11051	063504	010512			.WORD MSG1		
11052	063506	000000			.WORD 0		
11053	063510	016600	000020		MOV 20(SP),R0	: LUN,*	5898
11054	063514	104451			TRAP 51		
11055	063516	000416			BR 6\$		5899
11056	063520	020527	000003	5\$:	CMP R5,#3	: VALUE,*	5881
11057	063524	001016			BNE 8\$		
11058	063526	104455			TRAP 55		5903
11059	063530	011757			.WORD 11757		
11060	063532	010512			.WORD MSG1		
11061	063534	000000			.WORD 0		
11062	063536	016602	000020		MOV 20(SP),R2	: LUN,*	5904
11063	063542	112762	000006	032464	MOVB #6,WHY.DROPT(R2)		
11064	063550	010200			MOV R2,R0	: LUN,*	5905
11065	063552	104451			TRAP 51		
11066	063554	062706	000012	6\$:	ADD #12,SP		5906
11067	063560	000543		7\$:	BR 13\$		
11068	063562	004737	043010	8\$:	JSR PC,CHOOSE		5911
11069	063566	010066	000032		MOV R0,32(SP)	: *,COMMAND	
11070	063572	005002			CLR R2		5912
11071	063574	020027	042434		CMP R0,#READ	: COMMAND,*	
11072	063600	001015			BNE 9\$		
11073	063602	005202			INC R2		
11074	063604	013766	030710	000030	MOV RPTR,30(SP)	: *,PTR	5915
11075	063612	016646	000020		MOV 20(SP),-(SP)	: LUN,*	5916
11076	063616	010346			MOV R3,-(SP)	: WRDCNT,*	
11077	063620	013746	030710		MOV RPTR,-(SP)		
11078	063624	010446			MOV R4,-(SP)	: SECTOR,*	
11079	063626	004737	042434		JSR PC,READ		
11080	063632	000413			BR 10\$		
11081	063634	013766	030706	000030	MOV WPTR,30(SP)	: *,PTR	5920
11082	063642	016646	000020		MOV 20(SP),-(SP)	: LUN,*	5921
11083	063646	010346			MOV R3,-(SP)	: WRDCNT,*	
11084	063650	013746	030706		MOV WPTR,-(SP)		
11085	063654	010446			MOV R4,-(SP)	: SECTOR,*	
11086	063656	004737	042622		JSR PC,CHECK		
11087	063662	010005		10\$:	MOV R0,R5	: *,VALUE	
11088	063664	001102			BNE 14\$		5926
11089	063666	006002			ROR R2		5929
11090	063670	103402			BLO 12\$		
11091	063672	000137	064712	11\$:	JMP 27\$		
11092	063676	013746	030706	12\$:	MOV WPTR,-(SP)		5932
11093	063702	013746	030710		MOV RPTR,-(SP)		
11094	063706	010346			MOV R3,-(SP)	: WRDCNT,*	
11095	063710	004737	041226		JSR PC,DOUBLE.CHECK		
11096	063714	062706	000006		ADD #6,SP		
11097	063720	010066	000044		MOV R0,44(SP)	: *,DBL.VALUE	
11098	063724	001762			BEQ 11\$		
11099	063726	016646	000030		MOV 30(SP),-(SP)	: LUN,*	5935
11100	063732	004737	033436		JSR PC,SAYWHO		

Address	Hex	Hex	Hex	Label	Comment	Address	Time	Page
11102				:MLX4		23-Oct-1980 15:01:43	TOPS	
11103				:	TESTING RANDOM DATA	23-Oct-1980 14:57:05	PA:<	
11104								
11105	063736	012716	010640	MOV	#MSG5,(SP)			5936
11106	063742	012746	006514	MOV	#SAY1,-(SP)			
11107	063746	012746	000002	MOV	#2,-(SP)			
11108	063752	010600		MOV	SP,R0	: SP,*		
11109	063754	104414		TRAP	14			
11110	063756	016616	000052	MOV	52(SP),(SP)	: DBL.VALUE,*		5938
11111	063762	017646	000052	MOV	@52(SP),-(SP)	: DBL.VALUE,*		
11112	063766	012746	006332	MOV	#FMT12A,-(SP)			
11113	063772	012746	000003	MOV	#3,-(SP)			
11114	063776	010600		MOV	SP,R0	: SP,*		
11115	064000	104414		TRAP	14			
11116	064002	062766	010000 000060	ADD	#10000,60(SP)	: *,DBL.VALUE		5940
11117	064010	016616	000060	MOV	60(SP),(SP)	: DBL.VALUE,*		5941
11118	064014	017646	000060	MOV	@60(SP),-(SP)	: DBL.VALUE,*		
11119	064020	012746	006400	MOV	#FMT12B,-(SP)			
11120	064024	012746	000003	MOV	#3,-(SP)			
11121	064030	010600		MOV	SP,R0	: SP,*		
11122	064032	104414		TRAP	14			
11123	064034	016602	000052	MOV	52(SP),R2	: LUN,*		5943
11124	064040	112762	000010 032464	MOVB	#10,WHY.DROPT(R2)			
11125	064046	104455		TRAP	55			5944
11126	064050	011760		.WORD	11760			
11127	064052	010512		.WORD	MSG1			
11128	064054	000000		.WORD	0			
11129	064056	016600	000052	MOV	52(SP),R0	: LUN,*		5945
11130	064062	104451		TRAP	51			
11131	064064	062706	000044	ADD	#44,SP			5946
11132	064070	000521		BR	19\$			
11133	064072	020527	000001	CMP	R5,#1	: VALUE,*		5926
11134	064076	001035		BNE	15\$			
11135	064100	012746	000006	MOV	#6,-(SP)			5951
11136	064104	016646	000044	MOV	44(SP),-(SP)	: COMMAND,*		
11137	064110	016646	000034	MOV	34(SP),-(SP)	: LUN,*		
11138	064114	010346		MOV	R3,-(SP)	: WRDCNT,*		
11139	064116	016646	000050	MOV	50(SP),-(SP)	: PTR,*		
11140	064122	010446		MOV	R4,-(SP)	: SECTOR,*		
11141	064124	004737	043046	JSR	PC,RETRY			
11142	064130	062706	000014	ADD	#14,SP			
11143	064134	005700		TST	R0			
11144	064136	001655		BEQ	11\$			
11145	064140	016602	000030	MOV	30(SP),R2	: LUN,*		5954
11146	064144	112762	000004 032464	MOVB	#4,WHY.DROPT(R2)			
11147	064152	104455		TRAP	55			5955
11148	064154	011761		.WORD	11761			
11149	064156	010512		.WORD	MSG1			
11150	064160	000000		.WORD	0			
11151	064162	016600	000030	MOV	30(SP),R0	: LUN,*		5955
11152	064166	104451		TRAP	51			
11153	064170	000457		BR	18\$			5957
11154	064172	020527	000002	CMP	R5,#2	: VALUE,*		5926
11155	064176	001015		BNE	16\$			
11156	064200	016602	000030	MOV	30(SP),R2	: LUN,*		5962

Address	OpCode	Operand 1	Operand 2	Operand 3	Instruction	Comments	Sequence
11158					:MLX4		
11159					:	TESTING RANDOM DATA	23-Oct-1980 15:01:43 TOPS
11160							23-Oct-1980 14:57:05 PA:<
11161	064204	112762	000005	032464	MOVB #5,WHY.DROPT(R2)		
11162	064212	104455			TRAP 55	:	5963
11163	064214	011762			.WORD 11762		
11164	064216	010512			.WORD MSG1		
11165	064220	000000			.WORD 0		
11166	064222	016600	000030		MOV 30(SP),R0	: LUN,*	5964
11167	064226	104451			TRAP 51		
11168	064230	000437			BR 18\$	:	5965
11169	064232	020527	000003	16\$:	CMP R5,#3	: VALUE,*	5926
11170	064236	001014			BNE 17\$		
11171	064240	104455			TRAP 55	:	5969
11172	064242	011763			.WORD 11763		
11173	064244	010512			.WORD MSG1		
11174	064246	000000			.WORD 0		
11175	064250	016602	000030		MOV 30(SP),R2	: LUN,*	5970
11176	064254	112762	000006	032464	MOVB #6,WHY.DROPT(R2)		
11177	064262	010200			MOV R2,R0	: LUN,*	5971
11178	064264	104451			TRAP 51		
11179	064266	000420			BR 18\$	:	5972
11180	064270	020527	000004	17\$:	CMP R5,#4	: VALUE,*	5926
11181	064274	001021			BNE 20\$		
11182	064276	004737	035212		JSR PC,ISOLATE	:	5976
11183	064302	104455			TRAP 55	:	5977
11184	064304	011764			.WORD 11764		
11185	064306	010542			.WORD MSG2		
11186	064310	000000			.WORD 0		
11187	064312	016602	000030		MOV 30(SP),R2	: LUN,*	5978
11188	064316	112762	000007	032464	MOVB #7,WHY.DROPT(R2)		
11189	064324	010200			MOV R2,R0	: LUN,*	5979
11190	064326	104451			TRAP 51		
11191	064330	062706	000022	18\$:	ADD #22,SP	:	5980
11192	064334	000137	064754	19\$:	JMP 28\$		
11193	064340	020527	000005	20\$:	CMP R5,#5	: VALUE,*	5926
11194	064344	001162			BNE 27\$		
11195	064346	004737	035212		JSR PC,ISOLATE	:	5984
11196	064352	032737	000001	002260	BIT #1,ERROUT	:	5985
11197	064360	001423			BEQ 21\$		
11198	064362	017702	146040		MOV @ML.REG+42,R2	:	5986
11199	064366	006202			ASR R2		
11200	064370	006202			ASR R2		
11201	064372	006202			ASR R2		
11202	064374	006202			ASR R2		
11203	064376	006202			ASR R2		
11204	064400	006202			ASR R2		
11205	064402	042702	177700		BIC #177700,R2		
11206	064406	010246			MOV R2,-(SP)		
11207	064410	012746	006262		MOV #FMT10B,-(SP)		
11208	064414	012746	000002		MOV #2,-(SP)		
11209	064420	010600			MOV SP,R0	: SP,*	
11210	064422	104414			TRAP 14		
11211	064424	062706	000006		ADD #6,SP		
11212	064430	017766	145774	000034 21\$:	MOV @ML.REG+44,34(SP)	: *.OLDSEC	5983

23-Oct-1980 15:01:43 TOPS  
 23-Oct-1980 14:57:05 PA:<

Address	Hex	Hex	Hex	Instruction	Comments	Seq
11214						
11215						
11216						
11217	064436	017702	145764	MOV @ML.REG+42,R2		5989
11218	064442	006202		ASR R2		
11219	064444	006202		ASR R2		
11220	064446	006202		ASR R2		
11221	064450	006202		ASR R2		
11222	064452	006202		ASR R2		
11223	064454	006202		ASR R2		
11224	064456	042702	177700	BIC #177700,R2		
11225	064462	010266	0C0036	MOV R2,36(SP)	: *,OLDCHN	
11226	064466	012746	000001	MOV #1,-(SP)		5990
11227	064472	016646	000044	MOV 44(SP),-(SP)	: COMMAND,*	
11228	064476	016646	000034	MOV 34(SP),-(SP)	: LUN,*	
11229	064502	010346		MOV R3,-(SP)	: WRDCNT,*	
11230	064504	016646	000050	MOV 50(SP),-(SP)	: PTR,*	
11231	064510	010446		MOV R4,-(SP)	: SECTOR,*	
11232	064512	004737	043046	JSR PC,RETRY		
11233	064516	062706	000014	ADD #14,SP		
11234	064522	020027	000005	CMP R0,#5		
11235	064526	001052		BNE 24\$		
11236	064530	027766	145674 000034	CMP @ML.REG+44,34(SP)	: *,OLDSEC	5992
11237	064536	001035		BNE 23\$		
11238	064540	016600	000036	MOV 36(SP),R0	: OLDCHN,*	
11239	064544	017702	145656	MOV @ML.REG+42,R2		
11240	064550	006202		ASR R2		
11241	064552	006202		ASR R2		
11242	064554	006202		ASR R2		
11243	064556	006202		ASR R2		
11244	064560	006202		ASR R2		
11245	064562	006202		ASR R2		
11246	064564	042702	177700	BIC #177700,R2		
11247	064570	020200		CMP R2,#0		
11248	064572	001017		BNE 23\$		
11249	064574	032737	000001 002260	BIT #1,ERROUT		5995
11250	064602	001404		BEQ 22\$		
11251	064604	1C4456		TRAP 56		5997
11252	064606	011765		.WORD 11765		
11253	064610	010624		.WORD MSG4		
11254	064612	000000		.WORD 0		
11255	064614	016646	000030 22\$:	MOV 30(SP),-(SP)	: LUN,*	5998
11256	064620	013746	032360	MOV BOARD,-(SP)		
11257	064624	004737	035302	JSR PC,UP.HARD.COUNT		
11258	064630	000427		BR 26\$		5992
11259	064632	032737	000001 002260 23\$:	BIT #1,ERROUT		6002
11260	064640	001415		BEQ 25\$		
11261	064642	104457		TRAP 57		6004
11262	064644	011766		.WORD 11766		
11263	064646	010610		.WORD MSG3		
11264	064650	000000		.WORD 0		
11265	064652	000410		BR 25\$		6005
11266	064654	032737	000001 002260 24\$:	BIT #1,ERROUT		6009
11267	064662	001404		BEQ 25\$		
11268	064664	104457		TRAP 57		6011

```

11270 ;MLX4
11271 :
11272 : TESTING RANDOM DATA
11273 064666 011767 .WORD 11767
11274 064670 010610 .WORD MSG3
11275 064672 000000 .WORD 0
11276 064674 016646 000030 25$: MOV 30(SP),-(SP) ; LUN,* 6012
11277 064700 013746 032360 MOV BOARD,-(SP)
11278 064704 004737 035430 JSR PC,UP.SOFT.COUNT
11279 064710 022626 26$: CMP (SP)+,(SP)+ ;
11280 064712 010300 27$: MOV R3,RO ; WRDCNT,* 5983
11281 064714 006300 ASL RO ;
11282 064716 063700 030706 ADD WPTR,RO ;
11283 064722 010037 030706 MOV RO,WPTR ;
11284 064726 010316 000400 MOV R3,(SP) ; WRDCNT,* 6019
11285 064730 012746 005056 MOV #400,-(SP)
11286 064734 004737 060400 JSR PC,BLS$DIV ; SECTOR,*
11287 064740 010004 010004 MOV R4,RO ; *,SECTOR
11288 064742 062706 000024 ADD RO,R4 ;
11289 064744 000137 063306 JMP #24,SP ;
11290 064750 005266 000006 28$: INC 6(SP) ;
11291 064754 026666 000006 000010 29$: CMP 6(SP),10(SP) ; LUN
11292 064760 002002 063200 BGE 30$ ; LUN,* 5874
11293 064770 000137 000024 JMP 2$ ;
11294 064774 005266 000024 30$: INC 24(SP) ; COUNT 5873
11295 065000 026666 000024 000044 CMP 24(SP),44(SP) ; LUN,* 5862
11296 065006 003002 BGT 31$ ; COUNT,REPEAT
11297 065010 000137 063156 JMP 1$ ;
11298 065014 062706 000026 31$: ADD #26,SP ;
11299 065020 000207 RTS PC ;
11300 ;
11301 ;
11302 ; Routine Size: 482 words
11303 ; Maximum stack depth per invocation: 35 words
11308 ;
11309 ;
11310 : 6028

```

23-Oct-1980 15:01:43 TOPS  
 23-Oct-1980 14:57:05 PA:<

M  
R



```

11312 :MLX4
11313 :
11314 :
11315 : 6029 %SBTTL 'TESTING RANDOM DATA & WORD COUNTS'
11316 : 6030
11317 : 6031 ROUTINE RAND2(REPEAT): NOVALUE =
11318 : 6032 BEGIN !* 1 * START OF ROUTINE
11319 : 6033
11320 : 6034 !+
11321 : 6035 ROUTINE: RAND2(REPEAT)
11322 : 6036
11323 : 6037 PURPOSE: TO TEST ALL UNITS IN A SEQUENTIAL FASHION, BUT
11324 : 6038 USING RANDOM WORD COUNTS
11325 : 6039
11326 : 6040 ARGUMENT: REPEAT = NUMBER OF TIMES TO EXECUTE THIS ROUTINE
11327 : 6041 BEFORE RETURNING TO THE CALLER (OPT5).
11328 : 6042
11329 : 6043 NOTE. SINCE ONLY THE WORD COUNTS AND THE DATA ARE RANDOM,
11330 : 6044 THE OTHER TRANSFER VALUES MUST BE SET UP BY THIS
11331 : 6045 ROUTINE. THESE VALUES INCLUDE:
11332 : 6046 THE LUN, SECTOR AND BUFFER POINTERS.
11333 : 6047
11334 : 6048 THE CODE FOR 'RAND2' IN BRIEF:
11335 : 6049
11336 : 6050 BEGIN 1 (START OF ROUTINE)
11337 : 6051 SAY ROUTINE IS RUNNING
11338 : 6052 INCR COUNT FROM 1 TO REPEAT
11339 : 6053 : BEGIN 2 (START OF REPEAT LOOP FOR THE ROUTINE)
11340 : 6054 : INCR LUN FROM 0 TO LAST
11341 : 6055 : : BEGIN 3 (START OF LOGICAL UNIT SELECTION LOOP)
11342 : 6056 : : GENERATE THE RANDOM PATTERN
11343 : 6057 : : TESTLOOP:
11344 : 6058 : : : BEGIN 4 (START OF LOOP THAT COMPLETELY TESTS 1 UNIT)
11345 : 6059 : : : IF UNIT IS ACTIVE
11346 : 6060 : : : THEN
11347 : 6061 : : : : BEGIN 5 (START OF TEST FOR AN ACTIVE UNIT)
11348 : 6062 : : : : SECTOR = LOWEST
11349 : 6063 : : : : INITIALIZE THE WRITE AND READ BUFFER POINTERS
11350 : 6064 : : : : WHILE SECTOR LEQ HIGHEST DO
11351 : 6065 : : : : : BEGIN 6 (START OF A PASS THROUGH ALL SECTORS)
11352 : 6066 : : : : : CHOOSE A RANDOM WORD COUNT
11353 : 6067 : : : : : SET UP BUFFER POINTERS BEFORE TRANSFER
11354 : 6068 : : : : : CALCULATE NEXT STARTING SECTOR (BASED ON WORD COUNT)
11355 : 6069 : : : : : IF NEXT STARTING SECTOR GTR HIGHEST
11356 : 6070 : : : : : THEN ADJUST THE WORD COUNT & NEXT SECTOR SO THEY FIT
11357 : 6071 : : : : : : WITHIN THE TESTABLE SECTOR LIMITS
11358 : 6072 : : : : : WRITE
11359 : 6073 : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
11360 : 6074 : : : : : CHOOSE WHETHER TO WRITE CHECK OR READ
11361 : 6075 : : : : : DO THE WRITE CHECK OR READ
11362 : 6076 : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
11363 : 6077 : : : : : SECTOR = THE CALCULATED NEXT STARTING SECTOR
11364 : 6078 : : : : : CHANGE BUFFER POINTERS AFTER TRANSFER
11365 : 6079 : : : : : END 6 (END OF A PASS THROUGH ALL SECTORS)
11366 : 6080 : : : : END 5 (END OF TEST FOR AN ACTIVE UNIT)
  
```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.2LI.1 (49)

```

11368 ;MLX4
11369 :
11370 :
11371 : 6081 : : : END 4 (END TESTLOOP)
11372 : 6082 : : : END 3 (END OF LOGICAL UNIT SELECTION LOOP)
11373 : 6083 : : : END 2 (END OF REPEAT LOOP FOR THIS ROUTINE)
11374 : 6084 : RETURN
11375 : 6085 : END 1 (END OF ROUTINE)
11376 : 6086 : --
11377 : 6087 :
11378 : 6088 : LOCAL
11379 : 6089 : WRDCNT, SECTOR, NEXT_SECT, VALUE, OLDSEC, OLDCHN,
11380 : 6090 : PTR, COMMAND, DBL_VALUE;
11381 : 6091 :
11382 : 6092 : LABEL
11383 : 6093 : LOOP;
11384 : 6094 :
11385 : 6095 : PRINTB(SAY1,RTN5B);
11386 : 6096 : !'RAND2'
11387 : 6097 : INCR COUNT FROM 1 TO .REPEAT DO
11388 : 6098 : BEGIN !* 2 * START OF REPEAT LOOP FOR THIS ROUTINE
11389 : 6099 : INCR LUN FROM 0 TO (.L$UNIT - 1) DO
11390 : 6100 : BEGIN !* 3 * START OF LOGICAL UNIT SELECTION LOOP
11391 : 6101 : GEN5(); !RANDOM DATA PATTERN
11392 : 6102 : LOOP:
11393 : 6103 : BEGIN !* 4 * START OF THE LOOP FOR COMPLETELY TESTING 1 UNIT
11394 : 6104 : IF .DRIVE_STATUS[.LUN] EQL ACTIVE
11395 : 6105 : THEN
11396 : 6106 : BEGIN !* 5 * START OF TEST FOR AN ACTIVE UNIT
11397 : 6107 : L$LUN = .LUN;
11398 : 6108 : SECTOR = LOWEST;
11399 : 6109 : WPTR = WBUFF;
11400 : 6110 : RPTR = RBUFF;
11401 : 6111 : WHILE .SECTOR LEQA HIGHEST DO
11402 : 6112 : BEGIN !* 6 * START OF A PASS THROUGH ALL SECTORS
11403 : 6113 : WRDCNT = RNDWC(); !EXPECT VALUE BETWEEN 1 AND BUFSIZ
11404 : 6114 : SET PTRS(.WRDCNT);
11405 : 6115 : NEXT_SECT = .SECTOR + (.WRDCNT/256);
11406 : 6116 : IF .SECTOR EQL .NEXT_SECT
11407 : 6117 : THEN NEXT_SECT = .NEXT_SECT + 1;
11408 : 6118 : IF .NEXT_SECT GTRA HIGHEST
11409 : 6119 : THEN
11410 : 6120 : BEGIN
11411 : 6121 : WRDCNT = 256 * (HIGHEST - .SECTOR + 1);
11412 : 6122 : NEXT_SECT = HIGHEST + 1;
11413 : 6123 : END;
11414 : 6124 :
11415 : 6125 : VALUE = WRITE(.LUN,.WRDCNT,.WPTR,.SECTOR);
11416 : 6126 : !*
11417 : 6127 : !* SEE HOW SUCCESSFUL THE WRITE WAS:
11418 : 6128 : !*
11419 : 6129 : SELECTONE .VALUE OF !SEE 'SYSERR' FOR DEFINITION
11420 : 6130 : SET !OF ERROR # CONTAINED IN 'VALUE'
11421 : 6131 : L1] :
11422 : 6132 : BEGIN !* 6A * RETRY ALLOWED
  
```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (49)

11424 :MLX4  
11425 :  
11426 :  
11427 :  
11428 :  
11429 :  
11430 :  
11431 :  
11432 :  
11433 :  
11434 :  
11435 :  
11436 :  
11437 :  
11438 :  
11439 :  
11440 :  
11441 :  
11442 :  
11443 :  
11444 :  
11445 :  
11446 :  
11447 :  
11448 :  
11449 :  
11450 :  
11451 :  
11452 :  
11453 :  
11454 :  
11455 :  
11456 :  
11457 :  
11458 :  
11459 :  
11460 :  
11461 :  
11462 :  
11463 :  
11464 :  
11465 :  
11466 :  
11467 :  
11468 :  
11469 :  
11470 :  
11471 :  
11472 :  
11473 :  
11474 :  
11475 :  
11476 :  
11477 :  
11478 :

6133  
6134  
6135  
6136  
6137  
6138  
6139  
6140  
6141  
6142  
6143  
6144  
6145  
6146  
6147  
6148  
6149  
6150  
6151  
6152  
6153  
6154  
6155  
6156  
6157  
6158  
6159  
6160  
6161  
6162  
6163  
6164  
6165  
6166  
6167  
6168  
6169  
6170  
6171  
6172  
6173  
6174  
6175  
6176  
6177  
6178  
6179  
6180  
6181  
6182  
6183  
6184

TESTING RANDOM DATA & WORD COUNTS

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (49)

```
IF RETRY(SIX,WRITE,.LUN,.WRDCNT,.WPTR,.SECTOR) NEQ 0
THEN !THE RETRY FAILED -- SYSTEM FATAL ERROR
BEGIN
WHY DROPT[.LUN] = CODE_4;
ERRDF(5201,MSG1,0); !**** OPTION 5, RAND2 ERROR 01 ****
DODU(.LUN);
LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
END;
[2] : !* 6A *
BEGIN !* 6B * FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
WHY DROPT[.LUN] = CODE_5;
ERRDF(5202,MSG1,0); !**** OPTION 5, RAND2 ERROR 02 ****
DODU(.LUN);
LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
END; !* 6B *
[3] : !* 6C * FATAL DRIVE ERROR -- NO RETRY ALLOWED
ERRDF(5203,MSG1,0); !**** OPTION 5, RAND2 ERROR 03 ****
WHY DROPT[.LUN] = CODE_6;
DODU(.LUN);
LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
END; !* 6C *
TES;
COMMAND = CHOOSE();
IF .COMMAND EQL READ
THEN
BEGIN
PTR = .RPTR;
VALUE = READ(.LUN,.WRDCNT,.RPTR,.SECTOR);
END
ELSE
BEGIN
PTR = .WPTR;
VALUE = CHECK(.LUN,.WRDCNT,.WPTR,.SECTOR);
END;
!+
!SEE HOW SUCCESSFUL THE OPERATION WAS:
!-
SELECTONE .VALUE OF !SEE 'SYSERR' FOR DEFINITION
SET !OF ERROR # CONTAINED IN 'VALUE'
[0] :
IF .COMMAND EQL READ
THEN
BEGIN
IF (DBL_VALUE = DOUBLE_CHECK(.WPTR,.RPTR,.WRDCNT)) NEQ 0
THEN
BEGIN
SAYWHO(.LUN);
PRINTB(SAY1,MSG5);
```

11480 :MLX4  
11481 :  
11482 :  
11483 :  
11484 :  
11485 :  
11486 :  
11487 :  
11488 :  
11489 :  
11490 :  
11491 :  
11492 :  
11493 :  
11494 :  
11495 :  
11496 :  
11497 :  
11498 :  
11499 :  
11500 :  
11501 :  
11502 :  
11503 :  
11504 :  
11505 :  
11506 :  
11507 :  
11508 :  
11509 :  
11510 :  
11511 :  
11512 :  
11513 :  
11514 :  
11515 :  
11516 :  
11517 :  
11518 :  
11519 :  
11520 :  
11521 :  
11522 :  
11523 :  
11524 :  
11525 :  
11526 :  
11527 :  
11528 :  
11529 :  
11530 :  
11531 :  
11532 :  
11533 :  
11534 :

6185  
6186  
6187  
6188  
6189  
6190  
6191  
6192  
6193  
6194  
6195  
6196  
6197  
6198  
6199  
6200  
6201  
6202  
6203  
6204  
6205  
6206  
6207  
6208  
6209  
6210  
6211  
6212  
6213  
6214  
6215  
6216  
6217  
6218  
6219  
6220  
6221  
6222  
6223  
6224  
6225  
6226  
6227  
6228  
6229  
6230  
6231  
6232  
6233  
6234  
6235  
6236

TESTING RANDOM DATA & WORD COUNTS

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (49)

```
! 'ECC LOGIC FAILED TO DETECT DATA ERROR'  
PRINTB(FMT12A,..DBL VALUE,..DBL VALUE);  
! 'GOOD DATA: XXXXXX AT LOCATION YYYYYY'  
DBL VALUE = .DBL VALUE + BUFSIZ * 2;  
PRINTB(FMT12B,..DBL VALUE,..DBL VALUE);  
! 'BAD DATA: PPPPPP AT LOCATION QQQQQQ'  
WHY DROPT(.LUN) = CODE 8;  
ERRDF(5204,MSG1,0); !**** OPTION 5, RAND2 ERROR 04 ****  
DODU(.LUN);  
LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *  
END;  
END;  
[1] :  
BEGIN !* 6D * RETRY ALLOWED  
IF RETRY(SIX,.COMMAND,.LUN,.WRDCNT,.PTR,.SECTOR) NEQ 0  
THEN !THE RETRY FAILED -- SYSTEM FATAL ERROR  
BEGIN  
WHY DROPT(.LUN) = CODE 4;  
ERRDF(5205,MSG1,0); !**** OPTION 5, RAND2 ERROR 05 ****  
DODU(.LUN);  
LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *  
END;  
END; !* 6D *  
[2] :  
BEGIN !* 6E * FATAL CONTROLLER ERROR -- NO RETRY ALLOWED  
WHY DROPT(.LUN) = CODE 5;  
ERRDF(5206,MSG1,0); !**** OPTION 5, RAND2 ERROR 06 ****  
DODU(.LUN);  
LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *  
END; !* 6E *  
[3] :  
BEGIN !* 6F * FATAL DRIVE ERROR -- NO RETRY ALLOWED  
ERRDF(5207,MSG1,0); !**** OPTION 5, RAND2 ERROR 07 ****  
WHY DROPT(.LUN) = CODE_6;  
DODU(.LUN);  
LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *  
END; !* 6F *  
[4] :  
BEGIN !* 6G * UNRECOVERABLE DATA ERROR  
ISOLATE();  
ERRDF(5208,MSG2,0); !**** OPTION 5, RAND2 ERROR 08 ****  
WHY DROPT(.LUN) = CODE_7;  
DODU(.LUN);  
LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *  
END; !* 6G *  
[5] :  
BEGIN !* 6H * RECOVERABLE DATA ERROR  
ISOLATE();  
IF .ERROUT  
THEN PRINTB(FMT10B,.CHAN);  
! ' BIT QQ'  
OLDSEC = .MLEL;
```

```

11536 :MLX4
11537 :
11538 :
11539 : 6237 OLDCHN = .CHAN;
11540 : 6238 IF REPLY(ONE,.COMMAND,.LUN,.WRDCNT,.PTR,.SECTOR) EQL 5
11541 : 6239 THEN
11542 : 6240 IF ((.MLEL EQL .OLDSEC) AND (.CHAN EQL .OLDCHN))
11543 : 6241 THEN
11544 : 6242 BEGIN
11545 : 6243 IF .ERROUT
11546 : 6244 THEN
11547 : 6245 ERRHRD(5209,MSG4,0); !**** OPTION 5, RAND2 ERROR 09 ****
11548 : 6246 UP_HARD_COUNT(.LUN,.BOARD);
11549 : 6247 END
11550 : 6248 ELSE
11551 : 6249 BEGIN
11552 : 6250 IF .ERROUT
11553 : 6251 THEN
11554 : 6252 ERRSOFT(5210,MSG3,0); !**** OPTION 5, RAND2 ERROR 10 ****
11555 : 6253 UP_SOFT_COUNT(.LUN,.BOARD);
11556 : 6254 END
11557 : 6255 ELSE
11558 : 6256 BEGIN
11559 : 6257 IF .ERROUT
11560 : 6258 THEN
11561 : 6259 ERRSOFT(5211,MSG3,0); !**** OPTION 5, RAND2 ERROR 11 ****
11562 : 6260 UP_SOFT_COUNT(.LUN,.BOARD);
11563 : 6261 END;
11564 : 6262 END; !* 6H *
11565 : 6263
11566 : 6264 TES;
11567 : 6265
11568 : 6266 SECTOR = .NEXT_SECT;
11569 : 6267 WPTR = .WPTR + 2;
11570 : 6268 END; !* 6 * END OF A PASS THROUGH ALL SECTORS
11571 : 6269 END; !* 5 * END OF TEST FOR AN ACTIVE UNIT
11572 : 6270 END; !* 4 * END OF LOOP THAT COMPLETELY TESTS 1 UNIT
11573 : 6271 END; !* 3 * END OF LOGICAL UNIT SELECTION LOOP
11574 : 6272 END; !* 2 * END OF REPEAT LOOP FOR THIS ROUTINE
11575 : 6273 RETURN;
11576 : 6274 END; !* 1 * END OF ROUTINE

```

11581								
11585	065022	004137	005222					
11586	065026	162706	000022	RAND2:	JSR	R1,\$SAVE5	:	6031
11587	065032	012746	007314		SUB	#22,SP	:	
11588	065036	012746	006514		MOV	#RTN5B,-(SP)	:	6095
11589	065042	012746	000002		MOV	#SAY1,-(SP)	:	
11590	065046	010600			MOV	#2,-(SP)	:	
					MOV	SP,R0	:	SP,*

				:MLX4		TESTING RANDOM DATA & WORD COUNTS		23-Oct-1980 15:01:43	TOPS
				:				23-Oct-1980 14:57:05	PA:<
11592									
11593									
11594									
11595	065050	104414				TRAP	14		
11596	065052	005066	000016			CLR	16(SP)	: COUNT	6097
11597	065056	000137	066642			JMP	32\$		
11598	065062	013766	002012	000022	1\$:	MOV	L\$UNIT,22(SP)		6099
11599	065070	005001				CLR	R1	: LUN	
11600	065072	000137	066630			JMP	31\$		
11601	065076	004737	036710		2\$:	JSR	PC,GEN5		6101
11602	065102	010100				MOV	R1,RO	: LUN,*	6104
11603	065104	006200				ASR	RO		
11604	065106	006200				ASR	RO		
11605	065110	006200				ASR	RO		
11606	065112	062700	032460			ADD	#DRIVE.STATUS,RO		
11607	065116	010046				MOV	RO,-(SP)		
11608	065120	010146				MOV	R1,-(SP)	: LUN,*	
11609	065122	042716	177770			BIC	#177770,(SP)		
11610	065126	012746	000001			MOV	#1,-(SP)		
11611	065132	005046				CLR	-(SP)		
11612	065134	004737	004244			JSR	PC,BL\$GT2		
11613	065140	062706	000010			ADD	#10,SP		
11614	065144	005300				DEC	RO		
11615	065146	001166				BNE	9\$		
11616	065150	010137	002074			MOV	R1,L\$LUN	: LUN,*	6107
11617	065154	010100				MOV	R1,RO	: LUN,*	6108
11618	065156	006300				ASL	RO		
11619	065160	016004	032476			MOV	LOW.SECT(RO),R4	: *,SECTOR	
11620	065164	012737	010706	030706		MOV	#WBUF,WPTR		6109
11621	065172	012737	020706	030710		MOV	#RBUF,RPTR		6110
11622	065200	012766	032516	000020		MOV	#TOP.SECT,20(SP)		6111
11623	065206	060066	000020			ADD	RO,20(SP)		
11624	065212	020476	000020		3\$:	CMP	R4,@20(SP)	: SECTOR,*	
11625	065216	101142				BHI	9\$		
11626	065220	004737	062560			JSR	PC,RNDWC		6113
11627	065224	010003				MOV	RO,R3	: *,WRDCNT	
11628	065226	010346				MOV	R3,-(SP)	: WRDCNT,*	6114
11629	065230	004737	043770			JSR	PC,SET.PTRS		
11630	065234	010316				MOV	R3,(SP)	: WRDCNT,*	6115
11631	065236	012746	000400			MOV	#400,-(SP)		
11632	065242	004737	005056			JSR	PC,BL\$DIV		
11633	065246	060400				ADD	R4,RO	: SECTOR,*	
11634	065250	010066	000012			MOV	RO,12(SP)	: *,NEXT.SECT	
11635	065254	020400				CMP	R4,RO	: SECTOR,NEXT.SECT	6116
11636	065256	001002				BNE	4\$		
11637	065260	005266	000012			INC	12(SP)	: NEXT.SECT	6117
11638	065264	026676	000012	000024	4\$:	CMP	12(SP),@24(SP)	: NEXT.SECT,*	6118
11639	065272	101415				BLOS	5\$		
11640	065274	017600	000024			MOV	@24(SP),RO		6121
11641	065300	160400				SUB	R4,RO	: SECTOR,*	
11642	065302	000300				SWAB	RO		6120
11643	065304	105000				CLRB	RO		
11644	065306	010003				MOV	RO,R3	: *,WRDCNT	6121
11645	065310	062703	000400			ADD	#400,R3	: *,WRDCNT	
11646	065314	017666	000024	000012		MOV	@24(SP),12(SP)	: *,NEXT.SECT	6122

Address	Hex	Hex	Hex	Hex	Label	Code	Comment	Time	Page
11648					:MLX4			23-Oct-1980 15:01:43	TOPS
11649					:		TESTING RANDOM DATA & WORD COUNTS	23-Oct-1980 14:57:05	PA:<
11650									
11651	065322	005266	000012		5\$:	INC	12(SP) ; NEXT.SECT		
11652	065326	010116				MOV	R1,(SP) ; LUN,*		6125
11653	065330	010346				MOV	R3,-(SP) ; WRDCNT,*		
11654	065332	013746	030706			MOV	WPTR,-(SP)		
11655	065336	010446				MOV	R4,-(SP) ; SECTOR,*		
11656	065340	004737	042246			JSR	PC,WRITE		
11657	065344	010005				MOV	R0,R5 ; *,VALUE		
11658	065346	020527	000001			CMP	R5,#1 ; VALUE,*		6129
11659	065352	001031				BNE	6\$		
11660	065354	012746	000006			MOV	#6,-(SP) ;		6133
11661	065360	012746	042246			MOV	#WRITE,-(SP)		
11662	065364	010146				MOV	R1,-(SP) ; LUN,*		
11663	065366	010346				MOV	R3,-(SP) ; WRDCNT,*		
11664	065370	013746	030706			MOV	WPTR,-(SP)		
11665	065374	010446				MOV	R4,-(SP) ; SECTOR,*		
11666	065376	004737	043046			JSR	PC,RETRY		
11667	065402	062706	000014			ADD	#14,SP		
11668	065406	005700				TST	R0		
11669	065410	001446				BEQ	10\$		
11670	065412	112761	000004	032464		MOVB	#4,WHY.DROPT(R1) ; *,*(LUN)		6136
11671	065420	104455				TRAP	55 ;		6137
11672	065422	012121				.WORD	12121		
11673	065424	010512				.WORD	MSG1		
11674	065426	000000				.WORD	0		
11675	065430	010100				MOV	R1,R0 ; LUN,*		6138
11676	065432	104451				TRAP	51		
11677	065434	000431				BR	8\$ ;		6139
11678	065436	020527	000002		6\$:	CMP	R5,#2 ; VALUE,*		6129
11679	065442	001012				BNE	7\$		
11680	065444	112761	000005	032464		MOVB	#5,WHY.DROPT(R1) ; *,*(LUN)		6144
11681	065452	104455				TRAP	55 ;		6145
11682	065454	012122				.WORD	12122		
11683	065456	010512				.WORD	MSG1		
11684	065460	000000				.WORD	0		
11685	065462	010100				MOV	R1,R0 ; LUN,*		6146
11686	065464	104451				TRAP	51		
11687	065466	000414				BR	8\$ ;		6147
11688	065470	020527	000003		7\$:	CMP	R5,#3 ; VALUE,*		6129
11689	065474	001014				BNE	10\$		
11690	065476	104455				TRAP	55 ;		6151
11691	065500	012123				.WORD	12123		
11692	065502	010512				.WORD	MSG1		
11693	065504	000000				.WORD	0		
11694	065506	112761	000006	032464		MOVB	#6,WHY.DROPT(R1) ; *,*(LUN)		6152
11695	065514	010100				MOV	R1,R0 ; LUN,*		6153
11696	065516	104451				TRAP	51		
11697	065520	062706	000012		8\$:	ADD	#12,SP ;		6154
11698	065524	000535			9\$:	BR	15\$ ;		
11699	065526	004737	043010		10\$:	JSR	PC,CHOOSE ;		6159
11700	065532	010066	000024			MOV	R0,24(SP) ; *,COMMAND		
11701	065536	005002				CLR	R2 ;		6160
11702	065540	020027	042434			CMP	R0,#READ ; COMMAND,*		

Address	OpCode	Operand 1	Operand 2	Operand 3	Operand 4	Label	Instruction	Comments	Address	Time	Page
11704						:MLX4				23-Oct-1980 15:01:43	TOPS
11705						:				23-Oct-1980 14:57:05	PA:<
11706							TESTING RANDOM DATA & WORD COUNTS				
11707	065544	001014					BNE 11\$				
11708	065546	005202					INC R2				
11709	065550	013766	030710	000026			MOV RPTR,26(SP)	: *,PTR			6163
11710	065556	010146					MOV R1,-(SP)	: LUN,*			6164
11711	065560	010346					MOV R3,-(SP)	: WRDCNT,*			
11712	065562	013746	030710				MOV RPTR,-(SP)				
11713	065566	010446					MOV R4,-(SP)	: SECTOR,*			
11714	065570	004737	042434				JSR PC,READ				
11715	065574	000412					BR 12\$				
11716	065576	013766	030706	000026	11\$:		MOV WPTR,26(SP)	: *,PTR			6168
11717	065604	010146					MOV R1,-(SP)	: LUN,*			6169
11718	065606	010346					MOV R3,-(SP)	: WRDCNT,*			
11719	065610	013746	030706				MOV WPTR,-(SP)				
11720	065614	010446					MOV R4,-(SP)	: SECTOR,*			
11721	065616	004737	042622				JSR PC,CHECK				
11722	065622	010005			12\$:		MOV R0,R5	: *,VALUE			
11723	065624	001076					BNE 16\$				6174
11724	065626	006002					ROR R2				6177
11725	065630	103402					BLO 14\$				
11726	065632	000137	066604		13\$:		JMP 29\$				
11727	065636	013746	030706		14\$:		MOV WPTR,-(SP)				6180
11728	065642	013746	030710				MOV RPTR,-(SP)				
11729	065646	010346					MOV R3,-(SP)	: WRDCNT,*			
11730	065650	004737	041226				JSR PC,DOUBLE.CHECK				
11731	065654	062706	000006				ADD #6,SP				
11732	065660	010066	000032				MOV R0,32(SP)	: *,DBL.VALUE			
11733	065664	001762					BEQ 13\$				
11734	065666	010146					MOV R1,-(SP)	: LUN,*			6183
11735	065670	004737	033436				JSR PC,SAYWHO				
11736	065674	012716	010640				MOV #MSG5,(SP)				6184
11737	065700	012746	006514				MOV #SAY1,-(SP)				
11738	065704	012746	000002				MOV #2,-(SP)				
11739	065710	010600					MOV SP,R0	: SP,*			
11740	065712	104414					TRAP 14				
11741	065714	016616	000040				MOV 40(SP),(SP)	: DBL.VALUE,*			6186
11742	065720	017646	000040				MOV @40(SP),-(SP)	: DBL.VALUE,*			
11743	065724	012746	006332				MOV #FMT12A,-(SP)				
11744	065730	012746	000003				MOV #3,-(SP)				
11745	065734	010600					MOV SP,R0	: SP,*			
11746	065736	104414					TRAP 14				
11747	065740	062766	010000	000046			ADD #10000,46(SP)	: *,DBL.VALUE			6188
11748	065746	016616	000046				MOV 46(SP),(SP)	: DBL.VALUE,*			6189
11749	065752	017646	000046				MOV @46(SP),-(SP)	: DBL.VALUE,*			
11750	065756	012746	006400				MOV #FMT12B,-(SP)				
11751	065762	012746	000003				MOV #3,-(SP)				
11752	065766	010600					MOV SP,R0	: SP,*			
11753	065770	104414					TRAP 14				
11754	065772	112761	000010	032464			MOVB #10,WHY.DROPT(R1)	: *,*(LUN)			6191
11755	066000	104455					TRAP 55				6192
11756	066002	012124					.WORD 12124				
11757	066004	010512					.WORD MSG1				
11758	066006	000000					.WORD 0				



Address	OpCode	Operand 1	Operand 2	Operand 3	Label	Instruction	Comments	Address
11760					:MLX4			23-Oct-1980 15:01:43 TOPS
11761					:	TESTING RANDOM DATA & WORD COUNTS		23-Oct-1980 14:57:05 PA:<
11762								
11763	066010	010100				MOV R1,RO	: LUN,*	6193
11764	066012	104451				TRAP 51		
11765	066014	062706	000044			ADD #44,SP	:	6194
11766	066020	000506			15\$:	BR 21\$		
11767	066022	020527	000001		16\$:	CMP R5,#1	: VALUE,*	6174
11768	066026	001031				BNE 17\$		
11769	066030	012746	000006			MOV #6,-(SP)	:	6199
11770	066034	016646	000036			MOV 36(SP),-(SP)	: COMMAND,*	
11771	066040	010146				MOV R1,-(SP)	: LUN,*	
11772	066042	010346				MOV R3,-(SP)	: WRDCNT,*	
11773	066044	016646	000046			MOV 46(SP),-(SP)	: PTR,*	
11774	066050	010446				MOV R4,-(SP)	: SECTOR,*	
11775	066052	004737	043046			JSR PC,RETRY		
11776	066056	062706	000014			ADD #14,SP		
11777	066062	005700				TST R0		
11778	066064	001662				BEQ 13\$		
11779	066066	112761	000004	032464		MOVB #4,WHY.DROPT(R1)	: *,*(LUN)	6202
11780	066074	104455				TRAP 55	:	6203
11781	066076	012125				.WORD 12125		
11782	066100	010512				.WORD MSG1		
11783	066102	000000				.WORD 0		
11784	066104	010100				MOV R1,RO	: LUN,*	6204
11785	066106	104451				TRAP 51		
11786	066110	000450				BR 20\$	: VALUE,*	6205
11787	066112	020527	000002		17\$:	CMP R5,#2		6174
11788	066116	001012				BNE 18\$		
11789	066120	112761	000005	032464		MOVB #5,WHY.DROPT(R1)	: *,*(LUN)	6210
11790	066126	104455				TRAP 55	:	6211
11791	066130	012126				.WORD 12126		
11792	066132	010512				.WORD MSG1		
11793	066134	000000				.WORD 0		
11794	066136	010100				MOV R1,RO	: LUN,*	6212
11795	066140	104451				TRAP 51		
11796	066142	000433				BR 20\$	: VALUE,*	6213
11797	066144	020527	000003		18\$:	CMP R5,#3		6174
11798	066150	001012				BNE 19\$		
11799	066152	104455				TRAP 55	:	6217
11800	066154	012127				.WORD 12127		
11801	066156	010512				.WORD MSG1		
11802	066160	000000				.WORD 0		
11803	066162	112761	000006	032464		MOVB #6,WHY.DROPT(R1)	: *,*(LUN)	6218
11804	066170	010100				MOV R1,RO	: LUN,*	6219
11805	066172	104451				TRAP 51		
11806	066174	000416				BR 20\$	: VALUE,*	6220
11807	066176	020527	000004		19\$:	CMP R5,#4		6174
11808	066202	001016				BNE 22\$		
11809	066204	004737	035212			JSR PC,ISOLATE	:	6224
11810	066210	104455				TRAP 55	:	6225
11811	066212	012130				.WORD 12130		
11812	066214	010542				.WORD MSG2		
11813	066216	000000				.WORD 0		
11814	066220	112761	000007	032464		MOVB #7,WHY.DROPT(R1)	: *,*(LUN)	6225

Address	OpCode	Operand 1	Operand 2	Operand 3	Label	Instruction	Comments	Time	Page
11816								23-Oct-1980 15:01:43	TOPS
11817								23-Oct-1980 14:57:05	PA:<
11818									
11819	066226	010100				MOV R1,R0	; LUN,*		6227
11820	066230	104451				TRAP 51			
11821	066232	062706	000022			ADD #22,SP			6228
11822	066236	000573				BR 30\$			
11823	066240	020527	000005			CMP R5,#5	; VALUE,*		6174
11824	066244	001157				BNE 29\$			
11825	066246	004737	035212			JSR PC,ISOLATE			6232
11826	066252	032737	000001	002260		BIT #1,ERROUT			6233
11827	066260	001423				BEQ 23\$			
11828	066262	017702	144140			MOV @ML.REG+42,R2			6234
11829	066266	006202				ASR R2			
11830	066270	006202				ASR R2			
11831	066272	006202				ASR R2			
11832	066274	006202				ASR R2			
11833	066276	006202				ASR R2			
11834	066300	006202				ASR R2			
11835	066302	042702	177700			BIC #177700,R2			
11836	066306	010246				MOV R2,-(SP)			
11837	066310	012746	006262			MOV #FMT10B,-(SP)			
11838	066314	012746	000002			MOV #2,-(SP)			
11839	066320	010600				MOV SP,R0	; SP,*		
11840	066322	104414				TRAP 14			
11841	066324	062706	000006			ADD #6,SP			
11842	066330	017766	144074	000046	23\$:	MOV @ML.REG+44,46(SP)	; *.OLDSEC		6236
11843	066336	017702	144064			MOV @ML.REG+42,R2			6237
11844	066342	006202				ASR R2			
11845	066344	006202				ASR R2			
11846	066346	006202				ASR R2			
11847	066350	006202				ASR R2			
11848	066352	006202				ASR R2			
11849	066354	006202				ASR R2			
11850	066356	042702	177700			BIC #177700,R2			
11851	066362	010266	000050			MOV R2,50(SP)	; *.OLDCHN		
11852	066366	012746	000001			MOV #1,-(SP)			6238
11853	066372	016646	000036			MOV 36(SP),-(SP)	; COMMAND,*		
11854	066376	010146				MOV R1,-(SP)	; LUN,*		
11855	066400	010346				MOV R3,-(SP)	; WRDCNT,*		
11856	066402	016646	000046			MOV 46(SP),-(SP)	; PTR,*		
11857	066406	010446				MOV R4,-(SP)	; SECTOR,*		
11858	066410	004737	043046			JSR PC,RETRY			
11859	066414	062706	000014			ADD #14,SP			
11860	066420	020027	000005			CMP R0,#5			
11861	066424	001051				BNE 26\$			
11862	066426	027766	143776	000046		CMP @ML.REG+44,46(SP)	; *.OLDSEC		6240
11863	066434	001034				BNE 25\$			
11864	066436	016600	000050			MOV 50(SP),R0	; OLDCHN,*		
11865	066442	017702	143760			MOV @ML.REG+42,R2			
11866	066446	006202				ASR R2			
11867	066450	006202				ASR R2			
11868	066452	006202				ASR R2			
11869	066454	006202				ASR R2			
11870	066456	006202				ASR R2			

Address	Hex	Hex	Hex	Hex	Hex	Instruction	Comments	Address
11872								
11873								
11874								
11875	066460	006202				ASR R2		
11876	066462	042702	177700			BIC #177700,R2		
11877	066466	020200				CMP R2,R0		
11878	066470	001016				BNE 25\$		
11879	066472	032737	000001	002260		BIT #1,ERROUT	:	6243
11880	066500	001404				BEQ 24\$	:	
11881	066502	104456				TRAP 56	:	6245
11882	066504	012131				.WORD 12131		
11883	066506	010624				.WORD MSG4		
11884	066510	000000				.WORD 0		
11885	066512	010146			24\$:	MOV R1,-(SP)	: LUN,*	6246
11886	066514	013746	032360			MOV BOARD,-(SP)		
11887	066520	004737	035302			JSR PC,UP.HARD.COUNT		
11888	066524	000426				BR 28\$	:	6240
11889	066526	032737	000001	002260	25\$:	BIT #1,ERROUT	:	6250
11890	066534	001415				BEQ 27\$	:	
11891	066536	104457				TRAP 57	:	6252
11892	066540	012132				.WORD 12132		
11893	066542	010610				.WORD MSG3		
11894	066544	000000				.WORD 0		
11895	066546	000410				BR 27\$	:	6253
11896	066550	032737	000001	002260	26\$:	BIT #1,ERROUT	:	6257
11897	066556	001404				BEQ 27\$	:	
11898	066560	104457				TRAP 57	:	6259
11899	066562	012133				.WORD 12133		
11900	066564	010610				.WORD MSG3		
11901	066566	000000				.WORD 0		
11902	066570	010146			27\$:	MOV R1,-(SP)	: LUN,*	6260
11903	066572	013746	032360			MOV BOARD,-(SP)		
11904	066576	004737	035430			JSR PC,UP.SOFT.COUNT		
11905	066602	022626			28\$:	CMP (SP)+,(SP)+	:	6231
11906	066604	016604	000030		29\$:	MOV 30(SP),R4	: NEXT.SECT,SECTOR	6266
11907	066610	062737	000002	030706		ADD #2,WPTR	:	6267
11908	066616	062706	000022			ADD #22,SP	:	6112
11909	066622	000137	065212			JMP 3\$	:	6111
11910	066626	005201			30\$:	INC R1	: LUN	6099
11911	066630	020166	000022		31\$:	CMP R1,22(SP)	: LUN,*	
11912	066634	002002				BGE 32\$		
11913	066636	000137	065076			JMP 2\$		
11914	066642	005266	000016		32\$:	INC 16(SP)	: COUNT	6097
11915	066646	026666	000016	000046		CMP 16(SP),46(SP)	: COUNT,REPEAT	
11916	066654	003002				BGT 33\$		
11917	066656	000137	065062			JMP 1\$		
11918	066662	062706	000030		33\$:	ADD #30,SP	:	6031
11919	066666	000207				RTS PC		
11920								
11921								
11922								

: Routine Size: 467 words  
 : Maximum stack depth per invocation: 36 words

```

11928 :MLX4
11929 :
11930 :
11931 : 6275 %SBTTL 'TESTING RANDOM SECTORS, DATA, WORD COUNTS'
11932 : 6276
11933 : 6277 ROUTINE RAND3(REPEAT): NOVALUE =
11934 : 6278 BEGIN !* 1 * START OF ROUTINE
11935 : 6279
11936 : 6280 !++
11937 : 6281 ROUTINE: RAND3(REPEAT)
11938 : 6282
11939 : 6283 PURPOSE: TO CHECK RANDOM SECTORS
11940 : 6284
11941 : 6285 ARGUMENT: REPEAT = NUMBER OF TIMES TO EXECUTE ENTIRE ROUTINE
11942 : 6286
11943 : 6287 THE CODE FOR 'RAND3' IN BRIEF:
11944 : 6288
11945 : 6289 BEGIN 1 (START OF ROUTINE)
11946 : 6290 SAY ROUTINE IS RUNNING
11947 : 6291 INCR COUNT FROM 1 TO REPEAT
11948 : 6292 : BEGIN 2 (START OF REPEAT LOOP FOR THIS ROUTINE)
11949 : 6293 : GENERATE THE RANDOM PATTERN
11950 : 6294 : INCR LUN FROM 0 TO LAST
11951 : 6295 : : BEGIN 3 (START OF LOGICAL UNIT SELECTION LOOP)
11952 : 6296 : : INITIALIZE BUFFER POINTERS
11953 : 6297 : : TESTLOOP:
11954 : 6298 : : : BEGIN 4 (START OF LOOP THAT COMPLETELY TESTS 1 UNIT)
11955 : 6299 : : : IF UNIT IS ACTIVE
11956 : 6300 : : : THEN
11957 : 6301 : : : : BEGIN 5 (START OF TEST FOR AN ACTIVE UNIT)
11958 : 6302 : : : : TIMES = (HIGHEST - LOWEST)/2 + 1
11959 : 6303 : : : : INCR COUNT FROM 1 TO TIMES
11960 : 6304 : : : : : BEGIN 6 (START OF COUNTING LOOP FOR SECTOR SELECTION)
11961 : 6305 : : : : : CHOOSE A RANDOM WORD COUNT
11962 : 6306 : : : : : SET UP BUFFER POINTERS BEFORE TRANSFER
11963 : 6307 : : : : : CHOOSE A RANDOM SECTOR
11964 : 6308 : : : : : CALCULATE WHERE TRANSFER WILL END (BASED ON WORD COUNT)
11965 : 6309 : : : : : IF CALCULATED VALUE GTR HIGHEST
11966 : 6310 : : : : : : THEN ADJUST THE WORD COUNT SO IT FITS
11967 : 6311 : : : : : : : WITHIN THE TESTABLE SECTOR LIMITS
11968 : 6312 : : : : : WRITE
11969 : 6313 : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE LABEL)
11970 : 6314 : : : : : CHOOSE WHETHER TO WRITE CHECK OR READ
11971 : 6315 : : : : : DO THE WRITE CHECK OR READ
11972 : 6316 : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE LABEL)
11973 : 6317 : : : : : CHANGE BUFFER POINTERS AFTER TRANSFER
11974 : 6318 : : : : : END 6 (END OF COUNTING LOOP FOR SECTOR SELECTION)
11975 : 6319 : : : : : END 5 (END OF TEST FOR AN ACTIVE UNIT)
11976 : 6320 : : : : : END 4 (END OF TESTLOOP)
11977 : 6321 : : : : : END 3 (END OF LOGICAL UNIT SELECTION LOOP)
11978 : 6322 : : : : : END 2 (END OF REPEAT LOOP FOR THIS ROUTINE)
11979 : 6323 RETURN
11980 : 6324 END 1 (END OF ROUTINE)
11981 : 6325
11982 : 6326
  
```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (50)

```

11984 :MLX4
11985 :
11986 :
11987 : 6327 LOCAL
11988 : 6328 WRDCNT, SECTOR, TIMES, VALUE, OLDSEC, OLDCHN,
11989 : 6329 PTR, COMMAND, DBL_VALUE;
11990 : 6330
11991 : 6331 LABEL
11992 : 6332 LOOP;
11993 : 6333
11994 : 6334 PRINTB(SAY1,RTN5C);
11995 : 6335 !'RAND3'
11996 : 6336
11997 : 6337 INCR COUNT FROM 1 TO .REPEAT DO
11998 : 6338 BEGIN !* 2 * START OF REPEAT LOOP FOR THIS ROUTINE
11999 : 6339 INCR LUN FROM 0 TO (.L$UNIT - 1) DO
12000 : 6340 BEGIN !* 3 * START OF LOGICAL UNIT SELECTION LOOP
12001 : 6341 GEN5();
12002 : 6342 LOOP:
12003 : 6343 BEGIN !* 4 * START OF THE LOOP THAT COMPLETELY TESTS 1 UNIT
12004 : 6344 IF .DRIVE_STATUS[.LUN] EQL ACTIVE
12005 : 6345 THEN
12006 : 6346 BEGIN !* 5 * START OF TEST FOR AN ACTIVE UNIT
12007 : 6347 L$LUN = .LUN;
12008 : 6348 WPTR = WBUFF;
12009 : 6349 RPTR = RBUFF;
12010 : 6350 TIMES = (HIGHEST - LOWEST)/2 + 1;
12011 : 6351 INCR KOUNT FROM 1 TO .TIMES DO
12012 : 6352 BEGIN !* 6 * START OF COUNTING LOOP FOR SECTOR SELECTION
12013 : 6353 WRDCNT = RNDWC(); !VALUE BETWEEN 1 AND BUFSIZ
12014 : 6354 SET PTRS(.WRDCNT);
12015 : 6355 SECTOR = RNDSEC(.LUN); !VALUE BETWEEN LOWEST AND HIGHEST
12016 : 6356 IF (.SECTOR + .WRDCNT/256) GTRA HIGHEST
12017 : 6357 THEN WRDCNT = 256 * (HIGHEST - .SECTOR + 1);
12018 : 6358
12019 : 6359 VALUE = WRITE(.LUN,.WRDCNT,.WPTR,.SECTOR);
12020 : 6360 !+
12021 : 6361 ! SEE HOW SUCCESSFUL THE WRITE WAS:
12022 : 6362 !-
12023 : 6363 SELECTONE .VALUE OF !SEE 'SYSERR' FOR DEFINITION
12024 : 6364 SET !OF ERROR # CONTAINED IN 'VALUE'
12025 : 6365 [1] :
12026 : 6366 BEGIN !* 6A * RETRY ALLOWED
12027 : 6367 IF RETRY(SIX,WRITE,.LUN,.WRDCNT,.WPTR,.SECTOR) NEQ 0
12028 : 6368 THEN !THE RETRY FAILED -- SYSTEM FATAL ERROR
12029 : 6369 BEGIN
12030 : 6370 WHY DROPT[.LUN] = CODE_4;
12031 : 6371 ERRDF(5301,MSG1,0); !*** OPTION 5, RAND3 ERROR 01 ****
12032 : 6372 DODU(.LUN);
12033 : 6373 LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
12034 : 6374 END;
12035 : 6375 END; !* 6A *
12036 : 6376 [2] :
12037 : 6377 BEGIN !* 6B * FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
12038 : 6378 WHY_DROPT[.LUN] = CODE_5;
  
```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (50)

```

12040 :MLX4
12041 :
12042 :
12043 : 6379 ERRDF(5302,MSG1,0); .**** OPTION 5, RAND3 ERROR 02 ****
12044 : 6380 DODU(.LUN);
12045 : 6381 LEAVE LOOP; .JUMP JUST BEYOND END OF BLOCK * 4 *
12046 : 6382 END; !* 6B *
12047 : 6383 [3] :
12048 : 6384 BEGIN !* 6C * FATAL DRIVE ERROR -- NO RETRY ALLOWED
12049 : 6385 ERRDF(5303,MSG1,0); .**** OPTION 5, RAND3 ERROR 03 ****
12050 : 6386 WHY DROPT[.LUN] = CODE_6;
12051 : 6387 DODU(.LUN);
12052 : 6388 LEAVE LOOP; .JUMP JUST BEYOND END OF BLOCK * 4 *
12053 : 6389 END; !* 6C *
12054 : 6390
12055 : 6391 TES;
12056 : 6392
12057 : 6393 COMMAND = CHOOSE();
12058 : 6394 IF .COMMAND EQL READ
12059 : 6395 THEN
12060 : 6396 BEGIN
12061 : 6397 PTR = .RPTR;
12062 : 6398 VALUE = READ(.LUN,.WRDCNT,.RPTR,.SECTOR);
12063 : 6399 END
12064 : 6400 ELSE
12065 : 6401 BEGIN
12066 : 6402 PTR = .WPTR;
12067 : 6403 VALUE = CHECK(.LUN,.WRDCNT,.WPTR,.SECTOR);
12068 : 6404 END;
12069 : 6405 !+
12070 : 6406 !- SEE HOW SUCCESSFUL THE OPERATION WAS:
12071 : 6407 !-
12072 : 6408 SELECTONE .VALUE OF !SEE 'SYSERR' FOR DEFINITION
12073 : 6409 SET !OF ERROR # CONTAINED IN 'VALUE'
12074 : 6410 [0] :
12075 : 6411 IF .COMMAND EQL READ
12076 : 6412 THEN
12077 : 6413 BEGIN
12078 : 6414 IF (DBL_VALUE = DOUBLE_CHECK(.WPTR,.RPTR,.WRDCNT)) NEQ 0
12079 : 6415 THEN
12080 : 6416 BEGIN
12081 : 6417 SAYWHO(.LUN);
12082 : 6418 PRINTB(SAY1,MSG5);
12083 : 6419 !'ECC LOGIC FAILED TO DETECT DATA ERROR'
12084 : 6420 PRINTB(FMT12A,..DBL_VALUE,..DBL_VALUE);
12085 : 6421 !'GOOD DATA: XXXXXX AT LOCATION YYYYYY'
12086 : 6422 DBL_VALUE = .DBL_VALUE + BUFSIZ * 2;
12087 : 6423 PRINTB(FMT12B,..DBL_VALUE,..DBL_VALUE);
12088 : 6424 !'BAD DATA: PPPPPP AT LOCATION QQQQQQ'
12089 : 6425 WHY DROPT[.LUN] = CODE_8;
12090 : 6426 ERRDF(5304,MSG1,0); .**** OPTION 5, RAND3 ERROR 04 ****
12091 : 6427 DODU(.LUN);
12092 : 6428 LEAVE LOOP; .JUMP JUST BEYOND END OF BLOCK * 4 *
12093 : 6429 END;
12094 : 6430 END;

```

12096 : MLX4  
 12097 :  
 12098 :  
 12099 :  
 12100 :  
 12101 :  
 12102 :  
 12103 :  
 12104 :  
 12105 :  
 12106 :  
 12107 :  
 12108 :  
 12109 :  
 12110 :  
 12111 :  
 12112 :  
 12113 :  
 12114 :  
 12115 :  
 12116 :  
 12117 :  
 12118 :  
 12119 :  
 12120 :  
 12121 :  
 12122 :  
 12123 :  
 12124 :  
 12125 :  
 12126 :  
 12127 :  
 12128 :  
 12129 :  
 12130 :  
 12131 :  
 12132 :  
 12133 :  
 12134 :  
 12135 :  
 12136 :  
 12137 :  
 12138 :  
 12139 :  
 12140 :  
 12141 :  
 12142 :  
 12143 :  
 12144 :  
 12145 :  
 12146 :  
 12147 :  
 12148 :  
 12149 :  
 12150 :

TESTING RANDOM SECTORS, DATA, WORD COUNTS

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (50)

```

[1] :
  BEGIN !* 6D *          RETRY ALLOWED
  IF RETRY(SIX,.COMMAND,.LUN,.WRDCNT,.PTR,.SECTOR) NEQ 0
  THEN 'THE RETRY FAILED -- SYSTEM FATAL ERROR
      BEGIN
      WHY DROPT[.LUN] = CODE_4;
      ERRDF(5305,MSG1,0);  !**** OPTION 5, RAND3 ERROR 05 ****
      DODU(.LUN);
      LEAVE LOOP;          !JUMP JUST BEYOND END OF BLOCK * 4 *
      END;
  END; !* 6D *

[2] :
  BEGIN !* 6E *          FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
  WHY DROPT[.LUN] = CODE_5;
  ERRDF(5306,MSG1,0);  !**** OPTION 5, RAND3 ERROR 06 ****
  DODU(.LUN);
  LEAVE LOOP;          !JUMP JUST BEYOND END OF BLOCK * 4 *
  END;

[3] :
  BEGIN !* 6F *          FATAL DRIVE ERROR -- NO RETRY ALLOWED
  ERRDF(5307,MSG1,0);  !**** OPTION 5, RAND3 ERROR 07 ****
  WHY DROPT[.LUN] = CODE_6;
  DODU(.LUN);
  LEAVE LOOP;          !JUMP JUST BEYOND END OF BLOCK * 4 *
  END; !* 6F *

[4] :
  BEGIN !* 6G *          UNRECOVERABLE DATA ERROR
  ISOLATE();
  ERRDF(5308,MSG2,0);  !**** OPTION 5, RAND3 ERROR 08 ****
  WHY DROPT[.LUN] = CODE_7;
  DODU(.LUN);
  LEAVE LOOP;          !JUMP JUST BEYOND END OF BLOCK * 4 *
  END; !* 6G *

[5] :
  BEGIN !* 6H *          RECOVERABLE DATA ERROR
  ISOLATE();
  IF .ERROUT
  THEN PRINTB(FMT10B,.CHAN),
        !' BIT QQ'

  OLDSEC = .MLEL;
  OLDCHN = .CHAN;
  IF RETRY(ONE,CHECK,.LUN,.WRDCNT,.WPTR,.SECTOR) EQL 5
  THEN
    IF ((.MLEL EQL .OLDSEC) AND (.CHAN FQL .OLDCHN))
    THEN
      BEGIN
      IF .ERROUT
      THEN
        ERRHRD(5309,MSG4,0);  !**** OPTION 5, RAND3 ERROR 09 ****
        UP_HARD_COUNT(.LUN,.BOARD);
      END
    ELSE
  
```

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (50)

```

12152 :MLX4
12153 :
12154 :
12155 : 6483 BEGIN
12156 : 6484 IF .ERROUT
12157 : 6485 THEN
12158 : 6486 ERRSOFT(5310,MSG3,0); !**** OPTION 5, RAND3 ERROR 10 ****
12159 : 6487 UP_SOFT_COUNT(.LUN,.BOARD);
12160 : 6488 END
12161 : 6489 ELSE
12162 : 6490 BEGIN
12163 : 6491 IF .ERROUT
12164 : 6492 THEN
12165 : 6493 ERRSOFT(5311,MSG3,0); !**** OPTION 5, RAND3 ERROR 11 ****
12166 : 6494 UP_SOFT_COUNT(.LUN,.BOARD);
12167 : 6495 END;
12168 : 6496 END; !* 6H *
12169 : 6497
12170 : 6498 TES;
12171 : 6499
12172 : 6500 WPTR = .WPTR + 2;
12173 : 6501 END; !* 6 * END OF COUNTING LOOP FOR SECTOR SELECTION
12174 : 6502 END; !* 5 * END OF TEST FOR AN ACTIVE UNIT
12175 : 6503 END; !* 4 * END OF LOOP THAT COMPLETELY TESTS 1 UNIT
12176 : 6504 END; !* 3 * END OF LOGICAL UNIT SELECTION LOOP
12177 : 6505 END; !* 2 * END OF REPEAT LOOP FOR THIS ROUTINE
12178 : 6506 RETURN;
12179 : 6507 END; !* 1 * END OF ROUTINE
  
```

Address	Hex	Dec	Hex	Label	Instruction	Comment	Address
12183							
12184				.SBTTL	RAND3 TESTING RANDOM SECTORS, DATA, WORD COUNTS		
12188	066670	004137	005222	RAND3:	JSR R1,\$SAVE5		6277
12189	066674	162706	000024		SUB #24,SP		
12190	066700	012746	007322		MOV #RTNSC,-(SP)		6334
12191	066704	012746	006514		MOV #SAY1,-(SP)		
12192	066710	012746	000002		MOV #2,-(SP)		
12193	066714	010600			MOV SP,R0	: SP,*	
12194	066716	104414			TRAP 14		
12195	066720	005066	000030		CLR 30(SP)	: COUNT	6337
12196	066724	000137	070534		JMP 33\$		
12197	066730	013766	002012	1\$:	MOV L\$UNIT,12(SP)		6339
12198	066736	005004			CLR R4	: LUN	
12199	066740	000137	070522		JMP 32\$		
12200	066744	004737	036710	2\$:	JSR PC,GEN5		6341
12201	066750	010400			MOV R4,R0	: LUN,*	6344
12202	066752	006200			ASR R0		
12203	066754	006200			ASR R0		
12204	066756	006200			ASR R0		
12205	066760	062700	032460		ADD #DRIVE.STATUS,R0		
12206	066764	010046			MOV R0,-(SP)		





Address	Hex	Hex	Hex	Label	Instruction	Comments	Address	Address	Address
12264				:MLX4				23-Oct-1980	15:01:43
12265				:		TESTING RANDOM SECTORS, DATA, WORD COUNTS		23-Oct-1980	14:57:05
12266									TOPS
12267	067236	012746	042246		MOV	#WRITE,-(SP)			
12268	067242	010446			MOV	R4,-(SP)			
12269	067244	010346			MOV	R3,-(SP)			: LUN,*
12270	067246	013746	030706		MOV	WPTR,-(SP)			: WRDCNT,*
12271	067252	010146			MOV	R1,-(SP)			: SECTOR,*
12272	067254	004737	043046		JSR	PC,RETRY			
12273	067260	062706	000014		ADD	#14,SP			
12274	067264	005700			TST	R0			
12275	067266	001446			BEQ	9\$			
12276	067270	112764	000004	032464	MOVB	#4,WHY.DROPT(R4)			: *,*(LUN)
12277	067276	104455			TRAP	55			:
12278	067300	012265			.WORD	12265			
12279	067302	010512			.WORD	MSG1			
12280	067304	000000			.WORD	0			
12281	067306	010400			MOV	R4,R0			: LUN,*
12282	067310	104451			TRAP	51			
12283	067312	000431			BR	7\$			:
12284	067314	020527	000002	5\$:	CMP	R5,#2			: VALUE,*
12285	067320	001012			BNE	6\$			
12286	067322	112764	000005	032464	MOVB	#5,WHY.DROPT(R4)			: *,*(LUN)
12287	067330	104455			TRAP	55			:
12288	067332	012266			.WORD	12266			
12289	067334	010512			.WORD	MSG1			
12290	067336	000000			.WORD	0			
12291	067340	010400			MOV	R4,R0			: LUN,*
12292	067342	104451			TRAP	51			
12293	067344	000414			BR	7\$			:
12294	067346	020527	000003	6\$:	CMP	R5,#3			: VALUE,*
12295	067352	001014			BNE	9\$			
12296	067354	104455			TRAP	55			:
12297	067356	012267			.WORD	12267			
12298	067360	010512			.WORD	MSG1			
12299	067362	000000			.WORD	0			
12300	067364	112764	000006	032464	MOVB	#6,WHY.DROPT(R4)			: *,*(LUN)
12301	067372	010400			MOV	R4,R0			: LUN,*
12302	067374	104451			TRAP	51			
12303	067376	062706	000014	7\$:	ADD	#14,SP			:
12304	067402	000535		8\$:	BR	14\$			:
12305	067404	004737	043010	9\$:	JSR	PC,CHOOSE			:
12306	067410	010066	000040		MOV	R0,40(SP)			: *,COMMAND
12307	067414	005002			CLR	R2			:
12308	067416	020027	042434		CMP	R0,#READ			: COMMAND,*
12309	067422	001014			BNE	10\$			
12310	067424	005202			INC	R2			
12311	067426	013766	030710	000036	MOV	RPTR,36(SP)			: *,PTR
12312	067434	010446			MOV	R4,-(SP)			: LUN,*
12313	067436	010346			MOV	R3,-(SP)			: WRDCNT,*
12314	067440	013746	030710		MOV	RPTR,-(SP)			
12315	067444	010146			MOV	R1,-(SP)			: SECTOR,*
12316	067446	004737	042434		JSR	PC,READ			
12317	067452	000412			BR	11\$			
12318	067454	013766	030706	000036	MOV	WPTR,36(SP)			: *,PTR

Address	Hex	Hex	Hex	Hex	Code	Comment	Time	Page
12320					:MLX4		23-Oct-1980 15:01:43	TOPS
12321					:	TESTING RANDOM SECTORS, DATA, WORD COUNTS	23-Oct-1980 14:57:05	PA:<
12322								
12323	067462	010446			MOV	R4,-(SP) ; LUN,*		6403
12324	067464	010346			MOV	R3,-(SP) ; WRDCNT,*		
12325	067466	013746	030706		MOV	WPTR,-(SP)		
12326	067472	010146			MOV	R1,-(SP) ; SECTOR,*		
12327	067474	004737	042622		JSR	PC,CHECK		
12328	067500	010005		11\$:	MOV	R0,R5 ; *,VALUE		
12329	067502	001076			BNE	15\$		6408
12330	067504	006002			ROR	R2		6411
12331	067506	103402			BLO	13\$		
12332	067510	000137	070464	12\$:	JMP	28\$		
12333	067514	013746	030706	13\$:	MOV	WPTR,-(SP)		6414
12334	067520	013746	030710		MOV	RPTR,-(SP)		
12335	067524	010346			MOV	R3,-(SP) ; WRDCNT,*		
12336	067526	004737	041226		JSR	PC,DOUBLE.CHECK		
12337	067532	062706	000006		ADD	#6,SP		
12338	067536	010066	000052		MOV	R0,52(SP) ; *,DBL.VALUE		
12339	067542	001762			BEQ	12\$		
12340	067544	010446			MOV	R4,-(SP) ; LUN,*		6417
12341	067546	004737	033436		JSR	PC,SAYWHO		
12342	067552	012716	010640		MOV	#MSG5,(SP)		6418
12343	067556	012746	006514		MOV	#SAY1,-(SP)		
12344	067562	012746	000002		MOV	#2,-(SP)		
12345	067566	010600			MOV	SP,R0 ; SP,*		
12346	067570	104414			TRAP	14		
12347	067572	016616	000060		MOV	60(SP),(SP) ; DBL.VALUE,*		6420
12348	067576	017646	000060		MOV	@60(SP),-(SP) ; DBL.VALUE,*		
12349	067602	012746	006332		MOV	#FMT12A,-(SP)		
12350	067606	012746	070003		MOV	#3,-(SP)		
12351	067612	010600			MOV	SP,R0 ; SP,*		
12352	067614	104414			TRAP	14		
12353	067616	062766	010000	000066	ADD	#10000,66(SP) ; *,DBL.VALUE		6422
12354	067624	016616	000066		MOV	66(SP),(SP) ; DBL.VALUE,*		6423
12355	067630	017646	000066		MOV	@66(SP),-(SP) ; DBL.VALUE,*		
12356	067634	012746	006400		MOV	#FMT12B,-(SP)		
12357	067640	012746	000003		MOV	#3,-(SP)		
12358	067644	010600			MOV	SP,R0 ; SP,*		
12359	067646	104414			TRAP	14		
12360	067650	112764	000010	032464	MOVB	#10,WHY.DROPT(R4) ; *,*(LUN)		6425
12361	067656	104455			TRAP	55		6426
12362	067650	012270			.WORD	12270		
12363	067662	010512			.WORD	MSG1		
12364	067664	000000			.WORD	0		
12365	067666	010400			MOV	R4,R0 ; LUN,*		6427
12366	067670	104451			TRAP	51		
12367	067672	062706	000046		ADD	#46,SP		6428
12368	067676	000506		14\$:	BR	20\$		
12369	067700	020527	000001	15\$:	CMP	R5,#1 ; VALUE,*		6408
12370	067704	001031			BNE	16\$		
12371	067706	012746	000006		MOV	#6,-(SP)		6433
12372	067712	016646	000052		MOV	52(SP),-(SP) ; COMMAND,*		
12373	067716	010446			MOV	R4,-(SP) ; LUN,*		
12374	067720	010346			MOV	R3,-(SP) ; WRDCNT,*		

Address	Hex	Hex	Hex	Label	Instruction	Comments	Address
12376				:MLX4			23-Oct-1980 15:01:43 TOPS
12377				:	TESTING RANDOM SECTORS, DATA, WORD COUNTS		23-Oct-1980 14:57:05 PA:<
12378							
12379	067722	016646	000056		MOV 56(SP),-(SP)	: PTR,*	
12380	067726	010146			MOV R1,-(SP)	: SECTGR,*	
12381	067730	004737	043046		JSR PC,RETRY		
12382	067734	062706	000014		ADD #14,SP		
12383	067740	005700			TST R0		
12384	067742	001662			BEQ 12\$		
12385	067744	112764	000004 032464		MOVB #4,WHY.DROPT(R4)	: *,*(LUN)	6436
12386	067752	104455			TRAP 55	:	6437
12387	067754	012271			.WORD 12271		
12388	067756	010512			.WORD MSG1		
12389	067760	000000			.WORD 0		
12390	067762	010400			MOV R4,R0	: LUN,*	6438
12391	067764	104451			TRAP 51		
12392	067766	000450			BR 19\$	:	6439
12393	067770	020527	000002 16\$:		CMP R5,#2	: VALUE,*	6408
12394	067774	001012			BNE 17\$		
12395	067776	112764	000005 032464		MOVB #5,WHY.DROPT(R4)	: *,*(LUN)	6444
12396	070004	104455			TRAP 55	:	6445
12397	070006	012272			.WORD 12272		
12398	070010	010512			.WORD MSG1		
12399	070012	000000			.WORD 0		
12400	070014	010400			MOV R4,R0	: LUN,*	6446
12401	070016	104451			TRAP 51		
12402	070020	000433			BR 19\$	:	6447
12403	070022	020527	000003 17\$:		CMP R5,#3	: VALUE,*	6408
12404	070026	001012			BNE 18\$		
12405	070030	104455			TRAP 55	:	6451
12406	070032	012273			.WORD 12273		
12407	070034	010512			.WORD MSG1		
12408	070036	000000			.WORD 0		
12409	070040	112764	000006 032464		MOVB #6,WHY.DROPT(R4)	: *,*(LUN)	6452
12410	070046	010400			MOV R4,R0	: LUN,*	6453
12411	070050	104451			TRAP 51		
12412	070052	000416			BR 19\$	:	6454
12413	070054	020527	000004 18\$:		CMP R5,#4	: VALUE,*	6408
12414	070060	001017			BNE 21\$		
12415	070062	004737	035212		JSR PC,ISOLATE	:	6458
12416	070066	104455			TRAP 55	:	6459
12417	070070	012274			.WORD 12274		
12418	070072	010542			.WORD MSG2		
12419	070074	000000			.WORD 0		
12420	070076	112764	000007 032464		MOVB #7,WHY.DROPT(R4)	: *,*(LUN)	6460
12421	070104	010400			MOV R4,R0	: LUN,*	6461
12422	070106	104451			TRAP 51		
12423	070110	062706	000024 19\$:		ADD #24,SP	:	6462
12424	070114	000137	070520 20\$:		JMP 31\$		
12425	070120	020527	000005 21\$:		CMP R5,#5	: VALUE,*	6408
12426	070124	001157			BNE 28\$		
12427	070126	004737	035212		JSR PC,ISOLATE	:	6466
12428	070132	032737	000001 002260		BIT #1,ERROUT	:	6467
12429	070140	001423			BEQ 22\$	:	
12430	070142	017702	142260		MOV @ML.REG+42,R2	:	6463

23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<

```

12432           ;MLX4
12433           ;
12434           ;
12435 070146    006202    ASR      R2
12436 070150    006202    ASR      R2
12437 070152    006202    ASR      R2
12438 070154    006202    ASR      R2
12439 070156    006202    ASR      R2
12440 070160    006202    ASR      R2
12441 070162    042702    177700   BIC      #177700,R2
12442 070166    010246    MOV      R2,-(SP)
12443 070170    012746    006262   MOV      #FMT10B,-(SP)
12444 070174    012746    000002   MOV      #2,-(SP)
12445 070200    010600           MOV      SP,R0                ; SP,*
12446 070202    104414           TRAP     14
12447 070204    062706    000006   ADD      #6,SP
12448 070210    017766    142214   000042   22$:   MOV      @ML.REG+44,42(SP)      ; *,OLDSEC             6470
12449 070216    017702    142204           MOV      @ML.REG+42,R2        ;                               6471
12450 070222    006202    ASR      R2
12451 070224    006202    ASR      R2
12452 070226    006202    ASR      R2
12453 070230    006202    ASR      R2
12454 070232    006202    ASR      R2
12455 070234    006202    ASR      R2
12456 070236    042702    177700   BIC      #177700,R2
12457 070242    010266    000044   MOV      R2,44(SP)           ; *,OLDCHN
12458 070246    012746    000001   MOV      #1,-(SP)           ;                               6472
12459 070252    012746    042622   MOV      #CHECK,-(SP)
12460 070256    010446           MOV      R4,-(SP)           ; LUN,*
12461 070260    010346           MOV      R3,-(SP)           ; WRDCNT,*
12462 070262    013746    030706   MOV      WPTR,-(SP)
12463 070266    010146           MOV      R1,-(SP)           ; SECTOR,*
12464 070270    004737    043046   JSR      PC,RETRY
12465 070274    062706    000014   ADD      #14,SP
12466 070300    020027    000005   CMP      R0,#5
12467 070304    001051           BNE      25$
12468 070306    027766    142116   000042   CMP      @ML.REG+44,42(SP)      ; *,OLDSEC             6474
12469 070314    001034           BNE      24$
12470 070316    016600    000044   MOV      44(SP),R0           ; OLDCHN,*
12471 070322    017702    142100   MOV      @ML.REG+42,R2
12472 070326    006202    ASR      R2
12473 070330    006202    ASR      R2
12474 070332    006202    ASR      R2
12475 070334    006202    ASR      R2
12476 070336    006202    ASR      R2
12477 070340    006202    ASR      R2
12478 070342    042702    177700   BIC      #177700,R2
12479 070346    020200           CMP      R2,R0
12480 070350    001016           BNE      24$
12481 070352    032737    000001   002260   BIT      #1,ERROUT           ;                               6477
12482 070360    001404           BEQ      23$
12483 070362    104456           TRAP     56                   ;                               6479
12484 070364    012275           .WORD    12275
12485 070366    010624           .WORD    MSG4
12486 070370    000000           .WORD    0

```

```

12488 ;MLX4
12489 ;
12490 ;
12491 070372 010446 23$: MOV R4,-(SP) ; LUN,* 6480
12492 070374 013746 032360 MOV BOARD,-(SP)
12493 070400 004737 035302 JSR PC,UP.HARD.COUNT
12494 070404 000426 BR 27$ ; 6474
12495 070406 032737 000001 002260 24$: BIT #1,ERROUT ; 6484
12496 070414 001415 BEQ 26$ ;
12497 070416 104457 TRAP 57 ; 6486
12498 070420 012276 .WORD 12276
12499 070422 010610 .WORD MSG3
12500 070424 000000 .WORD 0
12501 070426 000410 BR 26$ ; 6487
12502 070430 032737 000001 002260 25$: BIT #1,ERROUT ; 6491
12503 070436 001404 BEQ 26$ ;
12504 070440 104457 TRAP 57 ; 6493
12505 070442 012277 .WORD 12277
12506 070444 010610 .WORD MSG3
12507 070446 000000 .WORD 0
12508 070450 010446 26$: MOV R4,-(SP) ; LUN,* 6494
12509 070452 013746 032360 MOV BOARD,-(SP)
12510 070456 004737 035430 JSR PC,UP.SOFT.COUNT
12511 070462 022626 27$: CMP (SP)+,(SP)+ ; 6465
12512 070464 062737 000002 030706 28$: ADD #2,WPTR ; 6500
12513 070472 062706 000020 ADD #20,SP ; 6352
12514 070476 005266 000012 29$: INC 12(SP) ; KOUNT 6351
12515 070502 026666 000012 000020 CMP 12(SP),20(SP) ; KOUNT,TIMES
12516 070510 003002 BGT 30$
12517 070512 000137 067114 JMP 3$
12518 070516 022626 30$: CMP (SP)+,(SP)+ ; 6346
12519 070520 005204 31$: INC R4 ; LUN 6339
12520 070522 020466 000012 32$: CMP R4,12(SP) ; LUN,*
12521 070526 002002 BGE 33$
12522 070530 000137 066744 JMP 2$
12523 070534 005266 000030 33$: INC 30(SP) ; COUNT 6337
12524 070540 026666 000030 000050 CMP 30(SP),50(SP) ; COUNT,REPEAT
12525 070546 003002 BGT 34$
12526 070550 000137 066730 JMP 1$
12527 070554 062706 000032 34$: ADD #32,SP ; 6277
12528 070560 000207 RTS PC

```

```

; Routine Size: 477 words
; Maximum stack depth per invocation: 38 words

```

```

12529
12530
12531
12536
12537

```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (51)

```

12539 :MLX4
12540 :
12541 :
12542 : 6508 %SBTTL 'TESTING RANDOM UNITS, SECTORS, DATA, WORD COUNTS'
12543 : 6509
12544 : 6510 ROUTINE RAND4(REPEAT): NOVALUE =
12545 : 6511 BEGIN !* 1 * START OF ROUTINE
12546 : 6512
12547 : 6513 !**
12548 : 6514 ROUTINE: RAND4(REPEAT)
12549 : 6515
12550 : 6516 PURPOSE: TO CHECK RANDOM UNITS
12551 : 6517
12552 : 6518 ARGUMENT: REPEAT = NUMBER OF TIMES TO EXECUTE ENTIRE ROUTINE
12553 : 6519
12554 : 6520 THE CODE FOR 'RAND4' IN BRIEF:
12555 : 6521
12556 : 6522 BEGIN 1 (START OF ROUTINE)
12557 : 6523 SAY ROUTINE IS RUNNING
12558 : 6524 INCR COUNT FROM 1 TO REPEAT
12559 : 6525 : BEGIN 2 (START OF REPEAT LOOP FOR THIS ROUTINE)
12560 : 6526 : GENERATE THE RANDOM PATTERN
12561 : 6527 : TIMES = NUMBER OF UNITS * 4
12562 : 6528 : INCR KOUNT FROM 1 TO TIMES
12563 : 6529 : : BEGIN 3 (START OF COUNTING LOOP FOR UNIT SELECTION)
12564 : 6530 : : CHOOSE A RANDOM LOGICAL UNIT WHICH IS ACTIVE
12565 : 6531 : : INITIALIZE BUFFER POINTERS
12566 : 6532 : : TESTLOOP:
12567 : 6533 : : : BEGIN 4 (START OF LOOP THAT COMPLETELY TESTS 1 UNIT)
12568 : 6534 : : : INCR KOUNT2 FROM 1 TO 10
12569 : 6535 : : : : BEGIN 5 (START OF COUNTING LOOP FOR SECTOR SELECTION)
12570 : 6536 : : : : CHOOSE A RANDOM WORD COUNT
12571 : 6537 : : : : SET UP BUFFER POINTERS BEFORE TRANSFER
12572 : 6538 : : : : CHOOSE A RANDOM SECTOR
12573 : 6539 : : : : CALCULATE WHERE TRANSFER WILL END (BASED ON WORD COUNT)
12574 : 6540 : : : : IF CALCULATED VALUE GTR HIGHEST
12575 : 6541 : : : : : THEN ADJUST THE WORD COUNT SO IT FITS
12576 : 6542 : : : : : : WITHIN THE TESTABLE SECTOR LIMITS
12577 : 6543 : : : : : WRITE
12578 : 6544 : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
12579 : 6545 : : : : : CHOOSE WHETHER TO WRITE CHECK OR READ
12580 : 6546 : : : : : DO THE WRITE CHECK OR READ
12581 : 6547 : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
12582 : 6548 : : : : : CHANGE BUFFER POINTERS AFTER TRANSFER
12583 : 6549 : : : : : END 5 (END OF COUNTING LOOP FOR SECTOR SELECTION)
12584 : 6550 : : : : : END 4 (END OF TESTLOOP)
12585 : 6551 : : : : : END 3 (END OF COUNTING LOOP FOR UNIT SELECTION)
12586 : 6552 : : : : : END 2 (END OF REPEAT LOOP FOR THIS ROUTINE)
12587 : 6553 : : : : : RETURN
12588 : 6554 : : : : : END 1 (END OF ROUTINE)
12589 : 6555 : : : : : --
12590 : 6556
12591 : 6557 LOCAL
12592 : 6558 LUN, WRDCNT, SECTOR, VALUE, OLDSEC, OLDCHN,
12593 : 6559 PTR, COMMAND, DBL_VALUE;
  
```

```

12595 :MLX4
12596 :
12597 :
12598 : 6560
12599 : 6561 LABEL
12600 : 6562 LOOP;
12601 : 6563
12602 : 6564 PRINTB(SAY1,RTNSD);
12603 : 6565 !'RAND4'
12604 : 6566 INCR COUNT FROM 1 TO .REPEAT DO
12605 : 6567 BEGIN !* 2 * START OF REPEAT LOOP FOR THIS ROUTINE
12606 : 6568 GEN5();
12607 : 6569 INCR COUNT FROM 1 TO (.NUM_DRIVES * 4) DO
12608 : 6570 BEGIN !* 3 * START OF COUNTING LOOP FOR UNIT SELECTION
12609 : 6571 LUN = RNDU();
12610 : 6572 L$LUN = .LUN;
12611 : 6573 WPTR = WBUFF;
12612 : 6574 RPTR = RBUFF;
12613 : 6575 LOOP:
12614 : 6576 BEGIN !* 4 * START OF LOOP THAT COMPLETELY TESTS 1 UNIT
12615 : 6577 INCR COUNT2 FROM 1 TO 10 DO
12616 : 6578 BEGIN !* 5 * START OF COUNTING LOOP FOR SECTOR SELECTION
12617 : 6579 WRDCNT = RNDWC(); !EXPECT VALUE BETWEEN 1 AND BUFSIZ
12618 : 6580 SET PTRS(.WRDCNT);
12619 : 6581 SECTOR = RNDSEC(.LUN);!EXPECT VALUE BETWEEN LOWEST AND HIGHEST
12620 : 6582 IF (.SECTOR + .WRDCNT/256) GTRA HIGHEST
12621 : 6583 THEN WRDCNT = 256 * (HIGHEST - .SECTOR + 1);
12622 : 6584
12623 : 6585 VALUE = WRITE(.LUN,.WRDCNT,.WPTR,.SECTOR);
12624 : 6586 !*
12625 : 6587 ! SEE HOW SUCCESSFUL THE WRITE WAS:
12626 : 6588 !-
12627 : 6589 SELECTONE .VALUE OF !SEE 'SYSERR' FOR DEFINITION
12628 : 6590 SET !OF ERROR # CONTAINED IN 'VALUE'
12629 : 6591 [1] :
12630 : 6592 BEGIN !* 5A * RETRY ALLOWED
12631 : 6593 IF RETRY(SIX,WRITE,.LUN,.WRDCNT,.WPTR,.SECTOR) NEQ 0
12632 : 6594 THEN !THE RETRY FAILED -- SYSTEM FATAL ERROR
12633 : 6595 BEGIN
12634 : 6596 WHY DROPT[.LUN] = CODE 4;
12635 : 6597 ERRDF(5401,MSG1,0); !**** OPTION 5, RAND4 ERROR 01 ****
12636 : 6598 DODU(.LUN);
12637 : 6599 LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
12638 : 6600 END;
12639 : 6601 END; !* 5A *
12640 : 6602 [2] :
12641 : 6603 BEGIN !* 5B * FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
12642 : 6604 WHY DROPT[.LUN] = CODE 5;
12643 : 6605 ERRDF(5402,MSG1,0); !**** OPTION 5, RAND4 ERROR 02 ****
12644 : 6606 DODU(.LUN);
12645 : 6607 LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
12646 : 6608 END; !* 5B *
12647 : 6609 [3] :
12648 : 6610 BEGIN !* 5C * FATAL DRIVE ERROR -- NO RETRY ALLOWED
12649 : 6611 ERRDF(5403,MSG1,0); !**** OPTION 5, RAND4 ERROR 03 ****

```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (51)



```

12651 :MLX4
12652 :
12653 :
12654 : 6612 WHY DROPT[.LUN] = CODE_6;
12655 : 6613 DODU(.LUN);
12656 : 6614 LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
12657 : 6615 END; !* 5C *
12658 : 6616
12659 : 6617 TES;
12660 : 6618
12661 : 6619 COMMAND = CHOOSE();
12662 : 6620 IF .COMMAND EQL READ
12663 : 6621 THEN
12664 : 6622 BEGIN
12665 : 6623 PTR = .RPTR;
12666 : 6624 VALUE = READ(.LUN,.WRDCNT,.RPTR,.SECTOR);
12667 : 6625 END
12668 : 6626 ELSE
12669 : 6627 BEGIN
12670 : 6628 PTR = .WPTR;
12671 : 6629 VALUE = CHECK(.LUN,.WRDCNT,.WPTR,.SECTOR);
12672 : 6630 END;
12673 : 6631 !+
12674 : 6632 ! SEE HOW SUCCESSFUL THE OPERATION WAS:
12675 : 6633 !-
12676 : 6634 SELECTONE .VALUE OF !SEE 'SYSERR' FOR DEFINITION
12677 : 6635 SET !OF ERROR # CONTAINED IN 'VALUE'
12678 : 6636 [0] :
12679 : 6637 IF .COMMAND EQL READ
12680 : 6638 THEN
12681 : 6639 BEGIN
12682 : 6640 IF (DBL_VALUE = DOUBLE_CHECK(.WPTR,.RPTR,.WRDCNT)) NEQ 0
12683 : 6641 THEN
12684 : 6642 BEGIN
12685 : 6643 SAYWHO(.LUN);
12686 : 6644 PRINTB(SAY1,MSG5);
12687 : 6645 !'ECC LOGIC FAILED TO DETECT DATA ERROR'
12688 : 6646 PRINTB(FMT12A,..DBL_VALUE,..DBL_VALUE);
12689 : 6647 !'GOOD DATA: XXXXXX AT LOCATION YYYYYY'
12690 : 6648 DBL_VALUE = .DBL_VALUE + BUFSIZ * 2;
12691 : 6649 PRINTB(FMT12B,..DBL_VALUE,..DBL_VALUE);
12692 : 6650 !'BAD DATA: PPPPP AT LOCATION QQQQQQ'
12693 : 6651 WHY DROPT[.LUN] = CODE 8;
12694 : 6652 ERRDF(5404,MSG1,0); !**** OPTION 5, RAND4 ERROR 04 ****
12695 : 6653 DODU(.LUN);
12696 : 6654 LEAVE LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
12697 : 6655 END;
12698 : 6656 END;
12699 : 6657 [1] :
12700 : 6658 BEGIN !* 5D * RETRY ALLOWED
12701 : 6659 IF RETRY(SIX,.COMMAND,.LUN,.WRDCNT,.PTR,.SECTOR) NEQ 0
12702 : 6660 THEN !THE RETRY FAILED -- SYSTEM FATAL ERROR
12703 : 6661 BEGIN
12704 : 6662 WHY DROPT[.LUN] = CODE 4;
12705 : 6663 ERRDF(5405,MSG1,0); !**** OPTION 5, RAND4 ERROR 05 ****

```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (51)

```

12707 :MLX4
12708 :
12709 :
12710 : 6664          DODU(.LUN);
12711 : 6665          LEAVE LOOP;          !JUMP JUST BEYOND END OF BLOCK * 4 *
12712 : 6666          END;
12713 : 6667          END;          !* 5D *
12714 : 6668          [2] :
12715 : 6669          BEGIN !* 5E *  FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
12716 : 6670          WHY DROPT[.LUN] = CODE_5;
12717 : 6671          ERRDF(5406,MSG1,0);  !**** OPTION 5, RAND4 ERROR 06 ****
12718 : 6672          DODU(.LUN);
12719 : 6673          LEAVE LOOP;          .JUMP JUST BEYOND END OF BLOCK * 4 *
12720 : 6674          END;          !* 5E *
12721 : 6675          [3] :
12722 : 6676          BEGIN !* 5F *  FATAL DRIVE ERROR -- NO RETRY ALLOWED
12723 : 6677          ERRDF(5407,MSG1,0);  !**** OPTION 5, RAND4 ERROR 07 ****
12724 : 6678          WHY DROPT[.LUN] = CODE_6;
12725 : 6679          DODU(.LUN);
12726 : 6680          LEAVE LOOP;          !JUMP JUST BEYOND END OF BLOCK * 4 *
12727 : 6681          END;          !* 5F *
12728 : 6682          [4] :
12729 : 6683          BEGIN !* 5G *  UNRECOVERABLE DATA ERROR
12730 : 6684          ISOLATE();
12731 : 6685          ERRDF(5408,MSG2,0);  !**** OPTION 5, RAND4 ERROR 08 ****
12732 : 6686          WHY DROPT[.LUN] = CODE_7;
12733 : 6687          DODU(.LUN);
12734 : 6688          LEAVE LOOP;          !JUMP JUST BEYOND END OF BLOCK * 4 *
12735 : 6689          END;          !* 5G *
12736 : 6690          [5] :
12737 : 6691          BEGIN !* 5H *  RECOVERABLE DATA ERROR
12738 : 6692          ISOLATE();
12739 : 6693          IF .ERROUT
12740 : 6694          THEN PRINTB(FMT10B,.CHAN);
12741 : 6695          !' BIT 00'
12742 : 6696          OLDSEC = .MLEL;
12743 : 6697          OLDCHN = .CHAN;
12744 : 6698          IF RETRY(ONE,.COMMAND,.LUN,.WRDCNT,.PTR,.SECTOR) EQL 5
12745 : 6699          THEN
12746 : 6700          IF ((.MLEL EQL .OLDSEC) AND (.CHAN EQL .OLDCHN))
12747 : 6701          THEN
12748 : 6702          BEGIN
12749 : 6703          IF .ERROUT
12750 : 6704          THEN
12751 : 6705          ERRHRD(5409,MSG4,0);  !**** OPTION 5, RAND4 ERROR 09 ****
12752 : 6706          UP_HARD_COUNT(.LUN,.BOARD);
12753 : 6707          END
12754 : 6708          ELSE
12755 : 6709          BEGIN
12756 : 6710          IF .ERROUT
12757 : 6711          THEN
12758 : 6712          ERRSOFT(5410,MSG3,0); !**** OPTION 5, RAND4 ERROR 10 ****
12759 : 6713          UP_SOFT_COUNT(.LUN,.BOARD);
12760 : 6714          END
12761 : 6715          ELSE

```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (51)

12763 :MLX4  
 12764 : TESTING RANDOM UNITS, SECTORS, DATA, WORD COUNT  
 12765 :  
 23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (51)

```

12766 :      6716          BEGIN
12767 :      6717          IF .ERROUT
12768 :      6718          THEN
12769 :      6719          ERRSOFT(5411,MSG3,0); !**** OPTION 5, RAND4 ERROR 11 ****
12770 :      6720          UP_SOFT_COUNT(.LUN,.BOARD);
12771 :      6721          END;
12772 :      6722          END; !* 5H *
12773 :      6723
12774 :      6724          TES;
12775 :      6725
12776 :      6726          WPTR = .WPTR + 2;
12777 :      6727          END; !* 5 * END OF COUNTING LOOP FOR SECTOR SELECTION
12778 :      6728          END; !* 4 * END OF LOOP THAT COMPLETELY TESTS 1 UNIT
12779 :      6729          END; !* 3 * END OF COUNTING LOOP FOR UNIT SELECTION
12780 :      6730          END; !* 2 * END OF REPEAT LOOP FOR THIS ROUTINE
12781 :      6731          RETURN;
12782 :      6732          END; !* 1 * END OF ROUTINE
  
```

Address	Hex	Hex	Hex	Label	Instruction	Comment	Address
12787				.SBTTL	RAND4 TESTING RANDOM UNITS, SECTORS, DATA, WORD COUNT		
12791	070562	004137	005222	RAND4:	JSR R1,\$SAVE5		6510
12792	070566	162706	000024		SUB #24,SP		
12793	070572	012746	007330		MOV #RTN5D,-(SP)		6564
12794	070576	012746	006514		MOV #SAY1,-(SP)		
12795	070602	012746	000002		MOV #2,-(SP)		
12796	070606	010600			MOV SP,R0	: SP,*	
12797	070610	104414			TRAP 14		
12798	070612	005066	000030		LLR 30(SP)	: COUNT	6566
12799	070616	000137	072356		JMP 30\$		
12800	070622	004737	036710	1\$:	JSR PC,GEN5		6568
12801	070626	013766	032474	000014	MOV NUM.DRIVES,14(SP)		6569
12802	070634	006366	000014		ASL 14(SP)		
12803	070640	006366	000014		ASL 14(SP)		
12804	070644	005066	000012		CLR 12(SP)	: KOUNT	
12805	070650	000560			BR 8\$		
12806	070652	004737	062714	2\$:	JSR PC,RNDU		6571
12807	070656	010001			MOV R0,R1	: *,LUN	
12808	070660	010137	002074		MOV R1,L\$LUN	: LUN,*	6572
12809	070664	012737	010706	030706	MOV #WBUFF,WPTR		6573
12810	070672	012737	020706	030710	MOV #RBUFF,RPTR		6574
12811	070700	010102			MOV R1,R2	: LUN,*	6582
12812	070702	006302			ASL R2		
12813	070704	012766	000001	000010	MOV #1,10(SP)	: *,KOUNT2	6577
12814	070712	004737	062560	3\$:	JSR PC,RNDWC		6579
12815	070716	010004			MOV R0,R4	: *,WRDCNT	
12816	070720	010446			MOV R4,-(SP)	: WRDCNT,*	6580
12817	070722	004737	043770		JSR PC,SET.PTRS		

Address	Hex	Hex	Hex	Instruction	Comments	Seq
12819						
12820						
12821						
12822	070726	010116		MOV R1,(SP)	: LUN,*	6581
12823	070730	004737	062614	JSR PC,RNDSEC		
12824	070734	010066	000010	MOV R0,10(SP)	: *,SECTOR	
12825	070740	010446		MOV R4,-(SP)	: WRDCNT,*	6582
12826	070742	012746	000400	MOV #400,-(SP)		
12827	070746	004737	005056	JSR PC,BLSDIV		
12828	070752	022626		CMP (SP)+,(SP)+		
12829	070754	066600	000010	ADD 10(SP),R0	: SECTOR,*	
12830	070760	020062	032516	CMP R0,TOP.SECT(R2)		
12831	070764	101411		BLOS 4\$		
12832	070766	016200	032516	MOV TOP.SECT(R2),R0	: SECTOR,*	6583
12833	070772	166600	000010	SUB 10(SP),R0	: SECTOR,*	
12834	070776	000300		SWAB R0		6582
12835	071000	105000		CLRB R0		
12836	071002	010004		MOV R0,R4	: *,WRDCNT	6583
12837	071004	062704	000400	ADD #400,R4	: *,WRDCNT	
12838	071010	010116		MOV R1,(SP)	: LUN,*	6585
12839	071012	010446		MOV R4,-(SP)	: WRDCNT,*	
12840	071014	013746	030706	MOV WPTR,-(SP)		
12841	071020	016646	000014	MOV 14(SP),-(SP)	: SECTOR,*	
12842	071024	004737	042246	JSR PC,WRITE		
12843	071030	010005		MOV R0,R5	: *,VALUE	
12844	071032	020527	000001	CMP R5,#1	: VALUE,*	6589
12845	071036	001032		BNE 5\$		
12846	071040	012746	000006	MOV #6,-(SP)		6593
12847	071044	012746	042246	MOV #WRITE,-(SP)		
12848	071050	010146		MOV R1,-(SP)	: LUN,*	
12849	071052	010446		MOV R4,-(SP)	: WRDCNT,*	
12850	071054	013746	030706	MOV WPTR,-(SP)		
12851	071060	016646	000030	MOV 30(SP),-(SP)	: SECTOR,*	
12852	071064	004737	043046	JSR PC,RETRY		
12853	071070	062706	000014	ADD #14,SP		
12854	071074	005700		TST R0		
12855	071076	001446		BEQ 9\$		
12856	071100	112761	000004 032464	MOVB #4,WHY.DROPT(R1)	: *,*(LUN)	6596
12857	071106	104455		TRAP 55		6597
12858	071110	012431		.WORD 12431		
12859	071112	010512		.WORD MSG1		
12860	071114	000000		.WORD 0		
12861	071116	010100		MOV R1,R0	: LUN,*	6598
12862	071120	104451		TRAP 51		
12863	071122	000431		BR 7\$		6599
12864	071124	020527	000002	CMP R5,#2	: VALUE,*	6589
12865	071130	001012		BNE 6\$		
12866	071132	112761	000005 032464	MOVB #5,WHY.DROPT(R1)	: *,*(LUN)	6604
12867	071140	104455		TRAP 55		6605
12868	071142	012432		.WORD 12432		
12869	071144	010512		.WORD MSG1		
12870	071146	000000		.WORD 0		
12871	071150	010100		MOV R1,R0	: LUN,*	6606
12872	071152	104451		TRAP 51		
12873	071154	000414		BR 7\$		6607

Address	OpCode	Operand 1	Operand 2	Operand 3	Label	Instruction	Comments	Line No.
12875					:MLX4			
12876					:			
12877					:	TESTING RANDOM UNITS, SECTORS, DATA, WORD COUNT	23-Oct-1980 15:01:43	TOPS
12878	071156	020527	000003	6\$:		CMP R5,#3		6589
12879	071162	001014				BNE 9\$		
12880	071164	104455				TRAP 55		6611
12881	071166	012433				.WORD 12433		
12882	071170	010512				.WORD MSG1		
12883	071172	000000				.WORD 0		
12884	071174	112761	000006 032464			MOVB #6,WHY.DROPT(R1)	: *,*(LUN)	6612
12885	071202	010100				MOV R1,R0	: LUN,*	6613
12886	071204	104451				TRAP 51		
12887	071206	062706	000010	7\$:		ADD #10,SP		6614
12888	071212	000537		8\$:		BR 14\$		
12889	071214	004737	043010	9\$:		JSR PC,CHOOSE		6619
12890	071220	010066	000034			MOV R0,34(SP)	: *,COMMAND	
12891	071224	005003				CLR R3		6620
12892	071226	020027	042434			CMP R0,#READ	: COMMAND,*	
12893	071232	001015				BNE 10\$		
12894	071234	005203				INC R3		
12895	071236	013766	030710 000032			MOV RPTR,32(SP)	: *,PTR	6623
12896	071244	010146				MOV R1,-(SP)	: LUN,*	6624
12897	071246	010446				MOV R4,-(SP)	: WRDCNT,*	
12898	071250	013746	030710			MOV RPTR,-(SP)		
12899	071254	016646	000024			MOV 24(SP),-(SP)	: SECTOR,*	
12900	071260	004737	042434			JSR PC,READ		
12901	071264	000413				BR 11\$		
12902	071266	013766	030706 000032	10\$:		MOV WPTR,32(SP)	: *,PTR	6628
12903	071274	010146				MOV R1,-(SP)	: LUN,*	6629
12904	071276	010446				MOV R4,-(SP)	: WRDCNT,*	
12905	071300	013746	030706			MOV WPTR,-(SP)		
12906	071304	016646	000024			MOV 24(SP),-(SP)	: SECTOR,*	
12907	071310	004737	042622			JSR PC,CHECK		
12908	071314	010005		11\$:		MOV R0,R5	: *,VALUE	
12909	071316	001076				BNE 15\$		6634
12910	071320	006003				ROR R3		6637
12911	071322	103402				BLO 13\$		
12912	071324	000137	072304	12\$:		JMP 28\$		
12913	071330	013746	030706	13\$:		MOV WPTR,-(SP)		6640
12914	071334	013746	030710			MOV RPTR,-(SP)		
12915	071340	010446				MOV R4,-(SP)	: WRDCNT,*	
12916	071342	004737	041226			JSR PC,DOUBLE.CHECK		
12917	071346	062706	000006			ADD #6,SP		
12918	071352	010066	000046			MOV R0,46(SP)	: *,DBL.VALUE	
12919	071356	001762				BEQ 12\$		
12920	071360	010146				MOV R1,-(SP)	: LUN,*	6643
12921	071362	004737	033436			JSR PC,SAYWHO		
12922	071366	012716	010640			MOV #MSG5,(SP)		6644
12923	071372	012746	006514			MOV #SAY1,-(SP)		
12924	071376	012746	000002			MOV #2,-(SP)		
12925	071402	010600				MOV SP,R0	: SP,*	
12926	071404	104414				TRAP 14		
12927	071406	016616	000054			MOV 54(SP),(SP)	: DBL.VALUE,*	6646
12928	071412	017646	000054			MOV 54(SP),-(SP)	: DBL.VALUE,*	
12929	071416	012746	006332			MOV #FMT12A,-(SP)		

MLX4 MACRO M1113 24-OCT-80 09:59 PAGE 337  
 RAND4 TESTING RANDOM UNITS, SECTORS, DATA, WORD COUNT

SEQ 0292

		MLX4	TESTING RANDOM UNITS, SECTORS, DATA, WORD COUNT	23-Oct-1980 15:01:43	TOPS
		:		23-Oct-1980 14:57:05	PA:<
12931		:			
12932		:			
12933		:			
12934	071422	012746	000003		
12935	071426	010600			
12936	071430	104414			
12937	071432	062766	010000 000062		6648
12938	071440	016616	000062		6649
12939	071444	017646	000062		
12940	071450	012746	006400		
12941	071454	012746	000003		
12942	071460	010600			
12943	071462	104414			
12944	071464	112761	000010 032464		6651
12945	071472	104455			6652
12946	071474	012434			
12947	071476	010512			
12948	071500	000000			
12949	071502	010100			6653
12950	071504	104451			
12951	071506	062706	000042		6654
12952	071512	000507	14\$:		
12953	071514	020527	000001 15\$:		6634
12954	071520	001032			
12955	071522	012746	000006		6659
12956	071526	016646	000046		
12957	071532	010146			
12958	071534	010446			
12959	071536	016646	000052		
12960	071542	016646	000040		
12961	071546	004737	043046		
12962	071552	062706	000014		
12963	071556	005700			
12964	071560	001661			
12965	071562	112761	000004 032464		6662
12966	071570	104455			6663
12967	071572	012435			
12968	071574	010512			
12969	071576	000000			
12970	071600	010100			6664
12971	071602	104451			
12972	071604	000450			6665
12973	071606	020527	000002 16\$:		6634
12974	071612	001012			
12975	071614	112761	000005 032464		6670
12976	071622	104455			6671
12977	071624	012436			
12978	071626	010512			
12979	071630	000000			
12980	071632	010100			6672
12981	071634	104451			
12982	071636	000433			6673
12983	071640	020527	000003 17\$:		6634
12984	071644	001012			
12985	071646	104455			6677

Address	Word	Value	Label	Instruction	Comment	Line
12987			:MLX4			
12988			:			
12989			:			
12990	071650	012437		.WORD	12437	
12991	071652	010512		.WORD	MSG1	
12992	071654	000000		.WORD	0	
12993	071656	112761	000006 032464	MOVB	#6,WHY.DROPT(R1)	: *,*(LUN) 6678
12994	071664	010100		MOV	R1,R0	: LUN,* 6679
12995	071666	104451		TRAP	51	
12996	071670	000416		BR	19\$	: 6680
12997	071672	020527	000004 18\$:	CMP	R5,#4	: VALUE,* 6634
12998	071676	001017		BNE	21\$	
12999	071700	004737	035212	JSR	PC,ISOLATE	: 6684
13000	071704	104455		TRAP	55	: 6685
13001	071706	012440		.WORD	12440	
13002	071710	010542		.WORD	MSG2	
13003	071712	000000		.WORD	0	
13004	071714	112761	000007 032464	MOVB	#7,WHY.DROPT(R1)	: *,*(LUN) 6686
13005	071722	010100		MOV	R1,R0	: LUN,* 6687
13006	071724	104451		TRAP	51	
13007	071726	062706	000020 19\$:	ADD	#20,SP	: 6688
13008	071732	000137	072336 20\$:	JMP	29\$	
13009	071736	020527	000005 21\$:	CMP	R5,#5	: VALUE,* 6634
13010	071742	001160		BNE	28\$	
13011	071744	004737	035212	JSR	PC,ISOLATE	: 6692
13012	071750	032737	000001 002760	BIT	#1,ERROUT	: 6693
13013	071756	001423		BEQ	22\$	
13014	071760	017703	140442	MOV	@ML.REG+42,R3	: 6694
13015	071764	006203		ASR	R3	
13016	071766	006203		ASR	R3	
13017	071770	006203		ASR	R3	
13018	071772	006203		ASR	R3	
13019	071774	006203		ASR	R3	
13020	071776	006203		ASR	R3	
13021	072000	042703	177700	BIC	#177700,R3	
13022	072004	010346		MOV	R3,-(SP)	
13023	072006	012746	006262	MOV	#FMT10B,-(SP)	
13024	072012	012746	000002	MOV	#2,-(SP)	
13025	072016	010600		MOV	SP,R0	: SP,*
13026	072020	104414		TRAP	14	
13027	072022	062706	000006	ADD	#6,SP	
13028	072026	017766	140376 000036 22\$:	MOV	@ML.REG+44,36(SP)	: *.OLDSEC 6696
13029	072034	017703	140366	MOV	@ML.REG+42,R3	: 6697
13030	072040	006203		ASR	R3	
13031	072042	006203		ASR	R3	
13032	072044	006203		ASR	R3	
13033	072046	006203		ASR	R3	
13034	072050	006203		ASR	R3	
13035	072052	006203		ASR	R3	
13036	072054	042703	177700	BIC	#177700,R3	
13037	072060	010366	000040	MOV	R3,40(SP)	: *.OLDCHN
13038	072064	012746	000001	MOV	#1,-(SP)	: 6698
13039	072070	016646	000046	MOV	46(SP),-(SP)	: COMMAND,*
13040	072074	010146		MOV	R1,-(SP)	: LUN,*
13041	072076	010446		MOV	R4,-(SP)	: WRDCNT,*

Address	Offset	Count	Code	Operation	Comments	Time	Page
13043						23-Oct-1980 15:01:43	TOPS
13044						23-Oct-1980 14:57:05	PA:<
13045							
13046	072100	016646	000052	MOV	52(SP),-(SP) ; PTR,*		
13047	072104	016646	000040	MOV	40(SP),-(SP) ; SECTOR,*		
13048	072110	004737	043046	JSR	PC,RETRY		
13049	072114	062706	000014	ADD	#14,SP		
13050	072120	020027	000005	CMP	R0,#5		
13051	072124	001051		BNE	25\$		
13052	072126	027766	140276	000036	CMP @ML.REG+44,36(SP) ; *,OLDSEC		6700
13053	072134	001034		BNE	24\$		
13054	072136	016600	000040	MOV	40(SP),R0 ; OLDCHN,*		
13055	072142	017703	140260	MOV	@ML.REG+42,R3		
13056	072146	006203		ASR	R3		
13057	072150	006203		ASR	R3		
13058	072152	006203		ASR	R3		
13059	072154	006203		ASR	R3		
13060	072156	006203		ASR	R3		
13061	072160	006203		ASR	R3		
13062	072162	042703	177700	BIC	#177700,R3		
13063	072166	020300		CMP	R3,R0		
13064	072170	001016		BNE	24\$		
13065	072172	032737	000001	002260	BIT #1,ERROUT ;		6703
13066	072200	001404		BEQ	23\$		
13067	072202	104456		TRAP	56 ;		6705
13068	072204	012441		.WORD	12441		
13069	072206	010624		.WORD	MSG4		
13070	072210	000000		.WORD	0		
13071	072212	010146		23\$: MOV	R1,-(SP) ; LUN,*		6706
13072	072214	013746	032360	MOV	BOARD,-(SP)		
13073	072220	004737	035302	JSR	PC,UP.HARD.COUNT		
13074	072224	000426		BR	27\$ ;		6700
13075	072226	032737	000001	002260	24\$: BIT #1,ERROUT ;		6710
13076	072234	001415		BEQ	26\$ ;		6712
13077	072236	104457		TRAP	57 ;		6713
13078	072240	012442		.WORD	12442		
13079	072242	010610		.WORD	MSG3		
13080	072244	000000		.WORD	0		
13081	072246	000410		BR	26\$ ;		6717
13082	072250	032737	000001	002260	25\$: BIT #1,ERROUT ;		6719
13083	072256	001404		BEQ	26\$ ;		6720
13084	072260	104457		TRAP	57 ;		6720
13085	072262	012443		.WORD	12443		
13086	072264	010610		.WORD	MSG3		
13087	072266	000000		.WORD	0		
13088	072270	010146		26\$: MOV	R1,-(SP) ; LUN,*		6691
13089	072272	013746	032360	MOV	BOARD,-(SP)		6726
13090	072276	004737	035430	JSR	PC,UP.SOFT.COUNT		6578
13091	072302	022626		27\$: CMP	(SP)+,(SP)+ ;		6577
13092	072304	062737	000002	030706	28\$: ADD #2,WPTR ;		6577
13093	072312	062706	000020	ADD	#20,SP ;		
13094	072316	005266	000010	INC	10(SP) ; KOUNT2		
13095	072322	026627	000010	000012	CMP 10(SP),#12 ; KOUNI2,*		
13096	072330	003002		BGT	29\$		
13097	072332	000137	070712	JMP	3\$		



```
13099 ;MLX4
13100 ; TESTING RANDOM UNITS, SECTORS, DATA, WORD COUNT 23-Oct-1980 15:01:43 TOPS
13101 ; 23-Oct-1980 14:57:05 PA:<
13102 072336 005266 000012 29$: INC 12(SP) ; KOUNT 6569
13103 072342 026666 000012 000014 CMP 12(SP),14(SP) ; KOUNT,*
13104 072350 003002 BGT 30$
13105 072352 000137 070652 JMP 2$
13106 072356 005266 000030 30$: INC 30(SP) ; COUNT 6566
13107 072362 026666 000030 000050 CMP 30(SP),50(SP) ; COUNT,REPEAT
13108 072370 003002 BGT 31$
13109 072372 000137 070622 JMP 1$
13110 072376 062706 000032 31$: ADD #32,SP ;
13111 072402 000207 RTS PC 6510
13112
13113 ; Routine Size: 457 words
13114 ; Maximum stack depth per invocation: 36 words
13119
13120
```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (52)

```

13122 :MLX4
13123 :
13124 :
13125 : 6733 %SBTTL 'DEFINITION OF OPTION 5'
13126 : 6734
13127 : 6735 ROUTINE OPT5: NOVALUE =
13128 : 6736 BEGIN !* 1 *
13129 : 6737
13130 : 6738 !++
13131 : 6739 ROUTINE: OPT5
13132 : 6740
13133 : 6741 PURPOSE: TO EXERCISE THE ML11 SYSTEMS UNDER TEST IN A RANDOM
13134 : 6742 MANNER, SO AS TO SIMULATE THE FLEXIBILITY OF TESTING
13135 : 6743 THAT WOULD BE DONE BY AN OPERATING SYSTEM.
13136 : 6744
13137 : 6745 THERE ARE 4 RANDOM TESTS WHICH ARE CALLED BY 'OPT5'
13138 : 6746 TO ACCOMPLISH ALL TESTING. IT IS THE RESPONSIBILITY
13139 : 6747 OF THIS ROUTINE TO DECIDE HOW MANY TIMES THOSE 4
13140 : 6748 RANDOM TESTS WILL BE EXECUTED. REFER TO 'RAND1' TO
13141 : 6749 'RAND4' ABOVE FOR MORE INFORMATION.
13142 : 6750 !--
13143 : 6751
13144 : 6752 LOCAL
13145 : 6753 REPEAT;
13146 : 6754
13147 : 6755 PRINTB(SAY2,WRD34,RTN5);
13148 : 6756 !'RUNNING OPT5'
13149 : 6757
13150 : 6758 REPEAT = 9 - .NUM_DRIVES; !THE MORE UNITS THERE ARE ON THE SYSTEM, THE
13151 : 6759 !FEWER TIMES EACH RANDOM TEST WILL BE RUN.
13152 : 6760
13153 : 6761 INCR LUN FROM 0 TO (.L$UNIT - 1) DO !SEE IF THERE ARE ANY UNITS
13154 : 6762 BEGIN !* 2 * !WHICH HAVE 64K CHIPS (ML11-B)
13155 : 6763 IF .DRIVE_STATUS[LUN] EQL ACTIVE
13156 : 6764 THEN
13157 : 6765 BEGIN !* 3 *
13158 : 6766 UNIT = .DRIVE;
13159 : 6767 IF .MLDT
13160 : 6768 THEN
13161 : 6769 BEGIN !* 4 *
13162 : 6770 REPEAT = 1; !THERE IS AT LEAST 1 UNIT WHICH IS AN ML11-B, SO
13163 : 6771 !LOWER THE REPEAT TIME FOR ALL TESTABLE UNITS.
13164 : 6772 EXITLOOP;
13165 : 6773 END; !* 4 *
13166 : 6774 END; !* 3 *
13167 : 6775 END; !* 2 *
13168 : 6776
13169 : 6777 !+
13170 : 6778 ! AT THIS POINT, 'REPEAT' IS BASED ON THE SYSTEM CONFIGURATION.
13171 : 6779 ! NOW WEIGHT THE LOOP COUNTS OF THE 4 RANDOM TESTS SO THAT RAND1
13172 : 6780 ! AND RAND2 ARE MUCH QUICKER THAN RAND3 AND RAND4. RAND3 IS LONGEST.
13173 : 6781 !-
13174 : 6782
13175 : 6783 RAND1(.REPEAT); !TEST USING RANDOM DATA
13176 : 6784 RAND2(.REPEAT); !TEST USING RANDOM DATA, WORD COUNTS

```

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (52)

13178 :MLX4  
13179 :  
13180 :  
13181 : 6785 RAND3(.REPEAT);  
13182 : 6786 RAND4((.REPEAT \* 16));  
13183 : 6787  
13184 : 6788 RETURN;  
13185 : 6789  
13186 : 6790 END; !\* 1 \*  
13190  
13191

!TEST USING RANDOM SECTORS, DATA, WORD COUNTS  
!TEST USING RANDOM UNITS, SECTORS, DATA, WORD COUNTS

13195	072404	004137	005202	OPT5:	.SBTTL	OPT5 DEFINITION OF OPTION 5		
13196	072410	012746	007300		JSR	R1,\$SAVE4	:	6735
13197	072414	012746	006766		MOV	#RTN5,-(SP)	:	6755
13198	072420	012746	006522		MOV	#WRD34,-(SP)		
13199	072424	012746	000003		MOV	#SAY2,-(SP)		
13200	072430	010600			MOV	#3,-(SP)		
13201	072432	104414			MOV	SP,R0	: SP,*	
13202	072434	012703	000011		TRAP	14		
13203	072440	163703	032474		MOV	#11,R3	: *,REPEAT	6758
13204	072444	013704	002012		SUB	NUM.DRIVES,R3	: *,REPEAT	
13205	072450	005001			MOV	L\$UNIT,R4	:	6761
13206	072452	000450			CLR	R1	: LUN	
13207	072454	010102		1\$:	BR	3\$		
13208	072456	006202			MOV	R1,R2	: LUN,*	6763
13209	072460	006202			ASR	R2		
13210	072462	006202			ASR	R2		
13211	072464	062702	032460		ASR	R2		
13212	072470	010246			ADD	#DRIVE.STATUS,R2		
13213	072472	010146			MOV	R2,-(SP)		
13214	072474	042716	177770		MOV	R1,-(SP)	: LUN,*	
13215	072500	012746	000001		BIC	#177770,(SP)		
13216	072504	005046			MOV	#1,-(SP)		
13217	072506	004737	004244		CLR	-(SP)		
13218	072512	062706	000010		JSR	PC,BL\$GT2		
13219	072516	005300			ADD	#10,SP		
13220	072520	001024			DEC	R0		
13221	072522	010102			BNE	2\$		
13222	072524	006302			MOV	R1,R2	: LUN,*	6766
13223	072526	016202	032440		ASL	R2		
13224	072532	016200	000006		MOV	PTABLE.ADDR(R2),R2		
13225	072536	042700	177770		MOV	6(R2),R0		
13226	072542	142777	000007	137624	BIC	#177770,R0		
13227	072550	150077	137620		BICB	#7,@ML.REG+10		
13228	072554	032777	000001	137630	BISB	R0,@ML.REG+10		
13229	072562	001403			BIT	#1,@ML.REG+26	:	6767
13230	072564	012703	000001		BEQ	2\$		
13231	072570	000403			MOV	#1,R3	: *,REPEAT	6770
13232	072572	005201		2\$:	BR	4\$	:	6772
					INC	P1	: LUN	6761

```

13234 ;MLX4
13235 ;
13236 ;
13237 072574 020104 3$: CMP R1,R4 ; LUN,*
13238 072576 002726 BLT 1$
13239 072600 010316 4$: MOV R3,(SP) ; REPEAT,* 6783
13240 072602 004737 063116 JSR PC,RAND1
13241 072606 010316 MOV R3,(SP) ; REPEAT,* 6784
13242 072610 004737 065022 JSR PC,RAND2
13243 072614 010316 MOV R3,(SP) ; REPEAT,* 6785
13244 072616 004737 066670 JSR PC,RAND3
13245 072622 010300 MOV R3,R0 , REPEAT,* 6786
13246 072624 006300 ASL R0
13247 072626 006300 ASL R0
13248 072630 006300 ASL R0
13249 072632 006300 ASL R0
13250 072634 010016 MOV R0,(SP)
13251 072636 004737 070562 JSR PC,RAND4
13252 072642 062706 000010 ADD #10,SP ; 6735
13253 072646 000207 RTS PC
13254
13255 ; Routine Size: 82 words
13256 ; Maximum stack depth per invocation: 13 words
13261
13262
  
```

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (53)

```

13264 :MLX4
13265 :
13266 :
13267 : 6791 %SBTTL 'THE OPTION SCHEDULER'
13268 : 6792
13269 : 6793 BGNTST;
13270 : 6794
13271 : 6795 !++
13272 : 6796 ROUTINE: T1
13273 : 6797
13274 : 6798 PURPOSE: THIS IS THE ONE MAIN TEST OF THE EXERCISER. IT LOOKS
13275 : 6799 AT THE AVAILABILITY OF TEST OPTIONS AND MAKES CALLS TO
13276 : 6800 THE OPTIONS WHEN APPROPRIATE.
13277 : 6801
13278 : 6802 THE QUICK VERIFY PASS INCLUDES THE FOLLOWING ROUTINES:
13279 : 6803
13280 : 6804 (1) INTEGRITY
13281 : 6805 (2) OPT1
13282 : 6806 (3) OPT2
13283 : 6807 (4) OPT3
13284 : 6808
13285 : 6809 SUBSEQUENT PASSES INCLUDE:
13286 : 6810
13287 : 6811 (1) OPT1
13288 : 6812 (2) OPT2
13289 : 6813 (3) OPT3
13290 : 6814 (4) OPT4
13291 : 6815 (5) OPT5
13292 : 6816 !--
13293 : 6817
13294 : 6818 IF .QUICK NEQ 0
13295 : 6819 THEN !THIS IS THE QUICK VERIFY PASS
13296 : 6820 BEGIN
13297 : 6821 PRINTB(CRLF);
13298 : 6822 PRINTB(SAY3,WRD2,PHR2,WRD4);
13299 : 6823 !'BEGIN QUICK VERIFY PASS'
13300 : 6824 INTEGRITY();
13301 : 6825 END;
13302 : 6826
13303 : 6827 IF .DROP1 EQL 0
13304 : 6828 THEN !OPTION 1 IS AVAILABLE
13305 : 6829 OPT1();
13306 : 6830
13307 : 6831 IF .DROP2 EQL 0
13308 : 6832 THEN !OPTION 2 IS AVAILABLE
13309 : 6833 OPT2();
13310 : 6834
13311 : 6835 IF .DROP3 EQL 0
13312 : 6836 THEN !OPTION 3 IS AVAILABLE
13313 : 6837 OPT3();
13314 : 6838
13315 : 6839 IF .QUICK EQL 0
13316 : 6840 THEN
13317 : 6841 BEGIN
13318 : 6842 IF .DROP4 EQL 0
  
```

23-Oct-1980 15:01:43 TOPS-20 Bliss-16 V2(206)  
 23-Oct-1980 14:57:05 PA:<ROSEN>MLX4.BLI.1 (53)

```

13320 ;MLX4
13321 ;
13322 ;
13323 ; 6843 THEN !OPTION 4 IS AVAILABLE
13324 ; 6844 OPT4();
13325 ; 6845
13326 ; 6846 IF .DROPS EQL 0
13327 ; 6847 THEN !OPTION 5 IS AVAILABLE
13328 ; 6848 OPT5();
13329 ; 6849 END;
13330 ; 6850
13331 ; 6851 ENDTST;
13335 ;
13336 ;
  
```

```

13340 072650 005737 030712 $T1: .SBTTL $T1 THE OPTION SCHEDULER ; 6818
13341 072654 001426 BEQ 1$ ;
13342 072656 012746 006510 MOV #CRLF,-(SP) ; 6821
13343 072662 012746 0000C1 MOV #1,-(SP) ;
13344 072666 010600 MOV SP,R0 ; SP,*
13345 072670 104414 TRAP 14 ;
13346 072672 012716 006634 MOV #WRD4,(SP) ; 6822
13347 072676 012746 007352 MOV #PHR2,-(SP) ;
13348 072702 012746 006622 MOV #WRD2,-(SP) ;
13349 072706 012746 006534 MOV #SAY3,-(SP) ;
13350 072712 012746 000004 MOV #4,-(SP) ;
13351 072716 010600 MOV SP,R0 ; SP,*
13352 072720 104414 TRAP 14 ;
13353 072722 004737 044070 JSR PC,INTEGRITY ; 6824
13354 072726 062706 000014 ADD #14,SP ; 6820
13355 072732 005737 002236 1$: TST DROP1 ; 6827
13356 072736 001002 BNE 2$ ;
13357 072740 004737 046316 JSR PC,OPT1 ; 6829
13358 072744 005737 002240 2$: TST DROP2 ; 6831
13359 072750 001002 BNE 3$ ;
13360 072752 004737 050232 JSR PC,OPT2 ; 6833
13361 072756 005737 002242 3$: TST DROP3 ; 6835
13362 072762 001002 BNE 4$ ;
13363 072764 004737 053560 JSR PC,OPT3 ; 6837
13364 072770 005737 030712 4$: TST QUICK ; 6839
13365 072774 001012 BNE 6$ ;
13366 072776 005737 002244 TST DROP4 ; 6842
13367 073002 001002 BNE 5$ ;
13368 073004 004737 055472 JSR PC,OPT4 ; 6844
13369 073010 005737 002250 5$: TST DROP5 ; 6846
13370 073014 001002 BNE 6$ ;
13371 073016 004737 072404 JSR PC,OPT5 ; 6848
13372 073022 000207 6$: RTS PC ; 6790
13373 ;
13374 ;
  
```

; Routine Size: 54 words

```
13376      ;MLX4
13377      ;
13378      ; THE OPTION SCHEDULER
13379      ; Maximum stack depth per invocation: 6 words
13384
13385
13389
13390      .SBTTL T1 THE OPTION SCHEDULER
13394 073024 T1::
13395 073024 004737 072650 1$: JSR PC,$T1
13396 073030 104466 TRAP 66
13397 073032 006000 ROR R0
13398 073034 103773 BLO 1$
13399 073036 000207 RTS PC
13400
13401      ; Routine Size: 6 words
13402      ; Maximum stack depth per invocation: 0 words
13407
13408
```

23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<

6849

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (54)

```
13410 :MLX4
13411 :
13412 :
13413 : 6852 %SBTTL 'END OF PASS SUMMARY'
13414 : 6853
13415 : 6854 ROUTINE EOP: NOVALUE =
13416 : 6855 BEGIN !* 1 *
13417 : 6856
13418 : 6857 !++
13419 : 6858 ROUTINE: EOP
13420 : 6859
13421 : 6860 PURPOSE: TO PRINT A STATUS REPORT FOR EACH DRIVE. IF AN
13422 : 6861 ARRAY HAS ANY HARD OR SOFT ERRORS, THEN ITS ERROR
13423 : 6862 COUNT WILL APPEAR, AND IF THE COUNT IS TOO HIGH,
13424 : 6863 THEN A DIAGNOSIS TO RUN THE ML11 PROM MAINTENANCE
13425 : 6864 PROGRAM WILL ALSO APPEAR.
13426 : 6865
13427 : 6866 THE FOLLOWING IS A SAMPLE REPORT FOR 2 DRIVES:
13428 : 6867
13429 : 6868
13430 : 6869 PERFORMANCE SUMMARY
13431 : 6870
13432 : 6871 NUMBER OF MBYTES TRANSFERRED:
13433 : 6872 1028 MBYTES WRITTEN
13434 : 6873 250 MBYTES READ
13435 : 6874 1145 MBYTES WRITE CHECKED
13436 : 6875
13437 : 6876 LOGICAL UNIT: 0 DRIVE: 1 SERIAL #: 1234
13438 : 6877 SOFT ERROR COUNT: 9
13439 : 6878 ARRAY 3: 9
13440 : 6879 HARD ERROR COUNT: 11
13441 : 6880 ARRAY 0: 6
13442 : 6881 ARRAY 10: 2
13443 : 6882 ARRAY 15: 3
13444 : 6883 TRANSFER RETRIES: 0
13445 : 6884
13446 : 6885 LOGICAL UNIT: 1 DRIVE: 1 SERIAL #: 9876
13447 : 6886 DRIVE DROPPED (CONTROLLER FATAL ERROR)
13448 : 6887 SOFT ERROR COUNT: 100
13449 : 6888 ARRAY 1: 9
13450 : 6889 ARRAY 13: 10 --> RUN ML11 PROM MAINTENANCE PROGRAM
13451 : 6890 ARRAY 14: 1
13452 : 6891 ARRAY 15: 80 --> RUN ML11 PROM MAINTENANCE PROGRAM
13453 : 6892 HARD ERROR COUNT: 2
13454 : 6893 ARRAY 14: 1
13455 : 6894 ARRAY 15: 1
13456 : 6895 TRANSFER RETRIES: 0
13457 : 6896 --
13458 : 6897
13459 : 6898
13460 : 6899 LOCAL
13461 : 6900 SFT_TOT, HRD_TOT, TRY_TOT;
13462 : 6901
13463 : 6902 IF .EOPSUM
13464 : 6903 THEN !THE OPERATOR HAS ALLOWED THE SUMMARY TO PRINT
```



23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
PA:<ROSEN>MLX4.BLI.1 (54)

13466 :MLX4

END OF PASS SUMMARY

```
13467 :
13468 :
13469 : 6904 BEGIN !* 2 *
13470 : 6905 PRINTB(CRLF);
13471 : 6906 PRINTB(SAY1,PHR17);
13472 : 6907 !'PERFORMANCE SUMMARY'
13473 : 6908 PRINTB(CRLF);
13474 : 6909 PRINTB(SAY1,PHR12);
13475 : 6910 !NUMBER OF MBYTES TRANSFERED:
13476 : 6911 PRINTB(FMT14,.WR MILLIONS,PHR19);
13477 : 6912 !'XXXXX MBYTES WRITTEN'
13478 : 6913 PRINTB(FMT14,.RD MILLIONS,PHR20);
13479 : 6914 !'XXXXX MBYTES READ'
13480 : 6915 PRINTB(FMT14,.WC MILLIONS,PHR13);
13481 : 6916 !'XXXXX MBYTES WRITE CHECKED'
13482 : 6917 INCR LUN FROM 0 TO (.LSUNIT-1) DO
13483 : 6918 BEGIN !* 3 *
13484 : 6919 UNIT = .DRIVE;
13485 : 6920 IF .DPR EQL 0
13486 : 6921 THEN PRINTB(FMT4A,PHR7,.LUN,WRD11,.DRIVE)
13487 : 6922 !'LOGICAL UNIT: X DRIVE: Y'
13488 : 6923 ELSE SAYWHO(.LUN);
13489 : 6924 !'LOGICAL UNIT: X DRIVE: Y SERIAL #: ZZZZ'
13490 : 6925 IF .DROPT_DRIVES[.LUN] EQL ACTIVE
13491 : 6926 THEN
13492 : 6927 BEGIN !* 4 *
13493 : 6928 PRINTB(SAY2,WRD11,WRD21);
13494 : 6929 !'DRIVE DROPPED'
13495 : 6930 SELECTONE .WHY_DROPT[.LUN] OF
13496 : 6931 SET
13497 : 6932 [CODE_1] : PRINTB(FMT2,CAUSE1);
13498 : 6933 !' (NOT POWERED UP)'
13499 : 6934 [CODE_2] : PRINTB(FMT2,CAUSE2);
13500 : 6935 !' (NOT AN ML11 UNIT)'
13501 : 6936 [CODE_3] : PRINTB(FMT2,CAUSE3);
13502 : 6937 !' (OPERATOR SELECTED TEST LIMITS INCORRECTLY)'
13503 : 6938 [CODE_4] : PRINTB(FMT2,CAUSE4);
13504 : 6939 !' (ALL RETRIES FAILED FOR A NON-FATAL ERROR)'
13505 : 6940 [CODE_5] : PRINTB(FMT2,CAUSE5);
13506 : 6941 !' (CONTROLLER FATAL ERROR)'
13507 : 6942 [CODE_6] : PRINTB(FMT2,CAUSE6);
13508 : 6943 !' (DRIVE FATAL ERROR)'
13509 : 6944 [CODE_7] : PRINTB(FMT2,CAUSE7);
13510 : 6945 !' (ECC HARD ERROR)'
13511 : 6946 [CODE_8] : PRINTB(FMT2,CAUSE8);
13512 : 6947 !' (ECC LOGIC FAILURE)'
13513 : 6948 TES;
13514 : 6949 END; * 4 *
13515 : 6950
13516 : 6951 SFT_TOT = 0;
13517 : 6952 HRD_TOT = 0;
13518 : 6953 TRY_TOT = 0;
13519 : 6954 INCR ARRAY FROM 0 TO 15 DO
13520 : 6955 BEGIN
```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (54)

```

13522 :MLX4
13523 :
13524 :
13525 :      6956      SFT_TOT = .SFT_TOT + .SOFTS[.LUN,.ARRAY,0,16,0];
13526 :      6957      HRD_TOT = .HRD_TOT + .HARDS[.LUN,.ARRAY,0,16,0];
13527 :      6958      TRY_TOT = .TRY_TOT + .TRIES[.LUN,.ARRAY,0,16,0];
13528 :      6959      END;
13529 :      6960      PRINTB(FMT7,PHR5,.SFT_TOT);
13530 :      6961      !' SOFT ERROR COUNT: DDDDD'
13531 :      6962      IF .SFT_TOT NEQ 0
13532 :      6963      THEN
13533 :      6964          INCR ARRAY FROM 0 TO 15 DO
13534 :      6965              IF .SOFTS[.LUN,.ARRAY,0,16,0] NEQ 0
13535 :      6966              THEN
13536 :      6967                  BEGIN
13537 :      6968                      PRINTB(FMT9,.ARRAY,.SOFTS[.LUN,.ARRAY,0,16,0]);
13538 :      6969                      !' ARRAY XX: YYYY'
13539 :      6970                      !+
13540 :      6971                      ! NOW SEE IF THRESHOLDS FOR SOFT ERRORS
13541 :      6972                      ! HAVE BEEN REACHED FOR THIS ARRAY BOARD:
13542 :      6973                      !-
13543 :      6974                      IF ((.ARR_TYP EQL 0) AND
13544 :      6975                          (.SOFTS[.LUN,.ARRAY,0,16,0] GEQ S16K_LIMIT))
13545 :      6976                      THEN PRINTB(FMT2,MSG2);
13546 :      6977                      !' --> RUN ML11 PROM MAINTENANCE PROGRAM'
13547 :      6978                      IF ((.ARR_TYP EQL 1) AND
13548 :      6979                          (.SOFTS[.LUN,.ARRAY,0,16,0] GEQ S64K_LIMIT))
13549 :      6980                      THEN PRINTB(FMT2,MSG2);
13550 :      6981                      !' --> RUN ML11 PROM MAINTENANCE PROGRAM'
13551 :      6982                      END;
13552 :      6983      PRINTB(FMT7,PHR18,.HRD_TOT);
13553 :      6984      !' HARD ERROR COUNT: DDDDD'
13554 :      6985      IF .HRD_TOT NEQ 0
13555 :      6986      THEN
13556 :      6987          INCR ARRAY FROM 0 TO 15 DO
13557 :      6988              IF .HARDS[.LUN,.ARRAY,0,16,0] NEQ 0
13558 :      6989              THEN
13559 :      6990                  BEGIN
13560 :      6991                      PRINTB(FMT9,.ARRAY,.HARDS[.LUN,.ARRAY,0,16,0]);
13561 :      6992                      !' ARRAY XX: YYYY'
13562 :      6993                      !+
13563 :      6994                      ! NOW SEE IF THRESHOLDS FOR HARD ERRORS
13564 :      6995                      ! HAVE BEEN REACHED FOR THIS ARRAY BOARD:
13565 :      6996                      !-
13566 :      6997                      IF ((.ARR_TYP EQL 0) AND
13567 :      6998                          (.HARDS[.LUN,.ARRAY,0,16,0] GEQ H16K_LIMIT))
13568 :      6999                      THEN PRINTB(FMT2,MSG2);
13569 :      7000                      !' --> RUN ML11 PROM MAINTENANCE PROGRAM'
13570 :      7001                      IF ((.ARR_TYP EQL 1) AND
13571 :      7002                          (.HARDS[.LUN,.ARRAY,0,16,0] GEQ H64K_LIMIT))
13572 :      7003                      THEN PRINTB(FMT2,MSG2);
13573 :      7004                      !' --> RUN ML11 PROM MAINTENANCE PROGRAM'
13574 :      7005                      END;
13575 :      7006      PRINTB(FMT7,PHR6,.TRY_TOT);
13576 :      7007      !' TRANSFER RETRIES: DDDDD'

```

23-Oct-1980 15:01:43  
23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
PA.<ROSEN>MLX4.BLI.1 (54)

```
13578 :MLX4
13579 :
13580 :
13581 : 7008 IF .TRY_TOT NEQ 0
13582 : 7009 THEN
13583 : 7010 INCR ARRAY FROM 0 TO 15 DO
13584 : 7011 IF .TRIESE[LUN,.ARRAY,0,16,0] NEQ 0
13585 : 7012 THEN PRINTB(FMT9,.ARRAY,.TRIESE[LUN,.ARRAY,0,16,0]);
13586 : 7013 !' ARRAY XX: YYYYY'
13587 : 7014 END; !* 3 *
13588 : 7015 PRINTB(CRLF);
13589 : 7016 END; !* 2 *
13590 : 7017
13591 : 7018 RETURN;
13592 : 7019
13593 : 7020 END; !* 1 *
```

```
13597
13598
13602 073040 004137 005222 EOP: .SBTTL EOP END OF PASS SUMMARY ; 6854
13603 073044 162706 000006 SUB R1,$SAVE5 ;
13604 073050 032737 000001 002256 BIT #6,SP ; 6902
13605 073056 001002 BNE 1$ ;
13606 073060 000137 074660 JMP 31$ ;
13607 073064 012746 006510 1$: MOV #CRLF,-(SP) ; 6905
13608 073070 012746 000001 MOV #1,-(SP) ;
13609 073074 010600 MOV SP,RO ; SP,*
13610 073076 104414 TRAP 14 ;
13611 073100 012716 007764 MOV #PHR17,(SP) ; 6906
13612 073104 012746 006514 MOV #SAY1,-(SP) ;
13613 073110 012746 000002 MOV #2,-(SP) ;
13614 073114 010600 MOV SP,RO ; SP,*
13615 073116 104414 TRAP 14 ;
13616 073120 012716 006510 MOV #CRLF,(SP) ; 6908
13617 073124 012746 000001 MOV #1,-(SP) ;
13618 073130 010600 MOV SP,RO ; SP,*
13619 073132 104414 TRAP 14 ;
13620 073134 012716 007624 MOV #PHR12,(SP) ; 6909
13621 073140 012746 006514 MOV #SAY1,-(SP) ;
13622 073144 012746 000002 MOV #2,-(SP) ;
13623 073150 010600 MOV SP,RO ; SP,*
13624 073152 104414 TRAP 14 ;
13625 073154 012716 010032 MOV #PHR19,(SP) ; 6911
13626 073160 013746 032336 MOV WR.MILLIONS,-(SP) ;
13627 073164 012746 006470 MOV #FMT14,-(SP) ;
13628 073170 012746 000003 MOV #3,-(SP) ;
13629 073174 010600 MOV SP,RO ; SP,*
13630 073176 104414 TRAP 14 ;
13631 073200 012716 010052 MOV #PHR20,(SP) ; 6913
13632 073204 013746 032344 MOV RD.MILLIONS,-(SP) ;
```



13690								23-Oct-1980 15:01:43	TOPS
13691								23-Oct-1980 14:57:05	PA:<
13692									
13693	073462	012746	000003		MOV	#3,-(SP)			
13694	073466	010600			MOV	SP,R0	; SP,*		
13695	073470	104414			TRAP	14			
13696	073472	005003			CLR	R3			6930
13697	073474	156103	032464		BISB	WHY.DROPT(R1),R3	; *(LUN),*		
13698	073500	020327	000001		CMP	R3,#1			
13699	073504	001011			BNE	5\$			
13700	073506	012746	010102		MOV	#CAUSE1,-(SP)			6932
13701	073512	012746	005602		MOV	#FMT2,-(SP)			
13702	073516	012746	000002		MOV	#2,-(SP)			
13703	073522	010600			MOV	SP,R0	; SP,*		
13704	073524	104414			TRAP	14			
13705	073526	000523			BR	12\$			
13706	073530	020327	000002	5\$:	CMP	R3,#2			6930
13707	073534	001011			RNE	6\$			
13708	073536	012746	010124		MOV	#CAUSE2,-(SP)			6934
13709	073542	012746	005602		MOV	#FMT2,-(SP)			
13710	073546	012746	000002		MOV	#2,-(SP)			
13711	073552	010600			MOV	SP,R0	; SP,*		
13712	073554	104414			TRAP	14			
13713	073556	000507			BR	12\$			
13714	073560	020327	000003	6\$:	CMP	R3,#3			6930
13715	073564	001011			BNE	7\$			
13716	073566	012746	010150		MOV	#CAUSE3,-(SP)			6936
13717	073572	012746	005602		MOV	#FMT2,-(SP)			
13718	073576	012746	000002		MOV	#2,-(SP)			
13719	073602	010600			MOV	SP,R0	; SP,*		
13720	073604	104414			TRAP	14			
13721	073606	000473			BR	12\$			
13722	073610	020327	000004	7\$:	CMP	R3,#4			6930
13723	073614	001011			BNE	8\$			
13724	073616	012746	010224		MOV	#CAUSE4,-(SP)			6938
13725	073622	012746	005602		MOV	#FMT2,-(SP)			
13726	073626	012746	000002		MOV	#2,-(SP)			
13727	073632	010600			MOV	SP,R0	; SP,*		
13728	073634	104414			TRAP	14			
13729	073636	000457			BR	12\$			
13730	073640	020327	000005	8\$:	CMP	R3,#5			6930
13731	073644	001011			BNE	9\$			
13732	073646	012746	010300		MOV	#CAUSE5,-(SP)			6940
13733	073652	012746	005602		MOV	#FMT2,-(SP)			
13734	073656	012746	000002		MOV	#2,-(SP)			
13735	073662	010600			MOV	SP,R0	; SP,*		
13736	073664	104414			TRAP	14			
13737	073666	000443			BR	12\$			
13738	073670	020327	000006	9\$:	CMP	R3,#6			6930
13739	073674	001011			BNE	10\$			
13740	073676	012746	010332		MOV	#CAUSE6,-(SP)			6942
13741	073702	012746	005602		MOV	#FMT2,-(SP)			
13742	073706	012746	000002		MOV	#2,-(SP)			
13743	073712	010600			MOV	SP,R0	; SP,*		
13744	073714	104414			TRAP	14			









13914  
13915  
13916  
13917 074640 012716 000510  
13918 074644 012746 000001  
13919 074650 010600  
13920 074652 104414  
13921 074654 062706 000042  
13922 074660 062706 000006  
13923 074664 000207  
13924  
13925  
13926  
13931  
13932

:MLX4  
:  
END OF PASS SUMMARY  
30\$: MOV #CRLF,(SP) ;  
MOV #1,-(SP) ;  
MOV SP,R0 ; SP,\*  
TRAP 14 ;  
ADD #42,SP ;  
31\$: ADD #6,SP ;  
RTS PC ;

23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<

7015

6904  
6854

: Routine Size: 459 words  
: Maximum stack depth per invocation: 39 words

```

13934 ;MLX4
13935 :
13936 :
13937 : 7021 %SBTTL 'CLEANUP CODING SECTION'
13938 : 7022
13939 : 7023 BGNCLN;
13940 : 7024
13941 : 7025 !+
13942 : 7026 ! THE CLEANUP CODING SECTION IS EXECUTED AFTER
13943 : 7027 ! EACH PASS THROUGH THE EXERCISER.
13944 : 7028 !-
13945 : 7029
13946 : 7030
13947 : 7031 EOP_COUNT = .EOP_COUNT + 1;
13948 : 7032
13949 : 7033 IF .EOP_COUNT EQL 1
13950 : 7034 THEN PRINTB(SAY3,WRD3,PHR2,WRD4)
13951 : 7035 !'END QUICK VERIFY PASS'
13952 : 7036 ELSE PRINTB(FMT3,WRD3,WRD4,.EOP_COUNT);
13953 : 7037 !'** END PASS XX **'
13954 : 7038 EOP();
13955 : 7039
13956 : 7040 RETURN;
13957 : 7041
13958 : 7042 ENDCLN;
  
```

23-Oct-1980 15:01:43  
 23-Oct-1980 14:57:05

TOPS-20 Bliss-16 V2(206)  
 PA:<ROSEN>MLX4.BLI.1 (55)

```

13962
13963
13967 074666 005237 030722
13968 074672 023727 030722 000001
13969 074700 001015
13970 074702 012746 006634
13971 074706 012746 007352
13972 074712 012746 006630
13973 074716 012746 006534
13974 074722 012746 000004
13975 074726 010600
13976 074730 104414
13977 074732 000414
13978 074734 013746 030722
13979 074740 012746 006634
13980 074744 012746 006630
13981 074750 012746 005610
13982 074754 012746 000004
13983 074760 010600
13984 074762 104414
13985 074764 004737 073040
13986 074770 062706 000012
13987 074774 000207
13988
  
```

.SBTTL	LCLEAN	LCLEAN	CLEANUP	CODING	SECTION	
INC	INC	EOP	COUNT			7031
CMP		EOP	COUNT,#1			7033
BNE		1\$				
MOV		#WRD4,-(SP)				7034
MOV		#PHR2,-(SP)				
MOV		#WRD3,-(SP)				
MOV		#SAY3,-(SP)				
MOV		#4,-(SP)				
MOV		SP,R0				: SP,*
TRAP		14				
BR		2\$				
MOV	1\$:	EOP	COUNT,-(SP)			7033
MOV		#WRD4,-(SP)				7036
MOV		#WRD3,-(SP)				
MOV		#FMT3,-(SP)				
MOV		#4,-(SP)				
MOV		SP,R0				: SP,*
TRAP		14				
JSR	2\$:	PC	EOP			7038
ADD		#12,SP				7020
RTS		PC				

```
13990 ;MLX4
13991 ;
13992 ;
13993 ; Routine Size: 36 words
13994 ; Maximum stack depth per invocation: 5 words
13999
14000
14004
14005 .SBTTL LSCLEAN CLEANUP CODING SECTION
14009 074776 LSCLEAN::
14010 074776 004737 074666 JSR PC,LCLEAN ;
14011 075002 104412 TRAP 12
14012 075004 000207 RTS PC
14013
14014 ; Routine Size: 4 words
14015 ; Maximum stack depth per invocation: 0 words
14020
14021
14022 : 7043
14023 : 7044 LASTAD;
14024 : 7045 BGNSETUP (0);
14025 : 7046 ENDSETUP;
14029
14030 075006 075012 BLSLAS::.WORD TSFREE
14031 075010 000000 .WORD <<TSFREE-<BLSLAS+4>>/2>
14032 075012 000000 TSFREE::.WORD 0
14033
14034
14035 075012 L$LAST== BLSLAS+4
14036 000000 T$PTHV== 0
14037
14038
14039 .SBT L SEND.LINK CLEANUP CODING SECTION
14043 075014 SEND.LINK::
14044 075014 000207 RTS PC ;
```

23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<

7040

7042

23-Oct-1980 15:01:43 TOPS  
23-Oct-1980 14:57:05 PA:<

```
14046 ;MLX4
14047 ;
14048 ; CLEANUP CODING SECTION
14049
14050 ; Routine Size: 1 word
14051 ; Maximum stack depth per invocation: 0 words
14052
14053
14054
14055
14056
14057
14058 : 7047 END
14059 : 7048
14060 : 7049 ELUDOM
14061
14062
14063
14064
14065 ; OTS external references
14066 .GLOBL BL$GT2, $$SAVE5, $$SAVE4, $$SAVE3
14067 .GLOBL $$SAVE2, BL$PU2, BL$SHF, BL$DIV
14068 .GLOBL BL$MOD, BL$MUL
14069
14070
14071
14072
14073
14074
14075 : 7050
14076 : 7051
14077 ; Size: 8779 code + 5384 data words
14078 ; Run Time: 01:54.5
14079 ; Elapsed Time: 09:06.0
14080 ; Memory Used: 150 pages
14081 ; Compilation Complete
14082
14083 000001 .END
```

ADR = 000020 G	C\$AUTO= 000061	DOUBLE 041226	F\$INIT= 000006	I\$MSG = 000041
ASSEMB= 000010	C\$BRK = 000022	DRIVE. 032460 G	F\$JMP = 000050	I\$PROT= 000040
BANK 032362	C\$BSEG= 000004	DROPNE 002234 G	F\$MOD = 000000	I\$PTAB= 000041
BASE.A 030724	C\$BSUB= 000002	DROPT. 032462 G	F\$MSG = 000011	I\$PWR = 000041
BIT0 = 000001 G	C\$CEFG= 000045	DROP1 002236 G	F\$PROT= 000021	I\$RPT = 000041
BIT00 = 000001 G	C\$CLCK= 000062	DROP2 002240 G	F\$PWR = 000017	I\$SEG = 000041
BIT01 = 000002 G	C\$CLEA= 000012	DROP3 002242 G	F\$RPT = 000012	I\$SETU= 000041
BIT02 = 000004 G	C\$CLOS= 000035	DROP4 002244 G	F\$SEG = 000003	I\$SFT = 000041
BIT03 = 000010 G	C\$CLP1= 000006	DROP5 002250 G	F\$SOFT= 000005	I\$SRV = 000041
BIT04 = 000020 G	C\$CVEC= 000036	ECCDIS 002254 G	F\$SRV = 000010	I\$SUB = 000041
BIT05 = 000040 G	C\$DCLN= 000044	EF.CON= 000036 G	F\$SUB = 000002	I\$TST = 000041
BIT06 = 000100 G	C\$DODU= 000051	EF.NEW= 000035 G	F\$SW = 000014	I.A.M.D 032354
BIT07 = 000200 G	C\$DRPT= 000024	EF.PWR= 000034 G	F\$TEST= 000001	J\$JMP = 000167
BIT08 = 000400 G	C\$DU = 000053	EF.RES= 000037 G	GEN1 036014	LAU 005536
BIT09 = 001000 G	C\$EDIT= 000003	EF.STA= 000040 G	GEN2 036522	LAUTO 005372
BIT1 = 000002 G	C\$ERDF= 000055	END.RB= 030706	GEN3 036556	LCLEAN 074666
BIT10 = 002000 G	C\$ERHR= 000056	END.WB= 020706	GEN4 036654	LDU 005404
BIT11 = 004000 G	C\$ERRO= 000060	EOP 073040 G	GEN5 036710	LIMIT 002222 G
BIT12 = 010000 G	C\$ERSF= 000054	EOPSUM 002256 G	GETCNT 036176	LINIT 035556
BIT13 = 020000 G	C\$ERSO= 000057	EOP.CO 030722	GET.WR 044034	LOE = 040000 G
BIT14 = 040000 G	C\$ESCA= 000010	ERRBLK 002200 G	G\$CNTO= 000200	LOT = 000010 G
BIT15 = 100000 G	C\$ESEG= 000005	ERRMSG 002176 G	G\$DELM= 000372	LOW.SE 032476 G
BIT2 = 000004 G	C\$ESUB= 000003	ERRNBR 002174 G	G\$DISP= 000003	LRPT 005354
BIT3 = 000010 G	C\$ETST= 000001	ERROUT 002260 G	G\$EXCP= 000400	LSECT 002226 G
BIT4 = 000020 G	C\$EXIT= 000032	ERRTYP 002172 G	G\$HILI= 000002	L\$ACP 002110 G
BIT5 = 000040 G	C\$GETB= 000026	EVL = 000004 G	G\$LOLI= 000001	L\$APT 002036 G
BIT6 = 000100 G	C\$GETW= 000027	E\$END = 002100	G\$NO = 000000	L\$AU 005540 G
BIT7 = 000200 G	C\$GMAN= 000043	E\$LOAD= 000035	G\$OFFS= 000400	L\$AUT 002070 G
BIT8 = 000400 G	C\$GPHR= 000042	FILLER 036356	G\$OF SI= 000376	L\$AUTO 005374 G
BIT9 = 001000 G	C\$GPLO= 000030	FMT1A = 005550	G\$PRMA= 000001	L\$CCP 002106 G
BL\$DIV 005056 G	C\$GPRI= 000040	FMT1B = 005564	G\$PRMD= 000002	L\$CLEA 074776 G
BL\$GT1 004122 G	C\$INIT= 000011	FMT10A= 006206	G\$PRML= 000000	L\$CO 002032 G
BL\$GI2 004244 G	C\$INLP= 000020	FMT10B= 006262	G\$RADA= 000140	L\$DEPO 002011 G
BL\$LAS 075006 G	C\$MANI= 000050	FMT11 = 006276	G\$RADB= 000000	L\$DESC 002130 G
BL\$MOD 005070 G	C\$MEM = 000031	FMT12A= 006332	G\$RADL= 000040	L\$DESP 002076 G
BL\$MUL 004632 G	C\$MSG = 000023	FMT12B= 006400	G\$RADL= 000120	L\$DEVP 002060 G
BL\$PU1 004406 G	C\$OPEN= 000034	FMT13 = 006446	G\$RADO= 000020	L\$DISP 002204 G
BL\$PU2 004502 G	C\$PNTB= 000014	FMT14 = 006470	G\$XFER= 000004	L\$DLY 002116 G
BL\$SHF 005102 G	C\$PNTF= 000017	FMT15 = 006502	G\$YES = 000010	L\$DTP 002040 G
BOARD 032360	C\$PNTS= 000016	FMT2 = 005602	HARDS 031332	L\$DTYP 002034 G
BOE = 000400 G	C\$PNTX= 000015	FMT3 = 005610	HELP = 000000	L\$DU 005526 G
BR.LEV 030730	C\$QIO = 000377	FMT4A = 005640	HOE = 100000 G	L\$DUT 002072 G
CAUSE1= 010102	C\$RDBU= 000007	FMT4B = 005674	IBE = 010000 G	L\$DVTY 002122 G
CAUSE2= 010124	C\$REFG= 000047	FMT4C = 005712	IDU = 000040 G	L\$EF 002052 G
CAUSE3= 010150	C\$RESE= 000033	FMT5 = 005742	IER = 020000 G	L\$ENVI 002044 G
CAUSE4= 010224	C\$REVI= 000003	FMT6 = 006020	INIT.A 033120	L\$ERRT 002172 G
CAUSE5= 010300	C\$RFLA= 000021	FMT7 = 006070	INTEGR 044070	L\$ETP 002102 G
CAUSE6= 010332	C\$RPT = 000025	FMT8 = 006110	ISOLAT 035212	L\$EXP1 002046 G
CAUSE7= 010356	C\$SEFG= 000046	FMT9 = 006156	ISR = 000100 G	L\$EXP4 002064 G
CAUSE8= 010400	C\$SPRI= 000041	FMTAU = 000015	IXE = 004000 G	L\$EXP5 002066 G
CHECK 042622	C\$SVEC= 000037	F\$AUTO= 000020	I\$AU = 000041	L\$HARD 002264 G
CHOOSE 043010	C\$TPRI= 000013	F\$BGN = 000040	I\$AUTO= 000041	L\$HIME 002120 G
CLRTBL 032572	DATA.C 030716	F\$CLEA= 000007	I\$CLN = 000041	L\$HPCP 002016 G
COMP.C 030720	DECODE 035106	F\$DU = 000016	I\$DU = 000041	L\$HPTP 002022 G
CONF IG 033714	DFPTBL 002210 G	F\$END = 000041	I\$HRD = 000041	L\$HW 002210 G
CRLF = 006510	DIAGMC= 000000	F\$HARD= 000004	I\$INIT= 000041	L\$ICP 002104 G
C\$AU = 000052	DIVMOD 004674	F\$HW = 000013	I\$MOD = 000041	L\$INIT 036004 G

L\$ADP	002026	G	ONEFIL=	000001	P.AAI	006020	P.ACN	007164	QH4	002417
L\$LAST=	075012	G	ONLY	002232	P.AAJ	006070	P.ACO	007172	QS1	002646
L\$LOAD	002100	G	OPT1	046316	P.AAK	006110	P.ACP	007176	QS10	003513
L\$LUN	002074	G	OPT2	050232	P.AAL	006156	P.ACQ	007202	QS11	003532
L\$MREV	002050	G	OPT3	053560	P.AAM	006206	P.ACR	007206	QS12	003614
L\$NAME	002000	G	OPT4	055472	P.AAN	006262	P.ACS	007212	QS13	003633
L\$PRIO	002042	G	OPT5	072404	P.AAO	006276	P.ACT	007216	QS14	003665
L\$PROT	004014	G	OSAPTS=	000001	P.AAP	006332	P.ACU	007250	QS15	003717
L\$PRT	002112	G	OSAU =	000001	P.AAQ	006400	P.ACV	007256	QS16	003764
L\$REPP	002062	G	OSBGNR=	000001	P.AAR	006446	P.ACW	007264	QS2	002713
L\$REV	002010	G	OSBGNS=	000001	P.AAS	006470	P.ACX	007272	QS3	003000
L\$RPT	005362	G	OSDU =	000001	P.AAT	006502	P.ACY	007300	QS4	003033
L\$SOFT	002440	G	OSERRT=	000001	P.AAU	006510	P.ACZ	007306	QS5	003065
L\$SPC	002056	G	OSGNSW=	000001	P.AAV	006514	P.ADA	007314	QS6	003132
L\$SPCP	002020	G	OSPOIN=	000001	P.AAW	006522	P.ADB	007322	QS7	003436
L\$SPTP	002024	G	OSSETU=	000001	P.AAX	006534	P.ADC	007330	QS8	003455
L\$STA	002030	G	PATTBL	032536	P.AAY	006552	P.ADD	007336	QS9	003474
L\$SW	002222	G	PATTER	030714	P.AAZ	006574	P.ADE	007352	QUICK	030712
L\$TEST	002114	G	PHR1 =	007336	P.ABA	006622	P.ADF	007370	RANDOM	005352
L\$TIML	002014	G	PHR10 =	007542	P.ABB	006630	P.ADG	007412	RAND1	063116
L\$UNIT	002012	G	PHR11 =	007562	P.ABC	006634	P.ADH	007426	RAND2	065022
L10000	002220		PHR12 =	007624	P.ABD	006642	P.ADI	007450	RAND3	066670
L10001	002262		PHR13 =	007662	P.ABE	006652	P.ADJ	007472	RAND4	070562
L10002	002330		PHR14 =	007710	P.ABF	006662	P.ADK	007510	RANGE	002224
L10003	002646		PHR15 =	007726	P.ABG	006670	P.ADL	007522	RBUFF	020706
MARPAT	002246	G	PHR16 =	007744	P.ABH	006700	P.ADM	007542	RCBUFF=	021706
MLB10 =	007114		PHR17 =	007764	P.ABI	006706	P.ADN	007562	RDBUFF=	020706
MLB11 =	007122		PHR18 =	010010	P.ABJ	006714	P.ADO	007624	RD.COU	032340
MLB12 =	007126		PHR19 =	010032	P.ABK	006722	P.ADP	007662	RD.MIL	032344
MLB13 =	007132		PHR2 =	007352	P.ABL	006734	P.ADQ	007710	RD.THO	032342
MLB14 =	007136		PHR20 =	010052	P.ABM	006744	P.ADR	007726	READ	042434
MLB15 =	007142		PHR3 =	007370	P.ABN	006754	P.ADS	007744	REFRES	002252
MLB16 =	007146		PHR4 =	007412	P.ABO	006760	P.ADT	007764	RETRY	043046
MLB17 =	007152		PHR5 =	007426	P.ABP	006766	P.ADU	010010	RETRYI	032356
MLB18 =	007156		PHR6 =	007450	P.ABQ	006776	P.ADV	010032	RE2	005250
MLB19 =	007164		PHR7 =	007472	P.ABR	007012	P.ADW	010052	RE3	005246
MLB2 =	007052		PHR8 =	007510	P.ABS	007022	P.ADX	010066	RE4	005244
MLB20 =	007172		PHR9 =	007522	P.ABT	007034	P.ADY	010070	RN	005256
MLB21 =	007176		PNT =	001000	P.ABU	007040	P.ADZ	010072	RNDSEC	062614
MLB22 =	007202		PRI =	002000	P.ABV	007046	P.AEA	010076	RNDU	062714
MLB23 =	007206		PRI00 =	000000	P.ABW	007052	P.AEB	010102	RNDWC	062560
MLB24 =	007212		PRI01 =	000040	P.ABX	007056	P.AEC	010124	RPTR	030710
MLB3 =	007056		PRI02 =	000100	P.ABY	007062	P.AED	010150	RTN0 =	007216
MLB4 =	007062		PRI03 =	000140	P.ABZ	007066	P.AEE	010224	RTN1 =	007250
MLB5 =	007066		PRI04 =	000200	P.ACA	007072	P.AEF	010300	RTN2 =	007256
MLB6 =	007072		PRI05 =	000240	P.ACB	007076	P.AEG	010332	RTN3 =	007264
MLB7 =	007076		PRI06 =	000300	P.ACC	007102	P.AEH	010356	RTN4 =	007272
MLB8 =	007102		PRI07 =	000340	P.ACD	007106	P.AEI	010400	RTN5 =	007300
MLB9 =	007106		PTABLE	032440	P.ACE	007114	P.AEJ	010424	RTNSA =	007306
ML.REG	032364	G	P.AAA	005550	P.ACF	007122	P.AEK	010512	RTNSB =	007314
MSG0 =	010424		P.AAB	005564	P.ACG	007126	P.AEL	010542	RTNSC =	007322
MSG1 =	010512		P.AAC	005602	P.ACH	007132	P.AEM	010610	RTNSD =	007330
MSG2 =	010542		P.AAD	005610	P.ACI	007136	P.AEN	010624	SAYWHO	033436
MSG3 =	010610		P.AAE	005640	P.ACJ	007142	P.AEO	010640	SAY1 =	006514
MSG4 =	010624		P.AAF	005674	P.ACK	007146	QH1	002330	SAY2 =	006522
MSG5 =	010640		P.AAG	005712	P.ACL	007152	QH2	002345	SAY3 =	006534
NUM.DR	032474	G	P.AAH	005742	P.ACM	007156	QH3	002367	SAY4 =	006552

SAY5 = 006574	TRT11 = 010076	T\$TEMP= 000000	WC.MIL 032352	WRD38 = 007034
SEED1 005344 G	TSECT 002230 G	T\$TEST= 000000	WC.THO 032350	WRD4 = 006634
SEED2 005346 G	T\$ARGC= 000002	T\$TSTM= 177777	WDBUFF= 010706	WRD40 = 007040
SEED3 005350 G	T\$CODE= 017130	T\$TSTS= 000000	WHY.DR 032464 G	WRD41 = 007046
SELPAT 036156	T\$ERRN= 000000	T\$SHAR= 010002	WPTR 030706	WRD6 = 006642
SERVIC 032562 G	T\$EXCP= 000000	T\$SHW = 010000	WRD11 = 006662	WRD7 = 006652
SET.PT 043770	T\$FREE 075012 G	T\$SPRO= 010004	WRD15 = 006670	WRITE 042246
SFPTBL 002222 G	T\$GMAN= 000000	T\$SSOF= 010003	WRD16 = 006700	WR.COU 032332
SOFTS 030732	T\$HILI= 000012	T\$SSW = 010001	WRD17 = 006706	WR.MIL 032336
START. 041324	T\$LAST= 000000	T1 073024 G	WRD18 = 006714	WR.THO 032334
SVCGBL= 000001	T\$LOLI= 000001	UAM = 000200 G	WRD19 = 006722	X\$ALWA= 000000
SVCINS= 000001	T\$LSYM= 010000	UP.HAR 035302	WRD2 = 006622	X\$FALS= 000040
SVCSUB= 000001	T\$NEST= 177777	UP.RD. 042062	WRD20 = 006734	X\$OFFS= 000400
SVCTAG= 000001	T\$NSO = 000000	UP.SOF 035430	WRD21 = 006744	X\$TRUE= 000020
SVCTST= 000001	T\$NS1 = 000021	UP.WC. 042154	WRD24 = 006754	\$END.L 075014 G
SYSERR 037000	T\$PTHV= 000000 G	UP.WR. 041770	WRD25 = 006760	\$PATCH 004022 G
\$LSYM= 010000	T\$PTNU= 000000	VEC 030726	WRD3 = 006630	\$SAVE2 005150 G
TOP.SE 032516 G	T\$SAVL= 177777	WAITER 041320	WRD34 = 006766	\$SAVE3 005164 G
TRIES 031732	T\$SEGL= 177777	WBUFF 010706	WRD35 = 006776	\$SAVE4 005202 G
TRT00 = 010066	T\$SUBN= 000000	WCBUFF= 011706	WRD36 = 007012	\$SAVE5 005222 G
TRT01 = 010070	T\$TAGL= 177777	WC.COU 032346	WRD37 = 007022	\$T1 072650
TRT10 = 010072	T\$TAGN= 010005			

. ABS. 075016 000  
000000 001  
ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 31277 WORDS ( 123 PAGES)  
DYNAMIC MEMORY: 21558 WORDS ( 82 PAGES)  
ELAPSED TIME: 00:16:12  
ML11X.BIN/EN:AMA,CZMLBA/CR/-SP=SVC/ML,ONEFILEX,MLX1,MLX2,OTS,RANDOM,MLX3,MLX4

CZMLBA  
 SYMBOL CROSS REFERENCE  
 SYMBOL VALUE  
 ADR = 000020 G  
 ASSEMB = 000010  
 BANK 032362  
 BASE.A 030724  
 BIT0 = 000001 G  
 BIT00 = 000001 G  
 BIT01 = 000002 G  
 BIT02 = 000004 G  
 BIT03 = 000010 G  
 BIT04 = 000020 G  
 BIT05 = 000040 G  
 BIT06 = 000100 G  
 BIT07 = 000200 G  
 BIT08 = 000400 G  
 BIT09 = 001000 G  
 BIT1 = 000002 G  
 BIT10 = 002000 G  
 BIT11 = 004000 G  
 BIT12 = 010000 G  
 BIT13 = 020000 G  
 BIT14 = 040000 G  
 BIT15 = 100000 G  
 BIT2 = 000004 G  
 BIT3 = 000010 G  
 BIT4 = 000020 G  
 BIT5 = 000040 G  
 BIT6 = 000100 G  
 BIT7 = 000200 G  
 BIT8 = 000400 G  
 BIT9 = 001000 G  
 BL\$DIV 005056 G  
 BL\$G11 004122 G  
 BL\$G12 004244 G  
 BL\$LAS 075006 G  
 BL\$MOD 005070 G  
 BL\$MUL 004632 G  
 BL\$PU1 004406 G  
 BL\$PU2 004502 G  
 BL\$SHF 005102 G  
 BOARD 032360  
 BOE = 000400 G  
 BR.LEV 030730  
 CAUSE1 = 010102  
 CAUSE2 = 010124

MACRO ON 24-OCT-80 AT 10:07

PAGE 1  
 CREF V01

G 9

SEQ 0318

REFERENCES  
 #120-1727  
 38-13 38-13  
 #119-1642 \*142-2840 \*143-2853 144-2904 205-5774 \*205-5786  
 #118-1607 \*129-2176 130-2225 141-2774  
 #120-1711  
 #120-1701  
 #120-1700  
 #120-1699  
 #120-1698  
 #120-1697  
 #120-1696  
 #120-1695  
 #120-1694  
 #120-1693  
 #120-1692  
 #120-1710  
 #120-1691  
 #120-1690  
 #120-1689  
 #120-1688  
 #120-1687  
 #120-1686  
 #120-1709  
 #120-1708  
 #120-1707  
 #120-1706  
 #120-1705  
 #120-1704  
 #120-1703  
 #120-1702  
 #72-1175 233-7240 248-8065 253-8331 264-8926 307-11286 313-11632 324-12229 324-12243  
 335-12827 359-14067  
 #48-170  
 #53-386 214-6242 227-6943 243-7783 249-8094 258-8632 276-9577 296-10682 302-10996  
 313-11612 324-12215 342-13217 351-13682 359-14066  
 #358-14030 358-14031 358-14035  
 #73-1206 159-3637 292-10502 294-10583 296-10667 296-10691 359-14068  
 #69-1027 138-2648 359-14068  
 #58-595  
 #64-833 83-198 83-209 85-311 126-2050 126-2057 151-3200 151-3209 159-3633  
 359-14067  
 48-182 48-185 48-189 48-193 49-218 54-408 54-411 54-415 54-419  
 54-440 58-598 58-602 58-608 59-635 64-842 64-846 64-852 65-884  
 #74-1241 129-2181 359-14067  
 #119-1641 \*143-2858 144-2905 205-5773 205-5782 \*205-5785 219-6478 219-6495 221-6622  
 221-6639 232-7210 233-7231 248-8039 248-8056 252-8296 253-8317 263-8896 263-8913  
 281-9859 281-9876 286-10138 286-10155 291-10418 291-10435 306-11256 307-11277 318-11886  
 318-11903 329-12492 329-12509 339-13072 339-13089  
 #121-1735  
 #118-1610 \*129-2178 129-2179  
 #123-1863 137-2594 352-13700  
 #123-1864 138-2621 352-13708



SYMBOL	VALUE	REFERENCES
CAUSE3	= 010150	#123-1865 139-2692 140-2723 352-13716
CAUSE4	= 010224	#123-1866 352-13724
CAUSE5	= 010300	#123-1867 352-13732
CAUSE6	= 010332	#123-1868 352-13740
CAUSE7	= 010356	#123-1869 353-13752
CAUSE8	= 010400	#123-1870 353-13760
CHECK	042622	#197-5380 199-5476 203-5664 203-5696 203-5707 219-6502 219-6507 221-6593 229-7040 245-7874 249-8131 260-8726 278-9690 283-9968 288-10248 303-11086 315-11721 326-12327 328-12459 336-12907
CHOOSE	043010	#199-5469 229-7022 244-7852 249-8112 260-8708 277-9664 282-9943 287-10222 303-11068 314-11699 325-12305 336-12889
CLRTBL	032572	#125-1996 150-3168
COMP.C	030720	#118-1603 *156-3521 159-3623
CONFIG	033714	#137-2568 151-3202
CRLF	= 006510	#121-1770 129-2188 130-2217 204-5741 204-5761 345-13342 350-13607 350-13616 356-13917
C\$AU	= 000052	#38-13
C\$AUTO	= 000061	#38-13
C\$BRK	= 000022	#38-13
C\$BSEG	= 000004	#38-13
C\$BSUB	= 000002	#38-13
C\$CEFG	= 000045	#38-13
C\$CLCK	= 000062	#38-13
C\$CLEA	= 000012	#38-13
C\$CLOS	= 000035	#38-13
C\$CLP1	= 000006	#38-13
C\$CVEC	= 000036	#38-13
C\$DCLN	= 000044	#38-13
C\$DODU	= 000051	#38-13
C\$DRPT	= 000024	#38-13
C\$DU	= 000053	#38-13
C\$EDIT	= 000003	#38-13 38-65
C\$ERDF	= 000055	#38-13
C\$ERHR	= 000056	#38-13
C\$ERRO	= 000060	#38-13
C\$ERSF	= 000054	#38-13
C\$ERSO	= 000057	#38-13
C\$ESCA	= 000010	#38-13
C\$ESEG	= 000005	#38-13
C\$ESUB	= 000003	#38-13
C\$ETST	= 000001	#38-13
C\$EXIT	= 000032	#38-13
C\$GETB	= 000026	#38-13
C\$GETW	= 000027	#38-13
C\$GMAN	= 000043	#38-13
C\$GPHR	= 000042	#38-13
C\$GPLO	= 000030	#38-13
C\$GPRI	= 000040	#38-13
C\$INIT	= 000011	#38-13
C\$INLP	= 000020	#38-13
C\$MANI	= 000050	#38-13
C\$MEM	= 000031	#38-13
C\$MSG	= 000023	#38-13

SYMBOL	CROSS REFERENCE	VALUE	REFERENCES
CSOPEN	=	000034	#38-13
CSPNTB	=	000014	#38-13
CSPNTF	=	000017	#38-13
CSPNTS	=	000016	#38-13
CSPNTX	=	000015	#38-13
CSQIO	=	000377	#38-13
CSRDBU	=	000007	#38-13
CSREFG	=	000047	#38-13
CSRESE	=	000033	#38-13 #38-13
CSREVI	=	000003	#38-13 38-65
CSRFLA	=	000021	#38-13
CSRPT	=	000025	#38-13
CSSEFG	=	000046	#38-13
CSSPRI	=	000041	#38-13
CSSVEC	=	000037	#38-13
CSTPRI	=	000013	#38-13
DATA.C		030716	#118-1601 *156-3520 159-3625
DECODE		035106	#142-2834 144-2901 205-5776
DFPTBL		002210	G #39-117
DIAGMC	=	000000	38-13 38-13
DIVMOD		004674	#70-1096 72-1176 73-1207
DOUBLE		041226	#180-4629 216-6321 229-7049 245-7882 249-8139 260-8735 278-9699 283-9977 288-10257
			303-11095 315-11730 326-12336 336-12916
DRIVE.		032460	G 82-176 83-192 #119-1649 126-2045 150-3177 214-6236 227-6937 243-7777 248-8084
			258-8626 276-9571 296-10676 301-10986 313-11606 323-12205 342-13211
DROPNE		002234	G #40-160 120-1680 126-2063
DROPT.		032462	G 82-176 83-203 #119-1652 126-2052 351-13676
DROP1		002236	G #40-161 120-1680 *127-2069 345-13355
DROP2		002240	G #40-162 120-1680 *127-2070 345-13358
DROP3		002242	G #40-163 120-1681 *127-2071 345-13361
DROP4		002244	G #40-164 120-1681 *127-2072 345-13366
DROP5		002250	G #40-166 120-1681 *127-2073 345-13369
ECCDIS		002254	G #40-169 120-1682 129-2183 172-4218 172-4250 173-4302 175-4391 *84-4865
EF.COM	=	000036	G #120-1714
EF.NEW	=	000035	G #120-1715
EF.PWR	=	000034	G #120-1716
EF.RES	=	000037	G #120-1713
EF.STA	=	000040	G #120-1712
END.RB	=	030706	#121-1748
END.WB	=	020706	#121-1747 206-5844
EOP		073040	G 79-32 79-40 #350-13602 357-13985
EOPSUM		002256	G #40-188 120-1682 350-13604
EOP.CO		030722	#118-1605 *126-2022 *357-13967 357-13968 357-13978
ERRBLK		002200	G #39-94
ERRMSG		002176	G #39-94
ERRNBR		002174	G #39-94
ERROUT		002260	G #40-190 120-1682 144-2902 173-4265 177-4535 194-5201 196-5302 198-5403 202-5647
			204-5733 204-5753 217-6414 218-6467 219-6481 219-6488 220-6562 221-6615 221-6625
			221-6632 231-7150 232-7203 232-7213 232-7220 247-7979 247-8028 248-8042 248-8049
			251-8236 252-8289 252-8299 252-8306 262-8836 263-8889 263-8899 263-8906 280-9799
			281-9852 281-9862 281-9869 285-10078 286-10131 286-10141 286-10148 290-10358 291-10411
			291-10421 291-10428 305-11196 306-11249 306-11259 306-11266 317-11826 318-11879 318-11889

SYMBOL	CROSS REFERENCE VALUE	REFERENCES	318-11896	327-12428	328-12481	329-12495	329-12502	338-13012	339-13065	339-13075	339-13082
ERRYP	= 002172 G	#39-94									
EVL	= 000004 G	#120-1725									
E\$END	= 002100	#38-13									
E\$LOAD	= 000035	#38-13	38-65								
FILLER	= 036356	#159-3615	161-3700	164-3839							
FMT1A	= 005550	#121-1750	243-7755								
FMT1B	= 005564	#121-1751	243-7764								
FMT10A	= 006206	#121-1762	144-2909								
FMT10B	= 006262	#121-1763	218-6429		220-6573	231-7161	247-7990	251-8247	262-8847	280-9810	285-10089
		290-10369	305-11207		317-11837	328-12443	338-13023				
FMT11	= 006276	#121-1764	139-2701		140-2732						
FMT12A	= 006332	#121-1765	216-6334		230-7066	245-7895	250-8156	261-8752	278-9712	283-9990	288-10270
		304-11112	315-11743		326-12349	336-12929					
FMT12B	= 006400	#121-1766	216-6341		230-7073	245-7902	250-8163	261-8759	278-9719	283-9997	288-10277
		304-11119	315-11750		326-12356	337-12940					
FMT13	= 006446	#121-1767	146-2989		148-3069						
FMT14	= 006470	#121-1768	350-13627		351-13637	351-13643					
FMT15	= 006502	#121-1769	173-4279		173-4287	173-4295	173-4305	174-4319	174-4327	174-4354	174-4362
		175-4374	175-4384		175-4394	175-4402	175-4410	175-4418	176-4430	176-4440	176-4448
		176-4456	176-4464		176-4472	176-4480	177-4494	177-4502	177-4510		
FMT2	= 005602	#121-1752	137-2595		138-2622	139-2693	140-2724	276-9588	277-9660	282-9939	287-10218
		352-13701	352-13709		352-13717	352-13725	352-13733	352-13741	353-13753	353-13761	354-13816
		354-13826	355-13865		355-13875						
FMT3	= 005610	#121-1753	357-13981								
FMT4A	= 005640	#121-1754	132-2329	137-2584	351-13664						
FMT4B	= 005674	#121-1755	132-2343	141-2776							
FMT4C	= 005712	#121-1756	132-2353								
FMT5	= 005742	#121-1757	140-2744								
FMT6	= 006020	#121-1758	185-4907	194-5209	196-5310	198-5411					
FMT7	= 006070	#121-1759	353-13788	354-13837	355-13886						
FMT8	= 006110	#121-1760	141-2770								
FMT9	= 006156	#121-1761	354-13807	354-13852	355-13900						
F\$AU	= 000015	#38-13									
F\$AUTO	= 000020	#38-13									
F\$BGN	= 000040	#38-13	38-39	41-256	42-300	44-376	44-408				
F\$CLEA	= 000007	#38-13									
F\$DU	= 000016	#38-13									
F\$END	= 000041	#38-13	38-13	38-13	38-13	38-13	38-13	38-13	38-13	38-13	38-13
		38-13	38-13	38-13	38-13	38-13	38-13	38-13	38-13	38-13	38-39
		41-274	43-337	44-408							
F\$HARD	= 000004	#38-13	41-256	41-274	42-312	42-314	42-317	42-320	43-323	43-328	
F\$HW	= 000013	#38-13	39-117	39-134							
F\$INIT	= 000006	#38-13									
F\$JMP	= 000050	#38-13									
F\$MOD	= 000000	#38-13	38-39	44-408							
F\$MSG	= 000011	#38-13									
F\$PROT	= 000021	#38-13	44-376	44-382							
F\$PWR	= 000017	#38-13									
F\$RPT	= 000012	#38-13									
F\$SEG	= 000003	#38-13									
F\$SOFT	= 000005	#38-13	42-300	42-312	42-314	42-317	42-320	43-323	43-328	43-337	

SYMBOL	CROSS REFERENCE	VALUE	REFERENCES
FSSRV	=	000010	#38-13
FSSUB	=	000002	#38-13
FSSW	=	000014	#38-13 40-143 40-192
FSTEST	=	000001	#38-13
GEN1		036014	#153-3313 228-6961
GEN2		036522	#161-3695 243-7769
GEN3		036556	#162-3763 214-6228 258-8618
GEN4		036654	#164-3834 275-9551 275-9555
GEN5		036710	#165-3895 301-10978 313-11601 323-12200 334-12800
GETCNT		036176	#156-3486 161-3696 164-3835
GET.WR		044034	#208-5893 215-6258 228-6964 243-7797 249-8108 259-8650 302-11010
GSCNTO	=	000200	#38-13
GSDERM	=	000372	#38-13
GSDISP	=	000003	#38-13
GSEXCP	=	000400	#38-13
GSHILI	=	000002	#38-13
GSLOLI	=	000001	#38-13
GSSNO	=	000000	#38-13
GSOFFS	=	000400	#38-13 41-269 41-270 41-271 41-272 42-311 42-313 42-315 42-316 42-318 42-319 43-322 43-324 43-325 43-326 43-327 43-329 43-330 43-331 43-332 43-333 43-334
GSOFSI	=	000376	#38-13 41-269 41-270 41-271 41-272 42-311 42-313 42-315 42-316 42-318 42-319 43-322 43-324 43-325 43-326 43-327 43-329 43-330 43-331 43-332 43-333 43-334
GSPRMA	=	000001	#38-13 41-269 41-270
GSPRMD	=	000002	#38-13 41-271 41-272 42-315 42-316 42-318 43-322 43-329 43-330
GSPRML	=	000000	#38-13 42-311 42-313 42-319 43-324 43-325 43-326 43-327 43-330 43-331 43-332 43-333 43-334
GSRADA	=	000140	#38-13
GSRADB	=	000000	#38-13
GSRADD	=	000040	#38-13 42-318 43-322 43-329
GSRADL	=	000120	#38-13 42-311 42-313 42-319 43-324 43-325 43-326 43-327 43-330 43-331 43-332 43-333 43-334
GSRADC	=	000020	#38-13 41-269 41-270 41-271 41-272 42-315 42-316
G\$XFER	=	000004	#38-13 42-312 42-314 42-317 42-320 43-323 43-328
G\$YES	=	000010	#38-13 41-269 41-270 41-271 41-272 42-311 42-313 42-315 42-316 42-318 42-319 43-322 43-324 43-325 43-326 43-327 43-329 43-330 43-331 43-332 43-333 43-334
HARDS		031332	#118-1613 *126-2014 145-2971 353-13781 354-13847
HELP	=	000000	#38-4 38-8 38-30 38-48 38-67 39-103 39-119 40-145 #41-197 41-258 41-276 42-302 44-339 44-384 44-401 44-413
HOE	=	100000	G #121-1742
IBE	=	010000	G #121-1739
IDU	=	000040	G #120-1728
IER	=	020000	G #121-1740
INIT.A		033120	#129-2173 150-3188
INTEGR		044070	#214-6217 345-13353
ISOLAT		035212	#144-2900 217-6401 217-6413 220-6550 220-6561 231-7136 231-7149 246-7962 246-7974 251-8223 251-8235 262-8822 262-8835 279-9782 280-9798 284-10060 285-10077 289-10340 290-10357 305-11182 305-11195 316-11809 317-11825 327-12415 327-12427 338-12999 338-13011
ISR	=	000100	G #121-1733
IXE	=	004000	G #121-1738

SYMBOL	CROSS REFERENCE	VALUE	REFERENCES
ISAU	=	000041	#38-13
ISAUTO	=	000041	#38-13
ISCLN	=	000041	#38-13
ISDU	=	000041	#38-13
ISHRD	=	000041	#41-256 #41-274
ISINIT	=	000041	#38-13
ISMOD	=	000041	#38-13 38-39 #38-39 44-408 #44-408
ISMSG	=	000041	#38-13
ISPROT	=	000040	#38-13 #44-376
ISPTAB	=	000041	#38-13
ISPR	=	000041	#38-13
ISRPT	=	000041	#38-13
ISSEG	=	000041	#38-13
ISSETU	=	000041	#38-13
ISSFT	=	000041	#42-300 #43-337
ISSRV	=	000041	#38-13
ISSUB	=	000041	#38-13
ISTST	=	000041	#38-13
I.A.M.D		032354	#119-1637 *123-1886 *185-4876 185-4882 185-4888
JSJMP	=	000167	#38-13
LAU		005536	#84-269 84-287
LAUTO		005372	#81-103 81-121
LCLEAN		074666	#357-13967 358-14010
LDU		005404	#82-184 83-232
LIMIT		002222 G	#40-153 120-1679 139-2659
LINIT		035556	#150-3160 151-3234
LOE	=	040000 G	#121-1741
LOT	=	000010 G	#120-1726
LW.SE		032476 G	#119-1659 *126-2039 *140-2715 140-2742 215-6253 228-6956 243-7792 249-8103 259-8645 276-9594 293-10566 302-11005 313-11619 324-12227
LRPT		005354	#79-40 79-59
LSECT		002226 G	#40-157 120-1679 *139-2671 *139-2679 140-2713 140-2715 140-2730
LSACP		002110 G	#38-65
LSAPT		002036 G	#38-65
LSAU		005540 G	38-65 #84-287
LSAUT		002070 G	#38-65
LSAUTO		005374 G	38-65 #81-121
LSCCP		002106 G	#38-65
LSCLEA		074776 G	38-65 #358-14009
LSCO		002032 G	#38-65
L\$DEPO		002011 G	#38-65
L\$DESC		002130 G	38-65 #39-85
L\$DESP		002076 G	#38-65
L\$DEVP		002060 G	#38-65
L\$DISP		002204 G	38-65 #39-101
L\$DLY		002116 G	#38-65
L\$DTP		002040 G	#38-65
L\$DTYP		002034 G	#38-65
L\$DU		005526 G	38-65 #83-232
L\$DUT		002072 G	#38-65
L\$DVTY		002122 G	38-65 #38-80
L\$EF		002052 G	#38-65

SYMBOL	CROSS REFERENCE	VALUE	REFERENCES
LSENVI		002044 G	#38-65
LSERRT		002172 G	38-65 #39-94
LSETP		002102 G	#38-65
LSEXP1		002046 G	#38-65
LSEXP4		002064 G	#38-65
LSEXP5		002066 G	#38-65
LSHARD		002264 G	38-65 41-256 #41-256
LSHIME		002120 G	#38-65
LSHPCP		002016 G	#38-65
LSHPTP		002022 G	#38-65
LSHW		002210 G	38-65 39-117 #39-117
LSICP		002104 G	#38-65
LSINIT		036004 G	38-65 #151-3234
LSLADP		002026 G	#38-65
LSLAST	=	075012 G	38-65 #358-14035
LSLOAD		002100 G	#38-65
LSLUN		002074 G	#38-65 82-176 82-184 83-194 83-199 83-205 120-1679 *126-2036 *150-3172 *214-6246 *228-6951 *243-7787 *249-8098 *259-8640 *276-9581 *302-11000 *313-11616 *324-12219 *334-12808
LSMREV		002050 G	#38-65
LSNAME		002000 G	#38-65
LSPRIO		002042 G	#38-65
LSPROT		004014 G	38-65 #44-376
LSPRT		002112 G	#38-65
LSREPP		002062 G	#38-65
LSREV		002010 G	#38-65
LSRPT		005362 G	38-65 #79-59
LSSOFT		002440 G	38-65 42-300 #42-300
LSSPC		002056 G	#38-65
LSSPCP		002020 G	#38-65
LSSPTP		002024 G	#38-65
LSSTA		00203^ G	#38-65
LSSW		002222 G	38-65 40-143 #40-143
LSTEST		002114 G	#38-65
LSTIML		002014 G	#38-65
LSUNIT		002012 G	#38-65 120-1679 125-1997 150-3169 214-6229 227-6930 243-7770 248-8077 258-8619 275-9560 296-10659 296-10666 296-10669 296-10690 301-10979 313-11598 323-12197 342-13204 351-13647
L10000		002220	39-117 #39-134
L10001		002262	40-143 #40-192
L10002		002330	41-256 #41-274
L10003		002646	42-300 #43-337
MARPAT		002246 G	#40-165 120-1681 275-9549 275-9552
MLB10	=	007114	#122-1810 175-4425
MLB11	=	007122	#122-1811 176-4439
MLB12	=	007126	#122-1812 176-4447
MLB13	=	007132	#122-1813 176-4455
MLB14	=	007136	#122-1814 176-4463
MLB15	=	007142	#122-1815 176-4471
MLB16	=	007146	#122-1816 176-4479
MLB17	=	007152	#122-1817 177-4493
MLB18	=	007156	#122-1818 177-4501

SYMBOL	CROSS REFERENCE	VALUE	REFERENCES
MLB19		= 007164	#122-1819 177-4509
MLB2		= 007052	#122-1802 174-4353
MLB20		= 007172	#122-1820 173-4278
MLB21		= 007176	#122-1821 174-4318
MLB22		= 007202	#122-1822 173-4286
MLB23		= 007206	#122-1823 173-4294
MLB24		= 007212	#122-1824 174-4326
MLB3		= 007056	#122-1803 174-4361
MLB4		= 007062	#122-1804 174-4369
MLB5		= 007066	#122-1805 175-4383
MLB6		= 007072	#122-1806 173-4304 175-4393
MLB7		= 007076	#122-1807 175-4401
MLB8		= 007102	#122-1808 175-4409
MLB9		= 007106	#122-1809 175-4417
ML.REG	032364	G	#119-1646 *130-2227 131-2300 132-2311 132-2314 132-2320 132-2341 137-2575 137-2576
			137-2577 138-2610 138-2631 138-2641 140-2748 142-2834 144-2900 144-2906 145-2973
			145-2977 147-3053 147-3057 171-4191 171-4194 171-4196 171-4198 171-4201 172-4207
			172-4209 172-4211 172-4213 172-4216 172-4221 172-4223 172-4225 172-4227 172-4229
			172-4231 172-4234 172-4236 172-4238 172-4241 172-4243 172-4245 172-4248 172-4253
			172-4255 173-4276 173-4284 173-4292 173-4300 173-4312 174-4324 174-4351 174-4359
			174-4367 175-4381 175-4389 175-4399 175-4407 175-4415 175-4423 176-4437 176-4445
			176-4453 176-4461 176-4469 176-4477 177-4491 177-4499 177-4507 184-4845 184-4846
			184-4851 184-4854 184-4855 184-4859 184-4860 184-4861 184-4864 185-4871 185-4872
			185-4873 185-4874 185-4875 185-4880 185-4885 217-6416 218-6434 218-6435 218-6454
			218-6457 220-6564 220-6578 220-6579 221-6602 221-6605 231-7152 231-7166 231-7167
			232-7190 232-7193 247-7981 247-7995 247-7996 247-8015 247-8018 251-8238 251-8252
			251-8253 252-8276 252-8279 262-8838 262-8852 262-8853 263-8876 263-8879 280-9801
			280-9815 280-9816 280-9835 280-9838 285-10080 285-10094 285-10095 285-10114 285-10117
			290-10360 290-10374 290-10375 290-10394 290-10397 305-11198 305-11212 306-11217 306-11236
			306-11239 317-11828 317-11842 317-11843 317-11862 317-11865 327-12430 328-12448 328-12449
			328-12468 328-12471 338-13014 338-13028 338-13029 339-13052 339-13055 342-13226 342-13227
			342-13228 351-13655 351-13656 351-13657 354-13811 354-13821 354-13856 355-13870
MSGU		= 010424	#123-1871 185-4911
MSG1		= 010512	#123-1872 215-6281 215-6291 215-6300 216-6348 217-6374 217-6384 217-6393 219-6519
			220-6533 220-6542 228-6990 228-7001 229-7014 230-7081 230-7103 230-7114 231-7129
			244-7826 244-7836 244-7845 245-7909 246-7935 246-7945 246-7954 250-8170 250-8192
			251-8206 251-8215 259-8676 259-8687 260-8700 261-8767 261-8789 261-8800 262-8813
			277-9628 277-9639 277-9648 278-9727 279-9753 279-9764 279-9773 281-9901 282-9916
			282-9925 283-10005 284-10031 284-10042 284-10051 286-10180 287-10195 287-10204 288-10285
			289-10311 289-10322 289-10331 302-11036 303-11051 303-11060 304-11127 304-11149 305-11164
			305-11173 314-11673 314-11683 314-11692 315-11757 316-11782 316-11792 316-11801 325-12279
			325-12289 325-12298 326-12363 327-12388 327-12398 327-12407 335-12859 335-12869 336-12882
			337-12947 337-12968 337-12978 338-12991
MSG2		= 010542	#123-1873 146-2987 148-3067 217-6404 220-6553 231-7139 246-7965 251-8226 262-8825
			279-9785 284-10063 289-10343 305-11185 316-11812 327-12418 338-13002 354-13815 354-13825
			355-13864 355-13874
MSG3		= 010610	#123-1874 219-6485 219-6492 221-6629 221-6636 232-7217 232-7224 248-8046 248-8053
			252-8303 252-8310 263-8903 263-8910 281-9866 281-9873 286-10145 286-10152 291-10425
			291-10432 306-11263 307-11274 318-11893 318-11900 329-12499 329-12506 339-13079 339-13086
MSG4		= 010624	#123-1875 218-6471 221-6619 232-7207 248-8036 252-8293 263-8893 281-9856 286-10135
			291-10415 306-11253 318-11883 328-12485 339-13069
MSG5		= 010640	#123-1876 216-6327 229-7055 245-7888 250-8149 260-8741 278-9705 283-9983 288-10263

CZMLBA SYMBOL	CROSS REFERENCE VALUE	REFERENCES
NUM.DR	032474 G	304-11105 315-11736 326-12342 336-12922
ONEFIL	= 000001	82-177 *83-210 #119-1657 *126-2023 150-3185 *150-3190 334-12801 342-13203
ONLY	002232 G	#2-29 36-1662 37-1663 38-34 40-194 41-220 44-426
OPT1	046316	#40-159 120-1680 139-2665 139-2675
OPT2	050232	#227-6921 345-13357
OPT3	053560	#242-7728 345-13360
OPT4	055472	#258-6608 345-13363
OPT5	072404	#275-9541 345-13368
OSAPTS	= 000001	#342-13195 345-13371
OSAU	= 000001	#38-13 #38-46 38-65
OSBGNR	= 000001	#38-13 #38-46 38-65
OSBGNS	= 000001	#38-13 #38-46 38-65
OSDU	= 000001	#38-13 #38-46 38-65
OSERRT	= 000001	#38-13 #38-46 38-65
OSGNSW	= 000001	#38-13 #38-46 38-65
OSPOIN	= 000001	#38-13 #38-46 38-46 38-65
OSSETU	= 000001	#38-13 #38-46 38-65
PATTBL	032536 G	#119-1663 156-3520 156-3521
PATTER	030714	#118-1600 *154-3398 154-3399 *154-3401 161-3695 *242-7736 242-7747 242-7749 243-7760
PHR1	= 007336	243-7768 #122-1835 185-4902 198-5409 203-5667
PHR10	= 007542	#123-1848 173-4269 174-4343
PHR11	= 007562	#123-1849 178-4543
PHR12	= 007624	#123-1850 350-13620
PHR13	= 007662	#123-1851 351-13641
PHR14	= 007710	#123-1852 139-2700 140-2729
PHR15	= 007726	#123-1853 140-2731
PHR16	= 007744	#123-1854 139-2698
PHR17	= 007764	#123-1855 350-13611
PHR18	= 010010	#123-1856 354-13836
PHR19	= 010032	#123-1857 350-13625
PHR2	= 007352	#122-1836 345-13347 357-13971
PHR20	= 010052	#123-1858 350-13631
PHR3	= 007370	#122-1837 130-2211
PHR4	= 007412	#122-1838 141-2775
PHR5	= 007426	#122-1839 353-13787
PHR6	= 007450	#122-1840 355-13885
PHR7	= 007472	#122-1841 132-2328 137-2583 351-13663
PHR8	= 007510	#123-1846 132-2342 132-2352
PHR9	= 007522	#123-1847 242-7750 243-7763
PNT	= 001000 G	#121-1736
PRI	= 002000 G	#121-1737
PRI00	= 000000 G	#120-1724
PRI01	= 000040 G	#120-1723
PRI02	= 000100 G	#120-1722
PRI03	= 000140 G	#120-1721
PRI04	= 000200 G	#120-1720
PRI05	= 000240 G	#120-1719
PRI06	= 000300 G	#120-1718
PRI07	= 000340 G	#120-1717
PTABLE	032440 G	#119-1647 132-2324 137-2572 137-2579 *150-3184 184-4842 184-4856 342-13223 351-13652



SYMBOL	VALUE	REFERENCES	
		351-13659	
P.AAA	005550	#107-965	121-1750
P.AAB	005564	#107-969	121-1751
P.AAC	005602	#107-974	121-1752
P.AAD	005610	#107-976	121-1753
P.AAE	005640	#107-984	121-1754
P.AAF	005674	#107-994	121-1755
P.AAG	005712	#107-999	121-1756
P.AAH	005742	#108-1011	121-1757
P.AAI	006020	#108-1027	121-1758
P.AAJ	006070	#108-1041	121-1759
P.AAK	006110	#108-1047	121-1760
P.AAL	006156	#109-1064	121-1761
P.AAM	006206	#109-1072	121-1762
P.AAN	006262	#109-1087	121-1763
P.AAO	006276	#109-1091	121-1764
P.AAP	006332	#109-1101	121-1765
P.AAQ	006400	#110-1118	121-1766
F.AAR	006446	#110-1131	121-1767
P.AAS	006470	#110-1137	121-1768
P.AAT	006502	#110-1141	121-1769
P.AAU	006510	#110-1143	121-1770
P.AAV	006514	#110-1145	121-1771
P.AAW	006522	#110-1147	121-1772
P.AAX	006534	#110-1151	121-1773
P.AAY	006552	#110-1156	121-1774
P.AAZ	006574	#110-1162	121-1775
P.ABA	006622	#111-1174	121-1776
P.ABB	006630	#111-1176	121-1777
P.ABC	006634	#111-1178	121-1778
P.ABD	006642	#111-1180	121-1779
P.ABE	006652	#111-1183	121-1780
P.ABF	006662	#111-1186	121-1781
P.ABG	006670	#111-1188	121-1782
P.ABH	006700	#111-1191	121-1783
P.ABI	006706	#111-1193	121-1784
P.ABJ	006714	#111-1195	121-1785
P.ABK	006722	#111-1197	122-1790
P.ABL	006734	#111-1201	122-1791
P.ABM	006744	#111-1204	122-1792
P.ABN	006754	#111-1207	122-1793
P.ABO	006760	#111-1209	122-1794
P.ABP	006766	#111-1211	122-1795
P.ABQ	006776	#111-1214	122-1796
P.ABR	007012	#111-1218	122-1797
P.ABS	007022	#111-1221	122-1798
P.ABT	007034	#111-1225	122-1799
P.ABU	007040	#112-1231	122-1800
P.ABV	007046	#112-1233	122-1801
P.ABW	007052	#112-1235	122-1802
P.ABX	007056	#112-1237	122-1803
P.ABY	007062	#112-1239	122-1804

SYMBOL	CROSS REFERENCE	VALUE	REFERENCES
P.ABZ		007066	#112-1241 122-1805
P.ACA		007072	#112-1243 122-1806
P.ACB		007076	#112-1245 122-1807
P.ACC		007102	#112-1247 122-1808
P.ACD		007106	#112-1249 122-1809
P.ACE		007114	#112-1251 122-1810
P.ACF		007122	#112-1253 122-1811
P.ACG		007126	#112-1255 122-1812
P.ACH		007132	#112-1257 122-1813
P.ACI		007136	#112-1259 122-1814
P.ACJ		007142	#112-1261 122-1815
P.ACK		007146	#112-1263 122-1816
P.ACL		007152	#112-1265 122-1817
P.ACM		007156	#112-1267 122-1818
P.ACN		007164	#112-1269 122-1819
P.ACO		007172	#112-1271 122-1820
P.ACP		007176	#112-1273 122-1821
P.ACQ		007202	#112-1275 122-1822
P.ACR		007206	#112-1277 122-1823
P.ACS		007212	#112-1279 122-1824
P.ACT		007216	#112-1281 122-1825
P.ACU		007250	#113-1294 122-1826
P.ACV		007256	#113-1296 122-1827
P.ACW		007264	#113-1298 122-1828
P.ACX		007272	#113-1300 122-1829
P.ACY		007300	#113-1302 122-1830
P.ACZ		007306	#113-1304 122-1831
P.ADA		007314	#113-1306 122-1832
P.ADB		007322	#113-1308 122-1833
P.ADC		007330	#113-1310 122-1834
P.ADD		007336	#113-1312 122-1835
P.ADE		007352	#113-1316 122-1836
P.ADF		007370	#113-1321 122-1837
P.ADG		007412	#113-1327 122-1838
P.ADH		007426	#113-1331 122-1839
P.ADI		007450	#113-1337 122-1840
P.ADJ		007472	#114-1347 122-1841
P.ADK		007510	#114-1352 123-1846
P.ADL		007522	#114-1356 123-1847
P.ADM		007542	#114-1362 123-1848
P.ADN		007562	#114-1368 123-1849
P.ADO		007624	#114-1380 123-1850
P.ADP		007662	#114-1390 123-1851
P.ADQ		007710	#115-1402 123-1852
P.ADR		007726	#115-1407 123-1853
P.ADS		007744	#115-1412 123-1854
P.ADT		007764	#115-1418 123-1855
P.ADU		010010	#115-1425 123-1856
P.ADV		010032	#115-1431 123-1857
P.ADW		010052	#115-1437 123-1858
P.ADX		010066	#115-1441 123-1859
P.ADY		010070	#115-1442 123-1860

CZMLBA  
 SYMBOL CROSS REFERENCE  
 SYMBOL VALUE  
 P.ADZ 010072  
 P.AEA 010076  
 P.AEB 010102  
 P.AEC 010124  
 P.AED 010150  
 P.AEE 010224  
 P.AEF 010300  
 P.AEG 010332  
 P.AEH 010356  
 P.AEI 010400  
 P.AEJ 010424  
 P.AEK 010512  
 P.AEL 010542  
 P.AEM 010610  
 P.AEN 010624  
 P.AEO 010640  
 QH1 002330  
 QH2 002345  
 QH3 002367  
 QH4 002417  
 QS1 002646  
 QS10 003513  
 QS11 003532  
 QS12 003614  
 QS13 003633  
 QS14 003665  
 QS15 003717  
 QS16 003764  
 QS2 002713  
 QS3 003000  
 QS4 003033  
 QS5 003065  
 QS6 003132  
 QS7 003436  
 QS8 003455  
 QS9 003474  
 QUICK 030712  
 RANDOM 005352  
 RAND1 063116  
 RAND2 065022  
 RAND3 066670  
 RAND4 070562  
 RANGE 002224  
 RBUFF 020706  
 RCBUFF = 021706  
 RDBUFF = 020706  
 RD.CO 032340  
 RD.MIL 032344  
 RD.THO 032342  
 READ 042434

CREATED BY MACRO ON 24-OCT-80 AT 10:07  
 REFERENCES  
 #115-1443 123-1861  
 #115-1445 123-1862  
 #115-1447 123-1863  
 #116-1457 123-1864  
 #116-1464 123-1865  
 #116-1479 123-1866  
 #116-1494 123-1867  
 #116-1503 123-1868  
 #117-1514 123-1869  
 #117-1520 123-1870  
 #117-1527 123-1871  
 #117-1545 123-1872  
 #117-1553 123-1873  
 #118-1570 123-1874  
 #118-1574 123-1875  
 #118-1578 123-1876  
 41-269 #41-283  
 41-270 #41-284  
 41-271 #41-285  
 41-272 #41-286  
 42-311 #44-345  
 43-327 #44-360  
 43-322 43-329 #44-361  
 43-330 #44-362  
 43-331 #44-363  
 43-332 #44-364  
 43-333 #44-365  
 43-334 #44-366  
 42-313 #44-346  
 42-315 #44-347  
 42-316 #44-348  
 42-318 #44-349  
 42-319 #44-350  
 43-324 #44-357  
 43-325 #44-358  
 43-326 #44-359  
 #118-1599 \*150-3167 \*151-3215 242-7737 248-8074 345-13340 345-13364  
 \*77-27 #77-37 120-1683 165-3903 199-5470 199-5472 \*292-10499 292-10500 \*294-10577  
 294-10581 \*296-10564 296-10665 \*296-10668 296-10672 296-10678 296-10686 296-10688 \*296-10692  
 #301-10969 343-13240  
 #312-11585 343-13242  
 #323-12188 343-13244  
 #334-12791 343-13251  
 #40-155 120-1679 139-2661  
 #118-1596 121-1745 121-1746 121-1748 216-6313 216-6319 216-6360 218-6448 228-6953  
 243-7789 249-8100 259-8642 302-11002 313-11621 324-12221 334-12810  
 #121-1746 275-9559  
 #121-1745 275-9558  
 #119-1625 \*126-2028 189-5036 \*189-5037 \*189-5041  
 #119-1629 \*126-2030 \*189-5045 \*189-5047 350-13632  
 #119-1627 \*126-2029 \*189-5040 189-5042 \*189-5044  
 #195-5279 199-5474 203-5674 204-5714 204-5726 216-6315 216-6357 218-6445 229-7025





CZMLBA		CREATED BY MACRO ON 24-OCT-80 AT 10:07		PAGE 15		H 10		SEQ 0332	
SYMBOL	CROSS REFERENCE	REFERENCES	CREF	V01					
SYMBOL	VALUE								
		43-333	43-333	43-333	43-333	43-333	43-333	43-333	43-334
		43-334	43-334	43-334	43-334	43-334	43-334	43-334	43-337
		43-337	43-337						
SVCSUB	= 000001	#38-13	#38-21						
SVCTAG	= 000001	#38-13	#38-23	39-134	40-192	41-274	43-337		
SVCTST	= 000001	#38-13	#38-20						
SYSERR	037000	#171-4184	185-4918						
SLSYM	= 010000	#38-13	#39-134	#40-192	#41-274	#43-337			
TOP.SE	032516	G #119-1661	138-2638	215-6257	228-6957	228-6963	243-7793	243-7796	249-8104
		259-8646	259-8649	276-9596	293-10568	294-10578	302-11006	302-11009	313-11622
		335-12830	335-12832						324-12224
TRIES	031732	#118-1614	*126-2015	*205-5784	353-13782	355-13896			
TRT00	= 010066	#123-1859	140-2752						
TRT01	= 010070	#123-1860	140-2756						
TRT10	= 010072	#123-1861	140-2760						
TRT11	= 010076	#123-1862	140-2764						
TSECT	002230	G #40-158	120-1680	*139-2672	*139-2673	*139-2680	*139-2681	139-2682	139-2684
		140-2713	140-2728						139-2699
T\$ARGC	= 000002	#38-65	38-65	#38-65	38-65	38-65	#38-65	38-65	38-65
		38-65	38-65	#38-65	38-65	38-65	#38-65	38-65	#38-65
T\$CODE	= 017130	#41-269	41-269	#41-269	41-269	#41-269	41-269	#41-270	41-270
		41-270	#41-270	41-270	#41-271	41-271	#41-271	41-271	#41-271
		#41-272	41-272	#41-272	41-272	#41-272	41-272	#42-311	42-311
		42-311	#42-311	42-311	#42-312	42-312	42-312	#42-312	42-312
		#42-312	42-312	#42-312	42-312	#42-313	42-313	#42-313	42-313
		42-313	#42-314	42-314	42-314	#42-314	42-314	42-314	#42-314
		#42-314	42-314	#42-315	42-315	#42-315	42-315	#42-315	42-315
		42-316	#42-316	42-316	#42-316	42-316	#42-317	42-317	42-317
		42-317	42-317	#42-317	42-317	#42-317	42-317	#42-318	42-318
		42-318	#42-318	42-318	#42-319	42-319	#42-319	42-319	#42-319
		#42-320	42-320	42-320	#42-320	42-320	42-320	#42-320	42-320
		42-320	#43-322	43-322	#43-322	43-322	#43-322	43-322	#43-323
		43-323	#43-323	43-323	43-323	#43-323	43-323	#43-323	43-323
		43-324	#43-324	43-324	#43-324	43-324	#43-325	43-325	#43-325
		#43-325	43-325	#43-326	43-326	#43-326	43-326	#43-326	43-326
		43-327	#43-327	43-327	#43-327	43-327	#43-328	43-328	43-328
		43-328	43-328	#43-328	43-328	#43-328	43-328	#43-329	43-329
		43-329	#43-329	43-329	#43-330	43-330	#43-330	43-330	#43-330
		#43-331	43-331	#43-331	43-331	#43-331	43-331	#43-332	43-332
		43-332	#43-332	43-332	#43-333	43-333	#43-333	43-333	#43-333
		#43-334	43-334	#43-334	43-334	#43-334	43-334	43-334	43-333
T\$ERRN	= 000000	#38-13							
T\$EXCP	= 000000	#41-269	41-269	#41-270	41-270	#41-271	41-271	#41-272	41-272
		42-315	#42-316	42-316	#42-318	42-318	#43-322	43-322	#43-329
T\$FREE	075012	G 358-14030	358-14031	#358-14032					
T\$GMAN	= 000000	#38-13							
T\$HILI	= 000012	#41-269	41-269	#41-270	41-270	#41-271	41-271	#41-272	41-272
		42-315	#42-316	42-316	#42-318	42-318	#43-322	43-322	#43-329
T\$LAST	= 000000	#38-13							
T\$LOLI	= 000001	#41-269	41-269	#41-270	41-270	#41-271	41-271	#41-272	41-272
		42-315	#42-316	42-316	#42-318	42-318	#43-322	43-322	#43-329
T\$LSYM	= 010000	#38-13	38-13	39-134	40-192	41-274	43-337		

CZMLBA  
SYMBOL CROSS REFERENCE  
SYMBOL VALUE  
T\$NEST = 177777

CREATED BY MACRO ON 24-OCT-80 AT 10:07

PAGE 16  
CREF V01

I 10

SEQ 0333

REFERENCES

T\$NSO = 000000  
T\$NS1 = 000021

T\$PTHV = 000000 G  
T\$PTNU = 000000  
T\$SAVL = 177777  
T\$SEGL = 177777  
T\$SUBN = 000000  
T\$TAGL = 177777  
T\$TAGN = 010005

T\$TEMP = 000000

#38-13	38-39	#38-39	38-39	39-117	#39-117	39-117	39-134	39-134
39-134	#39-134	40-143	#40-143	40-143	40-192	40-192	40-192	#40-192
41-256	#41-256	41-256	41-274	41-274	41-274	#41-274	42-300	#42-300
42-300	42-312	42-314	42-317	42-320	43-323	43-328	43-337	43-337
43-337	#43-337	44-376	#44-376	44-376	44-382	44-382	44-382	#44-382
44-408	44-408	44-408	#44-408					
#38-39	44-408							
#39-117	39-134	#40-143	40-192	#41-256	41-274	#42-300	42-312	42-314
42-317	42-320	43-323	43-328	43-337	#44-376	44-382		
38-65	#358-14036							
#38-13								
#38-13								
#38-13								
#38-13								
#38-13	39-117	39-117	#39-117	40-143	40-143	#40-143	41-256	41-256
#41-256	42-300	42-300	#42-300	44-376	44-376	#44-376		
#39-101	39-101	39-101	#39-101	#39-134	39-134	#40-192	40-192	#41-269
41-269	#41-269	41-269	#41-269	41-269	#41-270	41-270	#41-270	41-270
#41-270	41-270	#41-271	41-271	#41-271	41-271	#41-271	41-271	#41-272
41-272	#41-272	41-272	#41-272	41-272	#41-274	41-274	#42-311	42-311
#42-311	42-311	#42-311	42-311	#42-313	42-313	#42-313	42-313	#42-313
42-313	#42-315	42-315	#42-315	42-315	#42-315	42-315	#42-316	42-316
#42-316	42-316	#42-316	42-316	#42-318	42-318	#42-318	42-318	#42-318
42-318	#42-319	42-319	#42-319	42-319	#42-319	42-319	#43-322	43-322
#43-322	43-322	#43-322	43-322	#43-324	43-324	#43-324	43-324	#43-324
43-324	#43-325	43-325	#43-325	43-325	#43-325	43-325	#43-326	43-326
#43-326	43-326	#43-326	43-326	#43-327	43-327	#43-327	43-327	#43-327
43-327	#43-329	43-329	#43-329	43-329	#43-329	43-329	#43-330	43-330
#43-330	43-330	#43-330	43-330	#43-331	43-331	#43-331	43-331	#43-331
43-331	#43-332	43-332	#43-332	43-332	#43-332	43-332	#43-333	43-333
#43-333	43-333	#43-333	43-333	#43-334	43-334	#43-334	43-334	#43-334
43-334	#43-337	43-337	#44-382	44-382	#44-408	44-408		

T\$TEST = 000000  
T\$TSTM = 177777  
T\$TSTS = 000000  
T\$\$HAR = 010002  
T\$\$HW = 010000  
T\$\$PRO = 010004  
T\$\$SOF = 010003  
T\$\$SW = 010001  
T1 = 073024 G  
UAM = 000200 G  
UP.HAR = 035302  
  
UP.RD. = 042062  
UP.SOF = 035430  
  
UP.WC. = 042154  
UP.WR. = 041770  
VEC = 030726  
WAITER = 041320

#38-13								
#38-13								
#38-13								
#41-256	41-256	41-274						
#39-117	39-117	39-134						
#44-376								
#42-300	42-300	43-337						
#40-143	40-143	40-192						
39-101	#346-13394							
#121-1734								
#145-2963	219-6479	221-6623	232-7211	248-8040	252-8297	263-8897	281-9860	286-10139
291-10419	306-11257	318-11887	329-12493	339-13073				
#189-5033	196-5296							
#147-3043	219-6496	221-6640	233-7232	248-8057	253-8318	263-8914	281-9877	286-10156
291-10436	307-11278	318-11904	329-12510	339-13090				
#191-5096	198-5397							
#187-4970	194-5195							
#118-1609	*129-2177	130-2236						
#181-4688	184-4850	185-4884						

SYMBOL CROSS REFERENCE		VALUE	REFERENCES								
WBUFF	=	010706	#118-1595	121-1743	121-1744	121-1747	*153-3327	*153-3340	161-3697	*163-3772	*163-3773
			*163-3780	*165-3896	*165-3897	*165-3898	*165-3903	206-5846	215-6262	215-6272	216-6318
			219-6500	219-6510	221-6596	228-6952	243-7788	249-8099	259-8641	302-11001	313-11620
			324-12220	334-12809							
WCBUFF	=	011706	#121-1744	275-9554	275-9557						
WC.COU	=	032346	#119-1631	*126-2031	191-5099	*191-5100	*191-5104				
WC.MIL	=	032352	#119-1635	*126-2033	*191-5108	*191-5110	351-13642				
WC.THO	=	032350	#119-1633	*126-2032	*191-5103	191-5105	*191-5107				
WDBUFF	=	010706	#121-1743	275-9550	275-9556						
WHY.DR	=	032464	#119-1655	*126-2058	*138-2603	*138-2626	*139-2705	*140-2736	*215-6278	*215-6288	*215-6302
			*216-6345	*217-6371	*217-6381	*217-6395	*217-6406	*219-6516	*219-6526	*220-6544	*220-6555
			*228-6987	*228-6998	*229-7017	*230-7078	*230-7100	*230-7111	*231-7126	*231-7142	*244-7823
			*244-7833	*244-7847	*245-7906	*246-7932	*246-7942	*246-7956	*246-7967	*250-8167	*250-8189
			*251-8203	*251-8217	*251-8228	*259-8673	*259-8684	*260-8703	*261-8764	*261-8786	*261-8797
			*262-8816	*262-8828	*276-9621	*277-9636	*277-9651	*278-9724	*279-9750	*279-9761	*279-9776
			*279-9788	*281-9898	*282-9913	*282-9928	*283-10002	*284-10028	*284-10039	*284-10054	*284-10066
			*286-10177	*287-10192	*287-10207	*288-10282	*289-10308	*289-10319	*289-10334	*289-10346	*302-11033
			*302-11044	*303-11063	*304-11124	*304-11146	*305-11161	*305-11176	*305-11188	*314-11670	*314-11680
			*314-11694	*315-11754	*316-11779	*316-11789	*316-11803	*316-11814	*325-12276	*325-12286	*325-12300
			*326-12360	*327-12385	*327-12395	*327-12409	*327-12420	*335-12856	*335-12866	*336-12884	*337-12944
			*337-12965	*337-12975	*338-12993	*338-13004	352-13697				
WPTR	=	030706	#118-1597	206-5843	*206-5846	206-5847	*228-6952	228-6970	228-6980	229-7035	229-7038
			229-7046	233-7236	*233-7237	*243-7788	243-7803	244-7817	245-7869	245-7872	245-7879
			248-8061	*248-8062	*249-8099	249-8126	249-8129	249-8136	253-8327	*253-8328	*259-8641
			259-8656	259-8666	260-8721	260-8724	260-8732	264-8922	*264-8923	*302-11001	302-11016
			302-11026	303-11081	303-11084	303-11092	307-11282	*307-11283	*313-11620	314-11654	314-11664
			315-11716	315-11719	315-11727	*318-11907	*324-12220	324-12256	325-12270	325-12318	326-12325
			326-12333	328-12462	*329-12512	*334-12809	335-12840	335-12850	336-12902	336-12905	336-12913
			*339-13092								
WRD11	=	006662	#121-1781	132-2326	137-2581	137-2589	138-2616	139-2687	140-2718	351-13661	351-13687
WRD15	=	006670	#121-1782	144-2907	185-4904	194-5206	196-5307	198-5408			
WRD16	=	006700	#121-1783	185-4896	194-5207	203-5657					
WRD17	=	006706	#121-1784	185-4899	196-5308	203-5677					
WRD18	=	006714	#121-1785	202-5652	203-5666	203-5676					
WRD19	=	006722	#122-1790	204-5735			204-5736	204-5756			
WRD2	=	006622	#121-1776	203-5658	203-5668	203-5678	345-13348				
WRD20	=	006734	#122-1791	144-2908	204-5755						
WRD21	=	006744	#122-1792	137-2588	138-2615	139-2686	140-2717	351-13686			
WRD24	=	006754	#122-1793	276-9587	277-9659	287-10217					
WRD25	=	006760	#122-1794	282-9938							
WRD3	=	006630	#121-1777	357-13972	357-13980						
WRD34	=	006766	#122-1795	130-2200	214-6221	227-6924	242-7731	258-8611	275-9544	342-13197	
WRD35	=	006776	#122-1796	276-9582							
WRD36	=	007012	#122-1797	129-2187	130-2207						
WRD37	=	007022	#122-1798	129-2185	130-2209						
WRD38	=	007034	#122-1799	129-2194							
WRD4	=	006634	#121-1778	345-13346	357-13970	357-13979					
WRD40	=	007040	#122-1800	129-2195	130-2212						
WRD41	=	007046	#122-1801	129-2192							
WRD6	=	006642	#121-1779	138-2633							
WRD7	=	006652	#121-1780	138-2636							
WRITE	=	042246	#193-5178	202-5650	203-5687	203-5693	203-5702	204-5721	215-6264	215-6269	228-6972



SYMBOL	CROSS REFERENCE VALUE	REFERENCES
		228-6977 243-7805 244-7814 259-8658 259-8663 276-9606 276-9611 281-9883 281-9888
		286-10162 286-10167 302-11018 302-11023 314-11656 314-11661 324-12258 325-12267 335-12842
		335-12847
WR.COU	032332	#118-1615 *126-2025 187-4973 *187-4974 *187-4978
WR.MIL	032336	#119-1623 *126-2027 *187-4982 *187-4984 350-13626
WR.THO	032334	#118-1617 *126-2026 *187-4977 187-4979 *187-4981
X\$ALWA	= 000000	#38-13 42-317 43-323
X\$FALS	= 000040	#38-13 42-312 42-314
X\$OFFS	= 000400	#38-13 42-312 42-314 42-317 42-320 43-323 43-328
X\$TRUE	= 000020	#38-13 42-320 43-328
\$END.L	075014 G	#358-14043
\$PATCH	004022 G	#44-399
\$SAVE2	005150 G	69-1027 74-1274 #76-1350 156-3486 165-3895 193-5178 195-5279 197-5380 359-14067
\$SAVE3	005164 G	43-170 49-218 53-386 54-440 64-833 65-884 #76-1357 293-10563 359-14066
\$SAVE4	005202 G	#76-1365 125-1996 129-2174 150-3160 153-3313 184-4838 296-10658 342-13195 359-14066
\$SAVE5	005222 G	70-1096 74-1274 #76-1374 131-2299 137-2568 159-3615 171-4184 180-4630 202-5640
		214-6218 227-6921 242-7728 258-8608 275-9541 301-10969 312-11585 323-12188 334-12791
		350-13602 359-14066
\$T1	072650	#345-13340 346-13395

MACRO CROSS REFERENCE

MACRO NAME	REFERENCES									
BGNHRD	41-256									
BGNHW	#39-117									
BGNMOD	38-39									
BGNPRO	44-376									
BGNSFT	#42-300									
BGNSW	#40-143									
DESCRI	#39-85									
DEVTYP	#38-80									
DISPAT	39-101									
ENDHRD	#41-274									
ENDHW	39-134									
ENDMOD	44-408									
ENDPRO	#44-382									
ENDSFT	43-337									
ENDSW	40-192									
ERRTBL	39-94									
GPRMA	#41-269	#41-270								
GPRMD	41-271	41-272	42-315	42-316	42-318	43-322	43-329			
GPRML	#42-311	#42-313	#42-319	#43-324	#43-325	#43-326	#43-327	#43-330	#43-331	#43-332
	#43-333	#43-334								
HEADER	38-65									
MSBYTE	#38-65	#38-65	#38-65	#38-65						
MSCNTO	#41-269	41-269	#41-270	41-270	#41-271	41-271	#41-272	41-272	#42-311	42-311
	#42-313	42-313	#42-315	42-315	#42-316	42-316	#42-318	42-318	#42-319	42-319
	#43-322	43-322	#43-324	43-324	#43-325	43-325	#43-326	43-326	#43-327	43-327
	#43-329	43-329	#43-330	43-330	#43-331	43-331	#43-332	43-332	#43-333	43-333
	#43-334	43-334								
MSDATA	#38-65	38-65	38-65	38-65	38-65	38-65	38-65	38-65	38-65	38-65
	38-65	38-65	38-65	38-65	38-65	38-65	#38-65	38-65	38-65	38-65
	38-65	38-65	38-65	38-65	38-65	38-65	38-65	38-65	38-65	38-65
	38-65	38-65	38-65	38-65	38-65	38-65	38-65	38-65	38-65	#38-80
	38-80	#39-85	39-85							
MSDECLR	#39-134	39-134	#40-192	40-192	#41-274	41-274	#43-337	43-337	#44-382	44-382
	#44-408	44-408								
MSDEFA	#41-269	#41-269	#41-270	#41-270	#41-271	#41-271	#41-272	#41-272	#42-311	#42-311
	#42-313	#42-313	#42-315	#42-315	#42-316	#42-316	#42-318	#42-318	#42-319	#42-319
	#43-322	#43-322	#43-324	#43-324	#43-325	#43-325	#43-326	#43-326	#43-327	#43-327
	#43-329	#43-329	#43-330	#43-330	#43-331	#43-331	#43-332	#43-332	#43-333	#43-333
	#43-334	#43-334								
MSENDE	#39-134	#40-192	#41-274	#43-337	#44-408					
MSEXCP	#41-269	41-269	41-269	#41-270	41-270	41-270	#41-271	41-271	41-271	#41-272
	41-272	41-272	#42-315	42-315	42-315	#42-316	42-316	42-316	#42-318	42-318
	42-318	#43-322	43-322	43-322	#43-329	43-329	43-329	43-329		
MSGEN	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65
	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65
	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65
	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65
	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65
	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65
	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65
	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65
	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65
	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65
	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65
	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65	#38-65
	#39-94	#39-94	#39-101	#39-101	#39-117	#39-117	#39-117	#39-117	#39-134	#39-134



