

.REM 8

IDENTIFICATION

PRODUCT CODE: AC-S4288-MC
PRODUCT NAME: CZMLBBO ML11 PERFORMANCE EXERCISER
PRODUCT DATE: 19-MAR-82
MAINTAINER: MEMORY DIAGNOSTICS ENGINEERING
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, 1982 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL
DEC

PDP
DECUS

UNIBUS
DECTAPE

MASSBUS

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
7.0	MAINTENANCE HISTORY

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 ML11A OR ML11B DRIVES
5. 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,

1.4 DIAGNOSTIC HIERARCHY 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:DDDD	EXECUTE DDDDD PASSES (DDDD = 1 TO 64000)
/FLAGS:FLGS	SET SPECIFIED FLAGS (SEE SECTION 2.3)
/EOP:DDDD	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 TIM
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 OF 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
IBE*	INHIBIT ALL ERROR REPORTS EXCEPT FIRST LEVEL (FIRST LEVEL CONTAINS ERROR TYPE, NUMBER, PC, TEST AND UNIT)

IXE*	INHIBIT EXTENDED ERROR REPORTS (THOSE CALLED BY PRINTX MACRO'S)
PRI	DIRECT MESSAGES TO LINE PRINTER
BOE	'BELL' ON ERROR
ISR	INHIBIT STATISTICAL REPORTS
IDU	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 RM 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 ML119},
DEFAULT 0).

4. LAST SECTOR TO TEST (LAST) O?

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) D?

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).

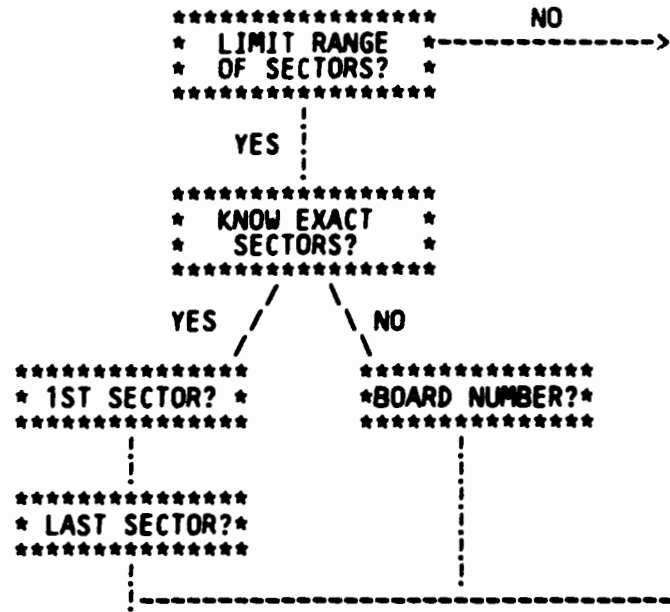
14. ENABLE END OF PASS SUMMARY PRINTOUT (N) L?

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

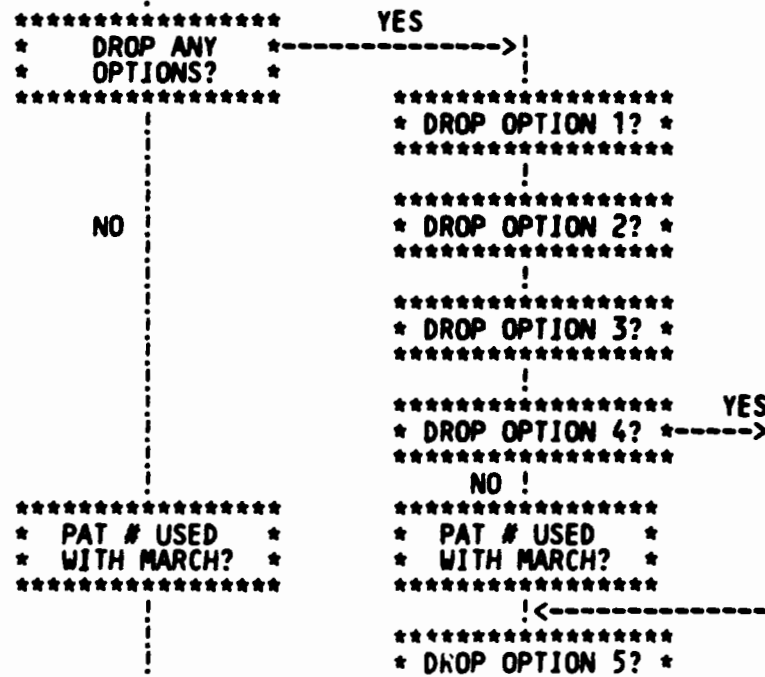
15. ENABLE ERROR PRINTOUTS (N) L?

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)



----->
-----<-----

* ENABL REFRESH *
* MARGINING? *

* DISABLE *
* ECC? *

* ENABLE *
* EOP SUMMARY? *

* ENABLE *
* PRINTOUTS? *

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:

<u>UNIT #</u>	<u>Q-FACTOR</u>
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 (0) ? 160000<CR>
SUB-DEVICE # (0) ? 0<CR>
Q-FACTOR (0) 0 ? 0<CR>

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

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

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

UNIT 4
CSR ADDRESS (0) ? 160000<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 (D) ? 8<CR>

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

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

UNIT 6
CSR ADDRESS (O) ? 160000<CR>
SUB-DEVICE # (O) ? 6,7<CR>
Q-FACTOR (O) 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 (O) ? 160000<CR>
SUB-DEVICE # (O) ? 0-7<CR>
Q-FACTOR (O) 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 ANY 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 FATA_ 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 ERROR DIAGNOSTIC

POSITION	TYPE	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 ****
```

!ERRSOF(310,MSG3,0); !**** OPTION 3 ERROR 10 ****
!ERRSOF(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 ****
!ERRSOF(410,MSG3,0); !**** OPTION 4 ERROR 10 ****
!ERRSOF(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 ****
!ERRSOF(421,MSG3,0); !**** OPTION 4 ERROR 21 ****
!ERRSOF(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 ****
!ERRDF(5309,MSG4,0): !**** OPTION 5, RAND3 ERROR 09 ****
!ERRSFT(5310,MSG3,0): !**** OPTION 5, RAND3 ERROR 10 ****
!ERRSFT(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 ****
!ERRDF(5409,MSG4,0): !**** OPTION 5, RAND4 ERROR 09 ****
!ERRSFT(5410,MSG3,0): !**** OPTION 5, RAND4 ERROR 10 ****
!ERRSFT(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 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)
: 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
: : : SECTOR = LOWEST
: : : WHILE SECTOR LEQ HIGHEST DO
: : : : BEGIN 15 (START OF SECTOR SELECTION LOOP)
: : : : GET WRDCNT
: : : : 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 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 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 M11 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 LUN 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 SELECT'ON)
: : : : 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)
```

7.0 MAINTENANCE HISTORY

MODIFIED BY: D.W. NEALE DATE: 18-FEB-82 VERSION: B

ALL FUNCTIONAL CHANGES TO THIS DIAGNOSTIC ARE LABELED WITH A COMMENT LINE OF 'VER CZMLBB' PRECEEDING ANY MODIFIED OR ADDED LINE/BLOCK OF CODE.

FUNCTIONAL CHANGES TO THIS DIAGNOSTIC INCLUDE:

1. MODIFYING CODE TO ENSURE QUALITY TESTING OF ML-11B AND ML-11A BLOCK MODE MEMORY SYSTEMS.
2. CORRECT EXERCISER FUNCTIONALITY TO CALL OUT ML11 PROM MAINTENANCE PROGRAM AFTER 10 UNIQUE FAILING SINGLE BIT ERRORS (SBE'S) HAVE BEEN DISCOVERED PER ARRAY MODULE.

VERSION 'A' PROM MAINTENANCE CALL OUT OCCURES AFTER ANY TEN SBE'S ARE DISCOVERED PER ARRAY MODULE.
3. PER REQUEST OF F/S AND MEMORY ENGINEERING, THE REPORTING OF 'MDPE' ERRORS DURING SBE'S CORRECTION WILL BE IGNORED.
4. ADDING TO THE REPORT SUMMARY CODE SECTION A TABLE OF SBE LOCATIONS AND PRINTING OF THIS TABLE DURING REPORT CODE PRINTING.

1
33
35 000000
36 002000
38
39 002000
40
41
42
43
44
45
46 002000
47
64
65 002000
002000
002000 103
002001 132
002002 115
002003 114
002004 102
002005 000
002006 000
002007 000
002010
002010 102
002011
002011 060
002012
002012 000000
002014
002014 003410
002016
002016 002266
002020
002020 002450
002022
002022 002210
002024
002024 002222
002026
002026 105050
002030
002030 000000
002032
002032 000000
002034
002034 000001
002036
002036 000000
002040
002040 002204
002042
002042 000000
002044
002044 000000

.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,B,0,1800.,1

LSNAME::
.ASCII /C/
.ASCII /Z/
.ASCII /M/
.ASCII /L/
.ASCII /B/
.BYTE 0
.BYTE 0
.BYTE 0
LSREV::
.ASCII /B/
LSDEPO::
.ASCII /O/
LSUNIT::
.WORD TSPTHV
LSTIML::
.WORD 1800.
LSHPCP::
.WORD LSHARD
LSSPCP::
.WORD LSSOFT
LSHPTP::
.WORD LSHW
LSSPTP::
.WORD LSSW
LSLADP::
.WORD LSLAST
LSSSTA::
.WORD 0
LSCO::
.WORD 0
LSDTYP::
.WORD 1
LSAPT::
.WORD 0
LSDTP::
.WORD LSDISPATCH
LSPRIO::
.WORD 0
LSENV1::
.WORD 0

```

002046
002046 000000
002050
002050 003
002051 003
002052
002052 000000
002054 000000
002056
002056 000000
002060
002060 002122
002062
002062 005474
002064
002064 000000
002066
002066 000000
002070
002070 005652
002072
002072 005640
002074
002074 000000
002076
002076 002130
002100
002100 104035
002102
002102 002172
002104
002104 040722
002106
002106 105034
002110
002110 005506
002112
002112 004126
002114
002114 000000
002116
002116 000000
002120
002120 000000
    
```

66
77
78
79
80

```

002122
002122 115 114 061
002122
    
```

...

NAMES OF DEVICES SUPPORTED BY THIS PROGRAM
DEVTYP <ML11>

```

LSEXP1:: .WORD 0
LSMREV:: .BYTE CS$REVISION
          .BYTE C$EDIT
LSEF:: .WORD 0
        .WORD 0
LSSPC:: .WORD 0
LSDEVP:: .WORD LSDVTYP
LSREPP:: .WORD LSRPT
LSEXP4:: .WORD 0
LSEXP5:: .WORD 0
LSAUT:: .WORD L$AU
LSDUT:: .WORD LSDU
LSLUN:: .WORD 0
L$DESP:: .WORD L$DESC
L$LOAD:: .WORD L$LOAD
          EMT ESLOAD
L$ETP:: .WORD L$ERRTBL
L$ICP:: .WORD L$INIT
L$CCP:: .WORD L$CLEAN
L$ACP:: .WORD L$AUTO
L$PRT:: .WORD L$PROT
L$TEST:: .WORD 0
L$DLY:: .WORD 0
L$HIME:: .WORD 0
    
```

```

LSDVTYP:: .ASCIZ /ML11/
          .EVEN
    
```



```

82      :
83      :      TEST DESCRIPTION
84      :
85      002130      :      DESCRIPT <CZMLBB ML11 PERFORMANCE EXERCISER>
      002130      :
      002130      103      132      115      :      L$DESC::
      :      .ASCIZ /CZMLBB ML11 PERFORM
      :      .EVEN

```

```

86      :
87      :
88      :
89      :
90      :      THE GLOBAL ERROR TABLE (INFORMATION
91      :      USED IN A CALL TO THE MACRO 'ERROR')
92      :
93      :
94      002172      :
      002172      :      ERRTBL
      002172      000000      :      L$ERRTBL::
      002174      000000      :      ERRTYP::      .WORD      0
      002176      000000      :      ERRNBR::      .WORD      0
      002200      000000      :      ERRMSG::      .WORD      0
      :      ERRBLK::      .WORD      0

```

```

95      :
96      :      ++
97      :      THE DISPATCH TABLE CONTAINS THE STARTING ADDRESS OF EACH TEST.
98      :      IT IS USED BY THE SUPERVISOR TO DISPATCH TO EACH TEST.
99      :      --
100     :
101     002202      :      DISPATCH 1
      002202      000001      :
      002204      :      L$DISPATCH::      .WORD      1
      002204      102712      :      .WORD      T1

```

```

102     :
103     :
104     :
105     :
106     :
107     :
108     :
109     :
110     :
111     :      ++
112     :      THE DEFAULT HARDWARE P-TABLE CONTAINS DEFAULT VALUES OF
113     :      THE TEST-DEVICE PARAMETERS. THE STRUCTURE OF THIS TABLE
114     :      IS IDENTICAL TO THE STRUCTURE OF THE HARDWARE P-TABLES,
115     :      AND IS USED AS A 'TEMPLATE' FOR BUILDING THE P-TABLES.
116     :      --

```

```

117     002206      :      BGNHW      DFPTBL
      002206      000004      :      L$HW::      .WORD      L10000-L$HW/2
      002210      :      DFPTBL::
      002210      :

```

```

118     :
119     :
120     :
121     :
122     :
123     :
124     :
125     :
126     :
127     :
128     :
129     002210      176400      :      .WORD      176400      :      CSR ADDRESS
130     002212      000204      :      .WORD      204      :      RH VECTOR ADDRESS
131     002214      000005      :      .WORD      5      :      BR LEVEL FOR INTERRUPT
132     002216      000000      :      .WORD      0      :      ML11 DRIVE NUMBER

```

```

133     :
134     002220      :      ENDHW
      002220      :      L10000:

```

136
137
138
139
140
141
142
143 002220
002220 000021
002222
002222

..**
: 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.
:--

BGNSW SFPTBL

.WORD L10001-L8SW/2
L8SW::
SFPTBL::

144
152
153 002222 000000
154
155 002224 000000
156
157 002226 000000
158 002230 000000
159 002232 000000
160 002234 000000
161 002236 000000
162 002240 000000
163 002242 000000
164 002244 000000
165 002246 000001
166 002250 000000
167 002252 000000
168
169 002254 000000
170

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

187
188 002256 000000
189
190 002260 000000
191
192 002262 000000
193
194
195
196 002264

EOPSUM:: .WORD 0 ;ENABLE EOP SUMMARIES? 0 = NO PRINTOUT
: 1 = YES
: ENABLE ERROR PRINTOUTS? 0 = NO PRINTOUT
ERROUT:: .WORD 0 ; 1 = YES
: ENABLE SOFT ERROR TESTING 0 = NO (DEFAULT)
EFNS21:: .WORD 0 ; 1 = YES
: THIS OPTION IS DESIGNED FOR MEMORY ENGINEERING
: FOR DMT PURPOSES

ENDSW

L10001:

197
222
248
249
250
251
252
253
254
255
256
257
258

;++
: 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.
:--

259 002264
002264 000022
002266

BGNHRD

.WORD L10002-L\$HARD/2
L\$HARD::

260
270
271
272 002266
002266 000031
002270 002332
002272 000000
002274 177777

GPRMA QH1,0,0,0,177777,YES

.WORD T\$CODE
.WORD QH1
.WORD T\$LLOLIM
.WORD T\$HILIM

273 002276
002276 001031
002300 002351
002302 000000
002304 000377

GPRMA QH2,2,0,0,377,YES

.WORD T\$CODE
.WORD QH2
.WORD T\$LLOLIM
.WORD T\$HILIM

274 002306
002306 002032
002310 002374
002312 000007
002314 000001
002316 000007

GPRMD QH3,4,0,7,1,7,YES

.WORD T\$CODE
.WORD QH3
.WORD 7
.WORD T\$LLOLIM
.WORD T\$HILIM

275 002320
002320 003032
002322 002425
002324 000007
002326 000000
002330 000007

GPRMD QH4,6,0,7,0,7,YES

.WORD T\$LODE
.WORD QH4
.WORD 7
.WORD T\$LLOLIM
.WORD T\$HILIM

276
277 002332

ENDHRD

.EVEN
L10002:

278 002332
285
286 002332 103 123 122
287 002351 111 116 124
288 002374 102 122 040
289 002425 104 122 111
290
291
292
293
294

QH1: .ASCIZ /CSR ADDRESS ?/
QH2: .ASCIZ /INTERRUPT VECTOR ?/
QH3: .ASCIZ /BR LEVEL FOR INTERRUPT ?/
QH4: .ASCIZ /DRIVE NUMBER ?/
.EVEN

;++
: THE SOFTWARE PARAMETER CODING SECTION CONTAINS MACROS

```

295      : THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES. THE
296      : MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
297      : INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES. THE
298      : MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
299      : WITH THE OPERATOR.
300      :--
301
302      BGNSFT
303      002446      000106
304      002446
305      002450
306      :--
307      :--
308      :--
309      :--
310      :--
311      :--
312      :--
313      002450      GPRML      QS1,0,1,YES
314      002450      000130
315      002452      002664
316      002454      000001
317      002456      XFERF      6$
318      002456      025044
319      002460      GPRML      QS2,2,1,YES
320      002460      001130
321      002462      002732
322      002464      000001
323      002466      XFERF      5$
324      002466      014044
325      002470      GPRMD      QS3,4,0,177777,0,177777,YES
326      002470      002032
327      002472      003007
328      002474      177777
329      002476      000000
330      002500      177777
331      002502      GPRMD      QS4,6,0,177777,0,177777,YES
332      002502      003032
333      002504      003047
334      002506      177777
335      002510      000000
336      002512      177777
337      002514      XFER      6$
338      002514      006004
339      002516      5$:      GPRMD      QS5,10,D,37,0,15.,YES
340      002516      004052
341      002520      003107
342      002522      000037
343      002524      000000
344      002526      000017
345      002530      6$:      GPRML      QS6,12,1,YES
346      002530      005130
347      002532      003147
348      002534      000001
349      002536      XFERT      7$
350      002536      007024
351      002540      GPRMD      QS11,24,D,77,1,10.,YES
352      002540      012052
353      002542      003554
354      002544      000077
355      002546      000001
356      002550      000012
357      002552      XFER      13$

```

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

.WORD T\$CODE
 .WORD QS1
 .WORD 1
 .WORD T\$CODE
 .WORD T\$CODE
 .WORD QS2
 .WORD 1
 .WORD T\$CODE
 .WORD T\$CODE
 .WORD QS3
 .WORD 177777
 .WORD T\$LOLIM
 .WORD T\$HILIM
 .WORD T\$CODE
 .WORD QS4
 .WORD 177777
 .WORD T\$LOLIM
 .WORD T\$HILIM
 .WORD T\$CODE
 .WORD QS5
 .WORD 37
 .WORD T\$LOLIM
 .WORD T\$HILIM
 .WORD T\$CODE
 .WORD QS6
 .WORD 1
 .WORD T\$CODE
 .WORD T\$CODE
 .WORD QS11
 .WORD 77
 .WORD T\$LOLIM
 .WORD T\$HILIM

325	002552	026004				.WORD	T\$CODE
	002554		7\$:	GPRML	QS7,14,1,YES		
	002554	006130				.WORD	T\$CODE
	002556	003454				.WORD	QS7
	002560	000001				.WORD	1
326	002562			GPRML	QS8,16,1,YES		
	002562	007130				.WORD	T\$CODE
	002564	003474				.WORD	QS8
	002566	000001				.WORD	1
327	002570			GPRML	QS9,20,1,YES		
	002570	010130				.WORD	T\$CODE
	002572	003514				.WORD	QS9
	002574	000001				.WORD	1
328	002576			GPRML	QS10,22,1,YES		
	002576	011130				.WORD	T\$CODE
	002600	003534				.WORD	QS10
	002602	000001				.WORD	1
329	002604			XFERT	12\$		
	002604	006024				.WORD	T\$CODE
330	002606			GPRMD	QS11,24,D,77,1,10.,YES		
	002606	012052				.WORD	T\$CODE
	002610	003554				.WORD	QS11
	002612	000077				.WORD	77
	002614	000001				.WORD	T\$LOLIM
	002616	000012				.WORD	T\$HILIM
331	002620		12\$:	GPRML	QS12,26,1,YES		
	002620	013130				.WORD	T\$CODE
	002622	003637				.WORD	QS12
	002624	000001				.WORD	1
332	002626		13\$:	GPRML	QS13,30,1,YES		
	002626	014130				.WORD	T\$CODE
	002630	003657				.WORD	QS13
	002632	000001				.WORD	1
333	002634			GPRML	QS14,32,1,YES		
	002634	015130				.WORD	T\$CODE
	002636	003724				.WORD	QS14
	002640	000001				.WORD	1
334	002642			GPRML	QS15,34,1,YES		
	002642	016130				.WORD	T\$CODE
	002644	003763				.WORD	QS15
	002646	000001				.WORD	1
335	002650			GPRML	QS16,36,1,YES		
	002650	017130				.WORD	T\$CODE
	002652	004030				.WORD	QS16
	002654	000001				.WORD	1
336	002656			GPRML	QS17,40,1,YES		
	002656	020130				.WORD	T\$CODE
	002660	004066				.WORD	QS17
	002662	000001				.WORD	1
337				.EVEN			
338							
339	002664			ENDSFT			
	002664					.EVEN	

L10003:

```

347 002664 114 111 115 QS1: .ASCIZ /LIMIT RANGE OF SECTORS TO BE TESTED ?/
348 002732 104 117 040 QS2: .ASCIZ /DO YOU KNOW EXACT RANGE OF SECTORS TO TEST ?/
349 003007 106 111 122 QS3: .ASCIZ /FIRST SECTOR TO BE TESTED ?/
350 003047 114 101 123 QS4: .ASCIZ /LAST SECTOR TO BE TESTED ?/
351 003107 127 110 111 QS5: .ASCIZ /WHICH BOARD (0-15) TESTED ?/
352 003147 124 110 105 QS6: .ASCII /THE PROGRAM OPTIONS INCLUDE:/
353 003203 012 015 040 .ASCII <12><15>/ 1. ADDRESS CHECK/
354 003231 012 015 040 .ASCII <12><15>/ 2. PATTERN TEST/
355 003256 012 015 040 .ASCII <12><15>/ 3. UNIQUE DATA CHECK/
356 003310 012 015 040 .ASCII <12><15>/ 4. MARCH TEST/
357 003333 012 015 040 .ASCII <12><15>/ 5. RANDOMNESS TESTS/
358 003364 012 015 104 .ASCIZ <12><15>/DO YOU WANT TO DROP ANY OF THESE FROM THE EXERCISER ?/
359 003454 104 122 117 QS7: .ASCIZ /DROP OPTION 1 ?/
360 003474 104 122 117 QS8: .ASCIZ /DROP OPTION 2 ?/
361 003514 104 122 117 QS9: .ASCIZ /DROP OPTION 3 ?/
362 003534 104 122 117 QS10: .ASCIZ /DROP OPTION 4 ?/
363 003554 120 101 124 QS11: .ASCIZ /PATTERN NUMBER (1-10) TO BE USED WITH MARCH TEST ?/
364 003637 104 122 117 QS12: .ASCIZ /DROP OPTION 5 ?/
365 003657 105 116 101 QS13: .ASCIZ /ENABLE REFRESH MARGINING ?/
366 003724 104 111 123 QS14: .ASCIZ /DISABLE ERROR CORRECTION ?/
367 003763 105 116 101 QS15: .ASCIZ /ENABLE END OF PASS SUMMARY PRINTOUT ?/
368 004030 105 116 101 QS16: .ASCIZ /ENABLE ERROR PRINTOUTS ?/
369 004066 105 116 101 QS17: .ASCIZ /ENABLE SOFT ERROR TESTING ?/
370 .EVEN
371
372
373
374
375
376
377
378
379 004126 BGNPROT L$PROT::
004126
380
381 004126 177777 -1 :OFFSET INTO P-TABLE FOR CSR ADDRESS
382 004130 177777 -1 :OFFSET INTO P-TABLE FOR MASSBUS ADDRESS
383 004132 177777 -1 :OFFSET INTO P-TABLE FOR DRIVE NUMBER
384
385 004134 ENDPROT
386
400
401
402 004134 $PATCH: .BLKW 32. ;THIS LEAVES ROOM FOR THE 22 ML-11 REGISTERS TOO
403
410
411 004234 ENDMOD
412
413
414
415
428
    
```

```

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

```

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          ;
9          ;OUTPUTS:                'RANDOM'--WORD CONTAINING RANDOM 16-BIT INTEGER VALUE
10         ;IMPLIED OUTPUTS:        NONE
11         ;CALLING SEQUENCE:       JSR PC,RN
12 005370 010046          RN::      MOV      R0,-(SP)
13 005372 016700 000062          MOV      SEED2,R0
14 005376 000241          CLC
15 005400 005367 000052          DEC      SEED1
16 005404 006100          ROL      R0
17 005406 006100          ROL      R0
18 005410 066700 000042          ADD      SEED1,R0
19 005414 066700 000042          ADD      SEED3,R0
20 005420 010067 000034          MOV      R0,SEED2
21 005424 006100          ROL      R0
22 005426 006100          ROL      R0
23 005430 066700 000026          ADD      SEED3,R0
24 005434 006100          ROL      R0
25 005436 006100          ROL      R0
26 005440 010067 000016          MOV      R0,SEED3
27 005444 016767 000010 000012  MOV      SEED2,RANDOM          ;SAVE NEW R.N.
28 005452 012600          MOV      (SP)+,R0
29 005454 000207          RTS      PC
30
31
32
33 005456 000000          SEED1:: .WORD 0
34 005460 001233          SEED2:: .WORD 1233
35 005462 007622          SEED3:: .WORD 7622
36
37 005464 000000          RANDOM:: .WORD 0

```

25-Mar-1982 22:23:37
25-Mar-1982 22:21:30

TOPS-20 Bliss-16 V2(212)
PA:<NEALE>MLX3.BLI.1 (1)

```

6 :MLX3
7 :
8 :
9 :      0001  MODULE MLX3 =
10 :      0002  BEGIN
11 :      0003
12 :      0004  REQUIRE 'BLSMAC.REQ';
13 :      1494
14 :      1495  %SBTTL 'REPORT CODING SECTION'
15 :      1496
16 :      1497  EXTERNAL ROUTINE
17 :      1498      EOP: NOVALUE;
18 :      1499
19 :      1500  BGNRPT;
20 :      1501
21 :      1502  EOP();
22 :      1503
23 :      1504  RETURN;
24 :      1505
25 :      1506  ENDRPT;

```

.GLOBL EOP

.SBTTL LRPT REPORT CODING SECTION

39	005466	004767	075234	LRPT:	JSR	PC,EOP	:	1502
40	005472	000207			RTS	PC	:	1498

: Routine Size: 3 words
: Maximum stack depth per invocation: 0 words

.SBTTL LSRPT REPORT CODING SECTION

58	005474	004767	177766	LSRPT::	JSR	PC,LRPT	:	1504
59				:MLX3				TOPS
60				:		REPORT CODING SECTION		PA:<

25-Mar-1982 22:23:37
25-Mar-1982 22:21:30

TRAP 25
RTS PC

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

```

61
62 005500 104425
63 005502 000207
64
65
66
67
68
69
70
71
72

```


25-Mar-1982 22:23:57 TOPS-20 Bliss-16 V2(212)
 25-Mar-1982 22:21:30 PA:<NEALE>MLX3.BLI.1 (2)

```

74 :MLX3
75 :
76 :
77 : 1507 %SBTTL 'AUTODROP SECTION'
78 : 1508
79 : 1509 !++
80 : 1510 ! THIS SECTION IS OPTIONALLY EXECUTED IMMEDIATELY AFTER THE INITIALIZATION
81 : 1511 ! CODE IF THE /ADR FLAG WAS SET. THE INTENT IS TO EXAMINE THE UNITS UNDER
82 : 1512 ! TEST TO SEE IF THEY WILL RESPOND. THOSE THAT DON'T RESPOND ARE DROPPED.
83 : 1513 !
84 : 1514 ! THIS FUNCTION IS AN INTEGRAL PART OF THE INITIALIZATION CODE ITSELF, AND
85 : 1515 ! THEREFORE A SEPARATE AUTODROP SECTION IS REDUNDANT AND NOT PROVIDED.
86 : 1516 !--
87 : 1517
88 : 1518 BGNAUTO;
89 : 1519
90 : 1520 RETURN;
91 : 1521
92 : 1522 ENDAUTO;
  
```

```

96 :
97 :
101 005504 000207          .SBTTL LAUTO AUTODROP SECTION          1506
102 : LAUTO: RTS PC ;
103 : ; Routine Size: 1 word
104 : ; Maximum stack depth per invocation: 0 words
109 :
110 :
114 :
115 :
119 005506 004767 177772 .SBTTL LSAUTO AUTODROP SECTION          1520
120 005512 104461 LSAUTO::JSR PC,LAUTO ;
121 005514 000207      TRAP 61
122 :      RTS PC
123 : ; Routine Size: 4 words
124 : ; Maximum stack depth per invocation: 0 words
  
```

130 :MLX3
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 :
178 :
179 :
180 :
181 :
182 :
183 :
184 :
185 :MLX3
186 :

DROP UNIT SECTION

25-Mar-1982 22:23:37
25-Mar-1982 22:21:30

TOPS-20 Bliss-16 V2(212)
PA:<NEALE>MLX3.BLI.1 (3)

1523 %SBTTL 'DROP UNIT SECTION'

1524

1525

1526

1527

1528

1529

1530

1531

1532

1533

1534

1535

1536

1537

1538

1539

1540

1541

1542

1543

1544

1545

1546

1547

1548

1549

1550

1551

1552

1553

1554

1555

1556

1557

1558

1559

1560

1561

1562

1563

1564

1565

1566

1567

1568

1569

1570

1571

1572

1573

1574

!++

THE DROP-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DRIVE TO NO LONGER BE TESTED. WHEN THIS HAPPENS, THE FOLLOWING ACTIONS WILL BE TAKEN:

(1) SET THE DRIVE STATUS TO INACTIVE AND SET THE DRIVE'S DROP FLAG.

(2) DECREMENT THE NUMBER OF ACTIVE DRIVES, AND IF ZERO ABORT PASS.

!+

DECLARE THE EXTERNAL SUMMARY REPORT CODE DATA STORAGE LOCATIONS TO THIS SECTION

EXTERNAL

L\$LUN,
DRIVE_STATUS:BITVECTOR[8],
DROPT_DRIVES:BITVECTOR[8],
NUM_DRIVES;

!DEFINES NUMBER OF UUT'S ATTACHED
!DRIVE STATUS FLAG REGISTER
!DRIVE DROP STATUS FLAG REGISTER
!INDICATES THE NUMBER OF ATTACHED UUT'S TO TEST

LITERAL

INACTIVE = 0,
ACTIVE = 1;

!CODE TO DESELECT UUT'S
!CODE TO SELECT UUT'S

!+

THIS SECTION IS CALLED FOR THE PURPOSE OF DROPPING THIS UUT FROM FURTHER TESTING.

THEREFORE CLEAR THIS UUT'S DRIVE ACTIVE FLAG BIT AND SET THIS UUT'S DRIVE DROPPED FLAG.

!-

DRIVE_STATUS[L\$LUN] = INACTIVE;
DROPT_DRIVES[L\$LUN] = ACTIVE;

!CLEAR THE DRIVE ACTIVE FLAG
!SET THE DRIVES DROPPED FLAG

!+

DECRIMENT THE GLOBAL VARIABLE 'NUM DRIVES' WHICH STORES THE NUMBER OF UUT'S WHICH ARE TESTED BY THIS EXERCISER.

IF THE NUMBER OF DRIVES REMAINING TO TEST IS ZERO THEN CALL THE CLEAN UP CODE SECTION AND EXIT THIS PROGRAM ELSE CONTINUE TESTING THE REMAINING ACTIVE UUT'S.

!-

DROP UNIT SECTION

25-Mar-1982 22:23:37
25-Mar-1982 22:21:30

TOPS-20 Bliss-16 V2(212)
PA:<NEALE>MLX3.BLI.1 (3)

```
187  
188 :  
189 : 1575  
190 : 1576 NUM_DRIVES = .NUM_DRIVES - 1; !DECRIENT THE NUMBER OF DRIVES REMAINING TO TEST  
191 : 1577  
192 : 1578  
193 : 1579 IF .NUM_DRIVES EQL 0 THEN DOCLN; !DO CLEAN UP CODE AND EXIT PROG IF ZERO  
194 : 1580  
195 : 1581 RETURN;  
196 : 1582  
197 : 1583 ENDDU;  
200  
201 .GLOBL L$LUN, DRIVE.STATUS, DROPT.DRIVES  
202 .GLOBL NUM.DRIVES  
203  
204  
205 .SBTTL LDU DROP UNIT SECTION
```

210	005516	016700	174352	LDU:	MOV	L\$LUN,RO	:	
211	005522	006200			ASR	RO		1562
212	005524	006200			ASR	RO		
213	005526	006200			ASR	RO		
214	005530	062700	034442		ADD	#DRIVE.STATUS,RO		
215	005534	010046			MOV	RO,--(SP)		
216	005536	016746	174332		MOV	L\$LUN,--(SP)		
217	005542	042716	177770		BIC	#177770,(SP)		
218	005546	012746	000001		MOV	#1,--(SP)		
219	005552	005046			CLR	--(SP)		
220	005554	004767	177034		JSR	PC,BL\$PU2		
221	005560	016700	174310		MOV	L\$LUN,RO	:	1563
222	005564	006200			ASR	RO		
223	005566	006200			ASR	RO		
224	005570	006200			ASR	RO		
225	005572	062700	034444		ADD	#DROPT.DRIVES,RO		
226	005576	010016			MOV	RO,(SP)		
227	005600	016746	174270		MOV	L\$LUN,--(SP)		
228	005604	042716	177770		BIC	#177770,(SP)		
229	005610	012746	000001		MOV	#1,--(SP)		
230	005614	011646			MOV	(SP),--(SP)		
231	005616	004767	176772		JSR	PC,BL\$PU2		
232	005622	005367	026630		DEC	MUM.DRIVES	:	1576
233	005626	001001			BNE	1\$:	1579
234	005630	104444			TRAP	44		
235	005632	062706	000016	1\$:	ADD	#16,SP	:	1522
236	005636	000207			RTS	PC		
237								
238								
239								
247								
248								
252								
253								
257	005640	004767	177652	L\$DU::	.SBTTL	L\$DU DROP UNIT SECTION	:	
258	005644	104453			JSR	PC,L\$DU		1581
259	005646	000207			TRAP	53		
260					RTS	PC		
261								
262								
267								
268								

: Routine Size: 41 words
 : Maximum stack depth per invocation: 7 words

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

25-Mar-1982 22:23:37
25-Mar-1982 22:21:30

TOPS-20 Bliss-16 V2(212)
PA:<NEALE>MLX3.BLI.1 (4)

270 :MLX3
271 :
272 :
273 : 1584
274 : 1585
275 : 1586
276 : 1587
277 : 1588
278 : 1589
279 : 1590
280 : 1591
281 : 1592
282 : 1593
283 : 1594
284 : 1595
285 : 1596
286 : 1597
287 : 1598
291 :
292 :
296 005650 000207
297 :

ADD UNIT SECTION

XSBTTL 'ADD UNIT SECTION'

!++
THE ADD UNIT SECTION CONTAINS ANY CODE THE PROGRAMMER WISHES
TO BE EXECUTED IN CONJUNCTION WITH THE ADDING OF A UNIT BACK
TO THE TEST CYCLE. SINCE THE INITIALIZATION CODE WILL HAVE
TO BE EXECUTED IMMEDIATELY AFTER AN 'ADD', THERE IS NO NEED
FOR ANY CODE IN THIS SECTION.
!--

BGNAU;

RETURN;

ENDAU;

LAU: .SBTTL LAU ADD UNIT SECTION
RTS PC ;

1583

299 ; Routine Size: 1 word
300 ; Maximum stack depth per invocation: 0 words
305
306
310

311
315 005652 004767 177772 LSAU:: .SBTTL LSAU ADD UNIT SECTION ; 1596
316 005656 104452 JSR PC,LAU
317 005660 000207 TRAP 52
RTS PC

318
319 ; Routine Size: 4 words
320 ; Maximum stack depth per invocation: 0 words
325 ; 1599

326 :MLX3
327 : ADD UNIT SECTION 25-Mar-1982 22:23:37 TOPS-20 Bliss-16 V2(212)
328 : 25-Mar-1982 22:21:30 PA:<NEALE>MLX3.BLI.1 (4)

329 : 1600
330 : 1601 END
331 : 1602
332 : 1603 ELUDOM
336

338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353

: OTS external references
 .GLOBL BL\$PU2

: 1604
: 1605
: Size: 62 code + 0 data words
: Run Time: 00:03.3
: Elapsed Time: 00:05.2
: Memory Used: 26 pages
: Compilation Complete

6	:MLX4			27-Mar-1982 19:24:42	TOPS-20 Bliss-16 V2(212)
7	:			27-Mar-1982 19:23:44	PA:<NEALE>MLX4.BLI.5 (1)
8	:				
9	:	0001	MODULE MLX4 =		
10	:	0002	BEGIN		
11	:	0003	:		
12	:	0004	PERTTY BLF COMMANDS		
13	:	0005	:		
14	:	0006	<BLF/NOERRORS>		
15	:	0007	<BLF/LOWERCASE_KEY>		
16	:	0008	:		
17	:	0009	:		
18	:	0010	require 'BLSMAC.REQ';		
19	:	1500	:		
20	:MLX4			27-Mar-1982 19:24:42	TOPS-20 Bliss-16 V2(212)
21	:		VARIABLES AND CONSTANTS	27-Mar-1982 19:23:44	PA:<NEALE>MLX4.BLI.5 (2)
22	:				
23	:	1501	%sbttl 'VARIABLES AND CONSTANTS'		
24	:	1502	:		
25	:	1503	literal		
26	:	1504	:		
27	:	1505	VER CZMLBB ADDED LITERAL TRUE, FALSE AND ZERO		
28	:	1506	:		
29	:	1507	ZERO = %o'000000',	!DEFINE ZERO DATA	
30	:	1508	TRUE = 1,	!LOGICAL TRUE INDICATOR	
31	:	1509	FALSE = 0,	!LOGICAL FALSE INDICATOR	
32	:	1510	:		
33	:	1511	OTHER LITERAL DEFINITIONS		
34	:	1512	:		
35	:	1513	ONE = 1,	!NUMBER OF TIMES TO RETRY FOR DATA ERROR	
36	:	1514	SIX = 6,	!NUMBER OF TIMES TO RETRY FOR SYSTEM ERROR	
37	:	1515	NUM_PATS = 10,	!NUMBER OF REGULAR PATTERNS	
38	:	1516	NUM_REGS = 22,	!NUMBER OF ML11 REGISTERS	
39	:	1517	TIMES_TO_LOOP = 2,	!LOOP READING CONSTANT	
40	:	1518	BUFSIZ = 256*8,	!2048 16-BIT WORDS IN A FULL BUFFER	
41	:	1519	256 = NUMBER OF WORDS IN A SECTOR		
42	:	1520	8 = NUMBER OF SECTORS IN THE BUFFER		
43	:	1521	:		
44	:	1522	ML11 DRIVE TYPES:		
45	:	1523	:		
46	:	1524	ML11A = %o'000110',	!16K CHIPS	
47	:	1525	ML11B = %o'000111',	!64K CHIPS	
48	:	1526	:		
49	:	1527	FUNCTION CODES:		
50	:	1528	:		
51	:	1529	DRV_CLR = %o'11',		
52	:	1530	WC_CMD = %o'51',		
53	:	1531	WR_CMD = %o'61',		
54	:	1532	RD_CMD = %o'71',		
55	:	1533	:		
56	:	1534	STATUS CODES:		
57	:	1535	:		
58	:	1536	INACTIVE = 0,		
59	:	1537	ACTIVE = 1,		
60	:	1538	:		
61	:	1539	ERROR THRESHOLD VALUES:		
62	:	1540	:		

63 :	1541	S16K_LIMIT = 10,	!SOFT ERROR, 16K ARRAYS
64 :	1542	H16K_LIMIT = 10,	!HARD ERROR, 16K ARRAYS
65 :	1543	S64K_LIMIT = 10,	!SOFT ERROR, 64K ARRAYS
66 :	1544	H64K_LIMIT = 10,	!HARD ERROR, 64K ARRAYS
67 :	1545		
68 :	1546	! SUMMARY OF ERROR CODES:	
69 :	1547	! :	
70 :	1548	! THE 'INTEGRITY' ROUTINE:	
71 :	1549	! :	
72 :	1550	! WRITE COMMAND:	
73 :	1551	!ERRDF(1,MSG1,0);	!**** INTEGRITY ROUTINE ERROR 01 ****
74 :	1552		

27-Mar-1982 19:24:42
27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
PA:<NEALE>MLX4.BLI.5 (2)

76 :MLX4
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 :
115 :
116 :
117 :
118 :
119 :
120 :
121 :
122 :
123 :
124 :
125 :
126 :
127 :
128 :
129 :
130 :

VARIABLES AND CONSTANTS

```

: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:
    
```

132 :MLX4
 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 :
 178 :
 179 :
 180 :
 181 :
 182 :
 183 :
 184 :
 185 :
 186 :

VARIABLES AND CONSTANTS

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (2)

```

!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 ****
  
```

188 :MLX4
189 :
190 :
191 :
192 :
193 :
194 :
195 :
196 :
197 :
198 :
199 :
200 :
201 :
202 :
203 :
204 :
205 :
206 :
207 :
208 :
209 :
210 :
211 :
212 :
213 :
214 :
215 :
216 :
217 :
218 :
219 :
220 :
221 :
222 :
223 :
224 :
225 :
226 :
227 :
228 :
229 :
230 :
231 :
232 :
233 :
234 :
235 :
236 :
237 :
238 :
239 :
240 :
241 :
242 :

VARIABLES AND CONSTANTS

27-Mar-1982 19:24:42
27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
PA:<NEALE>MLX4.BLI.5 (2)

```
1657 :ERRDF(419,MSG2,0): :**** OPTION 4 ERROR 19 ****
1658 :ERRHRD(420,MSG4,0): :**** OPTION 4 ERROR 20 ****
1659 :ERRSOFT(421,MSG3,0): :**** OPTION 4 ERROR 21 ****
1660 :ERRSOFT(422,MSG3,0): :**** OPTION 4 ERROR 22 ****
1661 :WRITE DATA:
1662 :ERRDF(423,MSG1,0): :**** OPTION 4 ERROR 23 ****
1663 :ERRDF(424,MSG1,0): :**** OPTION 4 ERROR 24 ****
1664 :ERRDF(425,MSG1,0): :**** OPTION 4 ERROR 25 ****
1665 :MARCHING UP:
1666 :CHECK OR READ DATA:
1667 :ERRDF(426,MSG1,0): :**** OPTION 4 ERROR 26 ****
1668 :ERRDF(427,MSG1,0): :**** OPTION 4 ERROR 27 ****
1669 :ERRDF(428,MSG1,0): :**** OPTION 4 ERROR 28 ****
1670 :ERRDF(429,MSG1,0): :**** OPTION 4 ERROR 29 ****
1671 :ERRDF(430,MSG2,0): :**** OPTION 4 ERROR 30 ****
1672 :ERRHRD(431,MSG4,0): :**** OPTION 4 ERROR 31 ****
1673 :ERRSOFT(432,MSG3,0): :**** OPTION 4 ERROR 32 ****
1674 :ERRSOFT(433,MSG3,0): :**** OPTION 4 ERROR 33 ****
1675 :
1676 : OPTION 5:
1677 :
1678 : 'RAND1' ROUTINE:
1679 : WRITE COMMAND:
1680 :ERRDF(5101,MSG1,0): :**** OPTION 5, RAND1 ERROR 01 ****
1681 :ERRDF(5102,MSG1,0): :**** OPTION 5, RAND1 ERROR 02 ****
1682 :ERRDF(5103,MSG1,0): :**** OPTION 5, RAND1 ERROR 03 ****
1683 :CHECK OR READ:
1684 :ERRDF(5104,MSG1,0): :**** OPTION 5, RAND1 ERROR 04 ****
1685 :ERRDF(5105,MSG1,0): :**** OPTION 5, RAND1 ERROR 05 ****
1686 :ERRDF(5106,MSG1,0): :**** OPTION 5, RAND1 ERROR 06 ****
1687 :ERRDF(5107,MSG1,0): :**** OPTION 5, RAND1 ERROR 07 ****
1688 :ERRDF(5108,MSG2,0): :**** OPTION 5, RAND1 ERROR 08 ****
1689 :ERRHRD(5109,MSG4,0): :**** OPTION 5, RAND1 ERROR 09 ****
1690 :ERRSOFT(5110,MSG3,0): :**** OPTION 5, RAND1 ERROR 10 ****
1691 :ERRSOFT(5111,MSG3,0): :**** OPTION 5, RAND1 ERROR 11 ****
1692 : 'RAND2' ROUTINE:
1693 : WRITE COMMAND:
1694 :ERRDF(5201,MSG1,0): :**** OPTION 5, RAND2 ERROR 01 ****
1695 :ERRDF(5202,MSG1,0): :**** OPTION 5, RAND2 ERROR 02 ****
1696 :ERRDF(5203,MSG1,0): :**** OPTION 5, RAND2 ERROR 03 ****
1697 :CHECK OR READ:
1698 :ERRDF(5204,MSG1,0): :**** OPTION 5, RAND2 ERROR 04 ****
1699 :ERRDF(5205,MSG1,0): :**** OPTION 5, RAND2 ERROR 05 ****
1700 :ERRDF(5206,MSG1,0): :**** OPTION 5, RAND2 ERROR 06 ****
1701 :ERRDF(5207,MSG1,0): :**** OPTION 5, RAND2 ERROR 07 ****
1702 :ERRDF(5208,MSG2,0): :**** OPTION 5, RAND2 ERROR 08 ****
1703 :ERRHRD(5209,MSG4,0): :**** OPTION 5, RAND2 ERROR 09 ****
1704 :ERRSOFT(5210,MSG3,0): :**** OPTION 5, RAND2 ERROR 10 ****
1705 :ERRSOFT(5211,MSG3,0): :**** OPTION 5, RAND2 ERROR 11 ****
1706 : 'RAND3' ROUTINE:
1707 : WRITE COMMAND:
1708 :ERRDF(5301,MSG1,0): :**** OPTION 5, RAND3 ERROR 01 ****
```

27-Mar-1982 19:24:42
27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
PA:<NEALE>MLX4.BLI.5 (2)

244 :MLX4
245 :
246 :
247 :
248 :
249 :
250 :
251 :
252 :
253 :
254 :
255 :
256 :
257 :
258 :
259 :
260 :
261 :
262 :
263 :
264 :
265 :
266 :
267 :
268 :
269 :
270 :
271 :
272 :
273 :
274 :
275 :
276 :
277 :
278 :
279 :
280 :
281 :
282 :
283 :
284 :
285 :
286 :
287 :
288 :
289 :
290 :
291 :
292 :
293 :
294 :
295 :
296 :
297 :
298 :

VARIABLES AND CONSTANTS

1709 !ERRDF(5302,MSG1,0): !**** OPTION 5, RAND3 ERROR 02 ****
1710 !ERRDF(5303,MSG1,0): !**** OPTION 5, RAND3 ERROR 03 ****
1711 !CHECK OR READ:
1712 !ERRDF(5304,MSG1,0): !**** OPTION 5, RAND3 ERROR 04 ****
1713 !ERRDF(5305,MSG1,0): !**** OPTION 5, RAND3 ERROR 05 ****
1714 !ERRDF(5306,MSG1,0): !**** OPTION 5, RAND3 ERROR 06 ****
1715 !ERRDF(5307,MSG1,0): !**** OPTION 5, RAND3 ERROR 07 ****
1716 !ERRDF(5308,MSG2,0): !**** OPTION 5, RAND3 ERROR 08 ****
1717 !ERRHRD(5309,MSG4,0): !**** OPTION 5, RAND3 ERROR 09 ****
1718 !ERRSOFT(5310,MSG3,0): !**** OPTION 5, RAND3 ERROR 10 ****
1719 !ERRSOFT(5311,MSG3,0): !**** OPTION 5, RAND3 ERROR 11 ****
1720 !'RAND4' ROUTINE:
1721 !WRITE COMMAND:
1722 !ERRDF(5401,MSG1,0): !**** OPTION 5, RAND4 ERROR 01 ****
1723 !ERRDF(5402,MSG1,0): !**** OPTION 5, RAND4 ERROR 02 ****
1724 !ERRDF(5403,MSG1,0): !**** OPTION 5, RAND4 ERROR 03 ****
1725 !CHECK OR READ:
1726 !ERRDF(5404,MSG1,0): !**** OPTION 5, RAND4 ERROR 04 ****
1727 !ERRDF(5405,MSG1,0): !**** OPTION 5, RAND4 ERROR 05 ****
1728 !ERRDF(5406,MSG1,0): !**** OPTION 5, RAND4 ERROR 06 ****
1729 !ERRDF(5407,MSG1,0): !**** OPTION 5, RAND4 ERROR 07 ****
1730 !ERRDF(5408,MSG2,0): !**** OPTION 5, RAND4 ERROR 08 ****
1731 !ERRHRD(5409,MSG4,0): !**** OPTION 5, RAND4 ERROR 09 ****
1732 !ERRSOFT(5410,MSG3,0): !**** OPTION 5, RAND4 ERROR 10 ****
1733 !ERRSOFT(5411,MSG3,0): !**** OPTION 5, RAND4 ERROR 11 ****

CODES FOR WHY A UNIT WAS DROPPED:

CODE_1 = 1, !DRIVE NOT POWERED UP
CODE_2 = 2, !DRIVE NOT AN ML11 UNIT
CODE_3 = 3, !OPERATOR SET TEST LIMITS INCORRECTLY
CODE_4 = 4, !FAILED ALL RETRIES FOR A NON-FATAL ERROR
CODE_5 = 5, !CONTROLLER FATAL ERROR
CODE_6 = 6, !DRIVE FATAL ERROR
CODE_7 = 7, !ECC HARD ERROR
CODE_8 = 8, !ECC LOGIC FAILED TO DETECT ERROR

FIELD DECLARTIONS

field

VER CZMLBB ADD FIELD DECLARATION

FIELD DECLARATION TO MAP THE SINGLE BIT
ERROR LOG TABLE.

SBE_TBL_MAP =

set
BANKS_SBE = [0, 0, 2, 0], !LOG BANK OF SBE
BRDS_SBE = [0, 2, 4, 0], !LOG BOARD OF SBE

27-Mar-1982 19:24:42 TOPS-20 Blss-16 V2(212)
27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (2)

300 :MLX4
301 :
302 :
303 :
304 :
305 :
306 :
307 :
308 :
309 :
310 :
311 :
312 :
313 :
314 :
315 :
316 :
317 :
318 :
319 :
320 :
321 :
322 :
323 :
324 :
325 :
326 :
327 :
328 :
329 :
330 :
331 :
332 :
333 :
334 :
335 :
336 :
337 :
338 :
339 :
340 :
341 :
342 :
343 :
344 :
345 :
346 :
347 :
348 :
349 :
350 :
351 :
352 :
353 :
354 :

VARIABLES AND CONSTANTS

1761 BITS_SBE = [0, 6, 7, 0]
1762 UNITS_SBE = [0, 13, 3, 0]
1763 SUMS_SBE = [1, 0, 16, 0]
1764 WRDS_0 = [0, 0, 16, 0]
1765 WRDS_1 = [1, 0, 16, 0]
1766 tes.

!LOG BIT OF SBE
!LOG UNIT OF SBE
!LOG NUMBER OF TIMES OCCURRING
!ACCESS FIRST WORD OF TABLE
!ACCESS SECOND WORD OF TABE

VER CZMLBB ADDED FIELD DECLARATION

FIELD DECLARATION FOR PROM MAINTENANCE
SUM TABLE

PM_SBE_MAP =
set
PM_SBE\$SUM = [0, 16, 0]
tes.

!ACCESS NUMBER OF SBE PER ARRAY

FIELD DECLARATION FOR PATTERN TABLE

PATMAP =
set
COUNT1 = [0, 0, 16, 0]
COUNT2 = [1, 0, 16, 0]
tes;

own

Single bit error logging and Prom Maint
program storage structures

VER CZMLBB ADDED FOLLOWING THREE DATA DECLARATIONS

SBE_LOG : blockvector [128, 2, word] field (SBE_TBL_MAP), !SBE LOCATION TABLE
PM_SBE_CNT : blockvector [8, 16, word] field (PM_SBE_MAP), !ARRAY SBE COUNT TALBE
SBE\$COUNT : word, !COUNTS NUMBER OF SBE LOCATION TABLE ENTRIES
BIT_NUM, !STORS FAILING SBE CHIP NUMBER

ML-11 EXERCISER STORAGE ALLOCATION

WBUFF : vector [BUFSIZ] volatile,
RBUFF : vector [BUFSIZ] volatile,
WPTR : volatile,
RPTR : volatile,
QUICK : volatile,
PATTERN : volatile,
DATA_COUNT : volatile,
COMP_COUNT : volatile,
EOP_COUNT : volatile,
BASE_ADDR,
VEC,
BR_LEVEL,
SOFTS : blockvector [8, 16],

!IMPORTANT -- WBUFF AND RBUFF MUST
!BE CONTIGUOUS AND IN THAT ORDER!!

!COUNTS FOR 8 LUNS, 16 ARRAYS EACH

356 :MLX4
 357 :
 358 :
 359 :
 360 :
 361 :
 362 :
 363 :
 364 :
 365 :
 366 :
 367 :
 368 :
 369 :
 370 :
 371 :
 372 :
 373 :
 374 :
 375 :
 376 :
 377 :
 378 :
 379 :
 380 :
 381 :
 382 :
 383 :
 384 :
 385 :
 386 :
 387 :
 388 :
 389 :
 390 :
 391 :
 392 :
 393 :
 394 :
 395 :
 396 :
 397 :
 398 :
 399 :
 400 :
 401 :
 402 :
 403 :
 404 :
 405 :
 406 :
 407 :
 408 :
 409 :
 410 :

VARIABLES AND CONSTANTS

HARDS : blockvector [8, 16],
 TRIES : blockvector [8, 16],
 WR_COUNT,
 WR_THOUSANDS,
 WR_MILLIONS,
 RD_COUNT,
 RD_THOUSANDS,
 RD_MILLIONS,
 WC_COUNT,
 WC_THOUSANDS,
 WC_MILLIONS,
 I AM DONE,
 RETRYING,
 BOARD,
 BANK;

global

ML_REG : vector [NUM_REGS] volatile,
 PTABLE_ADDR : vector [8] volatile,
 DRIVE_STATUS : bitvector [8] volatile,
 DROPT_DRIVES : bitvector [8] volatile,
 WHY_DROPT : vector [8, byte],
 NUM_DRIVES,
 LOW_SECT : vector [8] volatile,
 TOP_SECT : vector [8] volatile;

EQUALS:

macro

TO CALCULATE THE TEST RANGES FOR A PARTICULAR LOGICAL UNIT:

LOWEST =
 .LOW_SECT[.LUN]%, !THE FIRST SECTOR TO TEST
 HIGHEST =
 .TOP_SECT[.LUN]%, !THE LAST SECTOR TO TEST

! ADDRESS IN MAIN MEMORY THAT CONTAINS THE PHYSICAL DRIVE
 ! NUMBER THAT CORRESPONDS TO A PARTICULAR LOGICAL UNIT:

DRIVE =
 (.PTABLE_ADDR[.LUN] + 6)%,

ML-11 REGISTER NAMES:

MLCS1 =
 .ML_REG[0]%, !CONTROL AND STATUS REGISTER 1
 MLWC =
 .ML_REG[1]%, !WORD COUNT REGISTER
 MLBA =
 .ML_REG[2]%, !UNIBUS ADDRESS REGISTER
 MLDA =

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1987 19:23:44 PA:<NEALE>MLX4.BLI.5 (2)

!COUNTS FOR 8 LUNS, 16 ARRAYS EACH
 !COUNTS FOR 8 LUNS, 16 ARRAYS EACH
 !# BYTES TRANSFERED VIA 'WRITE'
 !THOUSANDS OF BYTES
 !MILLIONS OF BYTES
 !# BYTES TRANSFERED VIA 'READ'

!# BYTES TRANSFERED VIA 'WRITE CHECK'

412 :MLX4
 413 :
 414 :
 415 :
 416 : M 1865
 417 : M 1866
 418 : M 1867
 419 : M 1868
 420 : M 1869
 421 : M 1870
 422 : M 1871
 423 : M 1872
 424 : M 1873
 425 : M 1874
 426 : M 1875
 427 : M 1876
 428 : M 1877
 429 : M 1878
 430 : M 1879
 431 : M 1880
 432 : M 1881
 433 : M 1882
 434 : M 1883
 435 : M 1884
 436 : M 1885
 437 : M 1886
 438 : M 1887
 439 : M 1888
 440 : M 1889
 441 : M 1890
 442 : M 1891
 443 : M 1892
 444 : M 1893
 445 : M 1894
 446 : M 1895
 447 : M 1896
 448 : M 1897
 449 : M 1898
 450 : M 1899
 451 : M 1900
 452 : M 1901
 453 : M 1902
 454 : M 1903
 455 : M 1904
 456 : M 1905
 457 : M 1906
 458 : M 1907
 459 : M 1908
 460 : M 1909
 461 : M 1910
 462 : M 1911
 463 : M 1912
 464 : M 1913
 465 : M 1914
 466 : M 1915
 467 : M 1916

VARIABLES AND CONSTANTS

.ML REG[3]%,
 MLCS2 =
 .ML REG[4]%,
 MLDS =
 .ML REG[5]%,
 MLER =
 .ML REG[6]%,
 MLAS =
 .ML REG[7]%,
 MLPA =
 .ML REG[8]%,
 MLDB =
 .ML REG[9]%,
 MLMR =
 .ML REG[10]%,
 MLDT =
 .ML REG[11]%,
 MLSN =
 .ML REG[12]%,
 MLE1 =
 .ML REG[13]%,
 MLE2 =
 .ML REG[14]%,
 MLD1 =
 .ML REG[15]%,
 MLD2 =
 .ML REG[16]%,
 MLEE =
 .ML REG[17]%,
 MLEL =
 .ML REG[18]%,
 MLPD =
 .ML REG[19]%,
 MLBAE =
 .ML REG[20]%,
 MLCS3 =
 .ML REG[21]%,
 !
 ! BIT ASSIGNMENTS:
 !
 ! MLCS1 BITS:
 !
 SC =
 (MLCS1)<15,1>%,
 TRE =
 (MLCS1)<14,1>%,
 MCPE =
 (MLCS1)<13,1>%,
 DVA =
 (MLCS1)<11,1>%,
 RDY =

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (2)

!DESIRED ADDRESS REGISTER
 !CONTROL AND STATUS REGISTER 2
 !DRIVE STATUS REGISTER
 !ERROR REGISTER
 !ATTENTION SUMMARY REGISTER
 !PROM ADDRESS REGISTER
 !DATA BUFFER REGISTER
 !MAINTENANCE REGISTER
 !DRIVE TYPE REGISTER
 !SERIAL NUMBER REGISTER
 !ECC REGISTER 1
 !ECC REGISTER 2
 !DATA DIAGNOSTIC REGISTER 1
 !DATA DIAGNOSTIC REGISTER 2
 !ECC ERROR REGISTER
 !ECC ERROR LOCATION REGISTER
 !PROM DATA REGISTER
 !BUS ADDRESS EXTENSION REGISTER
 !CONTROL AND STATUS REGISTER 3
 !SPECIAL CONDITION
 !TRANSFER ERROR
 !MASSBUS CONTROL BUS PARITY ERROR
 !DRIVE AVAILABLE

468 :MLX4
 469 :
 470 :
 471 :
 472 :
 473 :
 474 :
 475 :
 476 :
 477 :
 478 :
 479 :
 480 :
 481 :
 482 :
 483 :
 484 :
 485 :
 486 :
 487 :
 488 :
 489 :
 490 :
 491 :
 492 :
 493 :
 494 :
 495 :
 496 :
 497 :
 498 :
 499 :
 500 :
 501 :
 502 :
 503 :
 504 :
 505 :
 506 :
 507 :
 508 :
 509 :
 510 :
 511 :
 512 :
 513 :
 514 :
 515 :
 516 :
 517 :
 518 :
 519 :
 520 :
 521 :
 522 :

VARIABLES AND CONSTANTS
 1917 (MLCS1)<7,1>%,
 M 1918 IE =
 1919 (MLCS1)<6,1>%,
 M 1920 FUNC =
 1921 (MLCS1)<0,6>%,
 1922 !
 1923 ! MLCS2 BITS:
 1924 !
 M 1925 DLT =
 1926 (MLCS2)<15,1>%,
 M 1927 WCE =
 1928 (MLCS2)<14,1>%,
 M 1929 PE =
 1930 (MLCS2)<13,1>%,
 M 1931 NED =
 1932 (MLCS2)<12,1>%,
 M 1933 NEM =
 1934 (MLCS2)<11,1>%,
 M 1935 PGE =
 1936 (MLCS2)<10,1>%,
 M 1937 MXF =
 1938 (MLCS2)<9,1>%,
 M 1939 MDPE =
 1940 (MLCS2)<8,1>%,
 M 1941 ORDY =
 1942 (MLCS2)<7,1>%,
 M 1943 IRDY =
 1944 (MLCS2)<6,1>%,
 M 1945 CLR =
 1946 (MLCS2)<5,1>%,
 M 1947 PAT =
 1948 (MLCS2)<4,1>%,
 M 1949 BAI =
 1950 (MLCS2)<3,1>%,
 M 1951 UNIT =
 1952 (MLCS2)<0,3>%,
 1953 !
 1954 ! MLDS BITS:
 1955 !
 M 1956 ATA =
 1957 (MLDS)<15,1>%,
 M 1958 ERR =
 1959 (MLDS)<14,1>%,
 M 1960 MOL =
 1961 (MLDS)<12,1>%,
 M 1962 LBT =
 1963 (MLDS)<10,1>%,
 M 1964 DPR =
 1965 (MLDS)<8,1>%,
 M 1966 DRY =
 1967 (MLDS)<7,1>%,
 M 1968 VV =

27-Mar-1982 19:24:42 TCPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (2)
 !READY
 !INTERRUPT ENABLE
 !FUNCTION (COMMAND) CODE AND GO BIT
 !DATA LATE
 !WRITE CHECK ERROR
 !PARITY ERROR
 !NON-EXISTENT DRIVE
 !NON-EXISTENT MEMORY
 !PROGRAM ERROR
 !MISSED TRANSFER
 !MASSBUS DATA BUS PARITY ERROR
 !OUTPUT READY
 !INPUT READY
 !CONTROLLER CLEAR
 !PARITY TEST
 !UNIBUS ADDRESS INCREMENT INHIBIT
 !UNIT SELECT
 !ATTENTION ACTIVE
 !ERROR SUMMARY
 !MEDIUM ON LINE
 !LAST BLOCK TRANSFERRED
 !DRIVE PRESENT
 !DRIVE READY

524 :MLX4
 525 :
 526 :
 527 :
 528 :
 529 :
 530 :
 531 : M
 532 :
 533 : M
 534 :
 535 : M
 536 :
 537 : M
 538 :
 539 : M
 540 :
 541 : M
 542 :
 543 : M
 544 :
 545 : M
 546 :
 547 : M
 548 :
 549 : M
 550 :
 551 : M
 552 :
 553 :
 554 :
 555 :
 556 : M
 557 :
 558 : M
 559 :
 560 : M
 561 :
 562 : M
 563 :
 564 : M
 565 :
 566 : M
 567 :
 568 : M
 569 :
 570 : M
 571 :
 572 : M
 573 :
 574 : M
 575 :
 576 : M
 577 :
 578 :

VARIABLES AND CONSTANTS

```

1969 (MLDS)<6,1>%
1970 !
1971 ! MLER BITS:
1972 !
1973 DCK =
1974 (MLER)<15,1>%
1975 UNS =
1976 (MLER)<14,1>%
1977 OPI =
1978 (MLER)<13,1>%
1979 IAE =
1980 (MLER)<10,1>%
1981 AOE =
1982 (MLER)<9,1>%
1983 ECH =
1984 (MLER)<6,1>%
1985 DPAR =
1986 (MLER)<5,1>%
1987 CPAR =
1988 (MLER)<3,1>%
1989 RMR =
1990 (MLER)<2,1>%
1991 ILR =
1992 (MLER)<1,1>%
1993 ILF =
1994 (MLER)<0,1>%
1995 !
1996 ! MLMR BITS:
1997 !
1998 SZ =
1999 (MLMR)<11,5>%
2000 ARR TYP =
2001 (MLMR)<10,1>%
2002 TRT =
2003 (MLMR)<8,2>%
2004 REF MAR =
2005 (MLMR)<7,1>%
2006 PROM RW =
2007 (MLMR)<6,1>%
2008 PROM DIS =
2009 (MLMR)<5,1>%
2010 DAT CLK =
2011 (MLMR)<4,1>%
2012 DAT DM =
2013 (MLMR)<3,1>%
2014 DCK EN =
2015 (MLMR)<2,1>%
2016 ECC DIS =
2017 (MLMR)<1,1>%
2018 ECC DM =
2019 (MLMR)<0,1>%
2020 !
    
```

```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (2)

!VOLUME VALID

!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

!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
    
```

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (2)

580 :MLX4
 581 :
 582 :
 583 :
 584 :
 585 :
 586 :
 587 :
 588 :
 589 :
 590 :
 591 :
 592 :
 593 :
 594 :
 595 :
 596 :
 597 :
 598 :
 599 :
 600 :
 601 :
 602 :
 603 :
 604 :
 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 :

VARIABLES AND CONSTANTS
 :
 : 2021 : MLSN BITS:
 : 2022 :
 M 2023 : SN3 =
 : 2024 : (MLSN)<12,4>%,
 M 2025 : SN2 =
 : 2026 : (MLSN)<8,4>%,
 M 2027 : SN1 =
 : 2028 : (MLSN)<4,4>%,
 M 2029 : SMO =
 : 2030 : (MLSN)<0,4>%,
 : 2031 :
 : 2032 : MLEE BITS:
 : 2033 :
 M 2034 : UNC =
 : 2035 : (MLEE)<15,1>%,
 M 2036 : SGL =
 : 2037 : (MLEE)<14,1>%,
 M 2038 : crc =
 : 2039 : (MLEE)<13,1>%,
 M 2040 : CHAN =
 : 2041 : (MLEE)<6,6>%,
 M 2042 : EFUN =
 : 2043 : (MLEE)<0,6>%;
 : 2044 :
 : 2045 : external
 : 2046 :
 : 2047 : HEADER INFORMATION:
 : 2048 :
 : 2049 : EFNS21,
 : 2050 : L\$UNIT,
 : 2051 : L\$LUN,
 : 2052 :
 : 2053 : ALL OF THE SOFTWARE P-TABLE LOCATIONS:
 : 2054 :
 : 2055 : LIMIT,
 : 2056 : RANGE,
 : 2057 : LSECT,
 : 2058 : TSECT,
 : 2059 : ONLY,
 : 2060 : DROPNE,
 : 2061 : DROP1,
 : 2062 : DROP2,
 : 2063 : DROP3,
 : 2064 : DROP4,
 : 2065 : DROPS,
 : 2066 : MARPAT,
 : 2067 : REFRESH,
 : 2068 : ECCDIS,
 : 2069 : EOPSUM,
 : 2070 : ERROUT,
 : 2071 :
 : 2072 : RANDOM NUMBER GENERATION VALUES:

!HIGH ORDER DECADE
 !THIRD DECADE
 !SECOND DECADE
 !LOW ORDER DECADE
 !UNCORRECTABLE ERROR
 !SINGLE ERROR
 !CRC ERROR
 !CHANNEL IN ERROR
 !ERROR FUNCTION

! VER CZMLBB EVENT FLAG TO TURN ON SBE TESTING

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (2)

636 :MLX4
 637 :
 638 :
 639 :
 640 :
 641 :
 642 :
 643 :
 644 :
 645 :
 646 :
 647 :
 648 :
 649 :
 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 :

VARIABLES AND CONSTANTS

2073 !
 2074 SEED1,
 2075 SEED2,
 2076 SEED3,
 2077 RANDOM;
 2078
 2079 external routine
 2080 RN : novalue;

!FOR 16-BIT RANDOM NUMBER GENERATION

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

global

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (2)

692 :MLX4
693 :
694 :
695 :
696 :
697 :
698 :
699 :
700 :
701 :
702 :
703 :
704 :
705 :
706 :
707 :
708 :
709 :
710 :
711 :
712 :
713 :
714 :
715 :
716 :
717 :
718 :
719 :
720 :

VARIABLES AND CONSTANTS

```
2125 !<BLF/NOFORMAT>  
2126 PATTBL: blockvector [NUM_PATS/2,2] field(PATMAP)  
2127 preset (  
2128 [0,COUNT1] = %decimal'0' , !FOR PATTERNS 1, -1, 6, -6  
2129 [0,COUNT2] = %decimal'1024' ,  
2130 [1,COUNT1] = %decimal'1' , !FOR PATTERNS 2, -2, 7, -7  
2131 [1,COUNT2] = %decimal'1' ,  
2132 [2,COUNT1] = %decimal'4' , !FOR PATTERNS 3, -3, 8, -8  
2133 [2,COUNT2] = %decimal'4' ,  
2134 [3,COUNT1] = %decimal'9' , !FOR PATTERNS 4, -4, 9, -9  
2135 [3,COUNT2] = %decimal'9' ,  
2136 [4,COUNT1] = %decimal'1024' , !FOR PATTERNS 5, -5, 10, -10  
2137 [4,COUNT2] = %decimal'1024'  
2138 );
```

!<BLF/FORMAT>

bind

DEFINITIONS OF LOCATIONS WITHIN THE WRITE AND READ BUFFERS:

```
2145 WDBUFF = WBUFF , !256-WORD WRITE DATA BUFFER  
2146 WCBUFF = WBUFF + 512, !256-WORD WRITE COMP BUFFER  
2147 RDBUFF = RBUFF , !256-WORD READ DATA BUFFER  
2148 RCBUFF = RBUFF + 512, !256-WORD READ COMP BUFFER  
2149 END_WBUFF = (WBUFF + BUFSIZ*2), !JUST BEYOND END OF FULL WRITE BUFFER  
2150 END_RBUFF = (RBUFF + BUFSIZ*2), !JUST BEYOND END OF FULL READ BUFFER
```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (3)

```
722 :MLX4
723 :
724 :
725 : 2151 %sbttl 'MESSAGES AND PRINT FORMATS'
726 : 2152
727 : 2153 SINGLE BIT ERROR LOGGING TABLE MESSAGE STRINGS
728 : 2154
729 : 2155 PHR21 = uplit (%asciz' SINGLE BIT ERROR LOG SUMMARY REPORT'),
730 : 2156 FMT16 = uplit (%asciz'%X%D1%A(D) %D2%A(D) %D1%A(D) %D2%A(D) %D6%A(D)'),
731 : 2157 SBES$HEADER = uplit (%asciz'UNIT #: ARRAY #: BANK #: BIT #: COUNT #:'),
732 : 2158
733 : 2159 SPECIAL PURPOSE FORMATS (WITH SAMPLE PRINTOUTS):
734 : 2160
735 : 2161 FMT1A = uplit (%asciz'%X%T%S2%D2'),
736 : 2162 !'PATTERN NUMBER XX'
737 : 2163 FMT1B = uplit (%asciz'%S7%T%S%A-%D2'),
738 : 2164 !' PATTERN NUMBER -XX'
739 : 2165 FMT2 = uplit (%asciz'%S2%T'),
740 : 2166 !' (NOT POWERED UP)'
741 : 2167 !' (NOT AN ML11 UNIT)'
742 : 2168 !' (OPERATOR SELECTED TEST LIMITS INCORRECTLY)'
743 : 2169 !' (ALL RETRIES FAILED FOR A NON-FATAL ERROR)'
744 : 2170 !' (CONTROLLER FATAL ERROR)'
745 : 2171 !' (DRIVE FATAL ERROR)'
746 : 2172 !' (ECC HARD ERROR)'
747 : 2173 !' UP'
748 : 2174 !' DOWN'
749 : 2175 !' --> RUN ML11 PROM MAINTENANCE PROGRAM'
750 : 2176 FMT3 = uplit (%asciz'%X%A**%S%T%S%T%D3%S%A**'),
751 : 2177 !'** END PASS X **'
752 : 2178 FMT4A = uplit (%asciz'%X2%T%A:%S%D1%S4%T%A:%S%D1'),
753 : 2179 !'LOGICAL UNIT: X DRIVE: Y'
754 : 2180 FMT4B = uplit (%asciz'%S4%T%A:%S%O6'),
755 : 2181 !' SERIAL #: ZZZZZZ'
756 : 2182 !' CSR ADDRESS: XXXXXX'
757 : 2183 FMT4C = uplit (%asciz'%S4%T%A:%S%D1%D1%D1%D1'),
758 : 2184 !' SERIAL #: DDDD'
759 : 2185 FMT5 = uplit (%asciz'%X%T%S3%A%SECTORS UNDER TEST:%S%O6%S%ATO%S%O6'),
760 : 2186 !'ML11-X SECTORS UNDER TEST: XXXXXX TO YYYYYY'
761 : 2187 FMT6 = uplit (%asciz'%X%A%BEGAN %D4%A WORD%S%T%A AT%S%T%S%O6'),
762 : 2188 !'BEGAN YYY WORD WRITE AT SECTOR ZZZZZZ'
763 : 2189 !'BEGAN YYY WORD READ AT SECTOR ZZZZZZ'
764 : 2190 !'BEGAN YYY WORD WRITE CHECK AT SECTOR ZZZZZZ'
765 : 2191 FMT7 = uplit (%asciz'%X%S2%T%A:%S%D5'),
766 : 2192 !' SOFT ERROR COUNT: DDDDD'
767 : 2193 !' HARD ERROR COUNT: DDDDD'
768 : 2194 !' TRANSFER RETRIES: DDDDD'
769 : 2195 FMT8 = uplit (%asciz'%X%A%TRANSFER RATE: %T%A MBYTES/SECOND'),
770 : 2196 !'TRANSFER RATE: X MBYTES/SECOND'
771 : 2197 FMT9 = uplit (%asciz'%X%S4%A%ARRAY%D3%A:%S%D5'),
772 : 2198 !' ARRAY XX: YYYYY'
773 : 2199 FMT10A = uplit (%asciz'%X%T%A:%S2%T%S%O6%S2%A%BOARD%D3%S2%A%BANK%D2'),
774 : 2200 !'FAILED: SECTOR XXXXXX BOARD YY BANK Z
775 : 2201 FMT10B = uplit (%asciz'%S2%A%BIT%D3'),
776 : 2202 !' BIT QQ'
```

LSAU ADD UNIT SECTION

27-Mar-1982 19:24:42
27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
PA:<NEALE>MLX4.BLI.5 (3)

778 :MLX4
779 :
780 :
781 :
782 :
783 :
784 :
785 :
786 :
787 :
788 :
789 :
790 :
791 :
792 :
793 :
794 :
795 :
796 :
797 :
798 :
799 :
800 :
801 :
802 :
803 :
804 :
805 :
806 :
807 :
808 :
809 :
810 :
811 :
812 :
813 :
814 :
815 :
816 :
817 :
818 :
819 :
820 :
821 :
822 :
823 :
824 :
825 :
826 :
827 :
828 :
829 :
830 :
831 :
832 :

MESSAGES AND PRINT FORMATS

FMT11 = uplit (%asciz'%NX%T%S%O6% EXCEEDS %T%S%O6%'),
! 'TOP SECTOR OF %XXXXX% EXCEEDS SYSTEM LIMIT OF %YYYYY%'
! 'LOW SECTOR OF %XXXXX% EXCEEDS TOP SECTOR OF %YYYYY%'
FMT12A = uplit (%asciz'%NX%AGOOD DATA: %O6% AT LOCATION %O6%'),
! 'GOOD DATA: %XXXXX% AT LOCATION %YYYYY%'
FMT12B = uplit (%asciz'%NX%ABAD DATA: %O6% AT LOCATION %O6%'),
! 'BAD DATA: %XXXXX% AT LOCATION %YYYYY%'
FMT13 = uplit (%asciz'%NX%AARRAY%D3%S2%T%'),
! 'ARRAY XX --> RUN ML11 FROM MAINTENANCE PROGRAM'
FMT14 = uplit (%asciz'%NX%D5%S%T%'),
! 'XXXXX MBYTES WRITTEN'
! 'XXXXX MBYTES READ'
! 'XXXXX MBYTES WRITE CHECKED'
FMT15 = uplit (%asciz'%S2%T%'),
! ' ECH'
! ' NED'
! (ANY OF THE ERROR BITS)
CRLF = uplit (%asciz'%N%'),

MESSAGE MAPS:

SAY1 = uplit (%asciz'%NX%T%'),
SAY2 = uplit (%asciz'%NX%T%S%T%'),
SAY3 = uplit (%asciz'%NX%T%S%T%S%T%'),
SAY4 = uplit (%asciz'%NX%T%S%T%S%T%S%T%'),
SAY5 = uplit (%asciz'%NX%T%S%T%S%T%S%T%S%T%'),

WORDS:

WRD2 = uplit (%asciz%'BEGIN%'),
WRD3 = uplit (%asciz%'END%'),
WRD4 = uplit (%asciz%'PASS%'),
WRD6 = uplit (%asciz%'ML11-A%'),
WRD7 = uplit (%asciz%'ML11-B%'),
WRD11 = uplit (%asciz%'DRIVE%'),
WRD15 = uplit (%asciz%'SECTOR%'),
WRD16 = uplit (%asciz%'WRITE%'),
WRD17 = uplit (%asciz%'READ%'),
WRD18 = uplit (%asciz%'RETRY%'),
WRD19 = uplit (%asciz%'SUCCEEDED%'),
WRD20 = uplit (%asciz%'FAILED%'),
WRD21 = uplit (%asciz%'DROPPED%'),
WRD24 = uplit (%asciz%'UP%'),
WRD25 = uplit (%asciz%'DOWN%'),
WRD34 = uplit (%asciz%'RUNNING%'),
WRD35 = uplit (%asciz%'MARCHING: %'),
WRD36 = uplit (%asciz%'ENABLED%'),
WRD37 = uplit (%asciz%'DISABLED%'),
WRD38 = uplit (%asciz%'ECC%'),
WRD40 = uplit (%asciz%'WITH%'),
WRD41 = uplit (%asciz%'AND%'),
!

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (3)

834 :MLX4
 835 :
 836 :
 837 :
 838 :
 839 :
 840 :
 841 :
 842 :
 843 :
 844 :
 845 :
 846 :
 847 :
 848 :
 849 :
 850 :
 851 :
 852 :
 853 :
 854 :
 855 :
 856 :
 857 :
 858 :
 859 :
 860 :
 861 :
 862 :
 863 :
 864 :
 865 :
 866 :
 867 :
 868 :
 869 :
 870 :
 871 :
 872 :
 873 :
 874 :
 875 :
 876 :
 877 :
 878 :
 879 :
 880 :
 881 :
 882 :
 883 :
 884 :
 885 :
 886 :
 887 :
 888 :

MESSAGES AND PRINT FORMATS

ML-11 BITS:

MLB2 = uplit (%asciz'NED'),
 MLB3 = uplit (%asciz'NEM'),
 MLB4 = uplit (%asciz'PGE'),
 MLB5 = uplit (%asciz'DLT'),
 MLB6 = uplit (%asciz'WCE'),
 MLB7 = uplit (%asciz'PE'),
 MLB8 = uplit (%asciz'MXF'),
 MLB9 = uplit (%asciz'MDPE'),
 MLB10 = uplit (%asciz'MCPE'),
 MLB11 = uplit (%asciz'LNS'),
 MLB12 = uplit (%asciz'IAE'),
 MLB13 = uplit (%asciz'AOE'),
 MLB14 = uplit (%asciz'RMR'),
 MLB15 = uplit (%asciz'ILR'),
 MLB16 = uplit (%asciz'ILF'),
 MLB17 = uplit (%asciz'OPI'),
 MLB18 = uplit (%asciz'DPAR'),
 MLB19 = uplit (%asciz'CPAR'),
 MLB20 = uplit (%asciz'DCK'),
 MLB21 = uplit (%asciz'ECH'),
 MLB22 = uplit (%asciz'CRC'),
 MLB23 = uplit (%asciz'SGL'),
 MLB24 = uplit (%asciz'UNC').

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'),
 RTN5B = uplit (%asciz'RAND2'),
 RTN5C = uplit (%asciz'RAND3'),
 RTN5D = uplit (%asciz'RAND4').

!RANDOM DATA
 !DATA & WORD COUNTS
 !DATA, WORD COUNTS & SECTORS
 !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').

890 :MLX4
891 :
892 :
893 : 2307
894 : 2308
895 : 2309
896 : 2310
897 : 2311
898 : 2312
899 : 2313
900 : 2314
901 : 2315
902 : 2316
903 : 2317
904 : 2318
905 : 2319
906 : 2320
907 : 2321
908 : 2322
909 : 2323
910 : 2324
911 : 2325
912 : 2326
913 : 2327
914 : 2328
915 : 2329
916 : 2330
917 : 2331
918 : 2332
919 : 2333
920 : 2334
921 : 2335
922 : 2336
923 : 2337
924 : 2338
925 : 2339
926 : 2340
927 : 2341
928 : 2342
929 : 2343

MESSAGES AND PRINT FORMATS

27-Mar-1982 19:24:42
27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
PA:<NEALE>MLX4.BLI.5 (3)

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'),

DROP MESSAGES:

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'(DPIVE 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');

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (4)

```

931 :MLX4
932 :
933 :
934 :      2344 %sbttl 'ML11 INTERRUPT SERVICE ROUTINE'
935 :      235  BGNSRV (SERVICE);
936 :      236  I AM DONE = ACTIVE;
937 :      247  -RDSRV;
  
```

```

941
942
943
944 005662      040      040      040 P.AAA: .ASCII / /
945 005665      040      040      011 .ASCII / /<11>
946 005670      011      123      111 .ASCII <11>/SI/
947 005673      116      107      114 .ASCII /NGL/
948 005676      105      040      102 .ASCII /E B/
949 005701      111      124      040 .ASCII /IT /
950 005704      105      122      122 .ASCII /ERR/
951 005707      117      122      040 .ASCII /OR /
952 005712      114      117      107 .ASCII /LOG/
953 005715      040      123      125 .ASCII / SU/
954 005720      115      115      101 .ASCII /MA/
955 005723      122      131      040 .ASCII /RY /
956 005726      122      105      120 .ASCII /REP/
957 005731      117      122      124 .ASCII /ORT/
958 005734      000      000      .ASCII <00><00>
959 005736      045      116      045 P.AAB: .ASCII /XZ/
960 005741      104      061      045 .ASCII /D1Z/
961 005744      101      050      104 .ASCII /A(D/
962 005747      051      040      040 .ASCII /) /
963 005752      040      040      040 .ASCII / /
964 005755      040      045      104 .ASCII / ZD/
965 005760      062      045      101 .ASCII /2ZA/
966 005763      050      104      051 .ASCII / (D)/
967 005766      040      040      040 .ASCII / /
968 005771      040      040      040 .ASCII / /
969 005774      045      104      061 .ASCII /ZD1/
970 005777      045      101      050 .ASCII /ZA(/
971 006002      104      051      040 .ASCII /D) /
972 006005      040      040      040 .ASCII / /
973 006010      040      040      045 .ASCII / Z/
974 006013      104      062      045 .ASCII /D2Z/
975 006016      101      050      104 .ASCII /A(D/
976 006021      051      040      040 .ASCII /) /
977 006024      040      040      040 .ASCII / /
978 006027      040      040      045 .ASCII / Z/
979 006032      104      066      045 .ASCII /D6Z/
980 006035      101      050      104 .ASCII /A(D/
981 006040      051      000      .ASCII /)/<00>
982 006042      125      116      111 P.AAC: .ASCII /UNI/
983 006045      124      040      043 .ASCII /T #/
  
```

```

984 :MLX4
985 :
986 :
987 006050      072      040      040 .ASCII /: /
988 006053      040      101      122 .ASCII / AR/
989 006056      122      101      131 .ASCII /RAY/
990 006061      040      043      072 .ASCII / #: /
  
```

27-Mar-1982 19:24:42 TOPS
 27-Mar-1982 19:23:44 PA:<

991	006064	040	040	040	.ASCII	/ /
992	006067	102	101	116	.ASCII	/BAN/
993	006072	113	040	043	.ASCII	/K #/
994	006075	072	040	040	.ASCII	/: /
995	006100	040	102	111	.ASCII	/BI/
996	006103	124	040	043	.ASCII	/T #/
997	006106	072	040	040	.ASCII	/: /
998	006111	040	040	040	.ASCII	/ /
999	006114	040	103	117	.ASCII	/CO/
1000	006117	125	116	124	.ASCII	/UNIT/
1001	006122	040	043	072	.ASCII	/ #:/
1002	006125	000			.ASCII	<00>
1003	006126	045	116	045	P.AAD:	.ASCII /XN%/
1004	006131	124	045	123		.ASCII /T%#/
1005	006134	062	045	104		.ASCII /2%D/
1006	006137	062	000	000		.ASCII /2/<00><00>
1007	006142	045	123	067	P.AAE:	.ASCII /%S7/
1008	006145	045	124	045		.ASCII /%T%/
1009	006150	123	045	101		.ASCII /%XA/
1010	006153	055	045	104		.ASCII /-%D/
1011	006156	062	000			.ASCII /2/<00>
1012	006160	045	123	062	P.AAF:	.ASCII /%S2/
1013	006163	045	124	000		.ASCII /%T/<00>
1014	006166	045	116	045	P.AAG:	.ASCII /XN%/
1015	006171	101	052	052		.ASCII /A**/
1016	006174	045	123	045		.ASCII /%S%/
1017	006177	124	045	123		.ASCII /T%#/
1018	006202	045	124	045		.ASCII /%T%/
1019	006205	104	063	045		.ASCII /D3%/
1020	006210	123	045	101		.ASCII /%XA/
1021	006213	052	052	000		.ASCII /**/<00>
1022	006216	045	116	062	P.AAH:	.ASCII /%N2/
1023	006221	045	124	045		.ASCII /%T%/
1024	006224	101	072	045		.ASCII /A:%/
1025	006227	123	045	104		.ASCII /%SD/
1026	006232	061	045	123		.ASCII /1%S/
1027	006235	064	045	124		.ASCII /4%T/
1028	006240	045	101	072		.ASCII /%A:/
1029	006243	045	123	045		.ASCII /%S%/
1030	006246	104	061	000		.ASCII /D1/<00>
1031	006251	000				.ASCII <00>
1032	006252	045	123	064	P.AAI:	.ASCII /%S4/
1033	006255	045	124	045		.ASCII /%T%/
1034	006260	101	072	045		.ASCII /A:%/
1035	006263	123	045	117		.ASCII /%S0/
1036	006266	066	000			.ASCII /6/<00>
1037	006270	045	123	064	P.AAJ:	.ASCII /%S4/
1038	006273	045	124	045		.ASCII /%T%/

27-Mar-1982 19:24:42 TUPS
27-Mar-1982 19:23:44 PA:<

```
1040          :MLX4
1041          :
1042          :
1043 006276    101    072    045    .ASCII /A:Z/
1044 006301    123    045    104    .ASCII /SXD/
1045 006304    061    045    104    .ASCII /1XD/
1046 006307    061    045    104    .ASCII /1XD/
1047 006312    061    045    104    .ASCII /1XD/
1048 006315    061    000    000    .ASCII /1/<00><00>
1049 006320    045    116    045    P.AAK: .ASCII /XNZ/
1050 006323    124    045    123    .ASCII /TXS/
1051 006326    063    045    101    .ASCII /3XA/
1052 006331    123    105    103    .ASCII /SEC/
1053 006334    124    117    122    .ASCII /TOR/
1054 006337    123    040    125    .ASCII /S U/
1055 006342    116    104    105    .ASCII /NDE/
1056 006345    122    040    124    .ASCII /R T/
1057 006350    105    123    124    .ASCII /EST/
1058 006353    072    045    123    .ASCII /:XS/
1059 006356    045    117    066    .ASCII /X06/
1060 006361    045    123    045    .ASCII /XSX/
1061 006364    101    124    117    .ASCII /ATO/
1062 006367    045    123    045    .ASCII /XSX/
1063 006372    117    066    000    .ASCII /06/<00>
1064 006375    000          .ASCII <00>
1065 006376    045    116    045    P.AAL: .ASCII /XNZ/
1066 006401    101    102    105    .ASCII /ABE/
1067 006404    107    101    116    .ASCII /GAN/
1068 006407    040    045    104    .ASCII / XD/
1069 006412    064    045    101    .ASCII /4XA/
1070 006415    040    127    117    .ASCII / WO/
1071 006420    122    104    045    .ASCII /RDZ/
1072 006423    123    045    124    .ASCII /SXT/
1073 006426    045    101    040    .ASCII /XA /
1074 006431    101    124    045    .ASCII /ATZ/
1075 006434    123    045    124    .ASCII /SXT/
1076 006437    045    123    045    .ASCII /XSX/
1077 006442    117    066    000    .ASCII /06/<00>
1078 006445    000          .ASCII <00>
1079 006446    045    116    045    P.AAM: .ASCII /XNZ/
1080 006451    123    062    045    .ASCII /S2Z/
1081 006454    124    045    101    .ASCII /TZA/
1082 006457    072    045    123    .ASCII /:XS/
1083 006462    045    104    065    .ASCII /XD5/
1084 006465    000          .ASCII <00>
1085 006466    045    116    045    P.AAN: .ASCII /XNZ/
1086 006471    101    124    122    .ASCII /ATR/
1087 006474    101    116    123    .ASCII /ANS/
1088 006477    106    105    122    .ASCII /FER/
1089 006502    040    122    101    .ASCII / RA/
1090 006505    124    105    072    .ASCII /TE:/
1091 006510    040    045    124    .ASCII / XT/
1092 006513    045    101    040    .ASCII /XA /
1093 006516    115    102    131    .ASCII /MBY/
1094 006521    124    105    123    .ASCII /TES/
```

```
1096          :MLX4
1097          :
1098          :
1099 006524    057    123    105          .ASCII <S7>/SE/
1100 006527    103    117    116          .ASCII /CON/
1101 006532    104    000          .ASCII /D/<00>
1102 006534    045    116    045 P.AAO: .ASCII /XNZ/
1103 006537    123    064    045          .ASCII /S4%/
1104 006542    101    101    122          .ASCII /AAR/
1105 006545    122    101    131          .ASCII /RAY/
1106 006550    045    104    063          .ASCII /XD3/
1107 006553    045    101    072          .ASCII /XA:/
1108 006556    045    123    045          .ASCII /XS%/
1109 006561    104    065    000          .ASCII /D5/<00>
1110 006564    045    116    045 P.AAP: .ASCII /XNZ/
1111 006567    124    045    101          .ASCII /TXA/
1112 006572    072    045    123          .ASCII /:XS/
1113 006575    062    045    124          .ASCII /2XT/
1114 006600    045    123    045          .ASCII /XS%/
1115 006603    117    066    045          .ASCII /O6%/
1116 006606    123    062    045          .ASCII /S2%/
1117 006611    101    102    117          .ASCII /ABO/
1118 006614    101    122    104          .ASCII /ARD/
1119 006617    045    104    063          .ASCII /XD3/
1120 006622    045    123    062          .ASCII /XS2/
1121 006625    045    101    102          .ASCII /XAB/
1122 006630    101    116    113          .ASCII /ANK/
1123 006633    045    104    062          .ASCII /XD2/
1124 006636    000    000          .ASCII <00><00>
1125 006640    045    123    062 P.AAQ: .ASCII /XS2/
1126 006643    045    101    102          .ASCII /XAB/
1127 006646    111    124    045          .ASCII /IT%/
1128 006651    104    063    000          .ASCII /D3/<00>
1129 006654    045    116    045 P.AAR: .ASCII /XNZ/
1130 006657    124    045    123          .ASCII /TXS/
1131 006662    045    117    066          .ASCII /XO6/
1132 006665    045    101    040          .ASCII /XA /
1133 006670    105    130    103          .ASCII /EXC/
1134 006673    105    105    104          .ASCII /EED/
1135 006676    123    040    045          .ASCII /S %/
1136 006701    124    045    123          .ASCII /TXS/
1137 006704    045    117    066          .ASCII /XO6/
1138 006707    000          .ASCII <00>
1139 006710    045    116    045 P.AAS: .ASCII /XNZ/
1140 006713    101    107    117          .ASCII /AGO/
1141 006716    117    104    040          .ASCII /OD /
1142 006721    104    101    124          .ASCII /DAT/
1143 006724    101    072    040          .ASCII /A: /
1144 006727    045    117    066          .ASCII /XO6/
1145 006732    045    101    040          .ASCII /XA /
1146 006735    101    124    040          .ASCII /AT /
1147 006740    114    117    103          .ASCII /LOC/
1148 006743    101    124    111          .ASCII /ATI/
1149 006746    117    116    040          .ASCII /ON /
1150 006751    045    117    066          .ASCII /XO6/
```

```

1152          :MLX4
1153          :
1154          :
1155 006754    000    000          .ASCII <00><00>
1156 006756    045    116    045 P.AAT: .ASCII /XNZ/
1157 006761    101    102    101    .ASCII /ABA/
1158 006764    104    040    104    .ASCII /D D/
1159 006767    101    124    101    .ASCII /ATA/
1160 006772    072    040    040    .ASCII /: /
1161 006775    045    117    066    .ASCII /X06/
1162 007000    045    101    040    .ASCII /XA /
1163 007003    101    124    040    .ASCII /AT /
1164 007006    114    117    103    .ASCII /LOC/
1165 007011    101    124    111    .ASCII /ATI/
1166 007014    117    116    040    .ASCII /ON /
1167 007017    045    117    066    .ASCII /X06/
1168 007022    000    000          .ASCII <00><00>
1169 007024    045    116    045 P.AAU: .ASCII /XNZ/
1170 007027    101    101    122    .ASCII /AAR/
1171 007032    122    101    131    .ASCII /RAY/
1172 007035    045    104    063    .ASCII /XD3/
1173 007040    045    123    062    .ASCII /XS2/
1174 007043    045    124    000    .ASCII /XT/<00>
1175 007046    045    116    045 P.AAV: .ASCII /XNZ/
1176 007051    104    065    045    .ASCII /D5X/
1177 007054    123    045    124    .ASCII /SXT/
1178 007057    000          .ASCII <00>
1179 007060    045    123    062 P.AAW: .ASCII /XS2/
1180 007063    045    124    000    .ASCII /XT/<00>
1181 007066    045    116    000 P.AAX: .ASCII /XN/<00>
1182 007071    000          .ASCII <00>
1183 007072    045    116    045 P.AAY: .ASCII /XNZ/
1184 007075    124    000    000    .ASCII /T/<00><00>
1185 007100    045    116    045 P.AAZ: .ASCII /XNZ/
1186 007103    124    045    123    .ASCII /TXS/
1187 007106    045    124    000    .ASCII /XT/<00>
1188 007111    000          .ASCII <00>
1189 007112    045    116    045 P.ABA: .ASCII /XNZ/
1190 007115    124    045    123    .ASCII /TXS/
1191 007120    045    124    045    .ASCII /XTX/
1192 007123    123    045    124    .ASCII /SXT/
1193 007126    000    000          .ASCII <00><00>
1194 007130    045    116    045 P.ABB: .ASCII /XNZ/
1195 007133    124    045    123    .ASCII /TXS/
1196 007136    045    124    045    .ASCII /XTX/
1197 007141    123    045    124    .ASCII /SXT/
1198 007144    045    123    045    .ASCII /XSX/
1199 007147    124    000    000    .ASCII /T/<00><00>
1200 007152    045    116    045 P.ABC: .ASCII /XNZ/
1201 007155    124    045    123    .ASCII /TXS/
1202 007160    045    124    045    .ASCII /XTX/
1203 007163    123    045    124    .ASCII /SXT/
1204 007166    045    123    045    .ASCII /XSX/
1205 007171    124    045    123    .ASCII /TXS/
1206 007174    045    124    000    .ASCII /XT/<00>

```

```
1208 ;MLX4
1209 ;
1210 ;
1211 007177 000
1212 007200 102 105 107 P.ABD: .ASCII <00>
1213 007203 111 116 000 .ASCII /BEG/
1214 007206 105 116 104 P.ABE: .ASCII /IN/<00>
1215 007211 000 .ASCII /END/
1216 007212 120 101 123 P.ABF: .ASCII <00>
1217 007215 123 000 000 .ASCII /PAS/
1218 007220 115 114 061 P.ABG: .ASCII /S/<00><00>
1219 007223 061 055 101 .ASCII /ML1/
1220 007226 000 000 .ASCII /1-A/
1221 007230 115 114 061 P.ABH: .ASCII <00><00>
1222 007233 061 055 102 .ASCII /ML1/
1223 007236 000 000 .ASCII /1-B/
1224 007240 104 122 111 P.ABI: .ASCII <00><00>
1225 007243 126 105 000 .ASCII /DRI/
1226 007246 123 105 103 P.ABJ: .ASCII /VE/<00>
1227 007251 124 117 122 .ASCII /SEC/
1228 007254 000 000 .ASCII /TOR/
1229 007256 127 122 111 P.ABK: .ASCII <00><00>
1230 007261 124 105 000 .ASCII /WRI/
1231 007264 122 105 101 P.ABL: .ASCII /TE/<00>
1232 007267 104 000 000 .ASCII /REA/
1233 007272 122 105 124 P.ABM: .ASCII /D/<00><00>
1234 007275 122 131 000 .ASCII /RET/
1235 007300 123 125 103 P.ABN: .ASCII /RY/<00>
1236 007303 103 105 105 .ASCII /SUC/
1237 007306 104 105 104 .ASCII /CEE/
1238 007311 000 .ASCII /DED/
1239 007312 106 101 111 P.ABO: .ASCII <00>
1240 007315 114 105 104 .ASCII /FAI/
1241 007320 000 000 .ASCII /LED/
1242 007322 104 122 117 P.ABP: .ASCII <00><00>
1243 007325 120 120 105 .ASCII /DRO/
1244 007330 104 000 .ASCII /PPE/
1245 007332 125 120 000 P.ABQ: .ASCII /D/<00>
1246 007335 000 .ASCII /UP/<00>
1247 007336 104 117 127 P.ABR: .ASCII <00>
1248 007341 116 000 000 .ASCII /DOW/
1249 007344 122 125 116 P.ABS: .ASCII /N/<00><00>
1250 007347 116 111 116 .ASCII /RUN/
1251 007352 107 000 .ASCII /NIN/
1252 007354 115 101 122 P.ABT: .ASCII /G/<00>
1253 007357 103 110 111 .ASCII /MAR/
1254 007362 116 107 072 .ASCII /CHI/
1255 007365 040 040 000 .ASCII /NG:/
1256 007370 105 116 101 P.ABU: .ASCII / /<00>
1257 007373 102 114 105 .ASCII /ENA/
1258 007376 104 000 .ASCII /BLE/
1259 007400 104 111 123 P.ABV: .ASCII /D/<00>
1260 007403 101 102 114 .ASCII /DIS/
1261 007406 105 104 000 .ASCII /ABL/
1262 007411 000 .ASCII /ED/<00>
      .ASCII <00>
```

```

1264          :MLX4
1265          :
1266          : ML11 INTERRUPT SERVICE ROUTINE
1267 007412    105    103    103 P.ABW: .ASCII /ECC/
1268 007415    000          .ASCII <00>
1269 007416    127    111    124 P.ABX: .ASCII /WIT/
1270 007421    110    000    000 .ASCII /H/<00><00>
1271 007424    101    116    104 P.ABY: .ASCII /AND/
1272 007427    000          .ASCII <00>
1273 007430    116    105    104 P.ABZ: .ASCII /NED/
1274 007433    000          .ASCII <00>
1275 007434    116    105    115 P.ACA: .ASCII /NEM/
1276 007437    000          .ASCII <00>
1277 007440    120    107    105 P.ACB: .ASCII /PGE/
1278 007443    000          .ASCII <00>
1279 007444    104    114    124 P.ACC: .ASCII /DLT/
1280 007447    000          .ASCII <00>
1281 007450    127    103    105 P.ACD: .ASCII /WCE/
1282 007453    000          .ASCII <00>
1283 007454    120    105    000 P.ACE: .ASCII /PE/<00>
1284 007457    000          .ASCII <00>
1285 007460    115    130    106 P.ACF: .ASCII /MXF/
1286 007463    000          .ASCII <00>
1287 007464    115    104    120 P.ACG: .ASCII /MDP/
1288 007467    105    000    000 .ASCII /E/<00><00>
1289 007472    115    103    120 P.ACH: .ASCII /MCP/
1290 007475    105    000    000 .ASCII /E/<00><00>
1291 007500    125    116    123 P.ACI: .ASCII /UNS/
1292 007503    000          .ASCII <00>
1293 007504    111    101    105 P.ACJ: .ASCII /IAE/
1294 007507    000          .ASCII <00>
1295 007510    101    117    105 P.ACK: .ASCII /AOE/
1296 007513    000          .ASCII <00>
1297 007514    122    115    122 P.ACL: .ASCII /RMR/
1298 007517    000          .ASCII <00>
1299 007520    111    114    122 P.ACM: .ASCII /ILR/
1300 007523    000          .ASCII <00>
1301 007524    111    114    106 P.ACN: .ASCII /ILF/
1302 007527    000          .ASCII <00>
1303 007530    117    120    111 P.ACO: .ASCII /OPI/
1304 007533    000          .ASCII <00>
1305 007534    104    120    101 P.ACP: .ASCII /DPA/
1306 007537    122    000    000 .ASCII /R/<00><00>
1307 007542    103    120    101 P.ACQ: .ASCII /CPA/
1308 007545    122    000    000 .ASCII /R/<00><00>
1309 007550    104    103    113 P.ACR: .ASCII /DCK/
1310 007553    000          .ASCII <00>
1311 007554    105    103    110 P.ACS: .ASCII /ECH/
1312 007557    000          .ASCII <00>
1313 007560    103    122    103 P.ACT: .ASCII /CRC/
1314 007563    000          .ASCII <00>
1315 007564    123    107    114 P.ACU: .ASCII /SGL/
1316 007567    000          .ASCII <00>
1317 007570    125    116    103 P.ACV: .ASCII /UNC/
1318 007573    000          .ASCII <00>

```



```

1320          :MLX4
1321          :
1322          :
1323 007574    103    117    115 P.ACW: .ASCII /COM/
1324 007577    115    101    116       .ASCII /MAN/
1325 007602    104    040    111       .ASCII /D I/
1326 007605    116    124    105       .ASCII /NTE/
1327 007610    107    122    111       .ASCII /GRI/
1328 007613    124    131    040       .ASCII /TY /
1329 007616    122    117    125       .ASCII /ROU/
1330 007621    124    111    116       .ASCII /TIN/
1331 007624    105    000          .ASCII /E/<00>
1332 007626    117    120    124 P.ACX: .ASCII /OPT/
1333 007631    061    000    000       .ASCII /1/<00><00>
1334 007634    117    120    124 P.ACY: .ASCII /OPT/
1335 007637    062    000    000       .ASCII /2/<00><00>
1336 007642    117    120    124 P.ACZ: .ASCII /OPT/
1337 007645    063    000    000       .ASCII /3/<00><00>
1338 007650    117    120    124 P.ADA: .ASCII /OPT/
1339 007653    064    000    000       .ASCII /4/<00><00>
1340 007656    117    120    124 P.ADB: .ASCII /OPT/
1341 007661    065    000    000       .ASCII /5/<00><00>
1342 007664    122    101    116 P.ADC: .ASCII /RAN/
1343 007667    104    061    000       .ASCII /D1/<00>
1344 007672    122    101    116 P.ADD: .ASCII /RAN/
1345 007675    104    062    000       .ASCII /D2/<00>
1346 007700    122    101    116 P.ADE: .ASCII /RAN/
1347 007703    104    063    000       .ASCII /D3/<00>
1348 007706    122    101    116 P.ADF: .ASCII /RAN/
1349 007711    104    064    000       .ASCII /D4/<00>
1350 007714    127    122    111 P.ADG: .ASCII /WRI/
1351 007717    124    105    040       .ASCII /TE /
1352 007722    103    110    105       .ASCII /CHE/
1353 007725    103    113    000       .ASCII /CK/<00>
1354 007730    121    125    111 P.ADH: .ASCII /QUI/
1355 007733    103    113    040       .ASCII /CK /
1356 007736    126    105    122       .ASCII /VER/
1357 007741    111    106    131       .ASCII /IFY/
1358 007744    000    000          .ASCII <00><00>
1359 007746    122    105    106 P.ADI: .ASCII /REF/
1360 007751    122    105    123       .ASCII /RES/
1361 007754    110    040    115       .ASCII /H M/
1362 007757    101    122    107       .ASCII /ARG/
1363 007762    111    116    111       .ASCII /INI/
1364 007765    116    107    000       .ASCII /NG/<00>
1365 007770    103    123    122 P.ADJ: .ASCII /CSR/
1366 007773    040    101    104       .ASCII / AD/
1367 007776    104    122    105       .ASCII /DRE/
1368 010001    123    123    000       .ASCII /SS/<00>
1369 010004    123    117    106 P.ADK: .ASCII /SOF/
1370 010007    124    040    105       .ASCII /T E/
1371 010012    122    122    117       .ASCII /RRO/
1372 010015    122    040    103       .ASCII /R C/
1373 010020    117    125    116       .ASCII /OUN/
1374 010023    124    000    000       .ASCII /T/<00><00>
  
```

27-Mar-1982 19:24:42 TOPS
27-Mar-1982 19:23:44 PA:<

```
1376          :MLX4
1377          :
1378          :
1379 010026    124    122    101 P.ADL: .ASCII /TRA/
1380 010031    116    123    106      .ASCII /NSF/
1381 010034    105    122    040      .ASCII /ER /
1382 010037    122    105    124      .ASCII /RET/
1383 010042    122    111    105      .ASCII /RIE/
1384 010045    123    000    000      .ASCII /S/<00><00>
1385 010050    114    117    107 P.ADM: .ASCII /LOG/
1386 010053    111    103    101      .ASCII /ICA/
1387 010056    114    040    125      .ASCII /L U/
1388 010061    116    111    124      .ASCII /NIT/
1389 010064    000    000      .ASCII <00><00>
1390 010066    123    105    122 P.ADN: .ASCII /SER/
1391 010071    111    101    114      .ASCII /IAL/
1392 010074    040    043    000      .ASCII / #/<00>
1393 010077    000      .ASCII <00>
1394 010100    120    101    124 P.ADO: .ASCII /PAT/
1395 010103    124    105    122      .ASCII /TER/
1396 010106    116    040    116      .ASCII /N N/
1397 010111    125    115    102      .ASCII /UMB/
1398 010114    105    122    000      .ASCII /ER/<00>
1399 010117    000      .ASCII <00>
1400 010120    105    122    122 P.ADP: .ASCII /ERR/
1401 010123    117    122    040      .ASCII /OR /
1402 010126    102    111    124      .ASCII /BIT/
1403 010131    123    040    123      .ASCII /S S/
1404 010134    105    124    072      .ASCII /ET:/
1405 010137    000      .ASCII <00>
1406 010140    123    103    040 P.ADQ: .ASCII /SC /
1407 010143    123    105    124      .ASCII /SET/
1408 010146    040    102    125      .ASCII / BU/
1409 010151    124    040    116      .ASCII /T N/
1410 010154    117    040    123      .ASCII /O S/
1411 010157    131    123    124      .ASCII /YST/
1412 010162    105    115    040      .ASCII /EM /
1413 010165    105    122    122      .ASCII /ERR/
1414 010170    117    122    123      .ASCII /ORS/
1415 010173    040    106    117      .ASCII / FO/
1416 010176    125    116    104      .ASCII /UND/
1417 010201    000      .ASCII <00>
1418 010202    116    125    115 P.ADR: .ASCII /NUM/
1419 010205    102    105    122      .ASCII /BER/
1420 010210    040    117    106      .ASCII / OF/
1421 010213    040    115    102      .ASCII / MB/
1422 010216    131    124    105      .ASCII /YTE/
1423 010221    123    040    124      .ASCII /S T/
1424 010224    122    101    116      .ASCII /RAN/
1425 010227    123    106    105      .ASCII /SFE/
1426 010232    122    105    104      .ASCII /RED/
1427 010235    072    000    000      .ASCII /:/<00><00>
1428 010240    115    102    131 P.ADS: .ASCII /MBY/
1429 010243    124    105    123      .ASCII /TES/
1430 010246    040    127    122      .ASCII / WR/
```

```

1432          :MLX4
1433          :
1434          :
1435 010251    111    124    105    .ASCII /ITE/
1436 010254    040    103    110    .ASCII / CH/
1437 010257    105    103    113    .ASCII /ECK/
1438 010262    105    104    000    .ASCII /ED/<00>
1439 010265    000          .ASCII <00>
1440 010266    124    117    120 P.ADT: .ASCII /TOP/
1441 010271    040    123    105    .ASCII / SE/
1442 010274    103    124    117    .ASCII /CTO/
1443 010277    122    040    117    .ASCII /R O/
1444 010302    106    000    .ASCII /F/<00>
1445 010304    114    117    127 P.ADU: .ASCII /LOW/
1446 010307    040    123    105    .ASCII / SE/
1447 010312    103    124    117    .ASCII /CTO/
1448 010315    122    040    117    .ASCII /R O/
1449 010320    106    000    .ASCII /F/<00>
1450 010322    123    131    123 P.ADV: .ASCII /SYS/
1451 010325    124    105    115    .ASCII /TEM/
1452 010330    040    114    111    .ASCII / LI/
1453 010333    115    111    124    .ASCII /MIT/
1454 010336    040    117    106    .ASCII / OF/
1455 010341    000          .ASCII <00>
1456 010342    120    105    122 P.ADW: .ASCII /PER/
1457 010345    106    117    122    .ASCII /FOR/
1458 010350    115    101    116    .ASCII /MAN/
1459 010353    103    105    040    .ASCII /CE /
1460 010356    123    125    115    .ASCII /SUM/
1461 010361    115    101    122    .ASCII /MAR/
1462 010364    131    000    .ASCII /Y/<00>
1463 010366    110    101    122 P.ADX: .ASCII /HAR/
1464 010371    104    040    105    .ASCII /D E/
1465 010374    122    122    117    .ASCII /RRO/
1466 010377    122    040    103    .ASCII /R C/
1467 010402    117    125    116    .ASCII /OUN/
1468 010405    124    000    000    .ASCII /T/<00><00>
1469 010410    115    102    131 P.ADY: .ASCII /MBY/
1470 010413    124    105    123    .ASCII /TES/
1471 010416    040    127    122    .ASCII / WR/
1472 010421    111    124    124    .ASCII /ITT/
1473 010424    105    116    000    .ASCII /EN/<00>
1474 010427    000          .ASCII <00>
1475 010430    115    102    131 P.ADZ: .ASCII /MBY/
1476 010433    124    105    123    .ASCII /TES/
1477 010436    040    122    105    .ASCII / RE/
1478 010441    101    104    000    .ASCII /AD/<00>
1479 010444    062    000    P.AEA: .ASCII /2/<00>
1480 010446    061    000    P.AEB: .ASCII /1/<00>
1481 010450    056    065    000 P.AEC: .ASCII /.5/<00>
1482 010453    000          .ASCII <00>
1483 010454    056    062    065 P.AED: .ASCII /.25/
1484 010457    000          .ASCII <00>
1485 010460    050    116    117 P.AEE: .ASCII /<NO/
1486 010463    124    040    120    .ASCII /T P/
    
```

```

1488      :MLX4
1489      :
1490
1491 01066 117 127 105 .ASCII /OWE/
1492 010471 122 105 104 .ASCII /RED/
1493 010474 040 125 120 .ASCII / UP/
1494 010477 051 000 000 .ASCII /)/<00><00>
1495 010502 050 116 117 P.AEF: .ASCII /(<NO/
1496 010505 124 040 101 .ASCII /T A/
1497 010510 116 040 115 .ASCII /N M/
1498 010513 114 061 061 .ASCII /L11/
1499 010516 040 125 116 .ASCII / UN/
1500 010521 111 124 051 .ASCII /IT)/
1501 010524 000 000 .ASCII <00><00>
1502 010526 050 117 120 P.AEG: .ASCII /(<OP/
1503 010531 105 122 101 .ASCII /ERA/
1504 010534 124 117 122 .ASCII /TOR/
1505 010537 040 123 105 .ASCII / SE/
1506 010542 114 105 103 .ASCII /LEC/
1507 010545 124 105 104 .ASCII /TED/
1508 010550 040 124 105 .ASCII / TE/
1509 010553 123 124 040 .ASCII /ST /
1510 010556 114 111 115 .ASCII /LIM/
1511 010561 111 124 123 .ASCII /ITS/
1512 010564 040 111 116 .ASCII / IN/
1513 010567 103 117 122 .ASCII /COR/
1514 010572 122 105 103 .ASCII /REC/
1515 010575 124 114 131 .ASCII /TLY/
1516 010600 051 000 .ASCII /)/<00>
1517 010602 050 101 114 P.AEH: .ASCII /(<AL/
1518 010605 114 040 122 .ASCII /L R/
1519 010610 105 124 122 .ASCII /ETR/
1520 010613 111 105 123 .ASCII /IES/
1521 010616 040 106 101 .ASCII / FA/
1522 010621 111 114 105 .ASCII /ILE/
1523 010624 104 040 106 .ASCII /D F/
1524 010627 117 122 040 .ASCII /OR /
1525 010632 101 040 116 .ASCII /A N/
1526 010635 117 116 055 .ASCII /ON-/
1527 010640 106 101 124 .ASCII /FAT/
1528 010643 101 114 040 .ASCII /AL /
1529 010646 105 122 122 .ASCII /ERR/
1530 010651 117 122 051 .ASCII /OR)/
1531 010654 000 000 .ASCII <00><00>
1532 010656 050 103 117 P.AEI: .ASCII /(<CO/
1533 010661 116 124 122 .ASCII /NTR/
1534 010664 117 114 114 .ASCII /OLL/
1535 010667 105 122 040 .ASCII /ER /
1536 010672 106 101 124 .ASCII /FAT/
1537 010675 101 114 040 .ASCII /AL /
1538 010700 105 122 122 .ASCII /ERR/
1539 010703 117 122 051 .ASCII /OR)/
1540 010706 000 000 .ASCII <00><00>
1541 010710 050 104 122 P.AEJ: .ASCII /(<DR/
1542 010713 111 126 105 .ASCII /IVE/

```

```

1544          ;MLX4
1545          ;
1546          ;
1547 010716   040   106   101   .ASCII / FA/
1548 010721   124   101   114   .ASCII /TAL/
1549 010724   040   105   122   .ASCII / ER/
1550 010727   122   117   122   .ASCII /ROR/
1551 010732   051   000         .ASCII /)/<00>
1552 010734   050   105   103   P.AEK: .ASCII /(EC/
1553 010737   103   040   110   .ASCII /C H/
1554 010742   101   122   104   .ASCII /ARD/
1555 010745   040   105   122   .ASCII / ER/
1556 010750   122   117   122   .ASCII /ROR/
1557 010753   051   000   000   .ASCII /)/<00><00>
1558 010756   050   105   103   P.AEL: .ASCII /(EC/
1559 010761   103   040   114   .ASCII /C L/
1560 010764   117   107   111   .ASCII /OGI/
1561 010767   103   040   106   .ASCII /C F/
1562 010772   101   111   114   .ASCII /AIL/
1563 010775   125   122   105   .ASCII /URE/
1564 011000   051   000         .ASCII /)/<00>
1565 011002   111   116   124   P.AEM: .ASCII /INT/
1566 011005   105   122   122   .ASCII /ERR/
1567 011010   125   120   124   .ASCII /UPT/
1568 011013   040   104   111   .ASCII / DI/
1569 011016   104   040   116   .ASCII /D N/
1570 011021   117   124   040   .ASCII /OT /
1571 011024   117   103   103   .ASCII /OCC/
1572 011027   125   122   054   .ASCII /UR,/
1573 011032   040   102   125   .ASCII / BU/
1574 011035   124   040   124   .ASCII /T T/
1575 011040   110   105   040   .ASCII /HE /
1576 011043   124   122   101   .ASCII /TRA/
1577 011046   116   123   106   .ASCII /NSF/
1578 011051   105   122   040   .ASCII /ER /
1579 011054   111   123   040   .ASCII /IS /
1580 011057   103   117   115   .ASCII /COM/
1581 011062   120   114   105   .ASCII /PLE/
1582 011065   124   105   000   .ASCII /TE/<00>
1583 011070   055   055   076   P.AEN: .ASCII /-->/
1584 011073   040   122   125   .ASCII / RU/
1585 011076   116   040   115   .ASCII /N M/
1586 011101   114   061   061   .ASCII /L11/
1587 011104   040   114   117   .ASCII / LO/
1588 011107   107   111   103   .ASCII /GIC/
1589 011112   040   124   105   .ASCII / TE/
1590 011115   123   124   000   .ASCII /ST/<00>
1591 011120   055   055   076   P.AEO: .ASCII /-->/
1592 011123   040   122   125   .ASCII / RU/
1593 011126   116   040   115   .ASCII /N M/
1594 011131   114   061   061   .ASCII /L11/
1595 011134   040   120   122   .ASCII / PR/
1596 011137   117   115   040   .ASCII /OM /
1597 011142   115   101   111   .ASCII /MAI/
1598 011145   116   124   105   .ASCII /NTE/
  
```

```

1600          :MLX4
1601          :
1602          :
1603 011150   116   101   116          .ASCII /NAN/
1604 011153   103   105   040          .ASCII /CE /
1605 011156   120   122   117          .ASCII /PRO/
1606 011161   107   122   101          .ASCII /GRA/
1607 011164   115   000          .ASCII /M/<00>
1608 011166   123   117   106 P.AEP: .ASCII /SOF/
1609 011171   124   040   105          .ASCII /T E/
1610 011174   122   122   117          .ASCII /RRO/
1611 011177   122   000   000          .ASCII /R/<00><00>
1612 011202   110   101   122 P.AEQ: .ASCII /HAR/
1613 011205   104   040   105          .ASCII /D E/
1614 011210   122   122   117          .ASCII /RRO/
1615 011213   122   000   000          .ASCII /R/<00><00>
1616 011216   105   103   103 P.AER: .ASCII /ECC/
1617 011221   040   114   117          .ASCII / LD/
1618 011224   107   111   103          .ASCII /GIC/
1619 011227   040   106   101          .ASCII / FA/
1620 011232   111   114   105          .ASCII /ILE/
1621 011235   104   040   124          .ASCII /D T/
1622 011240   117   040   104          .ASCII /O D/
1623 011243   105   124   105          .ASCII /ETE/
1624 011246   103   124   040          .ASCII /CT /
1625 011251   104   101   124          .ASCII /DAT/
1626 011254   101   040   105          .ASCII /A E/
1627 011257   122   122   117          .ASCII /RRO/
1628 011262   122   000          .ASCII /R/<00>
1629
1630
1631
1632 011264   SBE.LOG: .BLKW 400
1633 012264   PM.SBE.CNT:
1634 012264   .BLKW 200
1635 012664   SBES.COUNT:
1636 012664   .BLKW 1
1637 012666   BIT.NUM: .BLKW 1
1638 012670   WBUFF: .BLKW 4000
1639 022670   RBUFF: .BLKW 4000
1640 032670   WPTR: .BLKW 1
1641 032672   RPTR: .BLKW 1
1642 032674   QUICK: .BLKW 1
1643 032676   PATTERN: .BLKW 1
1644 032700   DATA.COUNT:
1645 032700   .BLKW 1
1646 032702   COMP.COUNT:
1647 032702   .BLKW 1
1648 032704   EOP.COUNT:
1649 032704   .BLKW 1
1650 032706   BASE.ADDR:
1651 032706   .BLKW 1
1652 032710   VEC: .BLKW 1
1653 032712   BR.LEVEL:

```

27-Mar-1982 19:24:42 TOPS
 27-Mar-1982 19:23:44 PA:<

```

1655          ;MLX4
1656          ;
1657          ;
1658 032712          .BLKW 1
1659 032714          SOFTS: .BLKW 200
1660 033314          HARDS: .BLKW 200
1661 033714          TRIES: .BLKW 200
1662 034314          WR.COUNT:
1663 034314          .BLKW 1
1664 034316          WR.THOUSANDS:
1665 034316          .BLKW 1
1666 034320          WR.MILLIONS:
1667 034320          .BLKW 1
1668 034322          RD.COUNT:
1669 034322          .BLKW 1
1670 034324          RD.THOUSANDS:
1671 034324          .BLKW 1
1672 034326          RD.MILLIONS:
1673 034326          .BLKW 1
1674 034330          WC.COUNT:
1675 034330          .BLKW 1
1676 034332          WC.THOUSANDS:
1677 034332          .BLKW 1
1678 034334          WC.MILLIONS:
1679 034334          .BLKW 1
1680 034336          I.AM.DONE:
1681 034336          .BLKW 1
1682 034340          RETRYING:
1683 034340          .BLKW 1
1684 034342          BOARD: .BLKW 1
1685 034344          BANK: .BLKW 1
1686
1687
1688
1689 034346          ML.REG: .BLKW 26
1690 034422          PTABLE.ADDR:
1691 034422          .BLKW 10
1692 034442          DRIVE.STATUS:
1693 034442          .BLKB 1
1694          .EVEN
1695 034444          DROPT.DRIVES:
1696 034444          .BLKB 1
1697          .EVEN
1698 034446          WHY.DROPT:
1699 034446          .BLKW 4
1700 034456          NUM.DRIVES:
1701 034456          .BLKW 1
1702 034460          LOW.SECT:
1703 034460          .BLKW 10
1704 034500          TOP.SECT:
1705 034500          .BLKW 10
1706 034520          PATTBL: .WORD 0
1707 034522          .WORD 2000
1708 034524          .WORD 1
  
```

000000
 002000
 000001

27-Mar-1982 19:24:42 TOPS
 27-Mar-1982 19:23:44 PA:<

```

1710          :MLX4
1711          :
1712          :
1713 034526 000001      .WORD 1
1714 034530 000004      .WORD 4
1715 034532 000004      .WORD 4
1716 034534 000011      .WORD 11
1717 034536 000011      .WORD 11
1718 034540 002000      .WORD 2000
1719 034542 002000      .WORD 2000
1720
1721
1722          .GLOBL EFNS21, L$UNIT, L$LUN, LIMIT, RANGE
1723          .GLOBL LSECT, TSECT, ONLY, DROPNE, DROP1
1724          .GLOBL DRCP2, DROP3, DROP4, DROP5, MARPAT
1725          .GLOBL REFRESH, ECCDIS, EOPSUM, ERROUT
1726          .GLOBL SEED1, SEED2, SEED3, RANDOM, RN
1727
1728
1729          100000      BIT15== -100000
1730          040000      BIT14== 40000
1731          020000      BIT13== 20000
1732          010000      BIT12== 10000
1733          004000      BIT11== 4000
1734          002000      BIT10== 2000
1735          001000      BIT09== 1000
1736          000400      BIT08== 400
1737          000200      BIT07== 200
1738          000100      BIT06== 100
1739          000040      BIT05== 40
1740          000020      BIT04== 20
1741          000010      BIT03== 10
1742          000004      BIT02== 4
1743          000002      BIT01== 2
1744          000001      BIT00== 1
1745          001000      BIT9== 1000
1746          000400      BIT8== 400
1747          000200      BIT7== 200
1748          000100      BIT6== 100
1749          000040      BIT5== 40
1750          000020      BIT4== 20
1751          000010      BIT3== 10
1752          000004      BIT2== 4
1753          000002      BIT1== 2
1754          000001      BIT0== 1
1755          000040      EF.START== 40
1756          000037      EF.RESTART== 37
1757          000036      EF.CONTINUE== 36
1758          000035      EF.NEW== 35
1759          000034      EF.PWR== 34
1760          000340      PRI07== 340
1761          000300      PRI06== 300
1762          000240      PRI05== 240
1763          000200      PRI04== 200
1764          000140      PRI03== 140
1765          :MLX4
1766          :
    
```

ML11 INTERRUPT SERVICE ROUTINE

27-Mar-1982 19:24:42 TOPS
 27-Mar-1982 19:23:44 PA:<


```

1767
1768      000100      PRI02==      100
1769      000040      PRI01==      40
1770      000000      PRI00==      0
1771      000004      EVL==        4
1772      000010      LOT==        10
1773      000020      ADR==        20
1774      000040      IOU==        40
1775      000100      ISR==        100
1776      000200      UAM==        200
1777      000400      BOE==        400
1778      001000      PNT==        1000
1779      002000      PRI==        2000
1780      004000      IXE==        4000
1781      010000      IBE==       10000
1782      020000      IER==       20000
1783      040000      LOE==       40000
1784      100000      HOE==      -100000
1785      012670      WDBUFF=      WBUFF
1786      013670      WCBUFF=     WCBUFF+1000
1787      022670      RDBUFF=      RBUFF
1788      023670      RCBUFF=     RCBUFF+1000
1789      022670      END.WBUFF=  WBUFF+10000
1790      032670      END.RBUFF=  RBUFF+10000
1791      005662      PHR21=      P.AAA
1792      005736      FMT16=      P.AAB
1793      006042      SBES$HEADER= P.AAC
1794      006126      FMT1A=      P.AAD
1795      006142      FMT1B=      P.AAE
1796      006160      FMT2=       P.AAF
1797      006166      FMT3=       P.AAG
1798      006216      FMT4A=      P.AAH
1799      006252      FMT4B=      P.AAI
1800      006270      FMT4C=      P.AAJ
1801      006320      FMT5=       P.AAK
1802      006376      FMT6=       P.AAL
1803      006446      FMT7=       P.AAM
1804      006466      FMT8=       P.AAN
1805      006534      FMT9=       P.AAO
1806      006564      FMT10A=     P.AAP
1807      006640      FMT10B=     P.AAQ
1808      006654      FMT11=      P.AAR
1809      006710      FMT12A=     P.AAS
1810      006756      FMT12B=     P.AAT
1811      007024      FMT13=      P.AAU
1812      007046      FMT14=      P.AAV
1813      007060      FMT15=      P.AAW
1814      007066      CRLF=       P.AAX
1815      007072      SAY1=       P.AAY
1816      007100      SAY2=       P.AAZ
1817      007112      SAY3=       P.ABA
1818      007130      SAY4=       P.ABB
1819      007152      SAY5=       P.ABC
1820      ;MLX4
1821      ;
1822
1823      007200      WRD2=       P.ABD

```

ML11 INTERRUPT SERVICE ROUTINE

27-Mar-1982 19:24:42
27-Mar-1982 19:23:44

TOPS
PA:<

1824	007206	WRD3=	P.ABE
1825	007212	WRD4=	P.ABF
1826	007220	WRD6=	P.ABG
1827	007230	WRD7=	P.ABH
1828	007240	WRD11=	P.ABI
1829	007246	WRD15=	P.ABJ
1830	007256	WRD16=	P.ABK
1831	007264	WRD17=	P.ABL
1832	007272	WRD18=	P.ABM
1833	007300	WRD19=	P.ABN
1834	007312	WRD20=	P.ABO
1835	007322	WRD21=	P.ABP
1836	007332	WRD24=	P.ABQ
1837	007336	WRD25=	P.ABR
1838	007344	WRD34=	P.ABS
1839	007354	WRD35=	P.ABT
1840	007370	WRD36=	P.ABU
1841	007400	WRD37=	P.ABV
1842	007412	WRD38=	P.ABW
1843	007416	WRD40=	P.ABX
1844	007424	WRD41=	P.ABY
1845	007430	MLB2=	P.ABZ
1846	007434	MLB3=	P.ACA
1847	007440	MLB4=	P.ACB
1848	007444	MLB5=	P.ACC
1849	007450	MLB6=	P.ACD
1850	007454	MLB7=	P.ACE
1851	007460	MLB8=	P.ACF
1852	007464	MLB9=	P.ACG
1853	007472	MLB10=	P.ACH
1854	007500	MLB11=	P.ACI
1855	007504	MLB12=	P.ACJ
1856	007510	MLB13=	P.ACK
1857	007514	MLB14=	P.ACL
1858	007520	MLB15=	P.ACM
1859	007524	MLB16=	P.ACN
1860	007530	MLB17=	P.ACO
1861	007534	MLB18=	P.ACP
1862	007542	MLB19=	P.ACQ
1863	007550	MLB20=	P.ACR
1864	007554	MLB21=	P.ACS
1865	007560	MLB22=	P.ACT
1866	007564	MLB23=	P.ACU
1867	007570	MLB24=	P.ACV
1868	007574	RTN0=	P.ACW
1869	007626	RTN1=	P.ACX
1870	007634	RTN2=	P.ACY
1871	007642	RTN3=	P.ACZ
1872	007650	RTN4=	P.ADA
1873	007656	RTN5=	P.ADB
1874	007664	RTN5A=	P.ADC
1875		;MLX4	
1876		;	
1877			
1878	007672	RTN5B=	P.ADD
1879	007700	RTN5C=	P.ADE
1880	007706	RTN5D=	P.ADF

ML11 INTERRUPT SERVICE ROUTINE

27-Mar-1982 19:24:42 TOPS
27-Mar-1982 19:23:44 PA:<

1881	007714	PHR1=	P.ADG
1882	007730	PHR2=	P.ADH
1883	007746	PHR3=	P.ADI
1884	007770	PHR4=	P.ADJ
1885	010004	PHR5=	P.ADK
1886	010026	PHR6=	P.ADL
1887	010050	PHR7=	P.ADM
1888	010066	PHR8=	P.ADN
1889	010100	PHR9=	P.ADO
1890	010120	PHR10=	P.ADP
1891	010140	PHR11=	P.ADQ
1892	010202	PHR12=	P.ADR
1893	010240	PHR13=	P.ADS
1894	010266	PHR14=	P.ADT
1895	010304	PHR15=	P.ADU
1896	010322	PHR16=	P.ADV
1897	010342	PHR17=	P.ADW
1898	010366	PHR18=	P.ADX
1899	010410	PHR19=	P.ADY
1900	010430	PHR20=	P.ADZ
1901	010444	TRT00=	P.AEA
1902	010446	TRT01=	P.AEB
1903	010450	TRT10=	P.AEC
1904	010454	TRT11=	P.AED
1905	010460	CAUSE1=	P.AEE
1906	010502	CAUSE2=	P.AEF
1907	010526	CAUSE3=	P.AEG
1908	010602	CAUSE4=	P.AEH
1909	010656	CAUSE5=	P.AEI
1910	010710	CAUSE6=	P.AEJ
1911	010734	CAUSE7=	P.AEK
1912	010756	CAUSE8=	P.AEL
1913	011002	MSG0=	P.AEM
1914	011070	MSG1=	P.AEN
1915	011120	MSG2=	P.AEO
1916	011166	MSG3=	P.AEP
1917	011202	MSG4=	P.AEQ
1918	011216	MSG5=	P.AER

.SBTTL SERVICE ML11 INTERRUPT SERVICE ROUTINE

1922
 1926 034544
 1927 034544 012767 000001 177564
 1928 034552 000002
 1929
 1930
 1931
 1932
 1933
 1934
 1939
 1940
 1941 :MLX4
 1942 :
 1943 :
 1944 : 2348 %sbttl 'ONCE-ONLY CODE'

SERVICE::
 MOV #1,I.AM.DONE
 RTI
 :
 :MLX4
 :
 :
 : Routine Size: 4 words
 : Maximum stack depth per invocation: 0 words

ML11 INTERRUPT SERVICE ROUTINE

2346
 2345
 TOPS
 PA:<
 27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (5)

```

1945 :      2349 routine CLRTBLS : novalue =
1946 :      2350 begin                               !* 1 *
1947 :      2351
1948 :      2352 !++
1949 :      2353 ROUTINE:      CLRTBLS
1950 :      2354
1951 :      2355 PURPOSE:      THIS ROUTINE IS CALLED BY THE INITIALIZATION CODE WHEN
1952 :      2356 A 'START' OR 'RESTART' COMMAND HAS BEEN USED TO BEGIN
1953 :      2357 THE PROGRAM.  THE PURPOSES OF THE ROUTINE ARE:
1954 :      2358
1955 :      2359 (1) TO INITIALIZE STATISTICS TABLES
1956 :      2360
1957 :      2361 (2) TO SET ALL DRIVE AND ARRAY STATUS LOCATIONS TO ACTIVE
1958 :      2362 NOTE:  EVEN IF A UNIT WAS DROPPED DURING A PREVIOUS
1959 :      2363 RUN, THE START OR RESTART COMMAND GUARANTEES
1960 :      2364 THAT THE DRIVE WILL ONCE AGAIN BE TESTABLE.
1961 :      2365
1962 :      2366 (3) TO ENABLE THE RUNNING OF ALL TEST OPTIONS IF THE
1963 :      2367 OPERATOR WANTED THEM ALL TO RUN.
1964 :      2368 --
1965 :      2369
1966 :      2370 !+
1967 :      2371 THIS IS THE CODE FOR PURPOSE 1:
1968 :      2372 !-
1969 :      2373
1970 :      2374 incr LUN from 0 to (.LSUNIT - 1) do
1971 :      2375 begin
1972 :      2376
1973 :      2377 incr ARRAY from 0 to 15 do
1974 :      2378 begin
1975 :      2379 SOFTS [.LUN, .ARRAY, 0, 16, 0] = 0;
1976 :      2380 HARDS [.LUN, .ARRAY, 0, 16, 0] = 0;
1977 :      2381 TRIES [.LUN, .ARRAY, 0, 16, 0] = 0;
1978 :      2382 end;
1979 :      2383
1980 :      2384 end;
1981 :      2385
1982 :      2386 SBES COUNT = 0;                               ! VER CZMLBB ADDED CLEARING THIS VARIABLE
1983 :      2387 EOP COUNT = 0;
1984 :      2388 NUM_DRIVES = 0;
1985 :      2389 RETRYING = INACTIVE;
1986 :      2390 WR_COUNT = 0;
1987 :      2391 WR_THOUSANDS = 0;
1988 :      2392 WR_MILLIONS = 0;
1989 :      2393 RD_COUNT = 0;
1990 :      2394 RD_THOUSANDS = 0;
1991 :      2395 RD_MILLIONS = 0;
1992 :      2396 WC_COUNT = 0;
1993 :      2397 WC_THOUSANDS = 0;
1994 :      2398 WC_MILLIONS = 0;
1995 :      2399 !+
1996 : MLX4
1997 :      ONCE-ONLY CODE
1998 :
1999 :      2400 !THIS IS THE CODE FOR PURPOSE 2:
2000 :      2401 !-
2001 :      2402
    
```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (5)

```

2002 :      2403      incr LUN from 0 to (.LSUNIT - 1) do
2003 :      2404          begin                               !* 2 *
2004 :      2405              L$[LUN] = .LUN;
2005 :      2406              LOW SECT [.LUN] = 0;
2006 :      2407              DRIVE_STATUS [.LUN] = ACTIVE;
2007 :      2408              DROPT_DRIVES [.LUN] = INACTIVE;
2008 :      2409              WHY DROPT [.LUN] = 0;
2009 :      2410          end;                               !* 2 *
2010 :      2411
2011 :      2412      !+
2012 :      2413      ! THIS IS THE CODE FOR PURPOSE 3:
2013 :      2414      !-
2014 :      2415
2015 :      2416      if .DROPNE eql 0
2016 :      2417      then                               !* 3 *
2017 :      2418          begin
2018 :      2419              DROP1 = 0;
2019 :      2420              DROP2 = 0;
2020 :      2421              DROP3 = 0;
2021 :      2422              DROP4 = 0;
2022 :      2423              DROP5 = 0;
2023 :      2424          end;                               !* 3 *
2024 :      2425
2025 :      2426      !+
2026 :      2427      ! VER CZMLBB ADDED CLEARING OF THIS STRUCTURE
2027 :      2428      !
2028 :      2429      ! The single bit error log table stores all
2029 :      2430      ! detected sbe's detected during the run time
2030 :      2431      ! of the exerciser. In this table is stored
2031 :      2432      ! the sbe's: unit #, board #, bank #, bit #
2032 :      2433      ! and a count of how many times this sbe has
2033 :      2434      ! reoccured.
2034 :      2435      !
2035 :      2436      ! This code initializes this structure to zeroes
2036 :      2437      ! before starting execution of the exerciser.
2037 :      2438      !-
2038 :      2439
2039 :      2440      incr index from 0 to 127 do           !Clear the single bit error log table
2040 :      2441          begin
2041 :      2442              SBE_LOG [.index, WRDS_0] = ZERO;       !Clear word zero
2042 :      2443              SBE_LOG [.index, WRDS_1] = ZERO;       !Clear word one
2043 :      2444          end;
2044 :      2445
2045 :      2446      !+
2046 :      2447      ! VER CZMLBB ADDED CLEARING OF THIS STRUCTURE
2047 :      2448      !
2048 :      2449      ! The prom maintenance program is called out
2049 :      2450      ! to be run on a particular array module when
2050 :      2451      ! that array module has collected 10 unique
    
```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (5)

```

2052 :MLX4
2053 :
2054 :
2055 : 2452 ! failing chips. This structure stores for
2056 : 2453 ! each array within each unit its total number
2057 : 2454 ! of unique failing chips.
2058 : 2455
2059 : 2456 ! This code initializes this structure to zeroes
2060 : 2457 ! before the exerciser is executed.
2061 : 2458 !-
2062 : 2459
2063 : 2460 incr UNIT_SEL from 0 to 7 do !Clear each units table
2064 : 2461
2065 : 2462 incr ARRAY_SEL from 0 to 15 do !Clear each array within each unit
2066 : 2463 PM_SBE_CNT [.UNIT_SEL, .ARRAY_SEL, PM_SBE$SUM] = ZERO; !Clear this array sum
2067 : 2464
2068 : 2465 return;
2069 : 2466 end;
    
```

```

2074 : .SBTTL CLRTBLS ONCE-ONLY CODE
2078 034554 004167 150534 CLRTBLS:JSR R1,$SAVE4 : 2349
2079 034560 016704 145226 MOV L$UNIT,R4 : 2374
2080 034564 005001 CLR R1 : LUN
2081 034566 000424 BR 3$ :
2082 034570 010100 1$: MOV R1,R0 : LUN,* 2379
2083 034572 006300 ASL R0
2084 034574 006300 ASL R0
2085 034576 006300 ASL R0
2086 034600 006300 ASL R0
2087 034602 005003 CLR R3 : ARRAY 2377
2088 034604 010002 2$: MOV R0,R2 : 2379
2089 034606 060302 ADD R3,R2 : ARRAY,*
2090 034610 006302 ASL R2
2091 034612 005062 032714 CLR SOFTS(R2)
2092 034616 005062 033314 CLR HARDS(R2) : 2380
2093 034620 005062 033714 CLR TRIES(R2) : 2381
2094 034626 005203 INC R3 : ARRAY 2377
2095 034630 020327 000017 CMP R3,#17 : ARRAY,*
2096 034634 003763 BLE 2$
2097 034636 005201 INC R1 : LUN 2374
2098 034640 020104 3$: CMP R1,R4 : LUN,*
2099 034642 002752 BLT 1$
2100 034644 005067 156014 CLR SBE$.COUNT : 2386
2101 034650 005067 176030 CLR EOP.COUNT : 2387
2102 034654 005067 177576 CLR NUM.DRIVES : 2388
2103 034660 005067 177454 CLR RETRYING : 2389
2104 034664 005067 177424 CLR WR.COUNT : 2390
2105 034670 005067 177422 CLR WR.THOUSANDS : 2391
2106 034674 005067 177420 CLR WR.MILLIONS : 2392
    
```

27-Mar-1982 19:24:42 TOPS
 27-Mar-1982 19:23:44 PA:<

Address	OpCode	Operand 1	Operand 2	Comment	Address	OpCode	Operand 1	Operand 2	Comment
2108				:MLX4					
2109				:					
2110									
2111	034700	005067	177416	CLR		RD.COUNT			2393
2112	034704	005067	177414	CLR		RD.THOUSANDS			2394
2113	034710	005067	177412	CLR		RD.MILLIONS			2395
2114	034714	005067	177410	CLR		WC.COUNT			2396
2115	034720	005067	177406	CLR		WC.THOUSANDS			2397
2116	034724	005067	177404	CLR		WC.MILLIONS			2398
2117	034730	005002		CLR		R2		LUN	2403
2118	034732	000445		BR		5\$			
2119	034734	010267	145134	4\$: MOV		R2,LSLUN		LUN,*	2405
2120	034740	010201		MOV		R2,R1		LUN,*	2406
2121	034742	006301		ASL		R1			
2122	034744	005061	034460	CLR		LOW.SECT(R1)			
2123	034750	010201		MOV		R2,R1		LUN,*	2407
2124	034752	006201		ASR		R1			
2125	034754	006201		ASR		R1			
2126	034756	006201		ASR		R1			
2127	034760	010146		MOV		R1,-(SP)			
2128	034762	062716	034442	ADD		#DRIVE.STATUS,(SP)			
2129	034766	010246		MOV		R2,-(SP)		LUN,*	
2130	034770	042716	177770	BIC		#177770,(SP)			
2131	034774	012746	000001	MOV		#1,-(SP)			
2132	035000	011646		MOV		(SP),-(SP)			
2133	035002	004767	147606	JSR		PC,BL\$PU2			
2134	035006	010116		MOV		R1,(SP)			2408
2135	035010	062716	034444	ADD		#DROPT.DRIVES,(SP)			
2136	035014	010246		MOV		R2,-(SP)		LUN,*	
2137	035016	042716	177770	BIC		#177770,(SP)			
2138	035022	012746	000001	MOV		#1,-(SP)			
2139	035026	005046		CLR		-(SP)			
2140	035030	004767	147560	JSR		PC,BL\$PU2			
2141	035034	105062	034446	CLRB		WHY.DROPT(R2)		*(LUN)	2409
2142	035040	062706	000016	ADD		#16,SP			2404
2143	035044	005202		INC		R2		LUN	2403
2144	035046	020204		5\$: CMP		R2,R4		LUN,*	
2145	035050	002731		BLT		4\$			
2146	035052	005767	145156	TST		DROPNE			2416
2147	035056	001012		BNE		6\$			
2148	035060	005067	145152	CLR		DROP1			2419
2149	035064	005067	145150	CLR		DROP2			2420
2150	035070	005067	145146	CLR		DROP3			2421
2151	035074	005067	145144	CLR		DROP4			2422
2152	035100	005067	145144	CLR		DROP5			2423
2153	035104	005003		6\$: CLR		R3		INDEX	2440
2154	035106	010302		7\$: MOV		R3,R2		INDEX,*	2442
2155	035110	006302		ASL		R2			
2156	035112	006302		ASL		R2			
2157	035114	005062	011264	CLR		SBE.LOG(R2)			
2158	035120	005062	011266	CLR		SBE.LOG+2(R2)			2443
2159	035124	005203		INC		R3		INDEX	2440
2160	035126	020327	000177	CMP		R3,#177		INDEX,*	
2161	035132	003765		BLE		7\$			
2162	035134	005003		CLR		R3		UNIT.SEL	2460

27-Mar-1982 19:24:42 TOPS
27-Mar-1982 19:23:44 PA:<

```

2164      ;MLX4
2165      ;
2166      ;
2167 035136 010300      8$:  MOV  R3,R0      ; UNIT.SEL,*      2463
2168 035140 006300      ASL  R0
2169 035142 006300      ASL  R0
2170 035144 006300      ASL  R0
2171 035146 006300      ASL  R0
2172 035150 005002      CLR  R2      ; ARRAY.SEL      2462
2173 035152 010001      9$:  MOV  R0,R1      ;
2174 035154 060201      ADD  R2,R1      ; ARRAY.SEL,*      2463
2175 035156 005301      ASL  R1
2176 035160 005061 012264 CLR  PM.SBE.CNT(R1)
2177 035164 005202      INC  R2      ; ARRAY.SEL      2462
2178 035166 020227 000017 CMP  R2,#17      ; ARRAY.SEL,*
2179 035172 003767      BLE  9$
2180 035174 005203      INC  R3      ; UNIT.SEL      2460
2181 035176 020327 000007 CMP  R3,#7      ; UNIT.SEL,*
2182 035202 003755      BLE  8$
2183 035204 000207      RTS  PC      ;
2184
2185      ; Routine Size: 141 words
2186      ; Maximum stack depth per invocation: 12 words
2191
2192

```


27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (6)

```

2194 :MLX4
2195 : ONCE-ONLY CODE
2196 :
2197 : 2467 routine INIT_ADDRESSES (PLOC) : novalue =
2198 : 2468 begin !* 1 *
2199 : 2469
2200 : 2470 !++
2201 : 2471 ROUTINE: INIT_ADDRESSES(PLOC)
2202 : 2472
2203 : 2473 PURPOSE: THIS ROUTINE IS CALLED ONLY ONCE DURING THE INITIALIZATION
2204 : 2474 CODE, EVEN IF THERE IS MORE THAN ONE DRIVE PRESENT. THE
2205 : 2475 PURPOSES OF THE ROUTINE ARE:
2206 : 2476
2207 : 2477 (1) TO OBTAIN HARDWARE P-TABLE INFORMATION FROM
2208 : 2478 MAIN MEMORY WHICH PERTAINS TO ALL DRIVES.
2209 : 2479
2210 : 2480 (2) TO SET UP THE ADDRESSES FOR THE 22 ML-11 REGISTERS.
2211 : 2481
2212 : 2482 (3) TO SET UP THE INTERRUPT SERVICE ROUTINE AT THE
2213 : 2483 CORRECT PRIORITY, AND LOWER THE CPU PRIORITY TO
2214 : 2484 ALLOW INTERRUPTS TO OCCUR.
2215 : 2485
2216 : 2486 ARGUMENT: PLOC = THE POINTER TO THE LOCATION IN MAIN MEMORY WHERE
2217 : 2487 THE HARDWARE P-TABLE IS TO BE FOUND.
2218 : 2488 !--
2219 : 2489
2220 : 2490 local
2221 : 2491 TEMP,
2222 : 2492 PRIORITY,
2223 : 2493 OFFSET;
2224 : 2494
2225 : 2495 !+
2226 : 2496 !THIS IS THE CODE FOR PURPOSE 1:
2227 : 2497 !-
2228 : 2498
2229 : 2499 BASE_ADDR = .(.PLOC + 0);
2230 : 2500 VEC = .(.PLOC + 2);
2231 : 2501 BR_LEVEL = .(.PLOC + 4);
2232 : 2502 PRIORITY = .BR_LEVEL^5;
2233 : 2503
2234 : 2504 if .ECCDIS
2235 : 2505 then
2236 : 2506 TEMP = WRD37 !ECC IS DISABLED
2237 : 2507 else
2238 : 2508 TEMP = WRD36; !ECC IS ENABLED (NORMAL OPERATION)
2239 : 2509
2240 : 2510 PRINTB (CRLF);
2241 : 2511 PRINTB (SAY5, WRD34, WRD40, WRD38, .TEMP, WRD41);
2242 : 2512 !'RUNNING WITH ECC ENABLED/DISABLED AND'
2243 : 2513
2244 : 2514 if .REFRESH
2245 : 2515 then
2246 : 2516 TEMP = WRD36 !REFRESH MARGINING IS ENABLED
2247 : 2517 else
2248 : 2518 TEMP = WRD37; !REFRESH MARGINING IS DISABLED (NORMAL OPERATION)
  
```

```

2250 ;MLX4
2251 : ONCE-ONLY CODE
2252 :
2253 : 2519
2254 : 2520 PRINTB (SAY3, WRD40, PHR3, .TEMP);
2255 : 2521 !'WITH REFRESH MARGINING ENABLED/DISABLED'
2256 : 2522 PRINTB (CRLF);
2257 : 2523
2258 : 2524 !+
2259 : 2525 !THIS IS THE CODE FOR PURPOSE 2:
2260 : 2526 !-
2261 : 2527
2262 : 2528 OFFSET = 0;
2263 : 2529
2264 : 2530 incr COUNT from 0 to (NUM_REGS - 1) do
2265 : 2531 begin !* 2 *
2266 : 2532 ML_REG [.COUNT] = .BASE_ADDR + .OFFSET;
2267 : 2533 OFFSET = .OFFSET + 2;
2268 : 2534 end; !* 2 *
2269 : 2535
2270 : 2536 !+
2271 : 2537 !THIS IS THE CODE FOR PURPOSE 3:
2272 : 2538 !-
2273 : 2539
2274 : 2540 SETPRI (PRI00);
2275 : 2541 SETVEC (.VEC, SERVICE, .PRIORITY);
2276 : 2542 return;
2277 : 2543 end; !* 1 *
2281 :
2282 :

```

Address	SBTTL	INIT.ADDRESSES	ONCE-ONLY CODE	Address
2286	035206			
2287	035206	004167 150102	JSR R1,\$SAVE4	2467
2288	035212	016602 000014	MOV 14(SP),R2	2499
2289	035216	011267 175464	MOV (R2),BASE_ADDR	
2290	035222	016267 000002 175460	MOV 2(R2),VEC	2500
2291	035230	016267 000004 175454	MOV 4(R2),BR.LEVEL	2501
2292	035236	016746 175450	MOV BR.LEVEL,-(SP)	2502
2293	035242	012746 000005	MOV #5,-(SP)	
2294	035246	004767 147742	JSR PC,BL\$SHF	
2295	035252	010003	MOV R0,R3	*.PRIORITY
2296	035254	032767 000001 144772	BIT #1,ECCDIS	2504
2297	035262	001403	BEQ 1\$	
2298	035264	012701 007400	MOV #WRD37,R1	*.TEMP 2506
2299	035270	000402	BR 2\$	2504
2300	035272	012701 007370	1\$: MOV #WRD36,R1	*.TEMP 2508
2301	035276	012716 007066	2\$: MOV #CRLF,(SP)	2510
2302	035302	012746 000001	MOV #1,-(SP)	
2303	035306	010600	MOV SP,R0	: SP,*
2304	035310	104414	TRAP 14	

Address	OpCode	Operand1	Operand2	Operand3	Comment	Line No.
2306						
2307						
2308						
2309	035312	012716	007424		MOV #WRD41,(SP)	
2310	035316	010146			MOV R1,-(SP)	
2311	035320	012746	007412		MOV #WRD38,-(SP)	
2312	035324	012746	007416		MOV #WRD40,-(SP)	
2313	035330	012746	007344		MOV #WRD34,-(SP)	
2314	035334	012746	007152		MOV #SAY5,-(SP)	
2315	035340	012746	000006		MOV #6,-(SP)	
2316	035344	010600			MOV SP,R0	
2317	035346	104414			TRAP 14	
2318	035350	032767	000001	144674	BIT #1,REFRESH	
2319	035356	001403			BEQ 3\$	
2320	035360	012701	007370		MOV #WRD36,R1	
2321	035364	000402			BR 4\$	
2322	035366	012701	007400		MOV #WRD37,R1	
2323	035372	010116			MOV R1,(SP)	
2324	035374	012746	007746		MOV #PHR3,-(SP)	
2325	035400	012746	007416		MOV #WRD40,-(SP)	
2326	035404	012746	007112		MOV #SAY3,-(SP)	
2327	035410	012746	000004		MOV #4,-(SP)	
2328	035414	010600			MOV SP,R0	
2329	035416	104414			TRAP 14	
2330	035420	012716	007066		MOV #CRLF,(SP)	
2331	035424	012746	000001		MOV #1,-(SP)	
2332	035430	010600			MOV SP,R0	
2333	035432	104414			TRAP 14	
2334	035434	005002			CLR R2	
2335	035436	005000			CLR R0	
2336	035440	010001			MOV R0,R1	
2337	035442	006301			ASL R1	
2338	035444	016704	175236		MOV BASE.ADDR,R4	
2339	035450	060204			ADD R2,R4	
2340	035452	010461	034346		MOV R4,ML.REG(R1)	
2341	035456	062702	000002		ADD #2,R2	
2342	035462	005200			INC R0	
2343	035464	020027	000025		CMP R0,#25	
2344	035470	003763			BLE 5\$	
2345	035472	005000			CLR R0	
2346	035474	104441			TRAP 41	
2347	035476	010316			MOV R3,(SP)	
2348	035500	012746	034544		MOV #SERVICE,-(SP)	
2349	035504	016746	175200		MOV VEC,-(SP)	
2350	035510	012746	000003		MOV #3,-(SP)	
2351	035514	104437			TRAP 37	
2352	035516	062706	000042		ADD #42,SP	
2353	035522	000207			RTS PC	
2354						
2355						
2356						

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

```

2362 :MLX4
2363 :
2364 :
2365 : 2544 Zsbttl 'DRIVE IDENTIFICATION ROUTINES'
2366 : 2545 routine SAYWHO (LUN) : novalue =
2367 : 2546 begin
2368 : 2547
2369 : 2548
2370 : 2549
2371 : 2550
2372 : 2551
2373 : 2552
2374 : 2553
2375 : 2554
2376 : 2555
2377 : 2556
2378 : 2557
2379 : 2558
2380 : 2559
2381 : 2560
2382 : 2561
2383 : 2562
2384 : 2563
2385 : 2564
2386 : 2565
2387 : 2566
2388 : 2567
2389 : 2568
2390 : 2569
2391 : 2570
2392 : 2571
2393 : 2572
2394 : 2573
2395 : 2574
2396 : 2575
2397 : 2576
2398 : 2577
2399 : 2578
2400 : 2579
2401 : 2580
2405 :
2406 :
2410 035524 004167 147604
2411 035530 017705 176642
2412 035534 006205
2413 035536 006205
2414 035540 006205
2415 035542 006205
2416 035544 000305
2417 :
2418 :
2419 :
2420 035546 042705 177760
2421 035552 017704 176620
2422 035556 000304
2423 035560 042704 177760
2424 035564 117701 176606
    
```

```

DRIVE IDENTIFICATION ROUTINES
    Zsbttl 'DRIVE IDENTIFICATION ROUTINES'
    routine SAYWHO (LUN) : novalue =
    begin
    ++
    ROUTINE: SAYWHO(LUN)
    PURPOSE: TO PRINT OUT AN IDENTIFICATION LINE WHICH INCLUDES:
              LOGICAL UNIT: X DRIVE: Y SERIAL #: ZZZZ
              THE UNIT'S SERIAL NUMBER IN EITHER BCD OR OCTAL FORMAT.
              BCD WILL BE PRINTED AS LONG AS THE DIGITS ARE ALL VALID.
    --
    local
    D3,
    D2,
    D1,
    D0;
    D3 = .SN3;
    D2 = .SN2;
    D1 = .SN1;
    D0 = .SN0;
    PRINTB (FMT4A, PHR7, .LUN, WRD11, .DRIVE);
    !'LOGICAL UNIT: X DRIVE: Y'
    if ((.D3 gtr 9) or (.D2 gtr 9) or (.D1 gtr 9) or (.D0 gtr 9))
    then
    PRINTB (FMT4B, PHR8, .MLSN)
    !' SERIAL #: ZZZZZZ'
    else
    PRINTB (FMT4C, PHR8, .D3, .D2, .D1, .D0);
    !' SERIAL #: DDDD'
    return;
    end;
    
```

```

.SBttl SAYWHO DRIVE IDENTIFICATION ROUTINES
SAYWHO: JSR R1,$SAVE5
MOV @ML.REG+30,R5
ASR R5
ASR R5
ASR R5
ASR R5
SWAB R5
;MLX4
:
DRIVE IDENTIFICATION ROUTINES
BIC #177760,R5
MOV @ML.REG+30,R4
SWAB R4
BIC #177760,R4
MOVB @ML.REG+30,R1
    
```

```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (7)
2545
2564
2565
2566
    
```

2425	035570	006201		ASR	R1	:	D1	
2426	035572	006201		ASR	R1	:	D1	
2427	035574	006201		ASR	R1	:	D1	
2428	035576	006201		ASR	R1	:	D1	
2429	035600	042701	177760	BIC	#177760,R1	:	*,D1	
2430	035604	117702	176566	MOVB	@ML,REG+30,R2	:	*,D0	2567
2431	035610	042702	177760	BIC	#177760,R2	:	*,D0	
2432	035614	016603	000016	MOV	16(SP),R3	:	LUN,*	2568
2433	035620	006303		ASL	R3			
2434	035622	016303	034422	MOV	PTABLE,ADDR(R3),R3			
2435	035626	016346	000006	MOV	6(R3),-(SP)			
2436	035632	012746	007240	MOV	#WRD11,-(SP)			
2437	035636	016646	000022	MOV	22(SP),-(SP)	:	LUN,*	
2438	035642	012746	010050	MOV	#PHR7,-(SP)			
2439	035646	012746	006216	MOV	#FMT4A,-(SP)			
2440	035652	012746	000005	MOV	#5,-(SP)			
2441	035656	010600		MOV	SP,R0	:	SP,*	
2442	035660	104414		TRAP	14			
2443	035662	020527	000011	CMP	R5,#11	:	D3,*	2571
2444	035666	003011		BGT	1\$			
2445	035670	020427	000011	CMP	R4,#11	:	D2,*	
2446	035674	003006		BGT	1\$			
2447	035676	020127	000011	CMP	R1,#11	:	D1,*	
2448	035702	003003		BGT	1\$			
2449	035704	020227	000011	CMP	R2,#11	:	D0,*	
2450	035710	003413		BLE	2\$			
2451	035712	017746	176460	1\$: MOV	@ML,REG+30,-(SP)	:		2573
2452	035716	012746	010066	MOV	#PHR8,-(SP)			
2453	035722	012746	006252	MOV	#FMT4B,-(SP)			
2454	035726	012745	000003	MOV	#3,-(SP)			
2455	035732	010600		MOV	SP,R0	:	SP,*	
2456	035734	104414		TRAP	14			
2457	035736	000416		BR	3\$			
2458	035740	010246		2\$: MOV	R2,-(SP)	:	D0,*	2571
2459	035742	010146		MOV	R1,-(SP)	:	D1,*	2576
2460	035744	010446		MOV	R4,-(SP)	:	D2,*	
2461	035746	010546		MOV	R5,-(SP)	:	D3,*	
2462	035750	012746	010066	MOV	#PHR8,-(SP)			
2463	035754	012746	006270	MOV	#FMT4C,-(SP)			
2464	035760	012746	000006	MOV	#6,-(SP)			
2465	035764	010600		MOV	SP,R0	:	SP,*	
2466	035766	104414		TRAP	14			
2467	035770	062706	000006	ADD	#6,SP			
2468	035774	062706	000024	3\$: ADD	#24,SP	:		2545
2469	036000	000207		RTS	PC			

: Routine Size: 87 words
 :MLX4
 : DRIVE IDENTIFICATION ROUTINES
 : Maximum stack depth per invocation: 19 words

27-Mar-1982 19:24:42 TOPS
 27-Mar-1982 19:23:44 PA:<

2482 :MLX4
 2483 :
 2484 :
 2485 : 2581 routine CONFIG (LUN) : novalue =
 27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (8)

```

2486 :      2582      begin                               !* 1 *
2487 :      2583
2488 :      2584      +-+
2489 :      2585      ROUTINE:      CONFIG(LUN)
2490 :      2586
2491 :      2587      PURPOSE:      CONFIG IS CALLED BY THE INITIALIZATION CODE FOR EACH DRIVE
2492 :      2588      WHICH SUCCESSFULLY RESPONDS TO A REQUEST FOR ITS HARDWARE
2493 :      2589      P-TABLE.  THE PURPOSES OF THIS ROUTINE ARE:
2494 :      2590
2495 :      2591      (1)  TO CHECK THAT THE DRIVE IS POWERED UP.
2496 :      2592
2497 :      2593      (2)  TO VERIFY THAT THE DRIVE IS AN ML11 UNIT.
2498 :      2594
2499 :      2595      (3)  TO VERIFY THAT THE OPERATOR DEFINED TEST RANGES ARE
2500 :      2596      WITHIN THE CALCULATED SYSTEM SIZE.
2501 :      2597
2502 :      2598      (4)  TO PRINT DRIVE IDENTIFICATION INFORMATION.
2503 :      2599
2504 :      2600      ARGUMENT:      LUN = THE LOGICAL UNIT NUMBER, COUNTING FROM 0 TO
2505 :      2601      NUMBER OF DRIVES MINUS 1.
2506 :      2602      --
2507 :      2603
2508 :      2604      local
2509 :      2605      TYPE,
2510 :      2606      TEMP,
2511 :      2607      TOP;
2512 :      2608
2513 :      2609      !+
2514 :      2610      !THIS IS THE CODE FOR PURPOSE 1:
2515 :      2611      !-
2516 :      2612
2517 :      2613      UNIT = .DRIVE;
2518 :      2614
2519 :      2615      if .DPR neq 1
2520 :      2616      then
2521 :      2617          begin
2522 :      2618              PRINTB (FMT4A, PHR7, .LUN, WRD11, .DRIVE);
2523 :      2619              !'LOGICAL UNIT: X  DRIVE: Y'
2524 :      2620              PRINTB (SAY2, WRD11, WRD21);
2525 :      2621              !'DRIVE DROPPED'
2526 :      2622              PRINTB (FMT2, CAUSE1);
2527 :      2623              !' (NOT POWERED UP)'
2528 :      2624              WHY DROPT [.LUN] = CODE_1;
2529 :      2625              DODD (.LUN);
2530 :      2626              return;
2531 :      2627              end;
2532 :      2628
2533 :      2629      !+
2534 :      2630      !THIS IS THE CODE FOR PURPOSE 2:
2535 :      2631      !-
2536 :      2632
    
```

```

2538 :MLX4
2539 :
2540 :
2541 : 2633 SAYWHO (.LUN);
2542 : 2634 !'LOGICAL UNIT: X DRIVE: Y SERIAL #: ZZZZ'
2543 : 2635 TYPE = .MLDT;
2544 : 2636
2545 : 2637 if ((.TYPE neq ML11A) and (.TYPE neq ML11B))
2546 : 2638 then
2547 : 2639 begin
2548 : 2640 PRINTB (SAY2, WRD11, WRD21);
2549 : 2641 !'DRIVE DROPPED'
2550 : 2642 PRINTB (FMT2, CAUSE2);
2551 : 2643 !' (NOT AN ML11 UNIT)';
2552 : 2644 WHY DROPT [.LUN] = CODE_2;
2553 : 2645 DODD (.LUN);
2554 : 2646 return;
2555 : 2647 end
2556 : 2648 else
2557 : 2649 begin !* 3 *
2558 : 2650
2559 : 2651 if .ARR_TYP eql 0
2560 : 2652 then
2561 : 2653 begin !* 4 *
2562 : 2654 TYPE = WRD6;
2563 : 2655 TOP = 1;
2564 : 2656 end !* 4 *
2565 : 2657 else
2566 : 2658 begin !* 5 *
2567 : 2659 TYPE = WRD7;
2568 : 2660 TOP = 4;
2569 : 2661 end; !* 5 *
2570 : 2662
2571 : 2663 !+
2572 : 2664 !THIS IS THE CODE FOR PURPOSE 3:
2573 : 2665 !-
2574 : 2666
2575 : 2667 TOP_SECT [.LUN] = (.TOP)*(512)*(SZ) - 1;
2576 : 2668
2577 : 2669 if .LIMIT
2578 : 2670 then
2579 : 2671
2580 : 2672 !+
2581 : 2673 ! THE OPERATOR HAS CHOSEN LIMITS:
2582 : 2674 !-
2583 : 2675
2584 : 2676 begin !* 6 *
2585 : 2677
2586 : 2678 if .RANGE eql 0
2587 : 2679 then
2588 : 2680
2589 : 2681 !+
2590 : 2682 !THE BOARD NUMBER (0-15) IS CONTAINED IN 'ONLY'
2591 : 2683 !-
2592 : 2684
    
```

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (8)

```

2594 :MLX4
2595 :
2596 :
2597 : 2685          begin
2598 : 2686          !* 10 *
2599 : 2687          !: VER CZMLBB CHANGED .TYPE TO .ARR_TYP
2600 : 2688          !:
2601 : 2689
2602 : 2690          if .ARR_TYP
2603 : 2691          then
2604 : 2692
2605 : 2693          !+
2606 : 2694          !: THE CHIPS ARE 64K
2607 : 2695          !-
2608 : 2696
2609 : 2697          begin
2610 : 2698          LSECT = 2048*.ONLY;
2611 : 2699          TSECT = .LSECT + 2047;
2612 : 2700          end
2613 : 2701          !* 7 *
2614 : 2702          else
2615 : 2703          !+
2616 : 2704          !: THE CHIPS ARE 16K
2617 : 2705          !-
2618 : 2706
2619 : 2707          begin
2620 : 2708          LSECT = 512*.ONLY;
2621 : 2709          TSECT = .LSECT + 511;
2622 : 2710          end;
2623 : 2711          !* 8 *
2624 : 2712          !* 10 *
2625 : 2713          end;
2626 : 2714          if .TSECT leqa .TOP_SECT [.LUN]
2627 : 2715          then
2628 : 2716          TOP_SECT [.LUN] = .TSECT
2629 : 2717          else
2630 : 2718          begin
2631 : 2719          PRINTB (SAY2, WRD11, WRD21);
2632 : 2720          !'DRIVE DROPPED'
2633 : 2721          PRINTB (FMT2, CAUSE3);
2634 : 2722          !' (OPERATOR SELECTED TEST LIMITS INCORRECTLY)'
2635 : 2723          PRINTB (FMT11, PHR14, .TSECT, PHR16, .TOP_SECT [.LUN]);
2636 : 2724          !'TOP SECTOR OF XXXXXX EXCEEDS SYSTEM LIMIT OF YYYYYY'
2637 : 2725          WHY DROPT [.LUN] = CODE_3;
2638 : 2726          DODD (.LUN);
2639 : 2727          return;
2640 : 2728          end;
2641 : 2729
2642 : 2730          if .LSECT leqa .TSECT
2643 : 2731          then
2644 : 2732          LOW_SECT [.LUN] = .LSECT
2645 : 2733          else
2646 : 2734          begin
2647 : 2735          PRINTB (SAY2, WRD11, WRD21);
2648 : 2736          !'DRIVE DROPPED'
    
```

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (8)


```

2650 :MLX4
2651 :
2652 :
2653 : 2737 PRINTB (FMT2, CAUSE3);
2654 : 2738 !' (OPERATOR SELECTED TEST LIMITS INCORRECTLY)'
2655 : 2739 PRINTB (FMT11, PHR15, .LSECT, PHR14, .TSECT);
2656 : 2740 !'LOW SECTOR OF XXXXXX EXCEEDS TOP SECTOR OF YYYYYY'
2657 : 2741 WHY DROPT [.LUN] = CODE_3;
2658 : 2742 DODD (.LUN);
2659 : 2743 return;
2660 : 2744 end;
2661 : 2745
2662 : 2746 end; !* 6 *
2663 : 2747
2664 : 2748 !+
2665 : 2749 !THIS IS THE CODE FOR PURPOSE 4:
2666 : 2750 !-
2667 : 2751
2668 : 2752 PRINTB (FMT5, .TYPE, LOWEST, HIGHEST);
2669 : 2753 !'ML11-X SECTORS UNDER TEST: XXXXXX TO YYYYYY'
2670 : 2754
2671 : 2755 selectone .TRT of
2672 : 2756 set
2673 : 2757
2674 : 2758 [0] :
2675 : 2759 TEMP = TRT00;
2676 : 2760
2677 : 2761 [1] :
2678 : 2762 TEMP = TRT01;
2679 : 2763
2680 : 2764 [2] :
2681 : 2765 TEMP = TRT10;
2682 : 2766
2683 : 2767 [3] :
2684 : 2768 TEMP = TRT11;
2685 : 2769 tes;
2686 : 2770
2687 : 2771 PRINTB (FMT8, .TEMP);
2688 : 2772 !'TRANSFER RATE: X MBYTES/SECOND'
2689 : 2773 PRINTB (FMT4B, PHR4, .BASE_ADDR);
2690 : 2774 !' CSR ADDRESS: XXXXXX'
2691 : 2775 end; !* 3 *
2692 : 2776
2693 : 2777 return;
2694 : 2778 end; !* 1 *
2698 :
2699 :

```

```

2703 036002 004167 147326 .SBTTL CONFIG DRIVE IDENTIFICATION ROUTINES
2704 036006 016603 000016 CONFIG: JSR R1,$SAVE5
MOV 16(SP),R3 ; LUN,*

```

2581
2613

```

2706          ;MLX4
2707          :
2708          :
2709 036012 010304      MOV      R3,R4
2710 036014 006304      ASL      R4
2711 036016 016401 034422  MOV      PTABLE.ADDR(R4),R1
2712 036022 016105 000006  MOV      6(R1),R5
2713 036026 042705 177770  BIC      #177770,R5
2714 036032 142777 000007 176316  BICB    #7,AML.REG+10
2715 036040 150577 176312  BISB    R5,AML.REG+10
2716 036044 032777 009400 176306  BIT     #400,AML.REG+12
2717 036052 001051      BNE
2718 036054 016401 034422  MOV      PTABLE.ADDR(R4),R1
2719 036060 016146 000006  MOV      6(R1),-(SP)
2720 036064 012746 007240  MOV      #WORD11, -(SP)
2721 036070 010346      MOV      R3, -(SP)
2722 036072 012746 010050  MOV      #PHR7, -(SP)
2723 036076 012746 006216  MOV      #FMT4A, -(SP)
2724 036102 012746 000005  MOV      #5, -(SP)
2725 036106 010600      MOV      SP,R0
2726 036110 104414      TRAP    14
2727 036112 012716 007322  MOV      #WORD21, (SP)
2728 036116 012746 007240  MOV      #WORD11, -(SP)
2729 036122 012746 007100  MOV      #SAY2, -(SP)
2730 036126 012746 000003  MOV      #3, -(SP)
2731 036132 010600      MOV      SP,R0
2732 036134 104414      TRAP    14
2733 036136 012716 010460  MOV      #CAUSE1, (SP)
2734 036142 012746 006160  MOV      #FMT2, -(SP)
2735 036146 012746 000002  MOV      #2, -(SP)
2736 036152 010600      MOV      SP,R0
2737 036154 104414      TRAP    14
2738 036156 112763 000001 034446  MOVB    #1,WHY.DROPT(R3)
2739 036164 010300      MOV      R3,R0
2740 036166 104451      TRAP    51
2741 036170 062706 000026  ADD     #26,SP
2742 036174 000207      RTS     PC
2743 036176 010346      MOV      R3, -(SP)
2744 036200 004767 177320  JSR     PC,SAYWHO
2745 036204 017702 176164  MOV     AML.REG+26,R2
2746 036210 020227 000110  CMP     R2,#110
2747 036214 001435      BEQ     2$
2748 036216 020227 000111  CMP     R2,#111
2749 036222 001432      BEQ     2$
2750 036224 012746 007322  MOV     #WORD21, -(SP)
2751 036230 012746 007240  MOV     #WORD11, -(SP)
2752 036234 012746 007100  MOV     #SAY2, -(SP)
2753 036240 012746 000003  MOV     #3, -(SP)
2754 036244 010600      MOV     SP,R0
2755 036246 104414      TRAP    14
2756 036250 012716 010502  MOV     #CAUSE2, (SP)
2757 036254 012746 006160  MOV     #FMT2, -(SP)
2758 036260 012746 000002  MOV     #2, -(SP)
2759 036264 010600      MOV     SP,R0
2760 036266 104414      TRAP    14
    
```

Address	OpCode	Operand1	Operand2	Operand3	Label	Instruction	Comments	Page
2762					:MLX4			
2763					:			
2764					:			
2765	036270	112763	000002	034446		MOVB #2,WHY.DROPT(R3)	:	2644
2766	036276	010300				MOV R3,R0	:	2645
2767	036300	104451				TRAP 51	:	
2768	036302	062706	000016			ADD #16,SP	:	2637
2769	036306	000207				RTS PC	:	2639
2770	036310	032777	002000	176054	2\$:	BIT #2000,@ML.REG+24	:	2651
2771	036316	001005				BNE 3\$:	
2772	036320	012702	007220			MOV #WRD6,R2	: *,TYPE	2634
2773	036324	012701	000001			MOV #1,R1	: *,TOP	2655
2774	036330	000404				BR 4\$:	2651
2775	036332	012702	007230		3\$:	MOV #WRD7,R2	: *,TYPE	2659
2776	036336	012701	000004			MOV #4,R1	: *,TOP	2660
2777	036342	012705	034500		4\$:	MOV #TOP,SECT,R5	:	2667
2778	036346	060405				ADD R4,R5	:	
2779	036350	010146				MOV R1,-(SP)	: TOP,*	2649
2780	036352	017701	176014			MOV @ML.REG+24,R1	:	
2781	036356	006201				ASR R1	:	
2782	036360	006201				ASR R1	:	
2783	036362	006201				ASR R1	:	
2784	036364	000301				SWAB R1	:	
2785	036366	042701	177740			BIC #177740,R1	:	
2786	036372	010146				MOV R1,-(SP)	:	
2787	036374	004767	146344			JSR PC,BLSMUL	:	
2788	036400	000300				SWAB R0	:	2667
2789	036402	105000				CLRB R0	:	
2790	036404	006300				ASL R0	:	
2791	036406	010001				MOV R0,R1	:	
2792	036410	005301				DEC R1	:	
2793	036412	010115				MOV R1,(R5)	:	
2794	036414	032767	000001	143600		BIT #1,LIMIT	:	2669
2795	036422	001575				BEQ 11\$:	
2796	036424	005767	143574			TST RANGE	:	2678
2797	036430	001037				BNE 6\$:	
2798	036432	032777	002000	175732		BIT #2000,@ML.REG+24	:	2690
2799	036440	001417				BEQ 5\$:	
2800	036442	016700	143564			MOV ONLY,R0	:	2698
2801	036446	000300				SWAB R0	:	
2802	036450	105000				CLRB R0	:	
2803	036452	006300				ASL R0	:	
2804	036454	006300				ASL R0	:	
2805	036456	006300				ASL R0	:	
2806	036460	010067	143542			MOV R0,LSECT	:	
2807	036464	010067	143540			MOV R0,TSECT	: LSECT,*	2699
2808	036470	062767	003777	143532		ADD #3777,TSECT	:	
2809	036476	000414				BR 6\$:	2690
2810	036500	016700	143526		5\$:	MOV ONLY,R0	:	2708
2811	036504	000300				SWAB R0	:	
2812	036506	105000				CLRB R0	:	
2813	036510	006300				ASL R0	:	
2814	036512	010067	143510			MOV R0,LSECT	:	
2815	036516	010067	143506			MOV R0,TSECT	: LSECT,*	2709
2816	036522	062767	000777	143500		ADD #777,TSECT	:	

Address	OpCode	Operand1	Operand2	Operand3	Label	Instruction	Comments	Address	Time	Page
2818									27-Mar-1982 19:24:42	TOPS
2819									27-Mar-1982 19:23:44	PA:<
2820										
2821	036530	026715	143474		6\$:	CMP TSECT,(R5)	:			2714
2822	036534	101003				BHI 7\$:			
2823	036536	016715	143466			MOV TSECT,(R5)	:			2716
2824	036542	000445				BR 8\$:			2714
2825	036544	012746	007322		7\$:	MOV #WRD21,-(SP)	:			2719
2826	036550	012745	007240			MOV #WRD11,-(SP)	:			
2827	036554	012746	007100			MOV #SAY2,-(SP)	:			
2828	036560	012746	000003			MOV #3,-(SP)	:			
2829	036564	010600				MOV SP,R0	: SP,*			
2830	036566	104414				TRAP 14	:			
2831	036570	012716	010526			MOV #CAUSE3,(SP)	:			2721
2832	036574	012746	006160			MOV #FMT2,-(SP)	:			
2833	036600	012746	000002			MOV #2,-(SP)	:			
2834	036604	010600				MOV SP,R0	: SP,*			
2835	036606	104414				TRAP 14	:			
2836	036610	011516				MOV (R5),(SP)	:			2723
2837	036612	012746	010322			MOV #PHR16,-(SP)	:			
2838	036616	016746	143406			MOV TSECT,-(SP)	:			
2839	036622	012746	010266			MOV #PHR14,-(SP)	:			
2840	036626	012746	006654			MOV #FMT11,-(SP)	:			
2841	036632	012746	000005			MOV #5,-(SP)	:			
2842	036636	010600				MOV SP,R0	: SP,*			
2843	036640	104414				TRAP 14	:			
2844	036642	112763	000003	034446		MOVB #3,WHY.DROPT(R3)	:			2725
2845	036650	010300				MOV R3,R0	:			2726
2846	036652	104451				TRAP 51	:			
2847	036654	000455				BR 10\$:			2714
2848	036656	026767	143344	143344	8\$:	CMP LSECT,TSECT	:			2730
2849	036664	101004				BHI 9\$:			
2850	036666	016764	143334	034460		MOV LSECT,LOW.SECT(R4)	:			2732
2851	036674	000450				BR 11\$:			2730
2852	036676	012746	007322		9\$:	MOV #WRD21,-(SP)	:			2735
2853	036702	012746	007240			MOV #WRD11,-(SP)	:			
2854	036706	012746	007100			MOV #SAY2,-(SP)	:			
2855	036712	012746	000003			MOV #3,-(SP)	:			
2856	036716	010600				MOV SP,R0	: SP,*			
2857	036720	104414				TRAP 14	:			
2858	036722	012716	010526			MOV #CAUSE3,(SP)	:			2737
2859	036726	012746	006160			MOV #FMT2,-(SP)	:			
2860	036732	012746	000002			MOV #2,-(SP)	:			
2861	036736	010600				MOV SP,R0	: SP,*			
2862	036740	104414				TRAP 14	:			
2863	036742	016716	143262			MOV TSECT,(SP)	:			2739
2864	036746	012746	010266			MOV #PHR14,-(SP)	:			
2865	036752	016746	143250			MOV LSECT,-(SP)	:			
2866	036756	012746	010304			MOV #PHR15,-(SP)	:			
2867	036762	012746	006654			MOV #FMT11,-(SP)	:			
2868	036766	012746	000005			MOV #5,-(SP)	:			
2869	036772	010600				MOV SP,R0	: SP,*			
2870	036774	104414				TRAP 14	:			
2871	036776	112763	000003	034446		MOVB #3,WHY.DROPT(R3)	:			2741
2872	037004	010300				MOV R3,R0	:			2742

Address	OpCode	Operand1	Operand2	Label	Instruction	Comments	Start	End	PA
2874									
2875									
2876									
2877	037006	104451			TRAP	51			
2878	037010	062706	000034	10\$:	ADD	#34,SP			2730
2879	037014	000207			RTS	PC			2734
2880	037016	011516		11\$:	MOV	(R5),(SP)			2752
2881	037020	016446	034460		MOV	LOW.SECT(R4),-(SP)			
2882	037024	010246			MOV	R2,-(SP)			
2883	037026	012746	006320		MOV	#FMT5,-(SP)			
2884	037032	012746	000004		MOV	#4,-(SP)			
2885	037036	010690			MOV	SP,R0			
2886	037040	104414			TRAP	14			
2887	037042	017701	175324		MOV	@ML.REG+24,R1			2755
2888	037046	000301			SWAB	R1			
2889	037050	042701	177774		BIC	#177774,R1			
2890	037054	001003			BNE	12\$			
2891	037056	012700	010444		MOV	#TRT00,R0			2759
2892	037062	000421			BR	15\$			2755
2893	037064	020127	000001	12\$:	CMP	R1,#1			
2894	037070	001003			BNE	13\$			
2895	037072	012700	010446		MOV	#TRT01,R0			2762
2896	037076	000413			BR	15\$			2755
2897	037100	020127	000002	13\$:	CMP	R1,#2			
2898	037104	001003			BNE	14\$			
2899	037106	012700	010450		MOV	#TRT10,R0			2765
2900	037112	000405			BR	15\$			2755
2901	037114	020127	000003	14\$:	CMP	R1,#3			
2902	037120	001002			BNE	15\$			
2903	037122	012700	010454		MOV	#TRT11,R0			2768
2904	037126	010016		15\$:	MOV	R0,(SP)			2771
2905	037130	012746	006466		MOV	#FMT8,-(SP)			
2906	037134	012746	000002		MOV	#2,-(SP)			
2907	037140	010600			MOV	SP,R0			
2908	037142	104414			TRAP	14			
2909	037144	016716	173536		MOV	BASE.ADDR,(SP)			2773
2910	037150	012746	007770		MOV	#PHR4,-(SP)			
2911	037154	012746	006252		MOV	#FMT4B,-(SP)			
2912	037160	012746	000003		MOV	#3,-(SP)			
2913	037164	010600			MOV	SP,R0			
2914	037166	104414			TRAP	14			
2915	037170	062706	000030		ADD	#30,SP			2581
2916	037174	000207			RTS	PC			
2917									
2918									
2919									
2924									
2925									

: Routine Size: 318 words
 : Maximum stack depth per invocation: 20 words

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (9)

```

2927 :MLX4
2928 :
2929 :
2930 : 2779 routine DECODE (SECT) : novalue =
2931 : 2780 begin
2932 : 2781
2933 : 2782 :++
2934 : 2783 ROUTINE: DECODE(SECT)
2935 : 2784
2936 : 2785 PURPOSE: TO INTERPRET THE FIELDS OF THE FAILING SECTOR
2937 : 2786 FOR BOTH 16K AND 64K CHIPS.
2938 : 2787
2939 : 2788 ARGUMENT: SECT = FAILING SECTOR NUMBER
2940 : 2789
2941 : 2790 RESULTS: (1) BANK = WHICH ROW OF CHIPS (0-4)
2942 : 2791
2943 : 2792 (2) BOARD = WHICH ARRAY MODULE (0-15)
2944 : 2793 --
2945 : 2794
2946 : 2795 if .MLDT
2947 : 2796 then !THE CHIPS ARE 64K
2948 : 2797 begin
2949 : 2798 BANK = .SECT<9, 2>;
2950 : 2799 BOARD = .SECT<11, 4>;
2951 : 2800 end
2952 : 2801 else !THE CHIPS ARE 16K
2953 : 2802 begin
2954 : 2803 BANK = .SECT<7, 2>;
2955 : 2804 BOARD = .SECT<9, 4>;
2956 : 2805 end;
2957 : 2806
2958 : 2807
2959 : 2808 ! VER CZMLBB ADDED LOADING OF BIT_NUM TO ROUTINE
2960 : 2809
2961 : 2810 BIT_NUM = .CHAN; !SAVE THE FAILING CHIP NUMBER
2962 : 2811 return;
2963 : 2812 end;
2967 :
2968 :
    
```

Address	Hex	Hex	Hex	Hex	Hex	Label	Instruction	Comment	Address
2972	037176	032777	000001	175170	DECODE:	.SBTTL	DECODE DRIVE IDENTIFICATION ROUTINES		
2973	037204	001415				BIT	#1,AML.REG+26	:	2795
2974	037206	016600	000002			BEQ	1\$		
2975	037212	006200				MOV	2(SP),R0	: SECT,*	2798
2976	037214	000300				ASR	R0		
2977	037216	042700	177774			SWAB	R0		
2978	037222	010067	175116			BIC	#177774,R0		
2979	037226	016600	000002			MOV	R0,BANK		
2980	037232	006200				MOV	2(SP),R0	: SECT,*	2799
2981	037234	006200				ASR	R0		
						ASR	R0		

27-Mar-1982 19:24:42 TOPS
 27-Mar-1982 19:23:44 PA:<

```

2983      ;MLX4
2984      ;
2985      ;
2986 037236 000412      BR      2$
2987 037240 016600 000002 1$:  MOV      2(SP),RO      ; SECT,*      2803
2988 037244 006100      ROL      RO
2989 037246 000300      SWAB     RO
2990 037250 042700 177774      BIC      #177774,RO
2991 037254 010067 175064      MOV      RO,BANK
2992 037260 016600 000002      MOV      2(SP),RO      ; SECT,*      2804
2993 037264 006200      ASR      RO
2994 037266 000300      SWAB     RO
2995 037270 042700 177760      BIC      #177760,RO
2996 037274 010067 175042      MOV      RO,BOARD
2997 037300 017700 175104      MOV      @ML.REG+42,RO      ;
2998 037304 006200      ASR      RO
2999 037306 006200      ASR      RO
3000 037310 006200      ASR      RO
3001 037312 006200      ASR      RO
3002 037314 006200      ASR      RO
3003 037316 006200      ASR      RO
3004 037320 042700 177700      BIC      #177700,RO
3005 037324 010067 153336      MOV      RO,BIT.NUM
3006 037330 000207      RTS      PC      ;
3007      ; Routine Size: 46 words
3008      ; Maximum stack depth per invocation: 0 words
3009
3014
3015
    
```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (10)

```

3017 :MLX4
3018 :
3019 : DRIVE IDENTIFICATION ROUTINES
3020 : 2813 routine ISOLATE : novalue =
3021 : 2814 begin
3022 : 2815
3023 : 2816 |++
3024 : 2817 | ROUTINE: ISOLATE
3025 : 2818 |
3026 : 2819 | PURPOSE: UPON THE DETECTION OF A DATA ERROR (EITHER RECOVERABLE OR
3027 : 2820 | NOT) TO EXAMINE THE SECTOR ADDRESS CONTAINED IN THE ECC
3028 : 2821 | ERROR LOCATION REGISTER AND PINPOINT THE LOCATION OF THE
3029 : 2822 | ERROR. THE SECTOR, BOARD (0-15) AND BANK (0-3) WILL BE
3030 : 2823 | REPORTED.
3031 : 2824 |---
3032 : 2825
3033 : 2826 DECODE (.MLEL);
3034 : 2827
3035 : 2828 if .ERROUT then PRINTB (FMT10A, WRD20, WRD15, .MLEL, .BOARD, .BANK);
3036 : 2829
3037 : 2830 !'FAILED: SECTOR XXXXXX BOARD YY BANK Z'
3038 : 2831 return;
3039 : 2832 end;
    
```

```

3043
3044 :SBTTL ISOLATE DRIVE IDENTIFICATION ROUTINES
3048 037332 017746 175054 ISOLATE:MOV @ML.REG+44,-(SP) 2826
3049 037336 004767 177634 JSR PC,DECODE
3050 037342 032767 000001 142710 BIT #1,ERROUT 2828
3051 037350 001422 BEQ 1$
3052 037352 016746 174766 MOV BANK,-(SP)
3053 037356 016746 174760 MOV BOARD,-(SP)
3054 037362 017746 175024 MOV @ML.REG+44,-(SP)
3055 037366 012746 007246 MOV #WRD15,-(SP)
3056 037372 012746 007312 MOV #WRD20,-(SP)
3057 037376 012746 006564 MOV #FMT10A,-(SP)
3058 037402 012746 000006 MOV #6,-(SP)
3059 037406 010600 MOV SP,R0 : SP,*
3060 037410 104414 TRAP 14
3061 037412 062706 000016 ADD #16,SP
3062 037416 005726 1$: TST (SP)+ : 2813
3063 037420 000207 RTS PC
3064
3065 : Routine Size: 28 words
3066 : Maximum stack depth per invocation: 8 words
    
```


27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (11)

3072 :MLX4
 3073 :
 3074 :
 3075 :
 3076 :
 3077 :
 3078 :
 3079 :
 3080 :
 3081 :
 3082 :
 3083 :
 3084 :
 3085 :
 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 :

DRIVE IDENTIFICATION ROUTINES

2833 routine UP_HARD_COUNT (LUN, ARRAY) : novalue =
 2834 begin

2836 !++
 2837 ROUTINE: UP_HARD_COUNT(LUN,ARRAY)

2838 PURPOSE: TO INCREMENT THE HARD ERROR COUNT FOR THE GIVEN
 2839 ARRAY, AND TO SEE IF THE HARD ERROR THRESHOLD HAS
 2840 BEEN REACHED.
 2841

2842 ARGUMENTS: LUN = LOGICAL UNIT WHICH HAS THE HARD ERROR
 2843 ARRAY = THE BOARD NUMBER (0-15)
 2844
 2845 --

2846 local ! VER CZMLBB ADDED THIS LOCAL STORAGE
 2847 SBE_EXIST; !Indicates if the searched sbe already exists

2848 HARDS [.LUN, .ARRAY, 0, 16, 0] = .HARDS [.LUN, .ARRAY, 0, 16, 0] + 1;

2850 VER CZMLBB ADDED FOLLOWING CODE TO LOG SBE'S

2851
 2852 First search the sbe log table and see if
 2853 this single bit error already exists in
 2854 the table. If it does exist then just
 2855 increment its occurrence count, else
 2856 record its failing location into the
 2857 table.
 2858

2859 Only log away 128 single bit errors.

2860 if .SBES_COUNT lss 127 !Have 128 errors been logged yet

2861 then
 2862 begin !There is room for at least one more sbe
 2863 SBE_EXIST = FALSE; !Init the existance flag to false

2864 !+
 2865 Search the single bit error log table
 2866 and see if this error already exist.
 2867 -

2868 incr index from 0 to .SBES_COUNT do
 2869 begin

2870 if (.SBE_LOG [.index, BITS_SBE] eql .BIT_NUM) and !Does this bit exist
 2871 (.SBE_LOG [.index, BNK\$SBE] eql .BANK) and !Does this bank exist
 2872 (.SBE_LOG [.index, BRD\$SBE] eql .BOARD) and !Does this board exist
 2873 (.SBE_LOG [.index, UNIT\$SBE] eql .LUN) !Does this unit exist

2874 then
 2875 begin !This sbe already exist so just up its occurrence count
 2876 SBE_LOG [.index, SUMS_SBE] = .SBE_LOG [.index, SUMS_SBE] + 1;
 2877 SBE_EXIST = TRUE; !Indicate that this sbe already exist

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (11)

3128 :MLX4
 3129 :
 3130 :
 3131 :
 3132 :
 3133 :
 3134 :
 3135 :
 3136 :
 3137 :
 3138 :
 3139 :
 3140 :
 3141 :
 3142 :
 3143 :
 3144 :
 3145 :
 3146 :
 3147 :
 3148 :
 3149 :
 3150 :
 3151 :
 3152 :
 3153 :
 3154 :
 3155 :
 3156 :
 3157 :
 3158 :
 3159 :
 3160 :
 3161 :
 3162 :
 3163 :
 3164 :
 3165 :
 3166 :
 3167 :
 3168 :
 3169 :
 3170 :
 3171 :
 3172 :
 3173 :
 3174 :
 3175 :
 3176 :
 3177 :
 3178 :
 3179 :
 3180 :
 3181 :
 3182 :

DRIVE IDENTIFICATION ROUTINES

```

exitloop;
end;

end;

+
Test to see if this sbe was already in the table
by testing the flag sbe_exist. If it did not exist then
up the Prom Maintenance count, load this sbe into the table
and test to see if this array module needs to be Prom Maintained.
If the sbe did exist then return from this routine.
-

if not .SBE_EXIST !Was this sbe found in the table
then
begin !It was not found in the table so up the pm count
PM_SBE_CNT [.LUN, .BOARD, PM_SBE$SUM] = .PM_SBE_CNT [.LUN, .BOARD, PM_SBE$SUM] + 1;
!Now see if the Prom Maint Program needs to be
run on this array module.

if .PM_SBE_CNT [.LUN, .BOARD, PM_SBE$SUM] eql 10 !Is the limit exceeded
then
begin !The limit was exceeded
SAYWHO (.LUN);
PRINTB (FMT13, .ARRAY, MSG2);
!'ARRAY XX --> RUN ML11 PROM MAINTENANCE PROGRAM'
end;

+
This sbe was not found in the table so
load this failing sbe into the table
at the bottom of the list.
-

SBE_LOG [.SBE$COUNT, BITS_SBE] = .BIT_NUM; !Load the failing bit
SBE_LOG [.SBE$COUNT, BNKS_SBE] = .BANK; !Load the failing bank
SBE_LOG [.SBE$COUNT, BRD$SBE] = .BOARD; !Load the failing board
SBE_LOG [.SBE$COUNT, UNITS_SBE] = .LUN; !Load the failing unit
SBE_LOG [.SBE$COUNT, SUMS_SBE] = 1; !Indicate this is the first one
SBE$COUNT = .SBE$COUNT + 1; !Up the count of unique sbe's detected
end;

end;

!
! VER COMMENTED OUT THIS CODE
! REPLACED BY ABOVE CODE
!
if (((.ARR_TYP eql 1) and (.HARDS [.LUN, .ARRAY, 0, 16, 0] eql H64K_LIMIT)) or ((.ARR_TYP eql 0) and (
.HARDS [.LUN, .ARRAY, 0, 16, 0] eql H16K_LIMIT)))

```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (11)

```

3184 ;MLX4
3185 ;
3186 ;
3187 : 2937 : then
3188 : 2938 : begin
3189 : 2939 : SAYWHO (.LUN);
3190 : 2940 : PRINTB (FMT13, .ARRAY, MSG2);
3191 : 2941 : !'ARRAY XX --> RUN ML11 PROM MAINTENANCE PROGRAM'
3192 : 2942 : end;
3193 : 2943 : return;
3194 : 2944 : end;
3198 ;
3199 ;
    
```

```

3203 037422 .SBTTL UP.HARD.COUNT DRIVE IDENTIFICATION ROUTINES
3204 037422 004167 145706 UP.HARD.COUNT: JSR R1,$SAVE5 ; 2833
3205 037426 016605 000020 MOV 20(SP),R5 ; LUN,* 2850
3206 037432 010500 MOV R5,R0
3207 037434 006300 ASL R0
3208 037436 006300 ASL R0
3209 037440 006300 ASL R0
3210 037442 006300 ASL R0
3211 037444 010002 MOV R0,R2
3212 037446 066602 000016 ADD 16(SP),R2 ; ARRAY,*
3213 037452 006302 ASL R2
3214 037454 005262 033314 INC HARDS(R2)
3215 037460 026727 153200 000177 CMP SBES.COUNT,#177 ; 2864
3216 037466 002401 BLT 1$
3217 037470 000207 RTS PC
3218 037472 005004 1$: CLR R4 ; SBE.EXIST 2867
3219 037474 005001 CLR R1 ; INDEX 2874
3220 037476 000463 BR 4$
3221 037500 010102 2$: MOV R1,R2 ; INDEX,* 2877
3222 037502 006302 ASL R2
3223 037504 006302 ASL R2
3224 037506 012703 011264 MOV #SBE.LOG,R3
3225 037512 060203 ADD R2,R3
3226 037514 016746 153146 MOV BIT.NUM,-(SP)
3227 037520 011346 MOV (R3),-(SP)
3228 037522 006216 ASR (SP)
3229 037524 006216 ASR (SP)
3230 037526 006216 ASR (SP)
3231 037530 006216 ASR (SP)
3232 037532 006216 ASR (SP)
3233 037534 006216 ASR (SP)
3234 037536 042716 177600 BIC #177600,(SP)
3235 037542 022626 CMP (SP)+,(SP)+
3236 037544 001037 BNE 3$
3237 037546 016746 174572 MOV BANK,-(SP) ; 2878
3238 037552 111346 MOVB (R3),-(SP)
    
```

Address	OpCode	Operand1	Operand2	Instruction	Comment	Address
3240						
3241						
3242						
3243	037554	042716	177774	BIC	#177774,(SP)	
3244	037560	022626		CMP	(SP)+,(SP)+	
3245	037562	001030		BNE	3\$	
3246	037564	016746	174552	MOV	BOARD,-(SP)	2879
3247	037570	111346		MOVB	(R3),-(SP)	
3248	037572	006216		ASR	(SP)	
3249	037574	006216		ASR	(SP)	
3250	037576	042716	177760	BIC	#177760,(SP)	
3251	037602	022626		CMP	(SP)+,(SP)+	
3252	037604	001017		BNE	3\$	
3253	037606	010546		MOV	R5,-(SP)	2880
3254	037610	011346		MOV	(R3),-(SP)	
3255	037612	006116		ROL	(SP)	
3256	037614	006116		ROL	(SP)	
3257	037616	006116		ROL	(SP)	
3258	037620	006116		ROL	(SP)	
3259	037622	042716	177770	BIC	#177770,(SP)	
3260	037626	022626		CMP	(SP)+,(SP)+	
3261	037630	001005		BNE	3\$	
3262	037632	005262	011266	INC	SBE.LOG+2(R2)	2883
3263	037636	012704	000001	MOV	#1,R4	2884
3264	037642	000404		BR	5\$	2885
3265	037644	005201		INC	R1	2874
3266	037646	020167	153012	CMP	R1,SBE\$.COUNT	INDEX,*
3267	037652	003712		BLE	2\$	INDEX,*
3268	037654	006004		ROR	R4	SBE.EXIST
3269	037656	103511		BLO	7\$	2898
3270	037660	010002		MOV	R0,R2	2901
3271	037662	066702	174454	ADD	BOARD,R2	
3272	037666	006302		ASL	R2	
3273	037670	005262	012264	INC	PM.SBE.CNT(R2)	
3274	037674	026227	012264 000012	CMP	PM.SBE.CNT(R2),#12	2907
3275	037702	001017		BNE	6\$	
3276	037704	010546		MOV	R5,-(SP)	2910
3277	037706	004767	175612	JSR	PC,SA:WHO	
3278	037712	012716	011120	MOV	#MSG2,(SP)	2911
3279	037716	016646	000020	MOV	20(SP),-(SP)	ARRAY,*
3280	037722	012746	007024	MOV	#FMT13,-(SP)	
3281	037726	012746	000003	MOV	#3,-(SP)	
3282	037732	010600		MOV	SP,R0	SP,*
3283	037734	104414		TRAP	14	
3284	037736	062706	000010	ADD	#10,SP	2909
3285	037742	016702	152716	MOV	SBE\$.COUNT,R2	2921
3286	037746	006302		ASL	R2	
3287	037750	006302		ASL	R2	
3288	037752	012701	011264	MOV	#SBE.LOG,R1	
3289	037756	060201		ADD	R2,R1	
3290	037760	016704	152702	MOV	BIT.NUM,R4	
3291	037764	000304		SWAB	R4	
3292	037766	106004		RORB	R4	
3293	037770	006004		ROR	R4	
3294	037772	006004		ROR	R4	

27-Mar-1982 19:24:42 TOPS
 27-Mar-1982 19:23:44 PA:<

3296								
3297				:MLX4				
3298				:		DRIVE IDENTIFICATION ROUTINES		
3299	037774	042704	160077		BIC	#160077,R4		
3300	040000	042711	017700		BIC	#17700,(R1)		
3301	040004	050411			BIS	R4,(R1)		
3302	040006	016704	174332		MOV	BANK,R4	:	2922
3303	040012	042704	177774		BIC	#177774,R4		
3304	040016	142711	000003		BICB	#3,(R1)		
3305	040022	150411			BISB	R4,(R1)		
3306	040024	016704	174312		MOV	BOARD,R4	:	2923
3307	040030	006304			ASL	R4		
3308	040032	006304			ASL	R4		
3309	040034	042704	177703		BIC	#177703,R4		
3310	040040	142711	000074		BICB	#74,(R1)		
3311	040044	150411			BISB	R4,(R1)		
3312	040046	006005			ROR	R5	:	2924
3313	040050	006005			ROR	R5		
3314	040052	006005			ROR	R5		
3315	040054	006005			ROR	R5		
3316	040056	042705	017777		BIC	#17777,R5		
3317	040062	042711	160000		BIC	#160000,(R1)		
3318	040066	050511			BIS	R5,(R1)		
3319	040070	012762	000001	011266	MOV	#1,SBE.LOG+2(R2)	:	2925
3320	040076	005267	152562		INC	SBE\$.COUNT	:	2926
3321	040102	000207		7\$:	RTS	PC	:	2833
3322								
3323					:	Routine Size: 153 words		
3324					:	Maximum stack depth per invocation: 10 words		
3329								
3330								

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (12)

```

3332 :MLX4
3333 :
3334 :
3335 : 2945 routine UP_SOFT_COUNT (LUN, ARRAY) : novalue =
3336 : 2946 begin
3337 : 2947
3338 : 2948 !++
3339 : 2949 ROUTINE: UP_SOFT_COUNT(LUN,ARRAY)
3340 : 2950
3341 : 2951 PURPOSE: TO INCREMENT THE SOFT ERROR COUNT FOR THE GIVEN
3342 : 2952 ARRAY, AND TO SEE IF THE SOFT ERROR THRESHOLD HAS
3343 : 2953 BEEN REACHED.
3344 : 2954
3345 : 2955 ARGUMENTS: LUN = LOGICAL UNIT WHICH HAS THE SOFT ERROR
3346 : 2956 ARRAY = THE BOARD NUMBER (0-15)
3347 : 2957 !--
3348 : 2958
3349 : 2959 local ! VER CZMLBB ADD THIS LOCAL STORAGE
3350 : 2960 SBE_EXIST; !Flag to indicate if sbe already exists
3351 : 2961
3352 : 2962 SOFTS [.LUN, .ARRAY, 0, 16, 0] = .SOFTS [.LUN, .ARRAY, 0, 16, 0] + 1;
3353 : 2963
3354 : 2964 ! VER CZMLBB ADDED FOLLOWING CODE TO LOG SBE'S
3355 : 2965
3356 : 2966 ! First search the sbe log table and see if
3357 : 2967 ! this single bit error already exists in
3358 : 2968 ! the table. If it does exist then just
3359 : 2969 ! increment its occurrence count, else
3360 : 2970 ! record its failing location into the
3361 : 2971 ! table.
3362 : 2972
3363 : 2973 ! Only log away 128 single bit errors.
3364 : 2974
3365 : 2975
3366 : 2976 if .SBE$_COUNT lss 127 !Have 128 errors been logged yet
3367 : 2977 then
3368 : 2978 begin !There is room for at least one more sbe
3369 : 2979 SBE_EXIST = FALSE; !Init the existence flage to false
3370 : 2980
3371 : 2981 !+
3372 : 2982 ! Search the single bit error log table
3373 : 2983 ! and see if this error already exist.
3374 : 2984 !-
3375 : 2985
3376 : 2986 incr index from 0 to .SBE$_COUNT do
3377 : 2987 begin
3378 : 2988
3379 : 2989 if (.SBE_LOG [.index, BITS_SBE] eql .BIT_NUM) and !Does this bit exist
3380 : 2990 (.SBE_LOG [.index, BANK$_SBE] eql .BANK) and !Does this bank exist
3381 : 2991 (.SBE_LOG [.index, BRD$_SBE] eql .BOARD) and !Does this board exist
3382 : 2992 (.SBE_LOG [.index, UNIT$_SBE] eql .LUN) !Does this unit exist
3383 : 2993 then
3384 : 2994 begin !This sbe already exist so just up its occurrence count
3385 : 2995 SBE_LOG [.index, SUM$_SBE] = .SBE_LOG [.index, SUM$_SBE] + 1;
3386 : 2996 SBE_EXIST = TRUE; !Indicate that this sbe already exist
    
```

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (12)

3388 :MLX4
 3389 :
 3390 :
 3391 :
 3392 :
 3393 :
 3394 :
 3395 :
 3396 :
 3397 :
 3398 :
 3399 :
 3400 :
 3401 :
 3402 :
 3403 :
 3404 :
 3405 :
 3406 :
 3407 :
 3408 :
 3409 :
 3410 :
 3411 :
 3412 :
 3413 :
 3414 :
 3415 :
 3416 :
 3417 :
 3418 :
 3419 :
 3420 :
 3421 :
 3422 :
 3423 :
 3424 :
 3425 :
 3426 :
 3427 :
 3428 :
 3429 :
 3430 :
 3431 :
 3432 :
 3433 :
 3434 :
 3435 :
 3436 :
 3437 :
 3438 :
 3439 :

DRIVE IDENTIFICATION ROUTINES

```

        exitloop;
    end;

    end;

+
Test to see if this sbe was already in the table
by testing the flag sbe_exist. If it did not exist then
enter this SBE into the table.
NOTE:
Soft errors are not counted toward the Prom Maintenance
call out.
-

if not .SBE_EXIST                !Was this sbe found in the table
then
begin                            !It was not found in the table so up the pm count

+
This sbe was not found in the table so
load this failing sbe into the table
at the bottom of the list.
-

SBE_LOG [.SBE$COUNT, BITS$SBE] = .BIT_NUM; !Load the failing bit
SBE_LOG [.SBE$COUNT, BNKS$SBE] = .BANK;    !Load the failing bank
SBE_LOG [.SBE$COUNT, BRDS$SBE] = .BOARD;  !Load the failing board
SBE_LOG [.SBE$COUNT, UNITS$SBE] = .LUN;   !Load the failing unit
SBE_LOG [.SBE$COUNT, SUMS$SBE] = 1;       !Indicate this is the first one
SBE$COUNT = .SBE$COUNT + 1;             !Up the count of unique sbe's detected
end;

end;

:
VER CZMLBB THIS CODE WAS COMMENTED OUT AND REPLACED BY THE
ABOVE CODE.

if (((.ARR_TYP eql 1) and (.SOFTS [.LUN, .ARRAY, 0, 16, 0] eql S64K_LIMIT)) or ((.ARR_TYP eql 0) and (
.SOFTS [.LUN, .ARRAY, 0, 16, 0] eql S16K_LIMIT)))
then
begin
SAYWHO (.LUN);
PRINTB (FMT13, .ARRAY, MSG2);
!'ARRAY XX --> RUN ML11 PROM MAINTENANCE PROGRAM'
end;
return;
end;
    
```

3444				:MLX4				
3445				:		DRIVE IDENTIFICATION ROUTINES		
3446								
3447								
3448								
3452	040104			UP.SOFT.COUNT:	.SBTTL	UP.SOFT.COUNT DRIVE IDENTIFICATION ROUTINES		
3453	040104	004167	145224		JSR	R1,\$SAVE5	:	2945
3454	040110	016605	000020		MOV	20(SP),R5	: LUN,*	2962
3455	040114	010500			MOV	R5,R0		
3456	040116	006300			ASL	R0		
3457	040120	006300			ASL	R0		
3458	040122	006300			ASL	R0		
3459	040124	006300			ASL	R0		
3460	040126	066600	000016		ADD	16(SP),R0	: ARRAY,*	
3461	040132	006300			ASL	R0		
3462	040134	005260	032714		INC	SOFTS(R0)		
3463	040140	016701	152520		MOV	SBE\$.COUNT,R1	:	2976
3464	040144	020127	000177		CMP	R1,#177		
3465	040150	002150			BGE	5\$		
3466	040152	005003			CLR	R3	: SBE.EXIST	2979
3467	040154	005000			CLR	R0	: INDEX	2986
3468	040156	000463			BR	3\$		
3469	040160	010004		1\$:	MOV	R0,R4	: INDEX,*	2989
3470	040162	006304			ASL	R4		
3471	040164	006304			ASL	R4		
3472	040166	012702	011264		MOV	#SBE.LOG,R2		
3473	040172	060402			ADD	R4,R2		
3474	040174	016746	152466		MOV	BIT.NUM,-(SP)		
3475	040200	011246			MOV	(R2),-(SP)		
3476	040202	006216			ASR	(SP)		
3477	040204	006216			ASR	(SP)		
3478	040206	006216			ASR	(SP)		
3479	040210	006216			ASR	(SP)		
3480	040212	006216			ASR	(SP)		
3481	040214	006216			ASR	(SP)		
3482	040216	042716	177600		BIC	#177600,(SP)		
3483	040222	022626			CMP	(SP)+,(SP)+		
3484	040224	001037			BNE	2\$		
3485	040226	016746	174112		MOV	BANK,-(SP)	:	2990
3486	040232	111246			MOVB	(R2),-(SP)		
3487	040234	042716	177774		BIC	#177774,(SP)		
3488	040240	022626			CMP	(SP)+,(SP)+		
3489	040242	001030			BNE	2\$		
3490	040244	016746	174072		MOV	BOARD,-(SP)	:	2991
3491	040250	111246			MOVB	(R2),-(SP)		
3492	040252	006216			ASR	(SP)		
3493	040254	006216			ASR	(SP)		
3494	040256	042716	177760		BIC	#177760,(SP)		
3495	040262	022626			CMP	(SP)+,(SP)+		
3496	040264	001017			BNE	2\$		
3497	040266	010546			MOV	R5,-(SP)	:	2992
3498	040270	011246			MOV	(R2),-(SP)		

27-Mar-1982 19:24:42 TOPS
 27-Mar-1982 19:23:44 PA:<

Address	OpCode	Operand1	Operand2	Comment	Label	Value
3500				:MLX4		
3501				:		
3502						
3503	040272	006116		ROL	(SP)	
3504	040274	006116		ROL	(SP)	
3505	040276	006116		ROL	(SP)	
3506	040300	006116		ROL	(SP)	
3507	040302	042716	177770	BIC	#177770,(SP)	
3508	040306	022626		CMP	(SP)+,(SP)+	
3509	040310	001005		BNE	2\$	
3510	040312	005264	011266	INC	SBE.LOG+2(R4)	2995
3511	040316	012703	000001	MCV	#1,R3	2996
3512	040322	000403		BR	4\$	2997
3513	040324	005200		2\$: INC	R0	INDEX
3514	040326	020001		3\$: CMP	R0,R1	INDEX,*
3515	040330	003713				
3516	040332	006003		4\$: ROR	R3	SBE.EXIST
3517	040334	103456				
3518	040336	006301		BLO	5\$	
3519	040340	006301		ASL	R1	3012
3520	040342	012700	011264	ASL	R1	
3521	040346	060100		MOV	#SBE.LOG,R0	
3522	040350	016704	152312	ADD	R1,R0	
3523	040354	000304		MOV	BIT.NUM,R4	
3524	040356	106004		SWAB	R4	
3525	040360	006004		RORB	R4	
3526	040362	006004		ROR	R4	
3527	040364	042704	160077	ROR	R4	
3528	040370	042710	017700	BIC	#160077,R4	
3529	040374	050410		BIC	#17700,(R0)	
3530	040376	016704	173742	BIS	R4,(R0)	
3531	040402	042704	177774	MOV	BANK,R4	3023
3532	040406	142710	000003	BIC	#177774,R4	
3533	040412	150410		BICB	#3,(R0)	
3534	040414	016704	173722	BISB	R4,(R0)	
3535	040420	006304		MOV	BOARD,R4	3024
3536	040422	006304		ASL	R4	
3537	040424	042704	177703	ASL	R4	
3538	040430	142710	000074	BIC	#177703,R4	
3539	040434	150410		BICB	#74,(R0)	
3540	040436	006005		BISB	R4,(R0)	
3541	040440	006005		ROR	R5	3025
3542	040442	006005		ROR	R5	
3543	040444	006005		ROR	R5	
3544	040446	042705	017777	BIC	#17777,R5	
3545	040452	042710	160000	BIC	#160000,(R0)	
3546	040456	050510		BIS	R5,(R0)	
3547	040460	012761	000001 011266	MOV	#1,SBE.LOG+2(R1)	3026
3548	040466	005267	152172	INC	SBE\$.COUNT	3027
3549	040472	000207		5\$: RTS	PC	2945
3550						
3551				: Routine Size: 124 words		
3552				: Maximum stack depth per invocation: 8 words		
3560						
3561						

```

3563 :MLX4
3564 :
3565 :
3566 : 3046 %sbttl 'INITIALIZATION CODE'
3567 : 3047 BGNINIT:
3568 : 3048
3569 : 3049
3570 : 3050
3571 : 3051
3572 : 3052
3573 : 3053
3574 : 3054
3575 : 3055
3576 : 3056
3577 : 3057
3578 : 3058
3579 : 3059
3580 : 3060
3581 : 3061
3582 : 3062
3583 : 3063
3584 : 3064
3585 : 3065
3586 : 3066
3587 : 3067
3588 : 3068
3589 : 3069
3590 : 3070
3591 : 3071
3592 : 3072
3593 : 3073
3594 : 3074
3595 : 3075
3596 : 3076
3597 : 3077
3598 : 3078
3599 : 3079
3600 : 3080
3601 : 3081
3602 : 3082
3603 : 3083
3604 : 3084
3605 : 3085
3606 : 3086
3607 : 3087
3608 : 3088
3609 : 3089
3610 : 3090
3611 : 3091
3612 : 3092
3613 : 3093
3614 : 3094
3615 : 3095
3616 : 3096
3617 : 3097

INITIALIZATION CODE
! * 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 ('LSUNIT').

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).

local
PLOC;
if ((READEF (EF_START)) or (READEF (EF_RESTART)))
then
begin
! * 2 *
QUICK = 1;
CLRTBLS ();

incr LUN from 0 to (.LSUNIT - 1) do
begin
! * 3 *
LSLUN = .LUN;

if GPHARD (.LUN, PLOC) neq 0
then
begin
! * 4 *
PTABLE_ADDR [.LUN] = .PLOC;

if .NUM_DRIVES eq 0 then INIT_ADDRESSES (.PLOC);

NUM_DRIVES = .NUM_DRIVES + 1;
DRIVE_STATUS [.LUN] = ACTIVE;
CONFIG (.LUN),
end
! * 4 *
else
DRIVE_STATUS [.LUN] = INACTIVE;

end;
! * 3 *
end;

```

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (13)

```

3619 :MLX4
3620 :      INITIALIZATION CODE
3621 :
3622 :      3098      end
3623 :      3099      else
3624 :      3100      QUICK = 0;
3625 :      3101
3626 :      3102      !CATEGORY 3 CODE WOULD GO HERE, BUT THERE IS NONE.
3627 :      3103      ENDINIT;
3631 :
3632 :
3633 :      .SBTTL      LINIT INITIALIZATION CODE
3634 :      LINIT:     JSR      R1,$SAVE4
3635 :                  MOV      #40,R0
3636 :                  TRAP     47
3637 :                  BCS      1$
3638 :                  MOV      #37,R0
3639 :                  TRAP     47
3640 :                  BHS      7$
3641 :                  MOV      #1,QUICK
3642 :                  JSR      PC,CLRTBLS
3643 :                  MOV      L$UNIT,R4
3644 :                  CLR      R3
3645 :                  BR       6$
3646 :                  MOV      R3,L$LUN
3647 :                  MOV      R3,R2
3648 :                  ASR      R2
3649 :                  ASR      R2
3650 :                  ASR      R2
3651 :                  ADD      #DRIVE.STATUS,R2
3652 :                  MOV      R3,R0
3653 :                  TRAP     42
3654 :                  MOV      R0,R1
3655 :                  BEQ      4$
3656 :                  MOV      R3,R0
3657 :                  ASL      R0
3658 :                  MOV      R1,PTABLE.ADDR(R0)
3659 :                  TST      NUM.DRIVES
3660 :                  BNE      3$
3661 :                  MOV      R1,-(SP)
3662 :                  JSR      PC,INIT.ADDRESSES
3663 :                  TST      (SP)+
3664 :                  INC      NUM.DRIVES
3665 :                  MOV      R2,-(SP)
3666 :                  MOV      R3,-(SP)
3667 :                  BIC      #177770,(SP)
3668 :                  MOV      #1,-(SP)
3669 :                  MOV      (SP),-(SP)
3670 :                  JSR      PC,BL$PU2
3671 :                  MOV      R3,(SP)
    
```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (13)

! * 2 *
 ! THIS IS CATEGORY 2 CODE
 ! * 1 *

3045
 3072
 3075
 3076
 3078
 3080
 3090
 3082
 3085
 3087
 3089
 3090
 3091

3675
 3676
 3677
 3678 040650 004767 175126
 3679 040654 000411
 3680 040656 010246
 3681 040660 010346
 3682 040662 042716 177770
 3683 040666 012746 000001
 3684 040672 005046
 3685 040674 004767 143714
 3686 040700 062706 000010
 3687 040704 005203
 3688 040706 020304
 3689 040710 002714
 3690 040712 000207
 3691 040714 005067 171754
 3692 040720 000207
 3693
 3694
 3695
 3700
 3701
 3705
 3706
 3710 040722 004767 177546
 3711 040726 104411
 3712 040730 000207
 3713
 3714
 3715
 3720
 3721

```

:MLX4
:
      JSR  PC,CONFIG
      BR   5$
4$:   MOV  R2,-(SP)
      MOV  R3,-(SP)
      BIC  #177770,(SP)
      MOV  #1,-(SP)
      CLR  -(SP)
      JSR  PC,BL$PU2
5$:   ADD  #10,SP
      INC  R3
6$:   CMP  R3,R4
      BLT  2$
      RTS  PC
7$:   CLR  QUICK
      RTS  PC
  
```

27-Mar-1982 19:24:42 TOPS
 27-Mar-1982 19:23:44 PA:<

3082
 3094

3079
 3078

3072
 3100
 3045

: Routine Size: 75 words
 : Maximum stack depth per invocation: 9 words

```

.SBTTL L$INIT INITIALIZATION CODE
L$INIT::JSR PC,LINIT
      TRAP 11
      RTS  PC
  
```

3100

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

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (14)

```

3723 :MLX4
3724 :
3725 :
3726 : 3104 %sbttl 'GENERATOR FOR OPTION 1'
3727 : 3105 routine GEN1 (SECTOR, COMP_FLAG) : novalue =
3728 : 3106 begin !* 1 *
3729 : 3107
3730 : 3108 !++
3731 : 3109 ROUTINE: GEN1(SECTOR,COMP_FLAG)
3732 : 3110
3733 : 3111 PURPOSE: TO GENERATE THE ENTIRE 4K-WORD WRITE BUFFER FOR OPTION 1.
3734 : 3112 THE GENERATION IS IN 16 GROUPS OF 256 WORDS (THE SIZE
3735 : 3113 OF 1 SECTOR). EACH GROUP OF 256 WORDS RECEIVES THE
3736 : 3114 SECTOR ADDRESS WHERE THE WORDS WILL BE TRANSFERRED.
3737 : 3115
3738 : 3116 ARGUMENTS: (1) SECTOR - CURRENT SECTOR NUMBER.
3739 : 3117
3740 : 3118 (2) COMP_FLAG - AN INDICATOR WHICH IS USED TO DECIDE
3741 : 3119 WHETHER THE LOCATION IN THE WRITE BUFFER
3742 : 3120 SHOULD BE COMPLEMENTED.
3743 : 3121 !--
3744 : 3122
3745 : 3123 local
3746 : 3124 OFFSET;
3747 : 3125
3748 : 3126 OFFSET = 0;
3749 : 3127
3750 : 3128 if .COMP_FLAG
3751 : 3129 then !GENERATE COMPLEMENT DATA
3752 : 3130
3753 : 3131 incru SECT from (.SECTOR) to ((.SECTOR) + 15) do
3754 : 3132 begin
3755 : 3133 BREAK;
3756 : 3134
3757 : 3135 incru COUNT from 1 to 256 do
3758 : 3136 begin !* 2A *
3759 : 3137 (WBUF + (.OFFSET)) = not (.SECT); !COMPLEMENT DATA
3760 : 3138 OFFSET = .OFFSET + 2;
3761 : 3139 end; !* 2A *
3762 : 3140
3763 : 3141 end
3764 : 3142
3765 : 3143 else !GENERATE REGULAR DATA
3766 : 3144
3767 : 3145 incru SECT from (.SECTOR) to ((.SECTOR) + 15) do
3768 : 3146 begin
3769 : 3147 BREAK;
3770 : 3148
3771 : 3149 incru COUNT from 1 to 256 do
3772 : 3150 begin !* 2B *
3773 : 3151 (WBUF + (.OFFSET)) = (.SECT); !REGULAR DATA
3774 : 3152 OFFSET = .OFFSET + 2;
3775 : 3153 end; !* 2B *
3776 : 3154
3777 : 3155 end;
  
```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (14)

```

3779 :MLX4
3780 :
3781 :
3782 : 3156
3783 : 3157
3784 : 3158
3788 :
    
```

```

return;
end;
!* 1 *
    
```

```

3789 .SBTTL GEN1 GENERATOR FOR OPTION 1
3793 040732 004167 144356 GEN1: JSR R1,$SAVE4 ; 3105
3794 040736 005002 CLR R2 ; OFFSET 3126
3795 040740 016601 000016 MOV 16(SP),R1 ; SECTOR,* 3131
3796 040744 010104 MOV R1,R4
3797 040746 062704 000017 ADD #17,R4
3798 040752 032766 000001 000014 BIT #1,14(SP) ; *,COMP.FLAG 3128
3799 040760 001437 BEQ 6$
3800 040762 010103 MOV R1,R3 ; *,SECT 3131
3801 040764 000416 BR 3$
3802 040766 104422 1$: TRAP 22 ; 3132
3803 040770 010301 MOV R3,R1 ; SECT,* 3137
3804 040772 005101 COM R1
3805 040774 012700 000001 MOV #1,R0 ; *,COUNT 3135
3806 041000 010162 012670 2$: MOV R1,WBUFF(R2) ; *,*(OFFSET) 3137
3807 041004 062702 000002 ADD #2,R2 ; *,OFFSET 3138
3808 041010 005200 INC R0 ; COUNT 3135
3809 041012 020027 000400 CMP R0,#400 ; COUNT,*
3810 041016 101770 BLOS 2$
3811 041020 005203 INC R3 ; SECT 3131
3812 041022 020304 3$: CMP R3,R4 ; SECT,*
3813 041024 101760 BLOS 1$
3814 041026 000207 RTS PC ; 3128
3815 041030 104422 4$: TRAP 22 ; 3146
3816 041032 012700 000001 MOV #1,R0 ; *,COUNT 3149
3817 041036 010162 012670 5$: MOV R1,WBUFF(R2) ; SECT,*(OFFSET) 3151
3818 041042 062702 000002 ADD #2,R2 ; *,OFFSET 3152
3819 041046 005200 INC R0 ; COUNT 3149
3820 041050 020027 000400 CMP R0,#400 ; COUNT,*
3821 041054 101770 BLOS 5$
3822 041056 005201 INC R1 ; SECT 3145
3823 041060 020104 6$: CMP R1,R4 ; SECT,*
3824 041062 101762 BLOS 4$
3825 041064 000207 RTS PC ; 3105
    
```

```

: Routine Size: 46 words
: Maximum stack depth per invocation: 5 words
    
```

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (15)

3834 .MLX4
 3835
 3836
 3837
 3838
 3839
 3840
 3841
 3842
 3843
 3844
 3845
 3846
 3847
 3848
 3849
 3850
 3851
 3852
 3853
 3854
 3855
 3856
 3857
 3858
 3859
 3860
 3861
 3865
 3866
 3870
 3871
 3872
 3873
 3874
 3875
 3876
 3877
 3882
 3883

PATTERN NUMBER SELECTION

```

3159 %sbttl 'PATTERN NUMBER SELECTION'
3160 routine SELPAT : novalue =
3161 begin
    
```

```

3162
3163 !++
3164 ROUTINE: SELPAT
3165
3166 PURPOSE: TO SELECT THE NEXT PATTERN NUMBER IN OPTION 2.
3167
3168 THIS ROUTINE AUTOMATICALLY COMPLEMENTS THE CURRENT
3169 PATTERN NUMBER AND EXAMINES THE SIGN OF THE RESULT.
3170 IF THE RESULT IS NEGATIVE, THEN IT IS TAKEN TO BE
3171 THE PATTERN NUMBER OF A COMPLEMENT PATTERN AND A
3172 'RETURN' IS EXECUTED. IF, HOWEVER, THE RESULT IS
3173 POSITIVE, THEN AN ENTIRELY NEW PATTERN NUMBER IS
3174 REQUIRED. THE NEW PATTERN NUMBER IS OBTAINED BY
3175 INCREMENTING THE OLD.
    
```

```

3176 !--
3177
3178 PATTERN = -(.PATTERN);
3179
3180 if .PATTERN geq 0 then PATTERN = .PATTERN + 1;
3181
3182 return;
3183 end;
    
```

```

.SBTTL SELPAT PATTERN NUMBER SELECTION
SELPAT: NEG PATTERN ;
          TST PATTERN ;
          BLT 1$ ;
          INC PATTERN ;
1$:      RTS PC ;
    
```

3178
 3180
 3160

: Routine Size: 8 words
 : Maximum stack depth per invocation: 0 words

3885 :MLX4
3886 :
3887 :
3888 :
3889 :
3890 :
3891 :
3892 :
3893 :
3894 :
3895 :
3896 :
3897 :
3898 :
3899 :
3900 :
3901 :
3902 :
3903 :
3904 :
3905 :
3906 :
3907 :
3908 :
3909 :
3910 :
3911 :
3912 :
3913 :
3914 :
3915 :
3916 :
3917 :
3918 :
3919 :
3920 :
3921 :
3922 :
3923 :
3924 :
3925 :
3926 :
3927 :
3928 :
3929 :
3930 :
3931 :
3932 :
3933 :
3934 :
3935 :
3936 :
3937 :
3938 :
3939 :
3940 :MLX4
3941 :

```
USING THE PATTERN TABLE

%sbttl '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;
        [5 to 10] :
          begin
            PATNUM = .PATTERN - 6;
            VALUE = %b'1111';
          end;
        [-10 to -6] :
          begin
            PATNUM = -(.PATTERN) - 6;

```

27-Mar-1982 19:24:42
27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
PA:<NEALE>MLX4.BLI.5 (16)

27-Mar-1982 19:24:42
27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
PA:<NEALE>MLX4.BLI.5 (16)


```

3942
3943 :      3236
3944 :      3237
3945 :      3238
3946 :      3239
3947 :      3240
3948 :      3241
3949 :      3242
3950 :      3243
3954
3955

```

```

VALUE = %b'0000';
end;
tes:
DATA_COUNT = .PATTBL [.PATNUM, COUNT1];
COMP_COUNT = .PATTBL [.PATNUM, COUNT2];
return .VALUE;
end;

```

```

3959 041106 004167 144150          GETCNTS: JSR   .SBTTL GETCNTS USING THE PATTERN TABLE          3185
3960 041112 016600 000010          MOV   R1,$SAVE2                                :          3212
3961 041116 003410          BLE   1$                                       : PATTERN,*
3962 041120 020027 000005          CMP   R0,#5                                    :
3963 041124 003005          BGT   1$                                       :
3964 041126 010001          MOV   R0,R1                                   : *,PATNUM          3217
3965 041130 005301          DEC   R1                                       : PATNUM
3966 041132 012702 000012          MOV   #12,R2                                  : *,VALUE          3218
3967 041134 000441          BR    4$                                       :          3212
3968 041140 020027 177773          1$:  CMP   R0,#-5                                :
3969 041144 002410          BLT   2$                                       :
3970 041146 005700          TST   R0                                       :
3971 041150 002006          BGE   2$                                       :
3972 041152 012701 177777          MOV   #-1,R1                                  : *,PATNUM          3223
3973 041156 160001          SUB   R0,R1                                   : *,PATNUM
3974 041160 012702 000005          MOV   #5,R2                                   : *,VALUE          3224
3975 041164 000426          BR    4$                                       :          3212
3976 041166 020027 000006          2$:  CMP   R0,#6                                :
3977 041172 002411          BLT   3$                                       :
3978 041174 020027 000012          CMP   R0,#12                                 :
3979 041200 003006          BGT   3$                                       :
3980 041202 010001          MOV   R0,R1                                   : *,PATNUM          3229
3981 041204 162701 000006          SUB   #6,R1                                   : *,PATNUM
3982 041210 012702 000017          MOV   #17,R2                                  : *,VALUE          3230
3983 041214 000412          BR    4$                                       :          3212
3984 041216 020027 177766          3$:  CMP   R0,#-12                               :
3985 041222 002407          BLT   4$                                       :
3986 041224 020027 177772          CMP   R0,#-6                                :
3987 041230 003004          BGT   4$                                       :
3988 041232 012701 177772          MOV   #-6,R1                                  : *,PATNUM          3235
3989 041236 160001          SUB   R0,R1                                   : *,PATNUM
3990 041240 005002          CLR   R2                                       : VALUE          3236
3991 041242 006301          4$:  ASL   R1                                       :          3240
3992 041244 006301          ASL   R1                                       :
3993 041246 016167 034520 171424      MOV   PATTBL(R1),DATA.COUNT                    :
3994 041254 016167 034522 171420      MOV   PATTBL+2(R1),COMP.COUNT                  :
3995          ;MLX4
3996          ;
3997          ;
3998 041262 010200          MOV   R2,R0                                   : VALUE,*          3186
3999 041264 000207          RTS   PC                                       :          3185

```

```

27-Mar-1982 19:24:42
27-Mar-1982 19:23:44

```

```

3241
TOPS
PA:<

```

```

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

```

```

4000
4001
4002
4007
4008

```

```

4010 :MLX4
4011 :
4012 :
4013 : 3244 %sbttl 'NIBBLE GENERATOR'
4014 : 3245 routine FILLER (BUFFER, WRDCNT, VALUE) : novalue =
4015 : 3246 begin !* 1 *
4016 : 3247
4017 : 3248
4018 : 3249 !**
4019 : 3250 ROUTINE: FILLER(BUFFER,WRDCNT,VALUE)
4020 : 3251 PURPOSE: TO LOAD A CHOSEN WRITE BUFFER, ONE NIBBLE AT A TIME, WITH A
4021 : 3252 PARTICULAR PATTERN, FOR A SPECIFIED NUMBER OF 16-BIT WORDS.
4022 : 3253 THIS ROUTINE IS THE HEART OF THE GENERATOR FOR OPTIONS 2 AND 4.
4023 : 3254
4024 : 3255 THE 'FILLER' ROUTINE IS A LOOP WHICH WILL ALTERNATE BETWEEN
4025 : 3256 THE TWO COUNTS AND FILL THE APPROPRIATE NUMBER OF NIBBLES OF
4026 : 3257 THE CHOSEN BUFFER WITH THE CONTENTS OF 'VALUE'. EVERY TIME
4027 : 3258 A COUNT IS EXHAUSTED, THE OTHER COUNT IS STARTED UP AGAIN,
4028 : 3259 AND EACH TIME A NEW COUNT IS BEGUN, 'VALUE' IS COMPLEMENTED.
4029 : 3260 NOTE THAT THE FIRST COUNT THAT WILL BE USED WILL ALWAYS BE
4030 : 3261 'DATA_COUNT', SO THAT THE BUFFER WILL BE REPETITIONS OF:
4031 : 3262 DATA, COMP, DATA, COMP ...
4032 : 3263
4033 : 3264 ARGUMENTS: (1) BUFFER = THE STARTING ADDRESS OF THE CHOSEN WRITE BUFFER.
4034 : 3265 (2) WRDCNT = THE NUMBER OF WORDS OF PATTERN TO BE GENERATED.
4035 : 3266 (3) VALUE = NIBBLE OF DATA TO BE PLACED IN THE WRITE BUFFER
4036 : 3267
4037 : 3268
4038 : 3269 --
4039 : 3270
4040 : 3271 local
4041 : 3272 NIBBLE,
4042 : 3273 OFFSET,
4043 : 3274 FLAG,
4044 : 3275 THE_COUNT;
4045 : 3276
4046 : 3277 FLAG = 0;
4047 : 3278 OFFSET = 0;
4048 : 3279 NIBBLE = 0;
4049 : 3280
4050 : 3281 while 1 do
4051 : 3282 begin !* 2 *
4052 : 3283 FLAG = not .FLAG; !FLIP THE FLAG
4053 : 3284 VALUE = not .VALUE; !AND THE DATA
4054 : 3285
4055 : 3286 if .FLAG eql 0 !CHOOSE A COUNT
4056 : 3287 then
4057 : 3288 THE_COUNT = .COMP_COUNT
4058 : 3289 else
4059 : 3290 THE_COUNT = .DATA_COUNT;
4060 : 3291
4061 : 3292 decr COUNT from .THE_COUNT to 1 do !EXHAUST THE COUNT
4062 : 3293 begin !* 3 *
4063 : 3294 (.BUFFER + (.OFFSET))<.NIBBLE, 4, 0> = .VALUE; !LOAD THE NIBBLE
4064 : 3295 NIBBLE = (((.NIBBLE) + 4) mod 16); !CHANGE NIBBLE POINTER
    
```

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (17)

```

4066 :MLX4
4067 :
4068 :
4069 : 3296
4070 : 3297 if .NIBBLE eql 0
4071 : 3298 then
4072 : 3299 begin !* 4 *
4073 : 3300 BREAK;
4074 : 3301 OFFSET = .OFFSET + 2;
4075 : 3302 WRDCNT = .WRDCNT - 1;
4076 : 3303
4077 : 3304 if .WRDCNT eql 0 then return;
4078 : 3305
4079 : 3306 end; !* 4 *
4080 : 3307
4081 : 3308 end; !* 3 *
4082 : 3309
4083 : 3310 end; !* 2 *
4084 : 3311
4085 : 3312 end; !* 1 *
    
```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (17)

```

4090 :
4094 041266 004167 144042 FILLER: .SBTTL FILLER NIBBLE GENERATOR
4095 041272 005001 JSR R1,$SAVE5
4096 041274 005005 CLR R1
4097 041276 005002 CLR R5
4098 041300 005101 CLR R2
4099 041302 005166 000016 1$: COM R1
4100 041306 005701 COM 16(SP)
4101 041310 001003 TST R1
4102 041312 016704 171364 BNE 2$
4103 041316 000402 MOV COMP.COUNT,R4
4104 041320 016704 171354 BR 3$
4105 041324 010403 2$: MOV DATA.COUNT,R4
4106 041326 003764 3$: MOV R4,R3
4107 041330 010546 4$: BLE 1$
4108 041332 066616 000024 MOV R5,-(SP)
4109 041336 010246 ADD 24(SP),(SP)
4110 041340 012746 000004 MOV R2,-(SP)
4111 041344 016646 000024 MOV #4,-(SP)
4112 041350 004767 143240 MOV 24(SP),-(SP)
4113 041354 010216 JSR PC,BLSPU2
4114 041356 062716 000004 MOV R2,(SP)
4115 041362 012746 000020 ADD #4,(SP)
4116 041366 004767 143610 JSR PC,BLSMOD
4117 041372 010002 MOV R0,R2
4118 041374 001011 BNE 6$
4119 041376 104422 TRAP 22
4120 041400 062705 000002 ADD #2,R5
    
```

```

: 3245
: FLAG 3277
: OFFSET 3278
: NIBBLE 3279
: FLAG 3283
: VALUE 3284
: FLAG 3286
:
: *,THE.COUNT 3288
:
: *,THE.COUNT 3286
: THE.COUNT,COUNT 3290
:
: OFFSET,* 3294
: BUFFER,*
: NIBBLE,*
:
: VALUE,*
: NIBBLE,* 3295
:
: *,NIBBLE 3297
:
: *,OFFSET 3299
:
: *,OFFSET 3301
    
```

4122
4123
4124
4125 041404 005366 000032
4126 041410 001003
4127 041412 062706 000012
4128 041416 000207
4129 041420 062706 000012
4130 041424 005303
4131 041426 001340
4132 041430 000723
4133
4134
4135
4140
4141

;MLX4
;
NIBBLE GENERATOR
DEC 32(SP)
BNE 6\$
ADD #12,SP
5\$: RTS PC
6\$: ADD #12,SP
DEC R3
BNE 4\$
BR 1\$

: WRDCNT
:
:
:
:
:
:
COUNT
:

27-Mar-1982 19:24:42 TOPS
27-Mar-1982 19:23:44 PA:<
3302
3304
3245
3304
3293
3292
3281

: Routine Size: 50 words
: Maximum stack depth per invocation: 11 words

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (18)

```

4143 :MLX4
4144 :
4145 :
4146 : 3313 %sbttl 'GENERATOR FOR OPTION 2'
4147 : 3314 routine GEN2 : novalue =
4148 : 3315 begin
4149 : 3316
4150 : 3317 !++
4151 : 3318 ROUTINE: GEN2
4152 : 3319
4153 : 3320 PURPOSE: TO GENERATE THE ENTIRE WRITE BUFFER (WHICH
4154 : 3321 WILL BE USED IN CONJUNCTION WITH OPTION 2)
4155 : 3322 BASED ON PATTERN TABLE ENTRIES.
4156 : 3323 !--
4157 : 3324
4158 : 3325 Local
4159 : 3326 VALUE:
4160 : 3327
4161 : 3328 VALUE = GETCNTS (.PATTERN);
4162 : 3329 FILLER (WBUFF, BUFSIZ, .VALUE);
4163 : 3330 return;
4164 : 3331 end;
  
```

```

4169 : .SBTTL GEN2 GENERATOR FOR OPTION 2
4173 041432 016746 171240 GEN2: MOV PATTERN,-(SP) ; 3328
4174 041436 004767 177444 JSR PC,GETCNTS ;
4175 041442 012716 012670 MOV #WBUFF,(SP) ; 3329
4176 041446 012746 004000 MOV #4000,-(SP) ;
4177 041452 010046 MOV R0,-(SP) ; VALUE,*
4178 041454 004767 177606 JSR PC,FILLER ;
4179 041460 062706 000006 ADD #6,SP ; 3314
4180 041464 000207 RTS PC
  
```

```

4181 :
4182 : ; Routine Size: 14 words
4183 : ; Maximum stack depth per invocation: 3 words
4188 :
4189 :
  
```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (19)

```

4191 :MLX4
4192 :
4193 :
4194 : 3332 %sbttl 'GENERATOR FOR OPTION 3'
4195 : 3333 routine GEN3 (COMP_FLAG) : novalue =
4196 : 3334 begin
4197 : 3335
4198 : 3336 !++
4199 : 3337 ROUTINE: GEN3(COMP_FLAG)
4200 : 3338
4201 : 3339 PURPOSE: TO GENERATE THE ENTIRE WRITE BUFFER WHICH WILL BE MADE
4202 : 3340 UP OF WORDS WHICH CONTAIN THE UNIQUE COUNT FROM ONE TO
4203 : 3341 BUFSIZ.
4204 : 3342
4205 : 3343 ARGUMENT: COMP_FLAG - THIS IS AN INDICATOR OF WHETHER OR NOT TO
4206 : 3344 COMPLEMENT EVERY WORD IN THE WRITE BUFFER.
4207 : 3345 !--
4208 : 3346
4209 : 3347 local
4210 : 3348 OFFSET;
4211 : 3349
4212 : 3350 OFFSET = 0;
4213 : 3351
4214 : 3352 if .COMP_FLAG
4215 : 3353 then !GENERATE COMPLEMENT DATA
4216 : 3354
4217 : 3355 incru COUNT from 1 to BUFSIZ do
4218 : 3356 begin
4219 : 3357 (WBUF + (.OFFSET)) = not (.COUNT); !COMPLEMENT DATA
4220 : 3358 OFFSET = .OFFSET + 2;
4221 : 3359 end
4222 : 3360
4223 : 3361 else !GENERATE REGULAR DATA
4224 : 3362
4225 : 3363 incru COUNT from 1 to BUFSIZ do
4226 : 3364 begin
4227 : 3365 (WBUF + (.OFFSET)) = (.COUNT); !REGULAR DATA
4228 : 3366 OFFSET = .OFFSET + 2;
4229 : 3367 end;
4230 : 3368
4231 : 3369 return;
4232 : 3370 end;
4236 :
4237 :

```

```

4241 041466 010146 GEN3: .SBTTL GEN3 GENERATOR FOR OPTION 3
4242 041470 005001 MOV R1,-(SP)
4243 041472 032766 000001 000004 CLR R1
4244 041500 001415 BEQ 2$
4245 041502 012700 000001 MOV #1,R0
:
: OFFSET 3333
: *.COMP.FLAG 3350
: *.COUNT 3355

```

```
4247          ;MLX4
4248          ;
4249          ;
4250 041506 010061 012670      1$:  MOV  R0,WBUFF(R1)      ; COUNT,*(OFFSET)      3357
4251 041512 005161 012670      COM  WBUFF(R1)      ; *(OFFSET)
4252 041516 062701 000002      ADD  #2,R1      ; *,OFFSET      3358
4253 041522 005200      INC  R0      ; COUNT      3355
4254 041524 020027 004000      CMP  R0,#4000      ; COUNT,*
4255 041530 101766      BLOS 1$
4256 041532 000412      BR   4$
4257 041534 012700 000001      2$:  MOV  #1,R0      ; *,COUNT      3352
4258 041540 010061 012670      3$:  MOV  R0,WBUFF(R1)      ; COUNT,*(OFFSET)      3363
4259 041544 062701 000002      ADD  #2,R1      ; *,OFFSET      3366
4260 041550 005200      INC  R0      ; COUNT      3363
4261 041552 020027 004000      CMP  R0,#4000      ; COUNT,*
4262 041556 101770      BLOS 3$
4263 041560 012601      4$:  MOV  (SP)+,R1      ;
4264 041562 000207      RTS  PC      ;
4265
4266          ; Routine Size: 31 words
4267          ; Maximum stack depth per invocation: 2 words
4272
4273
```

27-Mar-1982 19:24:42 TOPS
27-Mar-1982 19:23:44 PA:<

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (20)

```

4275 :MLX4
4276 :
4277 :
4278 : 3371 %sbttl 'GENERATOR FOR OPTION 4'
4279 : 3372 routine GEN4 (MARPAT, BUFFER) : novalue =
4280 : 3373 begin
4281 : 3374
4282 : 3375
4283 : 3376
4284 : 3377
4285 : 3378
4286 : 3379
4287 : 3380
4288 : 3381
4289 : 3382
4290 : 3383
4291 : 3384
4292 : 3385
4293 : 3386
4294 : 3387
4295 : 3388
4296 : 3389
4297 : 3390
4298 : 3391
4299 : 3392
4300 : 3393
4301 : 3394
    
```

```

++
ROUTINE: GEN4(MARPAT,BUFFER)
PURPOSE: THIS IS THE GENERATOR WHICH IS USED FOR OPTION 4, THE
MARCH TEST. IT BEGINS AT THE START OF THE CHOSEN
BUFFER AND GENERATES 256 WORDS IN THE EXACT WAY THAT
OPTION 2 DOES.
ARGUMENTS: (1) MARPAT - THE MARCH PATTERN NUMBER
(2) BUFFER - THE STARTING ADDRESS OF THE CHOSEN BUFFER
--
Local
VALUE;
VALUE = GETCNTS (.MARPAT);
FILLER (.BUFFER, 256, .VALUE);
return;
end;
    
```

```

4306 :
4310 041564 016646 000004 GEN4: .SBTTL GEN4 GENERATOR FOR OPTION 4
4311 041570 004767 177312 MOV 4(SP),-(SP) ; MARPAT,* 3391
4312 041574 016616 000004 JSR PC,GETCNTS
4313 041600 012746 000400 MOV 4(SP),(SP) ; BUFFER,* 3392
4314 041604 010046 MOV #400,-(SP) ; VALUE,*
4315 041606 004767 177454 JSR PC,FILLER
4316 041612 062706 000006 ADD #6,SP ; 3372
4317 041616 000207 RTS PC
    
```

```

4318 :
4319 : ; Routine Size: 14 words
4320 : ; Maximum stack depth per invocation: 3 words
4325 :
4326 :
    
```


27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (21)

```
4328 :MLX4
4329 :
4330 :
4331 : 3395 %sbttl 'GENERATOR FOR OPTION 5'
4332 : 3396 routine GEN5 : novalue =
4333 : 3397 begin
4334 : 3398
4335 : 3399 !++
4336 : 3400 ROUTINE: GEN5
4337 : 3401
4338 : 3402 PURPOSE: THIS IS THE GENERATOR WHICH IS USED FOR OPTION 5.
4339 : 3403 IT BEGINS AT THE START OF THE WRITE BUFFER, AND GENERATES
4340 : 3404 A FULL BUFFER OF RANDOM DATA. THE FIRST 3 WORDS OF THE BUFFER
4341 : 3405 RECEIVE THE 3 SEEDS WHICH WERE USED IN THE GENERATION.
4342 : 3406 !--
4343 : 3407
4344 : 3408 local
4345 : 3409 OFFSET;
4346 : 3410
4347 : 3411 (WBUF + 0) = .SEED1;
4348 : 3412 (WBUF + 2) = .SEED2;
4349 : 3413 (WBUF + 4) = .SEED3;
4350 : 3414 OFFSET = 6;
4351 : 3415
4352 : 3416 incru COUNT from 4 to BUFSIZ do
4353 : 3417 begin
4354 : 3418 RN ();
4355 : 3419 BREAK;
4356 : 3420 (WBUF + .OFFSET) = .RANDOM;
4357 : 3421 OFFSET = .OFFSET + 2;
4358 : 3422 end;
4359 : 3423
4360 : 3424 return;
4361 : 3425 end;
4365 :
```

!THE FIRST 3 WORDS OF THE
!WRITE BUFFER RECEIVE THE
!3 GENERATING SEEDS.

!NOTE: USING SIGNED VALUES (FULL 16 BITS)

```

4367      .SBTTL  GEN5 GENERATOR FOR OPTION 5
4371 041620 004167 143436      GEN5:  JSR    R1,$SAVE2      ;
4372 041624 016767 143626 151036      MOV    SEED1,WBUFF      ;
4373 041632 016767 143622 151032      MOV    SEED2,WBUFF+2    ;
4374 041640 016767 143616 151026      MOV    SEED3,WBUFF+4    ;
4375 041646 012701 000006      MOV    #6,R1           ; *,OFFSET
4376 041652 012702 000004      MOV    #4,R2           ; *,COUNT
4377 041656 004767 143506      1$:   JSR    PC,RN      ;
4378 041662 104422                TRAP   22              ;
4379 041664 016761 143574 012670      MOV    RANDOM,WBUFF(R1) ; *,*(OFFSET)
4380 041672 062701 000002      ADD    #2,R1           ; *,OFFSET
4381 041676 005202                INC    R2              ; COUNT
4382 041700 020227 004000      CMP    R2,#4000        ; COUNT,*
4383 041704 101764                BLOS   1$              ;
4384      :MLX4
4385      :
4386
4387 041706 000207                RTS    PC              ;
4388
4389      ; Routine Size: 28 words
4390      ; Maximum stack depth per invocation: 3 words
4395
4396
    
```

27-Mar-1982 19:24:42 TOPS
 27-Mar-1982 19:23:44 PA:<

3396

4398 :MLX4
4399 :
4400 :
4401 :
4402 :
4403 :
4404 :
4405 :
4406 :
4407 :
4408 :
4409 :
4410 :
4411 :
4412 :
4413 :
4414 :
4415 :
4416 :
4417 :
4418 :
4419 :
4420 :
4421 :
4422 :
4423 :
4424 :
4425 :
4426 :
4427 :
4428 :
4429 :
4430 :
4431 :
4432 :
4433 :
4434 :
4435 :
4436 :
4437 :
4438 :
4439 :
4440 :
4441 :
4442 :
4443 :
4444 :
4445 :
4446 :
4447 :
4448 :
4449 :
4450 :
4451 :
4452 :

SYSTEM ERROR DETECTOR

27-Mar-1982 19:24:42
27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
PA:<NEALE>MLX4.BLI.5 (22)

%sbttl 'SYSTEM ERROR DETECTOR'
routine SYSERR (LUN) =
begin

!* 1 *

++

ROUTINE: SYSERR(LUN)

PURPOSE: (1) TO SCAN FOR SYSTEM ERRORS AFTER A TRANSFER HAS ENDED.

(2) PRINT ERROR MESSAGES IF APPROPRIATE:

A) ERROR MESSAGES DURING RETRIES WILL NOT
BE PRINTED.

B) THE OPERATOR HAS THE ABILITY TO INHIBIT THE
PRINTING OF DATA ERRORS, BUT NOT SYSTEM ERRORS.

(3) DECIDE ON A VALUE TO RETURN WHICH WILL INDICATE THE
RELATIVE SEVERITY OF THE ERROR.

ARGUMENT: LUN = THE CURRENT LOGICAL UNIT NUMBER
[0 TO NUMBER OF DRIVES MINUS 1].

RESULT: VALUE RETURNED FOR THE SUBROUTINE SHOWS TYPE OF ERROR:
0 = NO ERRORS AT ALL
1 = RETRY ALLOWED FOR THESE TYPES OF ERRORS
2 = FATAL CONTROLLER ERROR
3 = FATAL DRIVE ERROR
4 = UNCORRECTABLE DATA ERROR
5 = CORRECTABLE DATA ERROR

FIGURE OUT WHICH VALUE TO RETURN BY THE FOLLOWING SCHEME:

- (1) IF THERE IS A CHOICE BETWEEN DATA ERROR AND SYSTEM ERROR,
FIRST CHOOSE THE DATA ERROR.
- (2) IF THERE IS A CHOICE BETWEEN FATAL AND NON-FATAL ERROR,
FIRST CHOOSE THE FATAL ERROR.
- (3) IF THERE IS A CHOICE BETWEEN CONTROLLER ERROR AND DRIVE ERROR,
FIRST CHOOSE CONTROLLER ERROR.

Label
PURPOSE_2_CODE;

Local
ERR2,
ERR3,
ERR4,
ERR5,
ERR6,
ERR7,
ERR8,

! VER CZMLBB ADDED THIS LOCAL

```

4454 :MLX4
4455 :
4456 :
4457 : 3478
4458 : 3479
4459 : 3480
4460 : 3481
4461 : 3482
4462 : 3483
4463 : 3484
4464 : 3485
4465 : 3486
4466 : 3487
4467 : 3488
4468 : 3489
4469 : 3490
4470 : 3491
4471 : 3492
4472 : 3493
4473 : 3494
4474 : 3495
4475 : 3496
4476 : 3497
4477 : 3498
4478 : 3499
4479 : 3500
4480 : 3501
4481 : 3502
4482 : 3503
4483 : 3504
4484 : 3505
4485 : 3506
4486 : 3507
4487 : 3508
4488 : 3509
4489 : 3510
4490 : 3511
4491 : 3512
4492 : 3513
4493 : 3514
4494 : 3515
4495 : 3516
4496 : 3517
4497 : 3518
4498 : 3519
4499 : 3520
4500 : 3521
4501 : 3522
4502 : 3523
4503 : 3524
4504 : 3525
4505 : 3526
4506 : 3527
4507 : 3528
4508 : 3529

SYSTEM ERROR DETECTOR

ERR9;

! VER CZMLBB ADDED THIS LOCAL

ERR2 = INACTIVE;
ERR3 = INACTIVE;
ERR4 = INACTIVE;
ERR5 = INACTIVE;
ERR6 = INACTIVE;
ERR7 = INACTIVE;
ERR8 = INACTIVE;
ERR9 = INACTIVE;

! VER CZMLBB
! VER CZMLBB

!+
! THIS IS THE CODE FOR PURPOSE 1:
!-

if .SC
then
begin
! THERE ARE SOME ERRORS -- FIND OUT WHAT KIND
!* 2 *
if ((.NED) or (.NEM) or (.PGE)) then ERR2 = ACTIVE;
if ((.DLT) or (.PE) or (.MXF) or (.MDPE) or (.MCPE)) then ERR3 = ACTIVE;
! VER ADDED THIS ERR9
if (.MDPE) and ( not .SGL) then ERR9 = ACTIVE;
! VER ADDED THIS ERR8
if ((.DLT) or (.PE) or (.MXF) or (.MCPE)) then ERR8 = ACTIVE;
if ((.WCE) and (.ECCDIS eql 0)) then ERR3 = ACTIVE;
if ((.UNS) or (.IAE) or (.AOE) or (.RMR) or (.ILR) or (.ILF)) then ERR4 = ACTIVE;
if ((.OPI) or (.DPAR) or (.CPAR)) then ERR5 = ACTIVE;
if ((.DCK) or (.crc) or (.SGL)) then ERR7 = ACTIVE;
if ((.WCE) and (.ECCDIS eql 1)) then ERR7 = ACTIVE;
if ((.ECH) or (.UNC)) then ERR6 = ACTIVE;
end
else
return 0;
!* 2 *
! THERE ARE NO ERRORS -- RETURN SUCCESSFULLY
! NO ERRORS AT ALL

!+
! THIS IS THE CODE FOR PURPOSE 2:
!-
LABEL:
BEGIN 3
    
```

4510 :MLX4
 4511 :
 4512 :
 4513 :
 4514 :
 4515 :
 4516 :
 4517 :
 4518 :
 4519 :
 4520 :
 4521 :
 4522 :
 4523 :
 4524 :
 4525 :
 4526 :
 4527 :
 4528 :
 4529 :
 4530 :
 4531 :
 4532 :
 4533 :
 4534 :
 4535 :
 4536 :
 4537 :
 4538 :
 4539 :
 4540 :
 4541 :
 4542 :
 4543 :
 4544 :
 4545 :
 4546 :
 4547 :
 4548 :
 4549 :
 4550 :
 4551 :
 4552 :
 4553 :
 4554 :
 4555 :
 4556 :
 4557 :
 4558 :
 4559 :
 4560 :
 4561 :
 4562 :
 4563 :
 4564 :

SYSTEM ERROR DETECTOR

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (22)

```

3530 |         IF RETRYING
3531 |         THEN LEAVE LABEL (NO PRINTOUTS AT ALL)
3532 |         ELSE
3533 |             BEGIN 4
3534 |                 IF ERROR PRINTOUTS ARE ALLOWED
3535 |                 THEN
3536 |                     BEGIN 5A
3537 |                         IDENTIFY THE LOGICAL UNIT
3538 |                         SCAN FOR & PRINT DATA ERRORS
3539 |                     END 5A
3540 |                 ELSE
3541 |                     BEGIN 5B
3542 |                         IF THERE ARE SYSTEM ERRORS TO REPORT
3543 |                         THEN IDENTIFY THE LOGICAL UNIT
3544 |                         ELSE LEAVE LABEL
3545 |                     END 5B
3546 |                 SCAN FOR & PRINT SYSTEM ERRORS
3547 |             END 4
3548 |         END 3
3549 |
3550 |
3551 | PURPOSE_2_CODE :
3552 |     begin                                     !* 3 *
3553 |
3554 |     if .RETRYING
3555 |     then
3556 |         leave PURPOSE_2_CODE                !(NO PRINTOUTS AT ALL)
3557 |     else
3558 |         begin                                 !* 4 *
3559 |
3560 |         if .ERROUT
3561 |         then
3562 |             begin                             !* 5A *
3563 |                 SAYWHO (.LUN);
3564 |                 PRINTB (SAY1, PHR10);
3565 |                 !'ERROR BITS SET:
3566 |
3567 |             if .ERR7
3568 |             then
3569 |                 begin
3570 |
3571 |                 if .DCK then PRINTB (FMT15, MLB20);
3572 |                 if .crc then PRINTB (FMT15, MLB22);
3573 |                 if .SGL then PRINTB (FMT15, MLB23);
3574 |
3575 |                 if ((.WCE) and (.ECCDIS eql 1)) then PRINTB (FMT15, MLB6);
3576 |
3577 |                 if ((.WCE) and (.ECCDIS eql 1)) then PRINTB (FMT15, MLB6);
3578 |
3579 |                 end;
3580 |
3581 |             if .ERR6
    
```

```

4566 :MLX4
4567 :
4568 :
4569 : 3582
4570 : 3583
4571 : 3584
4572 : 3585
4573 : 3586
4574 : 3587
4575 : 3588
4576 : 3589
4577 : 3590
4578 : 3591
4579 : 3592
4580 : 3593
4581 : 3594
4582 : 3595
4583 : 3596
4584 : 3597
4585 : 3598
4586 : 3599
4587 : 3600
4588 : 3601
4589 : 3602
4590 : 3603
4591 : 3604
4592 : 3605
4593 : 3606
4594 : 3607
4595 : 3608
4596 : 3609
4597 : 3610
4598 : 3611
4599 : 3612
4600 : 3613
4601 : 3614
4602 : 3615
4603 : 3616
4604 : 3617
4605 : 3618
4606 : 3619
4607 : 3620
4608 : 3621
4609 : 3622
4610 : 3623
4611 : 3624
4612 : 3625
4613 : 3626
4614 : 3627
4615 : 3628
4616 : 3629
4617 : 3630
4618 : 3631
4619 : 3632
4620 : 3633

SYSTEM ERROR DETECTOR

    then
    begin
        if .ECH then PRINTB (FMT15, MLB21);
        if .UNC then PRINTB (FMT15, MLB24);
    end;
    end;
else
    begin
        ! VER CZMLBB CHANGED ERR3 TO ERR8
        if ((.ERR2) or (.ERR8) or (.ERR4) or (.ERR5))
        then
            begin
                SAYWHO (.LUN);
                PRINTB (SAY1, PHR10);
                !'ERROR BITS SET:
            end
        else
            ! VER CZMLBB ADDED TEST OF ERR9 HERE
            if .ERR9
            then
                begin
                    SAYWHO (.LUN);
                    PRINTB (SAY1, PHR10);
                    !'ERROR BITS SET:
                end
            else
                begin
                    leave PURPOSE_2_CODE;
                end;
            end;
        end;
    end;
    if .ERR2
    then
        begin
            if .NED then PRINTB (FMT15, MLB2);
            if .NEM then PRINTB (FMT15, MLB3);
            if .PGE then PRINTB (FMT15, MLB4);
        end;
    if .ERR3
    then

```

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (22)

!* 5A *

!* 5B *

!* 5B *

27-Mar-1982 19:24:42
27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
PA:<NEALE>MLX4.BLI.5 (22)

```

4622 :MLX4
4623 :
4624 :
4625 : 3634
4626 : 3635
4627 : 3636
4628 : 3637
4629 : 3638
4630 : 3639
4631 : 3640
4632 : 3641
4633 : 3642
4634 : 3643
4635 : 3644 ! VER CZMLBB ADDED 'AND NOT .SGL'
4636 : 3645
4637 : 3646
4638 : 3647
4639 : 3648
4640 : 3649
4641 : 3650
4642 : 3651
4643 : 3652
4644 : 3653
4645 : 3654
4646 : 3655
4647 : 3656
4648 : 3657
4649 : 3658
4650 : 3659
4651 : 3660
4652 : 3661
4653 : 3662
4654 : 3663
4655 : 3664
4656 : 3665
4657 : 3666
4658 : 3667
4659 : 3668
4660 : 3669
4661 : 3670
4662 : 3671
4663 : 3672
4664 : 3673
4665 : 3674
4666 : 3675
4667 : 3676
4668 : 3677
4669 : 3678
4670 : 3679
4671 : 3680
4672 : 3681
4673 : 3682
4674 : 3683
4675 : 3684
4676 : 3685

```

SYSTEM ERROR DETECTOR

```

begin
if .DLT then PRINTB (FMT15, MLB5);
if ((.WCE) and (.ECCDIS eql 0)) then PRINTB (FMT15, MLB6);
if .PE then PRINTB (FMT15, MLB7);
if .MXF then PRINTB (FMT15, MLB8);
! VER CZMLBB ADDED 'AND NOT .SGL'
if ((.MDPE) and ( not .SGL)) then PRINTB (FMT15, MLB9);
if .MCPE then PRINTB (FMT15, MLB10);
end;
if .ERR4
then
begin
if .UNS then PRINTB (FMT15, MLB11);
if .IAE then PRINTB (FMT15, MLB12);
if .AOE then PRINTB (FMT15, MLB13);
if .RMR then PRINTB (FMT15, MLB14);
if .JLR then PRINTB (FMT15, MLB15);
if .JLF then PRINTB (FMT15, MLB16);
end;
if .ERR5
then
begin
if .OPI then PRINTB (FMT15, MLB17);
if .DPAR then PRINTB (FMT15, MLB18);
if .CPAR then PRINTB (FMT15, MLB19);
end;
end;
end;

```

!* 4 *

!* 3 *

27-Mar-1982 19:24:42 TOPS-20 BLISS-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (22)

```

4678 :MLX4
4679 :
4680 :
4681 : 3686 !+
4682 : 3687 ! THIS IS THE CODE FOR PURPOSE 3:
4683 : 3688 !-
4684 : 3689
4685 : 3690 if .ERR6 !UNRECOVERABLE DATA ERROR (NO RETRY ALLOWED)
4686 : 3691 then
4687 : 3692 return 4;
4688 : 3693
4689 : 3694 if .ERR7 !RECOVERABLE DATA ERROR (RETRY TO CLASSIFY)
4690 : 3695 then
4691 : 3696 return 5;
4692 : 3697
4693 : 3698 if .ERR2 !CONTROLLER FATAL ERROR
4694 : 3699 then
4695 : 3700 return 2;
4696 : 3701
4697 : 3702 if .ERR4 !DRIVE FATAL ERROR
4698 : 3703 then
4699 : 3704 return 3;
4700 : 3705
4701 : 3706 if (.ERR3 or .ERR5) !NON-FATAL ERRORS
4702 : 3707 then
4703 : 3708 return 1
4704 : 3709 else
4705 : 3710 begin
4706 : 3711
4707 : 3712 if ((.ERROUT) and ( not .RETRYING)) then PRINTB (SAY1, PHR11);
4708 : 3713
4709 : 3714 !'SC SET BUT NO SYSTEM ERRORS FOUND'
4710 : 3715 return 1;
4711 : 3716 end;
4712 : 3717
4713 : 3718 end;
    
```

!* 1 *

```

4718 :
4722 041710 004167 143420 SYSERR: .SBTTL SYSERR SYSTEM ERROR DETECTOR : 3427
4723 041714 005746 JSR R1,$SAVE5 :
4724 041716 005046 TST -(SP) : ERR2 3480
4725 041720 005046 CLR -(SP) : ERR3 3481
4726 041722 005001 CLR R1 : ERR4 3482
4727 041724 005002 CLR R2 : ERR5 3483
4728 041726 005066 000004 CLR 4(SP) : ERR6 3484
4729 041732 005003 CLR R3 : ERR7 3485
4730 041734 005004 CLR R4 : ERR8 3486
4731 041736 005005 CLR R5 : ERR9 3487
4732 041740 005777 172402 TST @ML.REG : 3493
4733 :
4734 : :MLX4
4735 : : SYSTEM ERROR DETECTOR
4736 041744 100402 BMI 1$
4737 041746 000167 002344 JMP 53$
4738 041752 032777 010000 172376 1$: BIT #10000,@ML.REG+10 : 3497
4739 041760 001010 BNE 2$
4740 041762 032777 004000 172366 BIT #4000,@ML.REG+10
    
```

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS
 PA:<

4741	041770	001004				BNE	2\$			
4742	041772	032777	002000	172356		BIT	#2000,AML.REG+10			
4743	042000	001403				BEQ	3\$			
4744	042002	012766	000001	000002	2\$:	MOV	#1,2(SP)	:	*.ERR2	
4745	042010	005777	172342		3\$:	TST	AML.REG+10	:		3499
4746	042014	100420				BMI	4\$			
4747	042016	032777	020000	172332		BIT	#20000,AML.REG+10			
4748	042024	001014				BNE	4\$			
4749	042026	032777	001000	172322		BIT	#1000,AML.REG+10			
4750	042034	001010				BNE	4\$			
4751	042036	032777	000400	172312		BIT	#400,AML.REG+10			
4752	042044	001004				BNE	4\$			
4753	042046	032777	020000	172272		BIT	#20000,AML.REG			
4754	042054	001402				BEQ	5\$			
4755	042056	012716	000001		4\$:	MOV	#1,(SP)	:	*.ERR3	
4756	042062	032777	000400	172266	5\$:	BIT	#400,AML.REG+10	:		3503
4757	042070	001406				BEQ	6\$			
4758	042072	032777	040000	172310		BIT	#40000,AML.REG+42			
4759	042100	001002				BNE	6\$			
4760	042102	012705	000001			MOV	#1,R5	:	*.ERR9	
4761	042106	005777	172244		6\$:	TST	AML.REG+10	:		3507
4762	042112	100414				BMI	7\$			
4763	042114	032777	020000	172234		BIT	#20000,AML.REG+10			
4764	042122	001010				BNE	7\$			
4765	042124	032777	001000	172224		BIT	#1000,AML.REG+10			
4766	042132	001004				BNE	7\$			
4767	042134	032777	020000	172204		BIT	#20000,AML.REG			
4768	042142	001402				BEQ	8\$			
4769	042144	012704	000001		7\$:	MOV	#1,R4	:	*.ERR8	
4770	042150	032777	040000	172200	8\$:	BIT	#40000,AML.REG+10	:		3509
4771	042156	001405				BEQ	9\$			
4772	042160	005767	140070			TST	ECCDIS			
4773	042164	001002				BNE	9\$			
4774	042166	012716	000001			MOV	#1,(SP)	:	*.ERR3	
4775	042172	032777	040000	172162	9\$:	BIT	#40000,AML.REG+14	:		3511
4776	042200	001024				BNE	10\$			
4777	042202	032777	002000	172152		BIT	#2000,AML.REG+14			
4778	042210	001020				BNE	10\$			
4779	042212	032777	001000	172142		BIT	#1000,AML.REG+14			
4780	042220	001014				BNE	10\$			
4781	042222	132777	000004	172132		BITB	#4,AML.REG+14			
4782	042230	001010				BNE	10\$			
4783	042232	132777	000002	172122		BITB	#2,AML.REG+14			
4784	042240	001004				BNE	10\$			
4785	042242	132777	000001	172112		BITB	#1,AML.REG+14			
4786	042250	001402				BEQ	11\$			
4787	042252	012701	000001		10\$:	MOV	#1,R1	:	*.ERR4	
4788					:MLX4					
4789					:					
4790							SYSTEM ERROR DETECTOR			
4791	042256	032777	020000	172076	11\$:	BIT	#20000,AML.REG+14	:		3513
4792	042264	001010				BNE	12\$			
4793	042266	132777	000040	172066		BITB	#40,AML.REG+14			
4794	042274	001004				BNE	12\$			
4795	042276	132777	000010	172056		BITB	#10,AML.REG+14			
4796	042304	001402				BEQ	13\$			
4797	042306	012702	000001		12\$:	MOV	#1,R2	:	*.ERR5	

27-Mar-1982 19:24:42 TOPS
27-Mar-1982 19:23:44 PA:<

4798	042312	005777	172044		13\$:	TST	@ML.REG+14	:		3515
4799	042316	100410				BMI	14\$			
4800	042320	032777	020000	172062		BIT	#20000,@ML.REG+42			
4801	042326	001004				BNE	14\$			
4802	042330	032777	040000	172052		BIT	#40000,@ML.REG+42			
4803	042336	001402				BEQ	15\$			
4804	042340	012703	000001		14\$:	MOV	#1,R3	:	*,ERR7	
4805	042344	032777	040000	172004	15\$:	BIT	#40000,@ML.REG+10	:		3517
4806	042352	001406				BEQ	16\$			
4807	042354	026727	137674	000001		JMP	ECCDIS,#1			
4808	042362	001002				BNE	16\$			
4809	042364	012703	000001			MOV	#1,R3	:	*,ERR7	
4810	042370	132777	000100	171764	16\$:	BITB	#100,@ML.REG+14	:		3519
4811	042376	001003				BNE	17\$			
4812	042400	005777	172004			TST	@ML.REG+42			
4813	042404	100003				BPL	18\$			
4814	042406	012766	000001	000004	17\$:	MOV	#1,4(SP)	:	*,ERR6	
4815	042414	032767	000001	171716	18\$:	BIT	#1,RETRYING	:		3554
4816	042422	001402				BEQ	19\$:		3556
4817	042424	000167	001520			JMP	47\$:		
4818	042430	032767	000001	137622	19\$:	BIT	#1,ERROUT	:		3560
4819	042436	001552				BEQ	25\$			
4820	042440	016646	000024			MOV	24(SP),-(SP)	:	LUN,*	3563
4821	042444	004767	173054			JSR	PC,SAYWHO			
4822	042450	012716	010120			MOV	#PHR10,(SP)	:		3564
4823	042454	012746	007072			MOV	#SAY1,-(SP)			
4824	042460	012746	000002			MOV	#2,-(SP)			
4825	042464	010600				MOV	SP,R0	:	SP,*	
4826	042466	104414				TRAP	14			
4827	042470	032703	000001			BIT	#1,R3	:	*,ERR7	3567
4828	042474	001473				BEQ	23\$			
4829	042476	005777	171660			TST	@ML.REG+14	:		3571
4830	042502	100012				BPL	20\$			
4831	042504	012746	007550			MOV	#MLB20,-(SP)			
4832	042510	012746	007060			MOV	#FMT15,-(SP)			
4833	042514	012746	000002			MOV	#2,-(SP)			
4834	042520	010600				MOV	SP,R0	:	SP,*	
4835	042522	104414				TRAP	14			
4836	042524	062706	000006			ADD	#6,SP			
4837	042530	032777	020000	171652	20\$:	BIT	#20000,@ML.REG+42	:		3573
4838	042536	001412				BEQ	21\$			
4839	042540	012746	007560			MOV	#MLB22,-(SP)			
4840	042544	012746	007060			MOV	#FMT15,-(SP)			
4841	042550	012746	000002			MOV	#2,-(SP)			
4842	042554	010600				MOV	SP,R0	:	SP,*	
4843										
4844					:MLX4					
4845					:					
4846	042556	104414				TRAP	14			
4847	042560	062706	000006			ADD	#6,SP			
4848	042564	032777	040000	171616	21\$:	BIT	#40000,@ML.REG+42	:		3575
4849	042572	001412				BEQ	22\$			
4850	042574	012746	007564			MOV	#MLB23,-(SP)			
4851	042600	012746	007060			MOV	#FMT15,-(SP)			
4852	042604	012746	000002			MOV	#2,-(SP)			
4853	042610	010600				MOV	SP,R0	:	SP,*	
4854	042612	104414				TRAP	14			

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS
 PA:<

4855	042614	062706	000006			ADD	#6,SP			
4856	042620	032777	040000	171530	22\$:	BIT	#40000,@ML.REG+10	:		3577
4857	042626	001416				BEQ	23\$			
4858	042630	026727	137420	000001		CMP	ECCDIS,#1			
4859	042636	001012				BNE	23\$			
4860	042640	012746	007450			MOV	#MLB6,-(SP)			
4861	042644	012746	007060			MOV	#FMT15,-(SP)			
4862	042650	012746	000002			MOV	#2,-(SP)			
4863	042654	010600				MOV	SP,R0	:	SP,*	
4864	042656	104414				TRAP	14			
4865	042660	062706	000006			ADD	#6,SP			
4866	042664	032766	000001	000012	23\$:	BIT	#1,12(SP)	:	*,ERR6	3581
4867	042672	001505				BEQ	29\$			
4868	042674	132777	000100	171460		BITB	#100,@ML.REG+14	:		3585
4869	042702	001412				BEQ	24\$			
4870	042704	012746	007554			MOV	#MLB21,-(SP)			
4871	042710	012746	007060			MOV	#FMT15,-(SP)			
4872	042714	012746	000002			MOV	#2,-(SP)			
4873	042720	010600				MOV	SP,R0	:	SP,*	
4874	042722	104414				TRAP	14			
4875	042724	062706	000006			ADD	#6,SP			
4876	042730	005777	171454		24\$:	TST	@ML.REG+42	:		3587
4877	042734	100064				BPL	29\$			
4878	042736	012746	007570			MOV	#MLB24,-(SP)			
4879	042742	012746	007060			MOV	#FMT15,-(SP)			
4880	042746	012746	000002			MOV	#2,-(SP)			
4881	042752	010600				MOV	SP,R0	:	SP,*	
4882	042754	104414				TRAP	14			
4883	042756	062706	000006			ADD	#6,SP			
4884	042762	000451				BR	29\$:		3562
4885	042764	032766	000001	000002	25\$:	BIT	#1,2(SP)	:	*,ERR2	3596
4886	042772	001010				BNE	26\$			
4887	042774	006004				ROR	R4	:	ERR8	
4888	042776	103406				BLO	26\$			
4889	043000	032701	000001			BIT	#1,R1	:	*,ERR4	
4890	043004	001003				BNE	26\$			
4891	043006	032702	000001			BIT	#1,R2	:	*,ERR5	
4892	043012	001415				BEQ	27\$			
4893	043014	016646	000024		26\$:	MOV	24(SP),-(SP)	:	LUN,*	3599
4894	043020	004767	172500			JSR	PC,SAYWHO			
4895	043024	012716	010120			MOV	#PHR10,(SP)	:		3600
4896	043030	012746	007072			MOV	#SAY1,-(SP)			
4897	043034	012746	000002			MOV	#2,-(SP)			
4898					:MLX4					
4899					:					
4900							SYSTEM ERROR DETECTOR			
4901	043040	010600				MOV	SP,R0	:	SP,*	
4902	043042	104414				TRAP	14			
4903	043044	000420				BR	29\$:		3598
4904	043046	006005			27\$:	ROR	R5	:	ERR9	3606
4905	043050	103402				BLO	28\$			
4906	043052	000167	001072			JMP	47\$			
4907	043056	016646	000024		28\$:	MOV	24(SP),-(SP)	:	LUN,*	3609
4908	043062	004767	172436			JSR	PC,SAYWHO			
4909	043066	012716	010120			MOV	#PHR10,(SP)	:		3610
4910	043072	012746	007072			MOV	#SAY1,-(SP)			
4911	043076	012746	000002			MOV	#2,-(SP)			

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS
 PA:<

4912	043102	010600			MOV	SP,R0		: SP,*	
4913	043104	104414			TRAP	14			
4914	043106	062706	000006	29\$:	ADD	#6,SP			3608
4915	043112	032766	000001	000002	BIT	#1,2(SP)		: *,ERR2	3620
4916	043120	001452			BEQ	32\$			
4917	043122	032777	010000	171226	BIT	#10000,@ML.REG+10		:	3624
4918	043130	001412			BEQ	30\$			
4919	043132	012746	007430		MOV	#MLB2,-(SP)			
4920	043136	012746	007060		MOV	#FMT15,-(SP)			
4921	043142	012746	000002		MOV	#2,-(SP)			
4922	043146	010600			MOV	SP,R0		: SP,*	
4923	043150	104414			TRAP	14			
4924	043152	062706	000006		ADD	#6,SP			
4925	043156	032777	004000	171172	BIT	#4000,@ML.REG+10		:	3626
4926	043164	001412			BEQ	31\$			
4927	043166	012746	007434		MOV	#MLB3,-(SP)			
4928	043172	012746	007060		MOV	#FMT15,-(SP)			
4929	043176	012746	000002		MOV	#2,-(SP)			
4930	043202	010600			MOV	SP,R0		: SP,*	
4931	043204	104414			TRAP	14			
4932	043206	062706	000006		ADD	#6,SP			
4933	043212	032777	002000	171136	BIT	#2000,@ML.REG+10		:	3628
4934	043220	001412			BEQ	32\$			
4935	043222	012746	007440		MOV	#MLB4,-(SP)			
4936	043226	012746	007060		MOV	#FMT15,-(SP)			
4937	043232	012746	000002		MOV	#2,-(SP)			
4938	043236	010600			MOV	SP,R0		: SP,*	
4939	043240	104414			TRAP	14			
4940	043242	062706	000006		ADD	#6,SP			
4941	043246	032716	000001	32\$:	BIT	#1,(SP)		: *,ERR3	3632
4942	043252	001532			BEQ	38\$			
4943	043254	005777	171076		TST	@ML.REG+10		:	3636
4944	043260	100012			BPL	33\$			
4945	043262	012746	007444		MOV	#MLB5,-(SP)			
4946	043266	012746	007060		MOV	#FMT15,-(SP)			
4947	043272	012746	000002		MOV	#2,-(SP)			
4948	043276	010600			MOV	SP,R0		: SP,*	
4949	043300	104414			TRAP	14			
4950	043302	062706	000006		ADD	#6,SP			
4951	043306	032777	040000	171042	BIT	#40000,@ML.REG+10		:	3638
4952	043314	001415			BEQ	34\$			
4953									
4954									
4955									
4956	043316	005767	136732		TST	ECCDIS			
4957	043322	001012			BNE	34\$			
4958	043324	012746	007450		MOV	#MLB6,-(SP)			
4959	043330	012746	007060		MOV	#FMT15,-(SP)			
4960	043334	012746	000002		MOV	#2,-(SP)			
4961	043340	010600			MOV	SP,R0		: SP,*	
4962	043342	104414			TRAP	14			
4963	043344	062706	000006		ADD	#6,SP			
4964	043350	032777	020000	171000	BIT	#20000,@ML.REG+10		:	3640
4965	043356	001412			BEQ	35\$			
4966	043360	012746	007454		MOV	#MLB7,-(SP)			
4967	043364	012746	007060		MOV	#FMT15,-(SP)			
4968	043370	012746	000002		MOV	#2,-(SP)			

:MLX4
:

SYSTEM ERROR DETECTOR

27-Mar-1982 19:24:42
27-Mar-1982 19:23:44

TOPS
PA:C

4969	043374	010600			MOV	SP,RO		: SP,*	
4970	043376	104414			TRAP	14			
4971	043400	062706	000006		ADD	#6,SP			
4972	043404	032777	001000	170744	35\$:	BIT	#1000,@ML.REG+10	:	3642
4973	043412	001412			BEQ	36\$			
4974	043414	012746	007460		MOV	#MLB8,-(SP)			
4975	043420	012746	007060		MOV	#FMT15,-(SP)			
4976	043424	012746	000002		MOV	#2,-(SP)			
4977	043430	010600			MOV	SP,RO		: SP,*	
4978	043432	104414			TRAP	14			
4979	043434	062706	000006		ADD	#6,SP			
4980	043440	032777	000400	170710	36\$:	BIT	#400,@ML.REG+10	:	3646
4981	043446	001416			BEQ	37\$			
4982	043450	032777	040000	170732		BIT	#40000,@ML.REG+42		
4983	043456	001012			BNE	37\$			
4984	043460	012746	007464		MOV	#MLB9,-(SP)			
4985	043464	012746	007060		MOV	#FMT15,-(SP)			
4986	043470	012746	000002		MOV	#2,-(SP)			
4987	043474	010600			MOV	SP,RO		: SP,*	
4988	043476	104414			TRAP	14			
4989	043500	062706	000006		ADD	#6,SP			
4990	043504	032777	020000	170634	37\$:	BIT	#20000,@ML.REG	:	3648
4991	043512	001412			BEQ	38\$			
4992	043514	012746	007472		MOV	#MLB10,-(SP)			
4993	043520	012746	007060		MOV	#FMT15,-(SP)			
4994	043524	012746	000002		MOV	#2,-(SP)			
4995	043530	010600			MOV	SP,RO		: SP,*	
4996	043532	104414			TRAP	14			
4997	043534	062706	000006		ADD	#6,SP			
4998	043540	032701	000001		38\$:	BIT	#1,R1	:	*,ERR4
4999	043544	001524			BEQ	44\$			
5000	043546	032777	040000	170606		BIT	#40000,@ML.REG+14	:	3656
5001	043554	001412			BEQ	39\$			
5002	043556	012746	007500		MOV	#MLB11,-(SP)			
5003	043562	012746	007060		MOV	#FMT15,-(SP)			
5004	043566	012746	000002		MOV	#2,-(SP)			
5005	043572	010600			MOV	SP,RO		: SP,*	
5006	043574	104414			TRAP	14			
5007	043576	062706	000006		ADD	#6,SP			
5008					;MLX4				
5009					;				
5010						SYSTEM ERROR DETECTOR			
5011	043602	032777	002000	170552	39\$:	BIT	#2000,@ML.REG+14	:	3658
5012	043610	001412			BEQ	40\$			
5013	043612	012746	007504		MOV	#MLB12,-(SP)			
5014	043616	012746	007060		MOV	#FMT15,-(SP)			
5015	043622	012746	000002		MOV	#2,-(SP)			
5016	043626	010600			MOV	SP,RO		: SP,*	
5017	043630	104414			TRAP	14			
5018	043632	062706	000006		ADD	#6,SP			
5019	043636	032777	001000	170516	40\$:	BIT	#1000,@ML.REG+14	:	3660
5020	043644	001412			BEQ	41\$			
5021	043646	012746	007510		MOV	#MLB13,-(SP)			
5022	043652	012746	007060		MOV	#FMT15,-(SP)			
5023	043656	012746	000002		MOV	#2,-(SP)			
5024	043662	010600			MOV	SP,RO		: SP,*	
5025	043664	104414			TRAP	14			

27-Mar-1982 19:24:42 TOPS
 27-Mar-1982 19:23:44 PA:<

5026	043666	062706	000006			ADD	#6,SP			
5027	043672	132777	000004	170462	41\$:	BITB	#4,@ML.REG+14	:		3662
5028	043700	001412				BEQ	42\$			
5029	043702	012746	007514			MOV	#MLB14,-(SP)			
5030	043706	012746	007060			MOV	#FMT15,-(SP)			
5031	043712	012746	000002			MOV	#2,-(SP)			
5032	043716	010600				MOV	SP,R0	:	SP,*	
5033	043720	104414				TRAP	14			
5034	043722	062706	000006			ADD	#6,SP			
5035	043726	132777	000002	170426	42\$:	BITB	#2,@ML.REG+14	:		3664
5036	043734	001412				BEQ	43\$			
5037	043736	012746	007520			MOV	#MLB15,-(SP)			
5038	043742	012746	007060			MOV	#FMT15,-(SP)			
5039	043746	012746	000002			MOV	#2,-(SP)			
5040	043752	010600				MOV	SP,R0	:	SP,*	
5041	043754	104414				TRAP	14			
5042	043756	062706	000006			ADD	#6,SP			
5043	043762	132777	000001	170372	43\$:	BITB	#1,@ML.REG+14	:		3666
5044	043770	001412				BEQ	44\$			
5045	043772	012746	007524			MOV	#MLB16,-(SP)			
5046	043776	012746	007060			MOV	#FMT15,-(SP)			
5047	044002	012746	000002			MOV	#2,-(SP)			
5048	044006	010600				MOV	SP,R0	:	SP,*	
5049	044010	104414				TRAP	14			
5050	044012	062706	000006			ADD	#6,SP			
5051	044016	032702	000001		44\$:	BIT	#1,R2	:	*,ERR5	3670
5052	044022	001452				BEQ	47\$			
5053	044024	032777	020000	170330		BIT	#20000,@ML.REG+14	:		3674
5054	044032	001412				BEQ	45\$			
5055	044034	012746	007530			MOV	#MLB17,-(SP)			
5056	044040	012746	007060			MOV	#FMT15,-(SP)			
5057	044044	012746	000002			MOV	#2,-(SP)			
5058	044050	010600				MOV	SP,R0	:	SP,*	
5059	044052	104414				TRAP	14			
5060	044054	062706	000006			ADD	#6,SP			
5061	044060	132777	000040	170274	45\$:	BITB	#40,@ML.REG+14	:		3676
5062	044066	001412				BEQ	46\$			
5063										
5064					:MLX4					
5065					:					
							SYSTEM ERROR DETECTOR			
5066	044070	012746	007534			MOV	#MLB18,-(SP)			
5067	044074	012746	007060			MOV	#FMT15,-(SP)			
5068	044100	012746	000002			MOV	#2,-(SP)			
5069	044104	010600				MOV	SP,R0	:	SP,*	
5070	044106	104414				TRAP	14			
5071	044110	062706	000006			ADD	#6,SP			
5072	044114	132777	000010	170240	46\$:	BITB	#10,@ML.REG+14	:		3678
5073	044122	001412				BEQ	47\$			
5074	044124	012746	007542			MOV	#MLB19,-(SP)			
5075	044130	012746	007060			MOV	#FMT15,-(SP)			
5076	044134	012746	000002			MOV	#2,-(SP)			
5077	044140	010600				MOV	SP,R0	:	SP,*	
5078	044142	104414				TRAP	14			
5079	044144	062706	000006			ADD	#6,SP			
5080	044150	032766	000001	000004	47\$:	BIT	#1,4(SP)	:	*,ERR6	3690
5081	044156	001403				BEQ	48\$			
5082	044160	012700	000004			MOV	#4,R0	:		3692

27-Mar-1982 19:24:42 TOPS
27-Mar-1982 19:23:44 PA:<

5083	044164	000455			BR	54\$				
5084	044166	006003		48\$:	ROR	R3		; ERR7		3694
5085	044170	103003			BCC	49\$				
5086	044172	012700	000005		MOV	#5,R0				3690
5087	044176	000450			BR	54\$				
5088	044200	032766	000001	000002	49\$:	BIT	#1,2(SP)		; *,ERR2	3698
5089	044206	001403			BEQ	50\$				
5090	044210	012700	000002		MOV	#2,R0				3700
5091	044214	000441			BR	54\$				
5092	044216	006001		50\$:	ROR	R1		; ERR4		3702
5093	044220	103003			BCC	51\$				
5094	044222	012700	000003		MOV	#3,R0				3704
5095	044226	000434			BR	54\$				
5096	044230	032716	000001	51\$:	BIT	#1,(SP)		; *,ERR3		3706
5097	044234	001024			BNE	52\$				
5098	044236	006002			ROR	R2		; ERR5		
5099	044240	103422			BLO	52\$				
5100	044242	032767	000001	136010	BIT	#1,ERROUT				3712
5101	044250	001416			BEQ	52\$				
5102	044252	032767	000001	170060	BIT	#1,RETRYING				
5103	044260	001012			BNE	52\$				
5104	044262	012746	010140		MOV	#PHR11,-(SP)				
5105	044266	012746	007072		MOV	#SAY1,-(SP)				
5106	044272	012746	000002		MOV	#2,-(SP)				
5107	044276	010600			MOV	SP,R0		; SP,*		
5108	044300	104414			TRAP	14				
5109	044302	062706	000006		ADD	#6,SP				
5110	044306	012705	000001	52\$:	MOV	#1,R5				3706
5111	044312	010500			MOV	R5,R0				3428
5112	044314	000401			BR	54\$				
5113	044316	005000		53\$:	CLR	R0				3427
5114	044320	062706	000006	54\$:	ADD	#6,SP				
5115	044324	000207			RTS	PC				

```

5116
5117      ; Routine Size: 647 words
5118      ;MLX4
5119      ;
5120      ; SYSTEM ERROR DETECTOR
5121      ;
5122      ; Maximum stack depth per invocation: 15 words
5126
5127

```

27-Mar-1982 19:24:42 TOPS
27-Mar-1982 19:23:44 PA:<

```

5129 :MLX4
5130 :
5131 :
5132 : 3719 %sbttl 'DATA COMPARISON ROUTINE'
5133 : 3720 routine DOUBLE_CHECK (W_POINTER, R_POINTER, COUNT) =
5134 : 3721 begin !* 1 *
5135 : 3722
5136 : 3723
5137 : 3724 ++ ROUTINE: DOUBLE_CHECK(W_POINTER,R_POINTER,COUNT)
5138 : 3725
5139 : 3726 PURPOSE: TO DOUBLE CHECK THE ECC DETECTION LOGIC AFTER A
5140 : 3727 SUCCESSFUL READ COMMAND. THE WRITE AND READ BUFFERS
5141 : 3728 ARE COMPARED, AND NO ERRORS SHOULD BE FOUND UNDER
5142 : 3729 NORMAL CIRCUMSTANCES.
5143 : 3730
5144 : 3731 ARGUMENTS: W_POINTER = POINTER TO THE WRITE BUFFER
5145 : 3732 R_POINTER = POINTER TO THE READ BUFFER
5146 : 3733 COUNT = THE NUMBER OF WORDS TO COMPARE
5147 : 3734
5148 : 3735 RESULTS: VALUE RETURNED IS EITHER:
5149 : 3736 0 = NO ERRORS WERE FOUND
5150 : 3737 N > 0 = ADDRESS IN WRITE BUFFER WHERE ERROR WAS FOUND
5151 : 3738 --
5152 : 3739
5153 : 3740 local
5154 : 3741 VALUE,
5155 : 3742 OFFSET,
5156 : 3743 GOOD,
5157 : 3744 BAD;
5158 : 3745
5159 : 3746 label
5160 : 3747 LOOP;
5161 : 3748
5162 : 3749 OFFSET = 0;
5163 : 3750 VALUE = 0;
5164 : 3751 LOOP :
5165 : 3752 begin !* 2 *
5166 : 3753
5167 : 3754 incr I from 1 to .COUNT do
5168 : 3755 begin !* 3 *
5169 : 3756 GOOD = (.W_POINTER + .OFFSET);
5170 : 3757 BAD = (.R_POINTER + .OFFSET);
5171 : 3758
5172 : 3759 if .GOOD eql .BAD
5173 : 3760 then
5174 : 3761 OFFSET = .OFFSET + 2
5175 : 3762 else
5176 : 3763 begin !* 4 *
5177 : 3764 VALUE = (.W_POINTER + .OFFSET); !ADDRESS OF GOOD
5178 : 3765 leave LOOP;
5179 : 3766 end; !* 4 *
5180 : 3767
5181 : 3768 end; !* 3 *
5182 : 3769
5183 : 3770 end; !* 2 *

```

27-Mar-1982 19:24:42

TOPS-20 Bliss-16 V2(212)
PA:<NEALE>MLX4.BLI.5 (23)

27-Mar-1982 19:23:44

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (23)

5185 :MLX4
 5186 : DATA COMPARISON ROUTINE
 5187 :
 5188 : 3771 return .VALUE;
 5189 : 3772 end;

!EITHER 0 OR THE ADDRESS OF GOOD DATA
 !* 1 *

5194 :
 5198 : .SBTTL DOUBLE.CHECK DATA COMPARISON ROUTINE

5198	044326		DOUBLE.CHECK:	JSR	R1,\$SAVE5	:		3720
5199	044326	004167		TST	-(SP)	:		
5200	044332	005746		CLR	R3	:	OFFSET	3749
5201	044334	005003		CLR	R1	:	VALUE	3750
5202	044336	005001		CLR	R2	:	I	3754
5203	044340	005002		BR	3\$:		
5204	044342	000417				:		
5205	044344	010304	1\$:	MOV	R3,R4	:	OFFSET,*	3756
5206	044346	066604	000024	ADD	24(SP),R4	:	W.POINTER,*	
5207	044352	011416		MOV	(R4),(SP)	:	*GOOD	
5208	044354	010305		MOV	R3,R5	:	OFFSET,*	3757
5209	044356	066605	000022	ADD	22(SP),R5	:	R.POINTER,*	
5210	044362	011500		MOV	(R5),R0	:	*BAD	
5211	044364	021600		CMP	(SP),R0	:	*BAD	3759
5212	044366	001003		BNE	2\$:		
5213	044370	062703	000002	ADD	#2,R3	:	*OFFSET	3761
5214	044374	000402		BR	3\$:		3759
5215	044376	010401		MOV	R4,R1	:	*VALUE	3764
5216	044400	000404		BR	4\$:		3765
5217	044402	005202		INC	R2	:	I	3754
5218	044404	020266	000020	CMP	R2,20(SP)	:	I,COUNT	
5219	044410	003755		BLE	1\$:		
5220	044412	010100		MOV	R1,R0	:	VALUE,*	3721
5221	044414	005726		TST	(SP)+	:		3720
5222	044416	000207		RTS	PC	:		

5223 :
 5224 : Routine Size: 29 words
 5225 : Maximum stack depth per invocation: 7 words
 5230 :
 5231 :

```
5233 :MLX4
5234 :
5235 :
5236 : 3773 %sbttl 'COMMAND INITIATION AND TERMINATION'
5237 : 3774 routine WAITER : novalue =
5238 : 3775 begin
5239 : 3776 !+
5240 : 3777 ! THIS ROUTINE DOES NOTHING, BUT IT IS CALLED BY 'START IT' TO
5241 : 3778 ! WASTE TIME WHILE WAITING TO BEGIN OR END AN ML11 TRANSFER.
5242 : 3779 !-
5243 : 3780 BREAK;
5244 : 3781 return;
5245 : 3782 end;
5249
5250
5254 044420 104422 .SBTTL WAITER COMMAND INITIATION AND TERMINATION
5255 044422 000207 WAITER: TRAP 22
5256 : RTS PC ;
5257 : ;
5258 : Routine Size: 2 words
5263 : Maximum stack depth per invocation: 0 words
5264
```

27-Mar-1982 19:24:42
27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
PA:<NEALE>MLX4.BLI.5 (24)

3775
3774

5266 :MLX4
5267 :
5268 :
5269 :
5270 :
5271 :
5272 :
5273 :
5274 :
5275 :
5276 :
5277 :
5278 :
5279 :
5280 :
5281 :
5282 :
5283 :
5284 :
5285 :
5286 :
5287 :
5288 :
5289 :
5290 :
5291 :
5292 :
5293 :
5294 :
5295 :
5296 :
5297 :
5298 :
5299 :
5300 :
5301 :
5302 :
5303 :
5304 :
5305 :
5306 :
5307 :
5308 :
5309 :
5310 :
5311 :
5312 :
5313 :
5314 :
5315 :
5316 :
5317 :
5318 :
5319 :
5320 :

3783
3784
3785
3786
3787
3788
3789
3790
3791
3792
3793
3794
3795
3796
3797
3798
3799
3800
3801
3802
3803
3804
3805
3806
3807
3808
3809
3810
3811
3812
3813
3814
3815
3816
3817
3818
3819
3820
3821
3822
3823
3824
3825
3826
3827
3828
3829
3830
3831
3832
3833
3834

COMMAND INITIATION AND TERMINATION

routine START_IT (COMMAND, LUN, WRDCNT, BUFFER, SECTOR) =
begin

++
ROUTINE: START_IT(COMMAND,LUN,WRDCNT,BUFFER,SECTOR)
PURPOSE: TO INITIATE A TRANSFER TO OR FROM THE ML11,
TO WAIT FOR THE TRANSFER TO COMPLETE, AND TO CALL
THE 'SYSERR' ROUTINE TO LOOK FOR RESULTING ERRORS.
ARGUMENTS: (1) COMMAND - THE FUNCTION CODE FOR THE TYPE OF TRANSFER
DESIRED. THIS CODE WILL BE SENT TO THE MLCS1
REGISTER TO START THE OPERATION, SINCE IT
ALSO CONTAINS THE GO BIT.
(2) LUN - LOGICAL UNIT NUMBER
(3) WRDCNT - NUMBER OF 16-BIT WORDS TO TRANSFER
(4) BUFFER - THE ADDRESS IN MAIN MEMORY OF THE SELECTED
WRITE OR READ BUFFER
(5) SECTOR THE TRANSFER'S STARTING ADDRESS IN THE ML11
RESULTS: VALUES RETURNED ARE IDENTICAL TO THOSE DEFINED ABOVE IN
THE 'SYSERR' ROUTINE.

--
Local
TEMP,
READY_BIT,
RTN;

Label
LOOP;

! WAIT FOR DRIVE READY:

UNIT = .DRIVE; !SELECT THE UNIT
READY_BIT = 0;

until .READY_BIT neq 0 do !WAIT FOR DRIVE READY
begin
WAITER ();
READY_BIT = .MLCS1 and BIT7;
end;

! INITIALIZE BEFORE EACH TRANSFER:

CLR = 1; !CONTROLLER CLEAR

27-Mar-1982 19:24:42
27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
PA:<NEALE>MLX4.BLI.5 (25)

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (25)

5322 :MLX4
5323 :
5324 :
5325 :
5326 :
5327 :
5328 :
5329 :
5330 :
5331 :
5332 :
5333 :
5334 :
5335 :
5336 :
5337 :
5338 :
5339 :
5340 :
5341 :
5342 :
5343 :
5344 :
5345 :
5346 :
5347 :
5348 :
5349 :
5350 :
5351 :
5352 :
5353 :
5354 :
5355 :
5356 :
5357 :
5358 :
5359 :
5360 :
5361 :
5362 :
5363 :
5364 :
5365 :
5366 :
5367 :
5368 :
5369 :
5370 :
5371 :
5372 :
5373 :
5374 :
5375 :
5376 :

COMMAND INITIATION AND TERMINATION

```

3835 DELAY (1); !DELAY FOR CLEAR TO COMPLETE
3836 MLCS1 = DRV_CLR; !DRIVE CLEAR
3837 DELAY (1); !DELAY FOR CLEAR TO COMPLETE
3838 UNIT = .DRIVE; !PUT BACK THE DRIVE NUMBER
3839
3840 SET UP THE REQUIRED ENABLE/DISABLE BITS:
3841
3842 DCK_EN = 1; !ALLOW REPORTING OF DATA CHECK ERRORS
3843
3844 if .REFRESH then REF_MAR = 1; !TURN ON REFRESH MARGINING IF OPERATOR SELECTED IT
3845
3846 if .ECCDIS then ECC_DIS = 1; !TURN OFF ECC IF OPERATOR SELECTED IT
3847
3848
3849 SEND REQUIRED INFORMATION TO DEVICE REGISTERS:
3850
3851 MLWC = -(.WRDCNT); !WORD COUNT REGISTER
3852 MLBA = .BUFFER; !BUS ADDRESS REGISTER (ADDRESS IN MAIN MEMORY)
3853 MLDA = .SECTOR; !DESIRED SECTOR ADDRESS REGISTER (ADDRESS IN ML11)
3854
3855 GO:
3856
3857 I_AM_DONE = INACTIVE;
3858 TEMP = .COMMAND + BIT6; !SET THE INTERRUPT ENABLE BIT WHILE LOADING THE
3859 MLCS1 = .TEMP; !CONTROL AND STATUS REGISTER WITH THE COMMAND
3860
3861 WAIT FOR DRIVE TO FINISH:
3862
3863 LOOP :
3864 begin
3865 READY_BIT = 0;
3866
3867 until .I_AM_DONE do
3868 begin
3869 WAITER ();
3870 READY_BIT = .MLCS1 and BIT7;
3871
3872 if ((.READY_BIT neq 0) and (.I_AM_DONE eql INACTIVE))
3873 then
3874 begin
3875 if ( not .RETRYING)
3876 then
3877 begin
3878 SAYWHO (.LUN);
3879
3880 if .COMMAND eql WR_CMD then RTN = WRD16;
3881
3882 if .COMMAND eql RD_CMD then RTN = WRD17;
3883
3884 if .COMMAND eql WC_CMD then RTN = PHR1;
3885
3886

```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (25)

5378 :MLX4
 5379 :
 5380 :
 5381 : 3887
 5382 : 3888
 5383 : 3889
 5384 : 3890
 5385 : 3891
 5386 : 3892
 5387 : 3893
 5388 : 3894
 5389 : 3895
 5390 : 3896
 5391 : 3897
 5392 : 3898
 5393 : 3899
 5394 : 3900
 5395 : 3901
 5399 :
 5400 :
 5401 :
 5402 :
 5403 :

COMMAND INITIATION AND TERMINATION

```

PRINTB (FMT6, .WRDCNT, .RTN, WRD15, .SECTOR);
!'BEGAN YYYY WORD (???) AT SECTOR ZZZZZZ'
!WHERE (???) IS EITHER 'WRITE', 'READ' OR 'WRITE CHECK'
PRINTB (SAY1, MSG0);
!INTERRUPT DID NOT OCCUR, BUT THE TRANSFER IS COMPLETE
end;
    
```

Leave LOOP;
 end;

end;

end;
 return SYSERR (.LUN);
 end;

!PASS ALONG THE ERROR VALUES THAT 'SYSERR' OBTAINED.

.GLOBL LSDLY

.SBTTL START.IT COMMAND INITIATION AND TERMINATION

5407 044424
 5408 044424 004167 140664
 5409 044430 005746
 5410 044432 016604 000024
 5411 044436 010401
 5412 044440 006301
 5413 044442 016102 034422
 5414 044446 016203 000006
 5415 044452 042703 177770
 5416 044456 142777 000007 167672
 5417 044464 150377 167666
 5418 044470 005003
 5419 044472 020707
 5420 044474 001007
 5421 044476 004767 177716
 5422 044502 017703 167640
 5423 044506 042703 177577
 5424 044512 000770
 5425 044514 152777 000040 167634 2\$:
 5426 044522 012700 000001
 5427 044526 001410 3\$:
 5428 044530 016702 135362
 5429 044534 001403
 5430 044536 005016 4\$:
 5431 044540 005302
 5432 044542 001375

START.IT:

```

JSR R1,SSAVE4 ;
TST -(SP)
MOV 24(SP),R4 ; LUN,*
MOV R4,R1
ASL R1
MOV PTABLE.ADDR(R1),R2
MOV 6(R2),R3
BIC #177770,R3
BICB #7,2ML.REG+10
BISB R3,2ML.REG+10
CLR R3 ; READY.BIT
CMP PC,PC ;
BNE 2$ 1$:
JSR PC,WAITER ;
MOV 2ML,REG,R3 ; *,READY.BIT
BIC #177577,R3 ; *,READY.BIT
BR 1$ ;
BISB #40,2ML.REG+10 ;
MOV #1,R0 ; *,SSTMP2
BEQ 6$ ;
MOV LSDLY,R2 ; *,SSTMP1
BEQ 5$ ;
CLR (SP) ; SSTMP
DEC R2 ; SSTMP1
BNE 4$ ;
    
```

3783
 3822
 3823
 3825
 3827
 3828
 3825
 3834
 3835

Address	Instruction	Hex	Hex	Hex	Label	Code	Comment	Value
5434					:MLX4			
5435					:		COMMAND INITIATION AND TERMINATION	
5436								
5437	044544	005300			5\$:	DEC R0	: \$\$TMP2	
5438	044546	000767				BR 3\$		
5439	044550	012777	000011	167570	6\$:	MOV #11,AML.REG	:	3836
5440	044556	012700	000001			MOV #1,R0	: *,\$\$TMP2	3837
5441	044562	001410			7\$:	BEQ 10\$		
5442	044564	016702	135326			MOV LSDLY,R2	: *,\$\$TMP1	
5443	044570	001403				BEQ 9\$		
5444	044572	005016			8\$:	CLR (SP)	: \$\$TMP	
5445	044574	005302				DEC R2	: \$\$TMP1	
5446	044576	001375				BNE 8\$		
5447	044600	005300			9\$:	DEC R0	: \$\$TMP2	
5448	044602	000767				BR 7\$		
5449	044604	016102	034422		10\$:	MOV PTABLE.ADDR(R1),R2	:	3838
5450	044610	016201	000006			MOV 6(R2),R1		
5451	044614	042701	177770			BIC #177770,R1		
5452	044620	142777	000007	167530		BICB #7,AML.REG+10		
5453	044626	150177	167524			BISB R1,AML.REG+10		
5454	044632	152777	000004	167532		BISB #4,AML.REG+24	:	3842
5455	044640	032767	000001	135404		BIT #1,REFRESH	:	3844
5456	044646	001403				BEQ 11\$		
5457	044650	152777	000200	167514		BISB #200,AML.REG+24		
5458	044656	032767	000001	135370	11\$:	BIT #1,ECCDIS	:	3846
5459	044664	001403				BEQ 12\$		
5460	044666	152777	000002	167476		BISB #2,AML.REG+24		
5461	044674	016677	000022	167446	12\$:	MOV 22(SP),AML.REG+2	: WRDCNT,*	3851
5462	044702	005477	167442			NEG AML.REG+2		
5463	044706	016677	000020	167436		MOV 20(SP),AML.REG+4	: BUFFER,*	3852
5464	044714	016677	000016	167432		MOV 16(SP),AML.REG+6	: SECTOR,*	3853
5465	044722	005067	167410			CLR I.AM.DONE	:	3857
5466	044726	016602	000026			MOV 26(SP),R2	: COMMAND,*	3858
5467	044732	010201				MOV R2,R1	: *,TEMP	
5468	044734	062701	000100			ADD #100,R1	: *,TEMP	
5469	044740	010177	167402			MOV R1,AML.REG	: TEMP,*	3859
5470	044744	005003				CLR R3	: READY.BIT	3865
5471	044746	032767	000001	167362	13\$:	BIT #1,I.AM.DONE	:	3867
5472	044754	001067				BNE 17\$		
5473	044756	004767	177436			JSR PC,WAITER	:	3869
5474	044762	017703	167360			MOV AML.REG,R3	: *,READY.BIT	3870
5475	044766	042703	177577			BIC #177577,R3	: *,READY.BIT	
5476	044772	001765				BEQ 13\$:	3872
5477	044774	005767	167336			TST I.AM.DONE		
5478	045000	001362				BNE 13\$		
5479	045002	032767	000001	167330		BIT #1,RETRYING	:	3876
5480	045010	001051				BNE 17\$		
5481	045012	010446				MOV R4,-(SP)	:	3879
5482	045014	004767	170504			JSR PC,SAYWHO		
5483	045020	020227	000061			CMP R2,#61	:	3881
5484	045024	001002				BNE 14\$		
5485	045026	012701	007256			MOV #WORD16,R1	: *,RTN	
5486	045032	020227	000071		14\$:	CMP R2,#71	:	3883
5487	045036	001002				BNE 15\$		
5488	045040	012701	007264			MOV #WORD17,R1	: *,RTN	

Address	OpCode	Operand 1	Operand 2	Operand 3	Label	Instruction	Comments	Address	Time	Page
5490					:MLX4			27-Mar-1982 19:24:42		TOPS
5491					:			27-Mar-1982 19:23:44		PA:<
5492						COMMAND INITIATION AND TERMINATION				
5493	045044	020227	000051		15\$:	CMP R2,#51	:			3885
5494	045050	001002				BNE 16\$				
5495	045052	012701	007714			MOV #PHR1,R1	:*,RTN			
5496	045056	016616	000020		16\$:	MOV 20(SP),(SP)	:SECTOR,*			3887
5497	045062	012746	007246			MOV #WRD15,-(SP)				
5498	045066	010146				MOV R1,-(SP)	:RTN,*			
5499	045070	016646	000030			MOV 30(SP),(SP)	:WRDCNT,*			
5500	045074	012746	006376			MOV #FMT6,-(SP)				
5501	045100	012746	000005			MOV #5,-(SP)				
5502	045104	010600				MOV SP,R0	:SP,*			
5503	045106	104414				TRAP 14				
5504	045110	012716	011002			MOV #MSG0,(SP)	:			3890
5505	045114	012746	007072			MOV #SAY1,-(SP)				
5506	045120	012746	000002			MOV #2,-(SP)				
5507	045124	010600				MOV SP,R0	:SP,*			
5508	045126	104414				TRAP 14				
5509	045130	062706	000020			ADD #20,SP	:			3878
5510	045134	010446			17\$:	MOV R4,-(SP)	:			3900
5511	045136	004767	174546			JSR PC,SYERR				
5512	045142	022626				CMP (SP)+,(SP)+	:			3783
5513	045144	000207				RTS PC				
5514										
5515										
5516										
5521										
5522										

: Routine Size: 169 words
 : Maximum stack depth per invocation: 14 words

```

5524 :MLX4
5525 : COUNTING BYTES TRANSFERED
5526 :
5527 : 3902 %sbttl 'COUNTING BYTES TRANSFERED'
5528 : 3903 routine UP_WR_COUNT (WORD_COUNT) : novalue =
5529 : 3904 begin
5530 : 3905 WR_COUNT = .WR_COUNT + (.WORD_COUNT*2);
5531 : 3906
5532 : 3907 if .WR_COUNT geq 20000
5533 : 3908 then
5534 : 3909 begin
5535 : 3910 WR_THOUSANDS = .WR_THOUSANDS + 20;
5536 : 3911 WR_COUNT = .WR_COUNT - 20000;
5537 : 3912
5538 : 3913 if .WR_THOUSANDS geq 1000
5539 : 3914 then
5540 : 3915 begin
5541 : 3916 WR_THOUSANDS = .WR_THOUSANDS - 1000;
5542 : 3917 WR_MILLIONS = .WR_MILLIONS + 1;
5543 : 3918
5544 : 3919 if .WR_MILLIONS lss 0 then WR_MILLIONS = 0;
5545 : 3920
5546 : 3921 end;
5547 : 3922
5548 : 3923 end;
5549 : 3924
5550 : 3925 return;
5551 : 3926 end;
5555 :
5556 :

```

```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (26)

```

Address	Hex	Dec	Hex	Dec	Hex	Dec	Instruction	Comment	Label
5560	045146	016600	000002				UP.WR.COUNT:		
5561	045146	016600	000002				MOV	2(SP),RO	3905
5562	045152	006300					ASL	RO	
5563	045154	066700	167134				ADD	WR.COUNT,RO	
5564	045160	010067	167130				MOV	RO,WR.COUNT	
5565	045164	020027	047040				CMP	RO,#47040	3907
5566	045170	002422					BLT	1\$	
5567	045172	062767	000024	167116			ADD	#24,WR.THOUSANDS	3910
5568	045200	162767	047040	167106			SUB	#47040,WR.COUNT	3911
5569	045206	026727	167104	001750			CMP	WR.THOUSANDS,#1750	3913
5570	045214	002410					BLT	1\$	
5571	045215	162767	001750	167072			SUB	#1750,WR.THOUSANDS	3916
5572	045224	005267	167070				INC	WR.MILLIONS	3917
5573	045230	100002					BPL	1\$	3919
5574	045232	005067	167062				CLR	WR.MILLIONS	
5575	045236	000207					1\$: RTS	PC	3903

```

5576 :
5577 : ; Routine Size: 29 words
5578 : ; Maximum stack depth per invocation: 0 words
5579 : ;MLX4
5580 : COUNTING BYTES TRANSFERED
5581 :
5586 :
5587 :

```

```

27-Mar-1982 19:24:42 TOPS
27-Mar-1982 19:23:44 PA:<

```


27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (27)

```

5589 ;MLX4
5590 ;          COUNTING BYTES TRANSFERED
5591 ;
5592 ;          3927 routine UP_RD_COUNT (WORD_COUNT) : novalue =
5593 ;          3928 begin
5594 ;          3929 RD_COUNT = .RD_COUNT + (.WORD_COUNT*2);
5595 ;          3930
5596 ;          3931 if .RD_COUNT geq 20000
5597 ;          3932 then
5598 ;          3933 begin
5599 ;          3934 RD_THOUSANDS = .RD_THOUSANDS + 20;
5600 ;          3935 RD_COUNT = .RD_COUNT - 20000;
5601 ;          3936
5602 ;          3937 if .RD_THOUSANDS geq 1000
5603 ;          3938 then
5604 ;          3939 begin
5605 ;          3940 RD_THOUSANDS = .RD_THOUSANDS - 1000;
5606 ;          3941 RD_MILLIONS = .RD_MILLIONS + 1;
5607 ;          3942
5608 ;          3943 if .RD_MILLIONS lss 0 then RD_MILLIONS = 0;
5609 ;          3944
5610 ;          3945 end;
5611 ;          3946
5612 ;          3947 end;
5613 ;          3948
5614 ;          3949 return;
5615 ;          3950 end;
5619 ;
5620 ;
    
```

		.SBTTL	UP.RD.COUNT COUNTING BYTES TRANSFERED		
5624	045240	UP.RD.COUNT:			
5625	045240	MOV	2(SP),R0	: WORD.COUNT,*	3929
5626	045244	ASL	R0		
5627	045246	ADD	RD.COUNT,R0		
5628	045252	MOV	R0,RD.COUNT		
5629	045256	CMP	R0,#47040	: RD.COUNT,*	3931
5630	045262	BLT	1\$		
5631	045264	ADD	#24,RD.THOUSANDS	:	3934
5632	045272	SUB	#47040,RD.COUNT	:	3935
5633	045300	CMP	RD.THOUSANDS,#1750	:	3937
5634	045306	BLT	1\$		
5635	045310	SUB	#1750,RD.THOUSANDS	:	3940
5636	045316	INC	RD.MILLIONS	:	3941
5637	045322	BPL	1\$:	3943
5638	045324	CLR	RD.MILLIONS	:	
5639	045330	1\$: RTS	PC	:	3927

5640 ;
 5641 ; Routine Size: 29 words
 5642 ; Maximum stack depth per invocation: 0 words
 5650 ;
 5651 ;

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (28)

```

5653 :MLX4
5654 :
5655 :
5656 : 3951 routine UP_WC_COUNT (WORD_COUNT) : novalue =
5657 : 3952 begin
5658 : 3953 WC_COUNT = .WC_COUNT + (.WORD_COUNT*2);
5659 : 3954
5660 : 3955 if .WC_COUNT geq 20000
5661 : 3956 then
5662 : 3957 begin
5663 : 3958 WC_THOUSANDS = .WC_THOUSANDS + 20;
5664 : 3959 WC_COUNT = .WC_COUNT - 20000;
5665 : 3960
5666 : 3961 if .WC_THOUSANDS geq 1000
5667 : 3962 then
5668 : 3963 begin
5669 : 3964 WC_THOUSANDS = .WC_THOUSANDS - 1000;
5670 : 3965 WC_MILLIONS = .WC_MILLIONS + 1;
5671 : 3966
5672 : 3967 if .WC_MILLIONS lss 0 then WC_MILLIONS = 0;
5673 : 3968
5674 : 3969 end;
5675 : 3970
5676 : 3971 end;
5677 : 3972
5678 : 3973 return;
5679 : 3974 end;
  
```

		.SBTTL UP.WC.COUNT COUNTING BYTES TRANSFERED			
Address	Hex	Dec	Hex	Op	Label
5688	045332	016600	000002	UP.WC.COUNT:	
5689	045332	006300		MOV	2(SP),RO ; WORD.COUNT,* 3953
5690	045336	066700	166764	ASL	RO
5691	045340	010067	166760	ADD	WC.COUNT,RO
5692	045344	020027	047040	MOV	RO,WC_COUNT
5693	045350	002422		CMP	RO,#47040 ; WC.COUNT,* 3955
5694	045354	062767	000024	BLT	1\$
5695	045356	162767	047040	ADD	#24,WC.THOUSANDS ; 3958
5696	045364	026727	166736	SUB	#47040,WC.COUNT ; 3959
5697	045372	002410	001750	CMP	WC.THOUSANDS,#1750 ; 3961
5698	045400	162767	001750	BLT	1\$
5699	045402	005267	166720	SUB	#1750,WC.THOUSANDS ; 3964
5700	045410	100002		INC	WC.MILLIONS ; 3965
5701	045414	005067	166712	BPL	1\$; 3967
5702	045416	000207		CLR	WC.MILLIONS
5703	045422			1\$: RTS	PC ; 3951

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

5704
 5705
 5706
 5714
 5715

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (29)

```

5717 :MLX4
5718 :
5719 :
5720 : 3975 %sbttl 'ML11 TRANSFER COMMANDS'
5721 : 3976 routine write (LUN, WRDCNT, BUFFER, SECTOR) =
5722 : 3977 begin
5723 : 3978
5724 : 3979 !++
5725 : 3980 ROUTINE: WRITE(LUN,WRDCNT,BUFFER,SECTOR)
5726 : 3981
5727 : 3982 PURPOSE: TO TRANSFER INFORMATION FROM MAIN MEMORY TO THE ML11
5728 : 3983 AND TO DECIDE ON THE ADVISABILITY OF A RETRY.
5729 : 3984
5730 : 3985 ARGUMENTS: SEE 'START_IT' ROUTINE ABOVE FOR DETAILS
5731 : 3986
5732 : 3987 RESULTS: UPON TERMINATION OF THE TRANSFER, ERROR DETECTION OCCURS
5733 : 3988 AND THE CONTENTS OF 'VALUE' BECOMES IMPORTANT IN TERMS OF
5734 : 3989 WHETHER THE TRANSFER WAS COMPLETELY SUCCESSFUL OR WHETHER A
5735 : 3990 RETRY SHOULD OR SHOULD NOT BE PERMITTED.
5736 : 3991
5737 : 3992 THE VALUES RETURNED FOR THIS SUBROUTINE ARE THE SAME AS
5738 : 3993 FOR THE SYSERR ROUTINE ABOVE.
5739 : 3994 !--
5740 : 3995
5741 : 3996 local
5742 : 3997 VALUE,
5743 : 3998 COMMAND;
5744 : 3999
5745 : 4000 COMMAND = WR_CMD;
5746 : 4001 VALUE = START_IT (.COMMAND, .LUN, .WRDCNT, .BUFFER, .SECTOR);
5747 : 4002
5748 : 4003 if ( not .RETRYING) then UP_WR_COUNT (.WRDCNT);
5749 : 4004
5750 : 4005 if .VALUE eql 0 then return 0; !NO ERRORS AT ALL
5751 : 4006
5752 : 4007 if ((.ERROUT) and ( not .RETRYING)) then PRINTB (FMT6, .WRDCNT, WRD16, WRD15, .SECTOR);
5753 : 4008
5754 : 4009 !'BEGAN YYYY WORD WRITE AT SECTOR ZZZZZZ'
5755 : 4010 return .VALUE;
5756 : 4011 end;

```

```

5761 :
5765 045424 004167 137632 WRITE: .SBTTL WRITE ML11 TRANSFER COMMANDS
5766 045430 012700 000061 JSR R1,$SAVE2 ;
5767 045434 010046 MOV #61,R0 ; *COMMAND
5768 045436 016646 000020 MOV R0,-(SP) ; COMMAND,*
5769 045442 016601 000020 MOV 20(SP),-(SP) ; LUN,*
5770 045446 010146 MOV 20(SP),R1 ; WRDCNT,*
5771 045450 016646 000020 MOV R1,-(SP) ; BUFFER,*

```

```

5773          ;MLX4
5774          ;
5775          ML11 TRANSFER COMMANDS
5776 045454 016646 000020          MOV      20(SP),-(SP)          ; SECTOR,*
5777 045460 004767 176740          JSR     PC,START.IT
5778 045464 010002          MOV     R0,R2          ; *,VALUE
5779 045466 032767 000001 166644  BIT     #1,RETRYING          ;
5780 045474 001004          BNE    1$
5781 045476 010146          MOV     R1,-(SP)
5782 045500 004767 177442          JSR     PC,UP.WR.COUNT
5783 045504 005726          TST    (SP)+
5784 045506 005702          1$:    TST    R2          ; VALUE          4003
5785 045510 001003          BNE    2$
5786 045512 062706 000012          ADD     #12,SP
5787 045516 000433          BR     4$
5788 045520 032767 000001 134532  2$:    BIT     #1,ERROUT          ;          4007
5789 045526 001423          BEQ    3$
5790 045530 032767 000001 166602  BIT     #1,RETRYING
5791 045536 001017          BNE    3$
5792 045540 016646 000022          MOV     22(SP),-(SP)          ; SECTOR,*
5793 045544 012746 007246          MOV     #WRD15,-(SP)
5794 045550 012746 007256          MOV     #WRD16,-(SP)
5795 045554 010146          MOV     R1,-(SP)
5796 045556 012746 006376          MOV     #FMT6,-(SP)
5797 045562 012746 000005          MOV     #5,-(SP)
5798 045566 010600          MOV     SP,R0          ; SP,*
5799 045570 104414          TRAP   14
5800 045572 062706 000014          ADD     #14,SP
5801 045576 062706 000012          3$:    ADD     #12,SP          ;          3976
5802 045602 010200          MOV     R2,R0          ; VALUE,*          3977
5803 045604 000207          RTS    PC
5804 045606 005000          4$:    CLR     R0          ;          3976
5805 045610 000207          RTS    PC
5806
5807          ; Routine Size: 59 words
5808          ; Maximum stack depth per invocation: 14 words
5813
5814

```

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (30)

5816 :MLX4
 5817 :
 5818 :
 5819 :
 5820 :
 5821 :
 5822 :
 5823 :
 5824 :
 5825 :
 5826 :
 5827 :
 5828 :
 5829 :
 5830 :
 5831 :
 5832 :
 5833 :
 5834 :
 5835 :
 5836 :
 5837 :
 5838 :
 5839 :
 5840 :
 5841 :
 5842 :
 5843 :
 5844 :
 5845 :
 5846 :
 5847 :
 5848 :
 5849 :
 5850 :
 5851 :
 5852 :
 5853 :
 5854 :
 5858 :
 5859 :

```

ML11 TRANSFER COMMANDS
routine read (LUN, WRDCNT, BUFFER, SECTOR) =
begin
++
ROUTINE:   READ(LUN,WRDCNT,BUFFER,SECTOR)
PURPOSE:   TO TRANSFER INFORMATION FROM THE ML11 TO MAIN MEMORY
            USING A 'READ' COMMAND.
ARGUMENTS: SEE 'START_IT' ROUTINE ABOVE FOR DETAILS
RESULTS:   UPON TERMINATION OF THE TRANSFER, ERROR DETECTION OCCURS
            AND THE CONTENTS OF 'VALUE' BECOMES IMPORTANT IN TERMS OF
            WHETHER THE TRANSFER WAS COMPLETELY SUCCESSFUL OR WHETHER A
            RETRY SHOULD OR SHOULD NOT BE PERMITTED.
            THE VALUES RETURNED FOR THIS SUBROUTINE ARE THE SAME AS
            FOR THE SYSERR ROUTINE ABOVE.
--
local
VALUE,
COMMAND;
COMMAND = RD_CMD;
VALUE = START_IT (.COMMAND, .LUN, .WRDCNT, .BUFFER, .SECTOR);
if ( not .RETRYING) then UP_RD_COUNT (.WRDCNT);
if .VALUE eql 0 then return 0;           !NO ERRORS AT ALL
if ((.ERROUT) and ( not .RETRYING)) then PRINTB (FMT6, .WRDCNT, WRD17, WRD15, .SECTOR);
!'BEGAN YYYY WORD READ AT SECTOR ZZZZZZ'
return .VALUE;
end;
    
```

5863	045612	004167	137444	READ:	.SBTTL	READ ML11 TRANSFER COMMANDS		
5864	045616	012700	000071		JSR	R1,\$SAVE2	:	4012
5865	045622	010046			MOV	#71,R0	:	* ,COMMAND
5866	045624	016646	000020		MOV	R0,-(SP)	:	COMMAND,*
5867	045630	016601	000020		MOV	20(SP),-(SP)	:	LUN,*
5868	045634	010146			MOV	20(SP),R1	:	WRDCNT,*
5869	045636	016646	000020		MOV	R1,-(SP)	:	BUFFER,*
5870	045642	016646	000020		MOV	20(SP),-(SP)	:	SECTOR,*

Address	Hex	Hex	Hex	Label	Instruction	Comments	Address
5872				:MLX4			
5873				:			
5874					ML11 TRANSFER COMMANDS		
5875	045646	004767	176552		JSR PC,START.IT		
5876	045652	010002			MOV R0,R2	: *,VALUE	
5877	045654	032767	000001 166456		BIT #1,RETRYING	:	4039
5878	045662	001004			BNE 1\$		
5879	045664	010146			MOV R1,-(SP)		
5880	045666	004767	177346		JSR PC,UP.RD.COUNT		
5881	045672	005726			TST (SP)+		
5882	045674	005702		1\$:	TST R2	: VALUE	4041
5883	045676	001003			BNE 2\$		
5884	045700	062706	000012		ADD #12,SP		
5885	045704	000433			BR 4\$		
5886	045706	032767	000001 134344	2\$:	BIT #1,ERROUT	:	4043
5887	045714	001423			BEQ 3\$		
5888	045716	032767	000001 166414		BIT #1,RETRYING		
5889	045724	001017			BNE 3\$		
5890	045726	016646	000022		MOV 22(SP),-(SP)	: SECTOR,*	
5891	045732	012746	007246		MOV #WRD15,-(SP)		
5892	045736	012746	007264		MOV #WRD17,-(SP)		
5893	045742	010146			MOV R1,-(SP)		
5894	045744	012746	006376		MOV #FMT6,-(SP)		
5895	045750	012746	000005		MOV #5,-(SP)		
5896	045754	010600			MOV SP,R0	: SP,*	
5897	045756	104414			TRAP 14		
5898	045760	062706	000014		ADD #14,SP		
5899	045764	062706	000012	3\$:	ADD #12,SP	:	4012
5900	045770	010200			MOV R2,R0	: VALUE,*	4013
5901	045772	000207			RTS PC		
5902	045774	005000		4\$:	CLR R0	:	4012
5903	045776	000207			RTS PC		

5904
 5905 ; Routine Size: 59 words
 5906 ; Maximum stack depth per invocation: 14 words
 5911
 5912

5914 :MLX4

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (31)

ML11 TRANSFER COMMANDS

5916 :
 5917 : 4048
 5918 : 4049
 5919 : 4050
 5920 : 4051
 5921 : 4052
 5922 : 4053
 5923 : 4054
 5924 : 4055
 5925 : 4056
 5926 : 4057
 5927 : 4058
 5928 : 4059
 5929 : 4060
 5930 : 4061
 5931 : 4062
 5932 : 4063
 5933 : 4064
 5934 : 4065
 5935 : 4066
 5936 : 4067
 5937 : 4068
 5938 : 4069
 5939 : 4070
 5940 : 4071
 5941 : 4072
 5942 : 4073
 5943 : 4074
 5944 : 4075
 5945 : 4076
 5946 : 4077
 5947 : 4078
 5948 : 4079
 5949 : 4080
 5950 : 4081
 5951 : 4082
 5952 : 4083
 5956 :
 5957 :

```

routine CHECK (LUN, WRDCNT, BUFFER, SECTOR) =
begin
  **
  ROUTINE:    CHECK(LUN,WRDCNT,BUFFER,SECTOR)
  PURPOSE:    TO TRANSFER INFORMATION FROM THE ML11 TO MAIN MEMORY
              USING A 'WRITE CHECK' COMMAND.
  ARGUMENTS:  SEE 'START_IT' ROUTINE ABOVE FOR DETAILS
  RESULTS:    UPON TERMINATION OF THE TRANSFER, ERROR DETECTION OCCURS
              AND THE CONTENTS OF 'VALUE' BECOMES IMPORTANT IN TERMS OF
              WHETHER THE TRANSFER WAS COMPLETELY SUCCESSFUL OR WHETHER A
              RETRY SHOULD OR SHOULD NOT BE PERMITTED.

              THE VALUES RETURNED FOR THIS SUBROUTINE ARE THE SAME AS
              FOR THE SYSERR ROUTINE ABOVE.
  --

```

Local
 VALUE,
 COMMAND;

```

COMMAND = WC CMD;
VALUE = START_IT (.COMMAND, .LUN, .WRDCNT, .BUFFER, .SECTOR);
if ( not .RETRYING) then UP_WC_COUNT (.WRDCNT);
if .VALUE eql 0 then return 0;           !NO ERRORS AT ALL
if ((.ERROUT) and ( not .RETRYING)) then PRINTB (FMT6, .WRDCNT, PHR1, WRD15, .SECTOR);
!'BEGAN YYYY WORD WRITE CHECK AT SECTOR ZZZZZ'
return .VALUE;
end;

```

5961 046000 004167 137256
 5962 046004 012700 000051
 5963 046010 010046
 5964 046012 016646 000020
 5965 046016 016601 000020
 5966 046022 010146
 5967 046024 016646 000020
 5968 046030 016646 000020

```

CHECK:  .SBTTL CHECK ML11 TRANSFER COMMANDS
        JSR   R1,$SAVE2
        MOV   #51,R0
        MOV   R0,-(SP)
        MOV   20(SP),-(SP)
        MOV   20(SP),R1
        MOV   R1,-(SP)
        MOV   20(SP),-(SP)
        MOV   20(SP),-(SP)

```

4048
 4072
 4073

```

: * ,COMMAND
: COMMAND,*
: LUN,*
: WRDCNT,*
: BUFFER,*
: SECTOR,*

```

```

5970                                     ;MLX4
5971                                     ;
5972                                     ;
5973 046034 004767 176364                JSR   PC,START.IT
5974 046040 010002                        MOV   R0,R2
5975 046042 032767 000001 166270        BIT   #1,RETRYING
5976 046050 001004                        BNE   1$
5977 046052 010146                        MOV   R1,-(SP)
5978 046054 004767 177252                JSR   PC,UP.WC.COUNT
5979 046060 005726                        TST   (SP)+
5980 046062 005702                        1$:  TST   R2
5981 046064 001003                        BNE   2$
5982 046066 062706 000012                ADD   #12,SP
5983 046072 000433                        BR    4$
5984 046074 032767 000001 134156 2$:  BIT   #1,ERROUT
5985 046102 001423                        BEQ   3$
5986 046104 032767 000001 166226        BIT   #1,RETRYING
5987 046112 001017                        BNE   3$
5988 046114 016646 000022                MOV   22(SP),-(SP)
5989 046120 012746 007246                MOV   #WORD15,-(SP)
5990 046124 012746 007714                MOV   #PHR1,-(SP)
5991 046130 010146                        MOV   R1,-(SP)
5992 046132 012746 006376                MOV   #FMT6,-(SP)
5993 046136 012746 000005                MOV   #5,-(SP)
5994 046142 010600                        MOV   SP,R0
5995 046144 104414                        TRAP  14
5996 046146 062706 000014                ADD   #14,SP
5997 046152 062706 000012 3$:  ADD   #12,SP
5998 046156 010200                        MOV   R2,R0
5999 046160 000207                        RTS   PC
6000 046162 005000                        4$:  CLR   R0
6001 046164 000207                        RTS   PC

```

```

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

```

6002
6003
6004
6009
6010

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (32)

6012 : MLX4
 6013 :
 6014 :
 6015 :
 6016 :
 6017 :
 6018 :
 6019 :
 6020 :
 6021 :
 6022 :
 6023 :
 6024 :
 6025 :
 6026 :
 6027 :
 6028 :
 6029 :
 6030 :
 6031 :
 6032 :
 6033 :
 6034 :
 6035 :
 6036 :
 6037 :
 6038 :
 6039 :
 6040 :
 6041 :
 6042 :
 6043 :
 6047 :
 6048 :
 6052 :
 6053 :
 6054 :
 6055 :
 6056 :
 6057 :
 6058 :
 6059 :
 6060 :
 6061 :
 6062 :
 6063 :
 6071 :
 6072 :

ML11 TRANSFER COMMANDS

```

routine CHOOSE =
begin
    ++
ROUTINE:    CHOOSE
PURPOSE:    TO DECIDE WHETHER TO DO A 'WRITE CHECK' OR A 'READ'.
             THE 'READ' WILL BE SELECTED APPROXIMATELY 25% OF THE
             TIME. THIS IS ACCOMPLISHED BY EXAMINING THE MOST AND
             THE LEAST SIGNIFICANT BITS OF A RANDOM NUMBER. IF
             BOTH ARE SET, THEN 'READ' IS CHOSEN.
RESULTS:    THE VALUE RETURNED FOR THIS ROUTINE IS THE ADDRESS OF
             THE CHOSEN COMMAND (EITHER 'READ' OR 'CHECK').
    --
Local
VALUE:
RN ();
if ((.RANDOM) and (.RANDOM lss 0))
then
    VALUE = read                !CHOOSE THE READ COMMAND
else
    VALUE = CHECK;              !CHOOSE THE WRITE CHECK COMMAND
return .VALUE;
end;
    
```

6052	046166	004767	137176	CHOOSE:	.SBTTL	CHOOSE ML11 TRANSFER COMMANDS		
6053	046172	032767	000001		PC,RN		:	4103
6054	046200	001406	137264		BIT	#1,RANDOM	:	4105
6055	046202	005767	137256		BEQ	1\$		
6056	046206	002003			TST	RANDOM		
6057	046210	012700	045612		BGE	1\$		
6058	046214	000207			MOV	#READ,RO	:	*.VALUE
6059	046216	012700	046000	1\$:	RTS	PC	:	4105
6060	046222	000207			MOV	#CHECK,RO	:	*.VALUE
					RTS	PC	:	4109
							:	4084

; Routine Size: 15 words
 ; Maximum stack depth per invocation: 0 words

6074 :MLX4
 6075 :
 6076 :
 6077 :
 6078 :
 6079 :
 6080 :
 6081 :
 6082 :
 6083 :
 6084 :
 6085 :
 6086 :
 6087 :
 6088 :
 6089 :
 6090 :
 6091 :
 6092 :
 6093 :
 6094 :
 6095 :
 6096 :
 6097 :
 6098 :
 6099 :
 6100 :
 6101 :
 6102 :
 6103 :
 6104 :
 6105 :
 6106 :
 6107 :
 6108 :
 6109 :
 6110 :
 6111 :
 6112 :
 6113 :
 6114 :
 6115 :
 6116 :
 6117 :
 6118 :
 6119 :
 6120 :
 6121 :
 6122 :
 6123 :
 6124 :
 6125 :
 6126 :
 6127 :
 6128 :

ML11 TRANSFER COMMANDS

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(2:2)
 PA:<NEALE>MLX4.BLI.5 (33)

routine RETRY (TIMES, COMMAND, LUN, WRDCNT, BUFFER, SECTOR) =
 begin !* 1 *

!+
 ROUTINE: RETRY(TIMES,COMMAND,LUN,WRDCNT,BUFFER,SECTOR)

- PURPOSE: (1) IF THE OPERATOR HAS ALLOWED ERROR PRINTOUTS, AND IF THE RETRY IS NOT TO CATEGORIZE ERRORS, THEN SAY THAT THE RETRY IS BEGINNING.
- (2) REISSUE THE WRITE, WRITE CHECK OR READ COMMAND UNTIL THE COMMAND SUCCEEDS OR UNTIL ALL PERMITTED RETRIES HAVE FAILED.
- (3) IF A 'BEGIN RETRY' MESSAGE WAS PRINTED, THEN A FINAL MESSAGE ABOUT THE SUCCESS OR FAILURE OF THE RETRIES SHOULD ALSO BE PRINTED.
- (4) INCREMENT THE RETRY COUNTER FOR EVERY RETRY DONE WHICH WAS NOT FOR ERROR CLASSIFICATION.

- ARGUMENTS: (1) TIMES - THE NUMBER OF RETRIES PERMITTED BEFORE IT IS CALLED A FAILURE.
- (2) OTHER ARGUMENTS ARE THE SAME AS FOR 'START_IT' ROUTINE ABOVE.

RESULTS: THE VALUES RETURNED FOR THIS SUBROUTINE ARE THE SAME AS FOR THE SYSERR ROUTINE ABOVE.

Label
 LOOP;

Local
 TEMP,
 VALUE,
 COUNT,
 TBOARD,
 TBANK;

!+
 !- THIS IS THE CODE FOR PURPOSE 1:
 !-

RETRYING = ACTIVE;

if ((.TIMES neq 1) and (.ERROUT))
 then
 begin !* 2 *

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (33)

```

6130 :MLX4
6131 :
6132 :
6133 : 4165 if .COMMAND eql write then PRINTB (SAY3, WRD2, WRD16, WRD18);
6134 : 4166 !BEGIN WRITE RETRY
6135 : 4167 !BEGIN WRITE RETRY
6136 : 4168
6137 : 4169 if .COMMAND eql CHECK then PRINTB (SAY3, WRD2, PHR1, WRD18);
6138 : 4170 !BEGIN WRITE CHECK RETRY
6139 : 4171 !BEGIN WRITE CHECK RETRY
6140 : 4172
6141 : 4173 if .COMMAND eql read then PRINTB (SAY3, WRD2, WRD17, WRD18);
6142 : 4174 !BEGIN READ RETRY
6143 : 4175 !BEGIN READ RETRY
6144 : 4176 end; !* 2 *
6145 : 4177
6146 : 4178 !+
6147 : 4179 ! THIS IS THE CODE FOR PURPOSE 2:
6148 : 4180 !-
6149 : 4181
6150 : 4182 LOOP :
6151 : 4183 begin !* 3 *
6152 : 4184
6153 : 4185 incr KOUNT from 1 to .TIMES do
6154 : 4186 begin !* 4 *
6155 : 4187
6156 : 4188 if .COMMAND eql write then VALUE = write (.LUN, .WRDCNT, .BUFFER, .SECTOR);
6157 : 4189
6158 : 4190 if .COMMAND eql CHECK
6159 : 4191 then
6160 : 4192 begin
6161 : 4193 write (.LUN, .WRDCNT, .BUFFER, .SECTOR);
6162 : 4194 VALUE = CHECK (.LUN, .WRDCNT, .BUFFER, .SECTOR);
6163 : 4195 end;
6164 : 4196
6165 : 4197 if .COMMAND eql read
6166 : 4198 then
6167 : 4199 begin
6168 : 4200 write (.LUN, .WRDCNT, (.BUFFER - BUFSIZ*2), .SECTOR);
6169 : 4201 VALUE = read (.LUN, .WRDCNT, .BUFFER, .SECTOR);
6170 : 4202 end;
6171 : 4203
6172 : 4204 !+
6173 : 4205 ! THIS IS THE CODE FOR PURPOSE 3:
6174 : 4206 !-
6175 : 4207
6176 : 4208 if .VALUE eql 0
6177 : 4209 then !THE RETRY WAS SUCCESSFUL
6178 : 4210 begin !* 5 *
6179 : 4211
6180 : 4212 if ((.TIMES neq 1) and (.ERROUT))
6181 : 4213 then
6182 : 4214 begin
6183 : 4215 PRINTB (SAY2, WRD18, WRD19);
6184 : 4216 PRINTB (CRLF);
    
```

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (33)

```

0186 :MLX4
6187 :
6188 :
6189 :         4217             !RETRY SUCCEEDED
6190 :         4218             end;
6191 :         4219
6192 :         4220             COUNT = .KOUNT;
6193 :         4221             leave LOOP;
6194 :         4222             end;
6195 :         4223
6196 :         4224             end;
6197 :         4225
6198 :         4226             !+
6199 :         4227             ! FALLS THROUGH HERE IF ALL RETRIES FAILED:
6200 :         4228             !-
6201 :         4229
6202 :         4230             if ((.TIMES neq 1) and (.ERROUT))
6203 :         4231             then
6204 :         4232             begin
6205 :         4233             PRINTB (SAY2, WRD18, WRD20);
6206 :         4234
6207 :         4235             ! VER CZMLBB COMMENTED OUT THIS PRINT CRLF
6208 :         4236             PRINTB (CRLF);
6209 :         4237             !RETRY FAILED
6210 :         4238             COUNT = .TIMES;
6211 :         4239             end;
6212 :         4240
6213 :         4241             end;
6214 :         4242
6215 :         4243             !+
6216 :         4244             ! THIS IS THE CODE FOR PURPOSE 4:
6217 :         4245             !-
6218 :         4246
6219 :         4247             if .TIMES neq 1
6220 :         4248             then
6221 :         4249             begin
6222 :         4250             TBOARD = .BOARD;
6223 :         4251             TBANK = .BANK;
6224 :         4252             DECODE (.SECTOR);
6225 :         4253             TRIES [.LUN, .BOARD, 0, 16, 0] = .TRIES [.LUN, .BOARD, 0, 16, 0] + .COUNT;
6226 :         4254             BOARD = .TBOARD;
6227 :         4255             BANK = .TBANK;
6228 :         4256             end;
6229 :         4257
6230 :         4258             RETRYING = INACTIVE;
6231 :         4259             return .VALUE;
6232 :         4260             end;
6233 :
6234 :
6235 :
6236 :
6237 :
    
```

.SBTTL RETRY ML11 TRANSFER COMMANDS

27-Mar-1982 19:24:42 TOPS
27-Mar-1982 19:23:44 PA:<

Address	Hex	Hex	Hex	Label	Command	Comments	Page
6242							
6243							
6244							
6245	046224	004167	137104				
6246	046230	012767	000001	166102	RETRY: JSR R1,\$SAVES		4113
6247	046236	016601	000030		MOV #1,RETRYING		4159
6248	046242	005046			MOV 30(SP),R1	: TIMES,*	4161
6249	046244	020127	000001		CLR -(SP)		
6250	046250	001472			CMP R1,#1		
6251	046252	005216			BEQ 3\$		
6252	046254	032767	000001	133776	INC (SP)		
6253	046262	001465			BIT #1,ERROUT		
6254	046264	016602	000030		BEQ 3\$: COMMAND,*	4165
6255	046270	020227	045424		MOV 30(SP),R2		
6256	046274	001016			CMP R2,#WRITE		
6257	046276	012746	007272		BNE 1\$		
6258	046302	012746	007256		MOV #WORD18,-(SP)		
6259	046306	012746	007200		MOV #WORD16,-(SP)		
6260	046312	012746	007112		MOV #WORD2,-(SP)		
6261	046316	012746	000004		MOV #SAY3,-(SP)		
6262	046322	010600			MOV #4,-(SP)		
6263	046324	104414			MOV SP,R0	: SP,*	
6264	046326	062706	000012		TRAP 14		
6265	046332	020227	046000	1\$:	ADD #12,SP		
6266	046336	001016			CMP R2,#CHECK		4169
6267	046340	012746	007272		BNE 2\$		
6268	046344	012746	007714		MOV #WORD18,-(SP)		
6269	046350	012746	007200		MOV #PHR1,-(SP)		
6270	046354	012746	007112		MOV #WORD2,-(SP)		
6271	046360	012746	000004		MOV #SAY3,-(SP)		
6272	046364	010600			MOV #4,-(SP)		
6273	046366	104414			MOV SP,R0	: SP,*	
6274	046370	062706	000012		TRAP 14		
6275	046374	020227	045612	2\$:	ADD #12,SP		
6276	046400	001016			CMP R2,#READ		4173
6277	046402	012746	007272		BNE 3\$		
6278	046406	012746	007264		MOV #WORD18,-(SP)		
6279	046412	012746	007200		MOV #WORD17,-(SP)		
6280	046416	012746	007112		MOV #WORD2,-(SP)		
6281	046422	012746	000004		MOV #SAY3,-(SP)		
6282	046426	010600			MOV #4,-(SP)		
6283	046430	104414			MOV SP,R0	: SP,*	
6284	046432	062706	000012		TRAP 14		
6285	046436	016602	000030	3\$:	ADD #12,SP		
6286	046442	005004			MOV 30(SP),R2	: COMMAND,*	4188
6287	046444	000543			CLR R4	: COUNT	4185
6288	046446	020227	045424	4\$:	BR 9\$		
6289	046452	001015			CMP R2,#WRITE		4188
6290	046454	016646	000026		BNE 5\$		
6291	046460	016646	000026		MOV 26(SP),-(SP)	: LUN,*	
6292	046464	016646	000026		MOV 26(SP),-(SP)	: WRDCNT,*	
6293	046470	016646	000026		MOV 26(SP),-(SP)	: BUFFER,*	
6294	046474	004767	176724		MOV 26(SP),-(SP)	: SECTOR,*	
6295	046500	010003			JSR PC,WRITE		
6296	046502	062706	000010		MOV R0,R3	: *,VALUE	
					ADD #10,SP		

Address	Hex	Hex	Hex	Label	Command	Comments	Address
6298				:MLX4			27-Mar-1982 19:24:42 TOPS
6299				:			27-Mar-1982 19:23:44 PA:<
6300					ML11 TRANSFER COMMANDS		
6301	046506	020227	046000	5\$:	CMP R2,#CHECK	:	4190
6302	046512	001027			BNE 6\$		
6303	046514	016646	000026		MOV 26(SP),-(SP)	: LUN,*	4193
6304	046520	016646	000026		MOV 26(SP),-(SP)	: WRDCNT,*	
6305	046524	016646	000026		MOV 26(SP),-(SP)	: BUFFER,*	
6306	046530	016646	000026		MOV 26(SP),-(SP)	: SECTOR,*	
6307	046534	004767	176664		JSR PC,WRITE		
6308	046540	016616	000036		MOV 36(SP),(SP)	: LUN,*	4194
6309	046544	016646	000034		MOV 34(SP),-(SP)	: WRDCNT,*	
6310	046550	016646	000034		MOV 34(SP),-(SP)	: BUFFER,*	
6311	046554	016646	000034		MOV 34(SP),-(SP)	: SECTOR,*	
6312	046560	004767	177214		JSR PC,CHECK		
6313	046564	010003			MOV R0,R3	: *,VALUE	
6314	046566	062706	000016		ADD #16,SP		4192
6315	046572	020227	045612	6\$:	CMP R2,#READ	:	4197
6316	046576	001031			BNE 7\$		
6317	046600	016646	000026		MOV 26(SP),-(SP)	: LUN,*	4200
6318	046604	016646	000026		MOV 26(SP),-(SP)	: WRDCNT,*	
6319	046610	016646	000026		MOV 26(SP),-(SP)	: BUFFER,*	
6320	046614	162716	010000		SUB #10000,(SP)		
6321	046620	016646	000026		MOV 26(SP),-(SP)	: SECTOR,*	
6322	046624	004767	17574		JSR PC,WRITE		
6323	046630	016616	000036		MOV 36(SP),(SP)	: LUN,*	4201
6324	046634	016646	000034		MOV 34(SP),-(SP)	: WRDCNT,*	
6325	046640	016646	000034		MOV 34(SP),-(SP)	: BUFFER,*	
6326	046644	016646	000034		MOV 34(SP),-(SP)	: SECTOR,*	
6327	046650	004767	176736		JSR PC,READ		
6328	046654	010003			MOV R0,R3	: *,VALUE	
6329	046656	062706	000016		ADD #16,SP		4199
6330	046662	005703		7\$:	TST R3	: VALUE	4208
6331	046664	001033			BNE 9\$		
6332	046666	032716	000001		BIT #1,(SP)	:	4212
6333	046672	001426			BEQ 8\$		
6334	046674	032767	000001	133356	BIT #1,ERRRUT		
6335	046702	001422			BEQ 8\$		
6336	046704	012746	007300		MOV #WRD19,-(SP)	:	4215
6337	046710	012746	007272		MOV #WRD18,-(SP)		
6338	046714	012746	007100		MOV #SAY2,-(SP)		
6339	046720	012746	000003		MOV #3,-(SP)		
6340	046724	010600			MOV SP,R0	: SP,*	
6341	046726	104414			TRAP 14		
6342	046730	012716	007066		MOV #CRLF,(SP)	:	4216
6343	046734	012746	000001		MOV #1,-(SP)		
6344	046740	010600			MOV SP,R0	: SP,*	
6345	046742	104414			TRAP 14		
6346	046744	062706	000012		ADD #12,SP		4214
6347	046750	010405		8\$:	MOV R4,R5	: KOUNT,COUNT	4220
6348	046752	000427			BR 10\$		4221
6349	046754	005204		9\$:	INC R4	: KOUNT	4185
6350	046756	020401			CMP R4,R1	: KOUNT,*	
6351	046760	003632			BLE 4\$		
6352	046762	032716	000001		BIT #1,(SP)	:	4230

6354					:MLX4				
6355					:		ML11 TRANSFER COMMANDS		
6356									
6357	046766	001454				BEQ	11\$		
6358	046770	032767	000001	133262		BIT	#1,ERROUT		
6359	046776	001415				BEQ	10\$		
6360	047000	012746	007312			MOV	#WRD20,-(SP)	:	4233
6361	047004	012746	007272			MOV	#WRD18,-(SP)		
6362	047010	012746	007100			MOV	#SAY2,-(SP)		
6363	047014	012746	000003			MOV	#3,-(SP)		
6364	047020	010600				MOV	SP,R0	: SP,*	
6365	047022	104414				TRAP	14		
6366	047024	010105				MOV	R1,R5	: *,COUNT	4238
6367	047026	062706	000010			ADD	#10,SP	:	4232
6368	047032	032716	000001		10\$:	BIT	#1,(SP)	:	4247
6369	047036	001430				BEQ	11\$		
6370	047040	016701	165276			MOV	BOARD,R1	: *,TBOARD	4250
6371	047044	016702	165274			MOV	BANK,R2	: *,TBANK	4251
6372	047050	016646	000020			MOV	20(SP),-(SP)	: SECTOR,*	4252
6373	047054	004767	170116			JSR	PC,DECODE		
6374	047060	016600	000030			MOV	30(SP),R0	: LUN,*	4253
6375	047064	006300				ASL	R0		
6376	047066	006300				ASL	R0		
6377	047070	006300				ASL	R0		
6378	047072	006300				ASL	R0		
6379	047074	066700	165242			ADD	BOARD,R0		
6380	047100	006300				ASL	R0		
6381	047102	060560	033714			ADD	R5,TRIES(R0)	: COUNT,*	
6382	047106	010167	165230			MOV	R1,BOARD	: TBOARD,*	4254
6383	047112	010267	165226			MOV	R2,BANK	: TBANK,*	4255
6384	047116	005726				TST	(SP)+	:	4249
6385	047120	005067	165214		11\$:	CLR	RETRYING	:	4258
6386	047124	010300				MOV	R3,R0	: VALUE,*	4114
6387	047126	005726				TST	(SP)+	:	4113
6388	047130	000207				RTS	PC		
6389									
6390									
6391									
6396									
6397									

: Routine Size: 227 words
: Maximum stack depth per invocation: 14 words

6399 :MLX4

27-Mar-1982 19:24:42

TOPS-20 Bliss-16 V2(212)

6400 :

TO SET UP BUFFER POINTERS BEFORE A TRANSFER

27-Mar-1982 19:23:44

PA:<NEALE>MLX4.BLI.5 (34)

6401 :

4261 %sbttl 'TO SET UP BUFFER POINTERS BEFORE A TRANSFER'

6403 :

4262 routine SET_PTRS (WRDCNT) : novalue =

6404 :

4263 begin

6405 :

4264

6406 :

4265

6407 :

4266

++ ROUTINE: SET_PTRS(WRDCNT)

6408 :

4267

6409 :

4268

PURPOSE: TO MAKE SURE THAT THE BUFFER POINTERS ARE SUITABLY PLACED BEFORE A TRANSFER, SO THAT THEY WILL SUPPORT THE CHOSEN WORD COUNT WITHIN THE BUFFER SPACE.

6410 :

4269

6411 :

4270

6412 :

4271

ARGUMENT: WRDCNT = THE NUMBER OF WORDS IN A TRANSFER.

6413 :

4272

6414 :

4273

RESULTS: 'WPTR' AND 'RPTR' ARE SET BACK TO THE START OF THE BUFFERS ONLY IF THE WORD COUNT WON'T FIT. OTHERWISE, THEY ARE LEFT UNCHANGED.

6415 :

4274

6416 :

4275

6417 :

4276

NOTE: THE WRITE BUFFER IS LOCATED BEFORE THE READ BUFFER.

6418 :

4277

6419 :

4278

6420 :

4279

6421 :

4280

if (.WPTR + .WRDCNT*2) geqa END_WBUFF then WPTR = WBUFF;

6422 :

4281

6423 :

4282

RPTR = .WPTR + (BUFSIZ*2);

6424 :

4283

return;

6425 :

4284

end;

6426 :

4285

6430

6431

.SBTTL SET_PTRS TO SET UP BUFFER POINTERS BEFORE A TRANSFER

6435

047132

SET_PTRS:

MOV 2(SP),R0 ; WRDCNT,*

4281

6436

047132

016600 000002

ASL R0

6437

047136

006300

ADD WPTR,R0

6438

047140

066700 163524

CMP R0,#END.WBUFF

6439

047144

020027 022670

BLO 1\$

6440

047150

103403

MOV #WBUFF,WPTR

6441

047152

012767 012670

163510

1\$:

MOV WPTR,R0 ;

4283

6442

047160

016700 163504

ADD #10000,R0 ;

6443

047164

062700 010000

MOV R0,RPTR ;

4262

6444

047170

010067 163476

RTS ;

6446

6447

6448

; Routine Size: 18 words

; Maximum stack depth per invocation: 0 words

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (35)

```

6454 :MLX4
6455 :
6456 :
6457 : 4286 %sbtll 'CHOOSING A WORD COUNT'
6458 : 4287 routine GET_WRCNT (SECTOR, LAST) =
6459 : 4288 begin
6460 : 4289
6461 : 4290 local
6462 : 4291 TEMP,
6463 : 4292 WRCNT;
6464 : 4293
6465 : 4294 TEMP = .LAST - .SECTOR + 1;
6466 : 4295
6467 : 4296
6468 : 4297
6469 : 4298
6470 : 4299
6471 : 4300 if (BUFSIZ/256) gtru .TEMP
6472 : 4301 then
6473 : 4302 WRCNT = .TEMP*256
6474 : 4303 else
6475 : 4304 WRCNT = BUFSIZ;
6476 : 4305
6477 : 4306 return .WRCNT;
6478 : end;
    
```

!NUMBER OF SECTORS LEFT TO TEST

! version czmlbb changed gtr to gtru

!LESS THAN A FULL BUFFER LEFT?

!YES -- USE AS LARGE A COUNT AS FITS

!NO -- USE THE ENTIRE BUFFER SIZE

6481								
6482				.SBTLL	GET.WRCNT CHOOSING A WORD COUNT			
6486	047176			GET.WRCNT:				
6487	047176	016600	000002	MOV	2(SP),R0	:	LAST,*	4294
6488	047202	166600	000004	SUB	4(SP),R0	:	SECTOR,*	
6489	047206	005200		INC	R0			
6490	047210	020027	000010	CMP	R0,#10	:	TEMP,*	4299
6491	047214	103003		BHIS	1\$			
6492	047216	000300		SWAB	R0	:		4301
6493	047220	105000		CLRB	R0			
6494	047222	000207		RTS	PC	:		4299
6495	047224	012700	004000	1\$: MOV	#4000,R0	:	*,WRCNT	4303
6496	047230	000207		RTS	PC	:		4287
6497								
6498								
6499								
6504								
6505								

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

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (36)

```

6507 :MLX4
6508 :
6509 :      COMMAND INTEGRITY ROUTINE
6510 :      4307 %sbtll 'COMMAND INTEGRITY ROUTINE'
6511 :      4308 routine INTEGRITY : novalue =
6512 :      4309      begin
6513 :      4310
6514 :      4311      +-
6515 :      4312      ROUTINE:      INTEGRITY
6516 :      4313
6517 :      4314      PURPOSE:      TO MAKE SURE THAT THE BASIC ML11 TRANSFER COMMANDS
6518 :      4315      WHICH WILL BE USED BY THE EXERCISER (WRITE,READ,
6519 :      4316      WRITE CHECK) WORK PROPERLY.
6520 :      4317
6521 :      4318      THE CODE FOR 'INTEGRITY' IN BRIEF:
6522 :      4319
6523 :      4320      BEGIN 1 (START OF ROUTINE)
6524 :      4321      SAY ROUTINE IS RUNNING
6525 :      4322      INCR COMPLEMENT FLAG FROM 0 TO 1
6526 :      4323      : BEGIN 2 (START OF COMPLEMENT FLAG SELECTION LOOP)
6527 :      4324      : GENERATE THE PATTERN
6528 :      4325      : INCR LOGICAL UNIT NUMBER FROM 0 TO LAST
6529 :      4326      : : BEGIN 3 (START OF LOGICAL UNIT SELECTION LOOP)
6530 :      4327      : : TESTLOOP:
6531 :      4328      : : : BEGIN 4 (START OF LOOP THAT COMPLETELY TESTS 1 UNIT)
6532 :      4329      : : : IF UNIT IS ACTIVE
6533 :      4330      : : : THEN
6534 :      4331      : : : : BEGIN 5 (START OF TEST FOR AN ACTIVE UNIT)
6535 :      4332      : : : : SECTOR = LOWEST
6536 :      4333      : : : : GET WRDCNT
6537 :      4334      : : : : WRITE
6538 :      4335      : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
6539 :      4336      : : : : READ
6540 :      4337      : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
6541 :      4338      : : : : WRITE CHECK
6542 :      4339      : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
6543 :      4340      : : : : END 5 (END OF TEST FOR AN ACTIVE UNIT)
6544 :      4341      : : : : END 4 (END OF LOOP THAT COMPLETELY TESTS 1 UNIT)
6545 :      4342      : : : END 3 (END OF LOGICAL UNIT SELECTION LOOP)
6546 :      4343      : : END 2 (END OF COMPLEMENT FLAG SELECTION LOOP)
6547 :      4344      RETURN
6548 :      4345      END 1 (END OF ROUTINE)
6549 :      4346      --
6550 :      4347
6551 :      4348      label
6552 :      4349      LOOP;
6553 :      4350
6554 :      4351      local
6555 :      4352      VALUE,
6556 :      4353      WRDCNT,
6557 :      4354      OLDSEC,
6558 :      4355      OLDCHN,
6559 :      4356      SECTOR,
6560 :      4357      DBL_VALUE;
6561 :      4358
    
```

257

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 v2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (36)

```

6563 :MLX4
6564 :
6565 :
6566 : 4359      PRINTB (SAY2, WRD34, RTNO);
6567 : 4360      !'RUNNING COMMAND INTEGRITY ROUTINE'
6568 : 4361
6569 : 4362      incr COMP_FLAG from 0 to 1 do
6570 : 4363      begin
6571 : 4364      GEN3 (.COMP_FLAG);
6572 : 4365
6573 : 4366      incr LUN from 0 to (.LSUNIT - 1) do
6574 : 4367      begin
6575 : 4368      LOOP :
6576 : 4369      begin
6577 : 4370
6578 : 4371      if .DRIVE_STATUS [.LUN] eql ACTIVE
6579 : 4372      then
6580 : 4373      begin
6581 : 4374      LSLUN = .LUN;
6582 : 4375      SECTOR = LOWEST;
6583 : 4376      WRDCNT = GET_WRDCNT (.SECTOR, HIGHEST);
6584 : 4377      VALUE = write (.LUN, .WRDCNT, WBUFF, LOWEST);
6585 : 4378
6586 : 4379      !+
6587 : 4380      ! SEE HOW SUCCESSFUL THE WRITE WAS:
6588 : 4381      !-
6589 : 4382
6590 : 4383      selectone .VALUE of
6591 : 4384      set
6592 : 4385
6593 : 4386      [1] :
6594 : 4387      begin
6595 : 4388
6596 : 4389      if RETRY (SIX, write, .LUN, .WRDCNT, WBUFF, LOWEST) neg 0
6597 : 4390      then
6598 : 4391      begin
6599 : 4392      WHY DROPT [.LUN] = CODE_4;
6600 : 4393      ERRDF (1, MSG1, 0); !*** INTEGRITY ROUTINE ERROR 01 ****
6601 : 4394      DODU (.LUN);
6602 : 4395      leave LOOP;
6603 : 4396      end;
6604 : 4397
6605 : 4398      !* 5A *
6606 : 4399
6607 : 4400      [2] :
6608 : 4401      begin
6609 : 4402      WHY DROPT [.LUN] = CODE_5;
6610 : 4403      ERRDF (2, MSG1, 0); !*** INTEGRITY ROUTINE ERROR 02 ****
6611 : 4404      DODU (.LUN);
6612 : 4405      leave LOOP;
6613 : 4406      end;
6614 : 4407
6615 : 4408      [3] :
6616 : 4409      begin
6617 : 4410      ERRDF (3, MSG1, 0); !* 5C * FATAL DRIVE ERROR -- NO RETRY ALLOWED
        !*** INTEGRITY ROUTINE ERROR 03 ****
    
```

```

6619 :MLX4
6620 :
6621 :
6622 : 4411
6623 : 4412
6624 : 4413
6625 : 4414
6626 : 4415
6627 : 4416
6628 : 4417
6629 : 4418
6630 : 4419
6631 : 4420
6632 : 4421
6633 : 4422
6634 : 4423
6635 : 4424
6636 : 4425
6637 : 4426
6638 : 4427
6639 : 4428
6640 : 4429
6641 : 4430
6642 : 4431
6643 : 4432
6644 : 4433
6645 : 4434
6646 : 4435
6647 : 4436
6648 : 4437
6649 : 4438
6650 : 4439
6651 : 4440
6652 : 4441
6653 : 4442
6654 : 4443
6655 : 4444
6656 : 4445
6657 : 4446
6658 : 4447
6659 : 4448
6660 : 4449
6661 : 4450
6662 : 4451
6663 : 4452
6664 : 4453
6665 : 4454
6666 : 4455
6667 : 4456
6668 : 4457
6669 : 4458
6670 : 4459
6671 : 4460
6672 : 4461
6673 : 4462

COMMAND INTEGRITY ROUTINE

WHY DROPT [.LUN] = CODE_6:
DODU (.LUN):
leave LOOP:
end:
!* 5C *
tes:

VALUE = read (.LUN, .WRDCNT, RBUFF, LOWEST);

!+
!- SEE HOW SUCCESSFUL THE READ WAS:
!-

selectone .VALUE of
set
!*SEE 'SYSERR' FOR DEFINITION
!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: 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 (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 ****
    
```

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (36)

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (36)

```
6675 :MLX4  
6676 :  
6677 :  
6678 : 4463  
6679 : 4464  
6680 : 4465  
6681 : 4466  
6682 : 4467  
6683 : 4468  
6684 : 4469  
6685 : 4470  
6686 : 4471  
6687 : 4472  
6688 : 4473  
6689 : 4474  
6690 : 4475  
6691 : 4476  
6692 : 4477  
6693 : 4478  
6694 : 4479  
6695 : 4480  
6696 : 4481  
6697 : 4482  
6698 : 4483  
6699 : 4484  
6700 : 4485  
6701 : 4486  
6702 : 4487  
6703 : 4488  
6704 : 4489  
6705 : 4490  
6706 : 4491  
6707 : 4492  
6708 : 4493  
6709 : 4494  
6710 : 4495  
6711 : 4496  
6712 : 4497  
6713 : 4498  
6714 : 4499  
6715 : 4500  
6716 : 4501  
6717 : 4502  
6718 : 4503  
6719 : 4504  
6720 : 4505  
6721 : 4506  
6722 : 4507  
6723 : 4508  
6724 : 4509  
6725 : 4510  
6726 : 4511  
6727 : 4512  
6728 : 4513  
6729 : 4514
```

```
COMMAND INTEGRITY ROUTINE  
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] :  
begin !* 5H * RECOVERABLE DATA ERROR  
ISOLATE ();  
if .ERROUT then PRINTB (FMT10B, .CHAN);  
!' BIT 00'  
OLDSEC = .MLEL;  
OLDCHN = .CHAN;  
if RETRY (ONE, read, .LUN, .WRDCNT, RBUFF, LOWEST) eql 5  
then  
if ((.MLEL eql .OLDSEC) and (.CHAN eql .OLDCHN))  
then  
begin  
if .ERROUT then ERRHRD (9, MSG4, 0);  
!**** INTEGRITY ROUTINE ERROR 09 ****  
UP_HARD_COUNT (.LUN, .BOARD);  
end  
else  
begin  
if .ERROUT then ERRSOFT (10, MSG3, 0);  
!**** INTEGRITY ROUTINE ERROR 10 ****  
UP_SOFT_COUNT (.LUN, .BOARD);  
end
```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (36)

6731 :MLX4
 6732 :
 6733 :
 6734 : 4515
 6735 : 4516
 6736 : 4517
 6737 : 4518
 6738 : 4519
 6739 : 4520
 6740 : 4521
 6741 : 4522
 6742 : 4523
 6743 : 4524
 6744 : 4525
 6745 : 4526
 6746 : 4527
 6747 : 4528
 6748 : 4529
 6749 : 4530
 6750 : 4531
 6751 : 4532
 6752 : 4533
 6753 : 4534
 6754 : 4535
 6755 : 4536
 6756 : 4537
 6757 : 4538
 6758 : 4539
 6759 : 4540
 6760 : 4541
 6761 : 4542
 6762 : 4543
 6763 : 4544
 6764 : 4545
 6765 : 4546
 6766 : 4547
 6767 : 4548
 6768 : 4549
 6769 : 4550
 6770 : 4551
 6771 : 4552
 6772 : 4553
 6773 : 4554
 6774 : 4555
 6775 : 4556
 6776 : 4557
 6777 : 4558
 6778 : 4559
 6779 : 4560
 6780 : 4561
 6781 : 4562
 6782 : 4563
 6783 : 4564
 6784 : 4565
 6785 : 4566

COMMAND INTEGRITY ROUTINE

```

else
begin
if .ERROUT then ERRSOFT (11, MSG3, 0);

!**** INTEGRITY ROUTINE ERROR 11 ****
UP_SOFT_COUNT (.LUN, .BOARD);
end;

end;          !* 5H *
tes;

VALUE = CHECK (.LUN, .WRDCNT, WBUFF, LOWEST);

!+
SEE HOW SUCCESSFUL THE WRITE CHECK WAS:
!-

selectone .VALUE of          !SEE 'SYSERR' FOR DEFINITION
set                          !OF ERROR # CONTAINED IN 'VALUE'

[1] :
begin                          !* 5I * RETRY ALLOWED
if RETRY (SIX, CHECK, .LUN, .WRDCNT, WBUFF, LOWEST) neg 0
then                          !THE RETRY FAILED -- SYSTEM FATAL ERROR
begin
WHY DROPT [.LUN] = CODE_4;
ERRDF (12, MSG1, 0);          !**** INTEGRITY ROUTINE ERROR 12 ****
DODU (.LUN);
leave LOOP;                  !JUMP JUST BEYOND END OF BLOCK * 4 *
end;

end;                          !* 5I *

[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 BEYOND 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] :
```

3
5
7
9
4

```

6787 :MLX4
6788 :
6789 :
6790 : 4567
6791 : 4568
6792 : 4569
6793 : 4570
6794 : 4571
6795 : 4572
6796 : 4573
6797 : 4574
6798 : 4575
6799 : 4576
6800 : 4577
6801 : 4578
6802 : 4579
6803 : 4580
6804 : 4581
6805 : 4582
6806 : 4583
6807 : 4584
6808 : 4585
6809 : 4586
6810 : 4587
6811 : 4588
6812 : 4589
6813 : 4590
6814 : 4591
6815 : 4592
6816 : 4593
6817 : 4594
6818 : 4595
6819 : 4596
6820 : 4597
6821 : 4598
6822 : 4599
6823 : 4600
6824 : 4601
6825 : 4602
6826 : 4603
6827 : 4604
6828 : 4605
6829 : 4606
6830 : 4607
6831 : 4608
6832 : 4609
6833 : 4610
6834 : 4611
6835 : 4612
6836 : 4613
6837 : 4614
6838 : 4615
6839 : 4616
6840 : 4617
6841 : 4618

COMMAND INTEGRITY ROUTINE

begin
ISOLATE ();
ERRDF (15, MSG2, 0);
WHY DROPT [.LUN] = CODE_7;
DODU (.LUN);
leave LOOP;
end;

[5] :
begin
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);

            UP HARD_COUNT (.LUN, .BOARD);
        end
    else
        begin
            if .ERROUT then ERRSOFT (17, MSG3, 0);

            UP SOFT_COUNT (.LUN, .BOARD);
        end
    else
        begin
            if .ERROUT then ERRSOFT (18, MSG3, 0);

            UP SOFT_COUNT (.LUN, .BOARD);
        end
    end;
end;

tes;

end;

```

```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (36)

!* 5L * UNRECOVERABLE DATA ERROR
!**** INTEGRITY ROUTINE ERROR 15 ****
!* JUMP JUST BEYOND END OF BLOCK * 4 *
!* 5L *

!* 5M * RECOVERABLE DATA ERROR

!**** INTEGRITY ROUTINE ERROR 16 ****
!**** INTEGRITY ROUTINE ERROR 17 ****
!**** INTEGRITY ROUTINE ERROR 18 ****

!* 5M *

!* 5 * END OF TEST FOR AN ACTIVE DRIVE

```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (36)

```

6843 :MLX4
6844 :
6845 :
6846 :         4619
6847 :         4620
6848 :         4621
6849 :         4622
6850 :         4623
6851 :         4624
6852 :         4625
6853 :         4626
6857 :
6858 :
    
```

COMMAND INTEGRITY ROUTINE

```

!* 4 * END OF LOOP THAT COMPLETELY TESTS 1 UNIT
!* 3 * END OF LOGICAL UNIT SELECTION LOOP
!* 2 * END OF COMPLEMENT FLAG SELECTION LOOP
!* 1 * END OF ROUTINE
    
```

```

6862 047232
6863 047232 004167 136076
6864 047236 162706 000012
6865 047242 012746 007574
6866 047246 012746 007344
6867 047252 012746 007100
6868 047256 012746 000003
6869 047262 010600
6870 047264 104414
6871 047266 005001
6872 047270 010146
6873 047272 004767 172170
6874 047276 016766 132510 000020
6875 047304 005004
6876 047306 000167 002110
6877 047312 010400
6878 047314 006200
6879 047316 006200
6880 047320 006200
6881 047322 062700 034442
6882 047326 010046
6883 047330 010446
6884 047332 042716 177770
6885 047336 012746 000001
6886 047342 005046
6887 047344 004767 135006
6888 047350 062706 000010
6889 047354 005300
6890 047356 001117
6891 047360 010467 132510
6892 047364 010400
6893 047366 006300
6894 047370 012702 034460
6895 047374 060002
6896 047376 011266 000022
6897 047402 016646 000022
    
```

.SBTTL INTEGRITY COMMAND INTEGRITY ROUTINE

```

INTEGRITY:
JSR R1,$SAVE5 ; 4308
SUB #12,SP
MOV #RTN0,-(SP) ; 4359
MOV #WRD34,-(SP)
MOV #SAY2,-(SP)
MOV #3,-(SP)
MOV SP,R0 ; SP,*
TRAP 14
CLR R1 ; COMP.FLAG 4362
MOV R1,-(SP) ; COMP.FLAG,* 4364
JSR PC,GEN3
MOV L$UNIT,20(SP) ; 4366
CLR R4 ; LUN
JMP 38$
2$: MOV R4,R0 ; LUN,* 4371
ASR R0
ASR R0
ASR R0
ADD #DRIVE.STATUS,R0
MOV R0,-(SP)
MOV R4,-(SP) ; LUN,*
BIC #177770,(SP)
MOV #1,-(SP)
CLR -(SP)
JSR PC,BL$GT2
ADD #10,SP
DEC R0
BNE 6$
MOV R4,L$LUN ; LUN,* 4374
MOV R4,R0 ; LUN,* 4375
ASL R0
MOV #LOW,SECT,R2
ADD R0,R2
MOV (R2),22(SP) ; *,SECTOR
MOV 22(SP),-(SP) ; SECTOR,* 4376
    
```


Address	Hex	Hex	Hex	Label	Instruction	Comments	Value
6899				:MLX4			
6900				:			
6901					COMMAND INTEGRITY ROUTINE		
6902	047406	016046	034500		MOV TOP,SECT(R0),-(SP)		
6903	047412	004767	177560		JSR PC,GET.WRDCNT		
6904	047416	010003			MOV R0,R3	: *,WRDCNT	
6905	047420	010416			MOV R4,(SP)	: LUN,*	4377
6906	047422	010346			MOV R3,-(SP)	: WRDCNT,*	
6907	047424	012746	012670		MOV #B,BUFF,-(SP)		
6908	047430	011246			MOV (R2),-(SP)		
6909	047432	004767	175766		JSR PC,WRITE		
6910	047436	010005			MOV R0,R5	: *,VALUE	
6911	047440	020527	000001		CMP R5,#1	: VALUE,*	4383
6912	047444	001031			BNE 3\$		
6913	047446	012746	000006		MOV #6,-(SP)	:	4389
6914	047452	012746	045424		MOV #WRITE,-(SP)		
6915	047456	010446			MOV R4,-(SP)	: LUN,*	
6916	047460	010346			MOV R3,-(SP)	: WRDCNT,*	
6917	047462	012746	012670		MOV #B,BUFF,-(SP)		
6918	047466	011246			MOV (R2),-(SP)		
6919	047470	004767	176530		JSR PC,RETRY		
6920	047474	062706	000014		ADD #14,SP		
6921	047500	005700			TST R0		
6922	047502	001446			BEQ 7\$		
6923	047504	112764	000004 034446		MOVB #4,WHY.DROPT(R4)	: *,*(LUN)	4392
6924	047512	104455			TRAP 55	:	4393
6925	047514	000001			.WORD 1		
6926	047516	011070			.WORD MSG1		
6927	047520	000000			.WORD 0		
6928	047522	010400			MOV R4,R0	: LUN,*	4394
6929	047524	104451			TRAP 51		
6930	047526	000431			BR 5\$		4395
6931	047530	020527	000002	3\$:	CMP R5,#2	: VALUE,*	4383
6932	047534	001012			BNE 4\$		
6933	047536	112764	000005 034446		MOVB #5,WHY.DROPT(R4)	: *,*(LUN)	4402
6934	047544	104455			TRAP 55	:	4403
6935	047546	000002			.WORD 2		
6936	047550	011070			.WORD MSG1		
6937	047552	000000			.WORD 0		
6938	047554	010400			MOV R4,R0	: LUN,*	4404
6939	047556	104451			TRAP 51		
6940	047560	000414			BR 5\$		4405
6941	047562	020527	000003	4\$:	CMP R5,#3	: VALUE,*	4383
6942	047566	001014			BNE 7\$		
6943	047570	104455			TRAP 55	:	4410
6944	047572	000003			.WORD 3		
6945	047574	011070			.WORD MSG1		
6946	047576	000000			.WORD 0		
6947	047600	112764	000006 034446		MOVB #6,WHY.DROPT(R4)	: *,*(LUN)	4411
6948	047606	010400			MOV R4,R0	: LUN,*	4412
6949	047610	104451			TRAP 51		
6950	047612	062706	000012	5\$:	ADD #12,SP	:	4413
6951	047616	000502		6\$:	BR 8\$		
6952	047620	010416		7\$:	MOV R4,(SP)	: LUN,*	4417
6953	047622	010346			MOV R3,-(SP)	: WRDCNT,*	

Address	Hex	Hex	Hex	Label	Command	Comments	Address
6955				:MLX4			
6956				:	COMMAND INTEGRITY ROUTINE		
6957							
6958	047624	012746	022670	MOV	#RBUF,-(SP)		
6959	047630	011246		MOV	(R2),-(SP)		
6960	047632	004767	175754	JSR	PC,READ		
6961	047636	010005		MOV	R0,R5	: *.VALUE	
6962	047640	001072		BNE	9\$:	4423
6963	047642	012746	012670	MOV	#RBUF,-(SP)	:	4428
6964	047646	012746	022670	MOV	#RBUF,-(SP)		
6965	047652	010346		MOV	R3,-(SP)	: WRDCNT,*	
6966	047654	004767	174446	JSR	PC,DOUBLE.CHECK		
6967	047660	062706	000006	ADD	#6,SP		
6968	047664	010066	000032	MOV	R0,32(SP)	: *.DBL.VALUE	
6969	047670	001477		BEQ	10\$		
6970	047672	010446		MOV	R4,-(SP)	: LUN,*	4431
6971	047674	004767	165624	JSR	PC,SAYWHO		
6972	047700	012716	011216	MOV	#MSG5,(SP)	:	4432
6973	047704	012746	007072	MOV	#SAY1,-(SP)		
6974	047710	012746	000002	MOV	#2,-(SP)		
6975	047714	010600		MOV	SP,R0	: SP,*	
6976	047716	104414		TRAP	14		
6977	047720	016616	000040	MOV	40(SP),(SP)	: DBL.VALUE,*	4434
6978	047724	017646	000040	MOV	@40(SP),-(SP)	: DBL.VALUE,*	
6979	047730	012746	006710	MOV	#FMT12A,-(SP)		
6980	047734	012746	000003	MOV	#3,-(SP)		
6981	047740	010600		MOV	SP,R0	: SP,*	
6982	047742	104414		TRAP	14		
6983	047744	062766	010000 000046	ADD	#10000,46(SP)	: *.DBL.VALUE	4436
6984	047752	016616	000046	MOV	46(SP),(SP)	: DBL.VALUE,*	4437
6985	047756	017646	000046	MOV	@46(SP),-(SP)	: DBL.VALUE,*	
6986	047762	012746	006756	MOV	#FMT12B,-(SP)		
6987	047766	012746	000003	MOV	#3,-(SP)		
6988	047772	010600		MOV	SP,R0	: SP,*	
6989	047774	104414		TRAP	14		
6990	047776	112764	000010 034446	MOVB	#10,WHY.DROPT(R4)	: *,*(LUN)	4439
6991	050004	104455		TRAP	55	:	4440
6992	050006	000004		.WORD	4		
6993	050010	011070		.WORD	MSG1		
6994	050012	000000		.WORD	0		
6995	050014	010400		MOV	R4,R0	: LUN,*	4441
6996	050016	104451		TRAP	51		
6997	050020	062706	000042	ADD	#42,SP	:	4442
6998	050024	000510		BR	16\$		
6999	050026	020527	000001	8\$: 9\$: CMP	R5,#1	: VALUE,*	4423
7000	050032	001033		BNE	12\$		
7001	050034	012746	000006	MOV	#6,-(SP)		4448
7002	050040	012746	045612	MOV	#READ,-(SP)		
7003	050044	010446		MOV	R4,-(SP)	: LUN,*	
7004	050046	010346		MOV	R3,-(SP)	: WRDCNT,*	
7005	050050	012746	022670	MOV	#RBUF,-(SP)		
7006	050054	011246		MOV	(R2),-(SP)		
7007	050056	004767	176142	JSR	PC,RETRY		
7008	050062	062706	000014	ADD	#14,SP		
7009	050066	005700		TST	R0		

Address	OpCode	Operand 1	Operand 2	Label	Instruction	Comment	Address
7011					:MLX4		
7012						COMMAND INTEGRITY ROUTINE	
7013							
7014	050070	001002		10\$:	BNE	11\$	
7015	050072	000167	000520		JMP	24\$	
7016	050076	112764	000004	034446	11\$:	MOV B #4,WHY.DROPT(R4)	: *,*(LUN)
7017	050104	104455			TRAP	55	: 4451
7018	050106	000005			.WORD	5	: 4452
7019	050110	011070			.WORD	MSG1	
7020	050112	000000			.WORD	0	
7021	050114	010400			MOV	R4,R0	: LUN,*
7022	050116	104451			TRAP	51	: 4453
7023	050120	000450			BR	15\$: 4454
7024	050122	020527	000002	12\$:	CMP	R5,#2	: VALUE,*
7025	050126	001012			BNE	13\$: 4423
7026	050130	112764	000005	034446	MOV B #5,WHY.DROPT(R4)		: *,*(LUN)
7027	050136	104455			TRAP	55	: 4461
7028	050140	000006			.WORD	6	: 4462
7029	050142	011070			.WORD	MSG1	
7030	050144	000000			.WORD	0	
7031	050146	010400			MOV	R4,R0	: LUN,*
7032	050150	104451			TRAP	51	: 4463
7033	050152	000433			BR	15\$: 4464
7034	050154	020527	000003	13\$:	CMP	R5,#3	: VALUE,*
7035	050160	001012			BNE	14\$: 4423
7036	050162	104455			TRAP	55	: 4469
7037	050164	000007			.WORD	7	
7038	050166	011070			.WORD	MSG1	
7039	050170	000000			.WORD	0	
7040	050172	112764	000006	034446	MOV B #6,WHY.DROPT(R4)		: *,*(LUN)
7041	050200	010400			MOV	R4,R0	: LUN,*
7042	050202	104451			TRAP	51	: 4470
7043	050204	000416			BR	15\$: 4471
7044	050206	020527	000004	14\$:	CMP	R5,#4	: VALUE,*
7045	050212	001017			BNE	17\$: 4472
7046	050214	004767	167112		JSR	PC,ISOLATE	: 4477
7047	050220	104455			TRAP	55	: 4478
7048	050222	000010			.WORD	10	
7049	050224	011120			.WORD	MSG2	
7050	050226	000000			.WORD	0	
7051	050230	112764	000007	034446	MOV B #7,WHY.DROPT(R4)		: *,*(LUN)
7052	050236	010400			MOV	R4,R0	: LUN,*
7053	050240	104451			TRAP	51	: 4480
7054	050242	062706	000020	15\$:	ADD	#20,SP	: 4481
7055	050246	000167	001146	16\$:	JMP	37\$: 4423
7056	050252	020527	000005	17\$:	CMP	R5,#5	: VALUE,*
7057	050256	001157			BNE	24\$: 4486
7058	050260	004767	167046		JSR	PC,ISOLATE	: 4488
7059	050264	032767	000001	131766	BIT	#1,ERRRUT	: 4488
7060	050272	001423			BEQ	18\$	
7061	050274	017700	164110		MOV	#ML.REG+42,R0	
7062	050300	006200			ASR	R0	
7063	050302	006200			ASR	R0	
7064	050304	006200			ASR	R0	
7065	050306	006200			ASR	R0	

27-Mar-1982 19:24:42 TOPS
 27-Mar-1982 19:23:44 PA:<

Address	Op	Op1	Op2	Op3	Op4	Op5	Op6	Op7	Op8	Op9	Op10	Op11	Op12	Op13	Op14	Op15	Op16	Op17	Op18	Op19	Op20	Op21	Op22	Op23	Op24	Op25	
7067																											
7068																											
7069																											
7070	050310	006200																									
7071	050312	006200																									
7072	050314	042700	177700																								
7073	050320	010046																									
7074	050322	012746	006640																								
7075	050326	012746	000002																								
7076	050332	010600																									
7077	050334	104414																									
7078	050336	062706	000006																								
7079	050342	017766	164044	000036	18\$:																						
7080	050350	017700	164034																								
7081	050354	006200																									
7082	050356	006200																									
7083	050360	006200																									
7084	050362	006200																									
7085	050364	006200																									
7086	050366	006200																									
7087	050370	042700	177700																								
7088	050374	010066	000034																								
7089	050400	012746	000001																								
7090	050404	012746	045612																								
7091	050410	010446																									
7092	050412	010346																									
7093	050414	012746	022670																								
7094	050420	011246																									
7095	050422	004767	175576																								
7096	050426	062706	000014																								
7097	050432	020027	000005																								
7098	050436	001051																									
7099	050440	027766	163746	000036																							
7100	050446	001034																									
7101	050450	016646	000034																								
7102	050454	017700	163730																								
7103	050460	006200																									
7104	050462	006200																									
7105	050464	006200																									
7106	050466	006200																									
7107	050470	006200																									
7108	050472	006200																									
7109	050474	042700	177700																								
7110	050500	020026																									
7111	050502	001016																									
7112	050504	032767	000001	131546																							
7113	050512	001404																									
7114	050514	104456																									
7115	050516	000011																									
7116	050520	011202																									
7117	050522	000000																									
7118	050524	010446			19\$:																						
7119	050526	016746	163610																								
7120	050532	004767	166664																								
7121	050536	000426																									

4491
4492

4494

4497

4501

4504

4497

7
PS
:<
33
58
32
47
50
51
52
53
54
55
59
58
14
3

Address	Instruction	Operand	Label	Hex	Symbol	Op	Comments	Seq
7123								
7124								
7125								
7126	050540	032767	000001	131512	20\$:	BIT	#1,ERROUT	4509
7127	050546	001415				BEQ	22\$	
7128	050550	104457				TRAP	57	
7129	050552	000012				.WORD	12	
7130	050554	011166				.WORD	MSG3	
7131	050556	000000				.WORD	0	
7132	050560	000410				BR	22\$	
7133	050562	032767	000001	131470	21\$:	BIT	#1,ERROUT	4512
7134	050570	001404				BEQ	22\$	4518
7135	050572	104457				TRAP	57	
7136	050574	000013				.WORD	13	
7137	050576	011166				.WORD	MSG3	
7138	050600	000000				.WORD	0	
7139	050602	010446			22\$:	MOV	R4,-(SP)	4521
7140	050604	016746	163532			MOV	BOARD,-(SP)	
7141	050610	004767	167270			JSR	PC,UP.SOFT.COUNT	
7142	050614	022626			23\$:	CMP	(SP)+,(SP)+	4525
7143	050616	010416			24\$:	MOV	R4,(SP)	4527
7144	050620	010346				MOV	R3,-(SP)	
7145	050622	012746	012670			MOV	#WBUF,-(SP)	
7146	050626	011246				MOV	(R2),-(SP)	
7147	050630	004767	175144			JSR	PC,CHECK	
7148	050634	010005				MOV	R0,R5	
7149	050636	020527	000001			CMP	R5,#1	4533
7150	050642	001031				BNE	25\$	
7151	050644	012746	000006			MOV	#6,-(SP)	4539
7152	050650	012746	046000			MOV	#CHECK,-(SP)	
7153	050654	010446				MOV	R4,-(SP)	
7154	050656	010346				MOV	R3,-(SP)	
7155	050660	012746	012670			MOV	#WBUF,-(SP)	
7156	050664	011246				MOV	(R2),-(SP)	
7157	050666	004767	175332			JSR	PC,RETRY	
7158	050672	062706	000014			ADD	#14,SP	
7159	050676	005700				TST	R0	
7160	050700	001462				BEQ	28\$	
7161	050702	112764	000004	034446		MOVB	#4,WHY.DROPT(R4)	4542
7162	050710	104455				TRAP	55	4543
7163	050712	000014				.WORD	14	
7164	050714	011070				.WORD	MSG1	
7165	050716	000000				.WORD	0	
7166	050720	010400				MOV	R4,R0	4544
7167	050722	104451				TRAP	51	
7168	050724	000450				BR	28\$	4545
7169	050726	020527	000002		25\$:	CMP	R5,#2	4533
7170	050732	001012				BNE	26\$	
7171	050734	112764	000005	034446		MOVB	#5,WHY.DROPT(R4)	4552
7172	050742	104455				TRAP	55	4553
7173	050744	000015				.WORD	15	
7174	050746	011070				.WORD	MSG1	
7175	050750	000000				.WORD	0	
7176	050752	010400				MOV	R4,R0	4554
7177	050754	104451				TRAP	51	

1
2
3

Address	Instruction	Comments	Labels	Operation	Value	Address	Instruction	Comments	Labels	Operation	Value
7179											
7180											
7181											
7182	050756	000433		BR	28\$						4555
7183	050760	020527	000003	26\$: CMP	R5,#3					VALUE,*	4533
7184	050764	001012		BNE	27\$						
7185	050766	104455		TRAP	55						4560
7186	050770	000016		.WORD	16						
7187	050772	011070		.WORD	MSG1						
7188	050774	000000		.WORD	0						
7189	050776	112764	000006	034446	MOV B	#6,WHY.DROPT(R4)				*,*(LUN)	4561
7190	051004	010400		MOV	R4,RO					LUN,*	4562
7191	051006	104451		TRAP	51						
7192	051010	000416		BR	28\$						4563
7193	051012	020527	000004	27\$: CMP	R5,#4					VALUE,*	4533
7194	051016	001014		BNE	29\$						
7195	051020	004767	166306	JSR	PC,ISOLATE						4568
7196	051024	104455		TRAP	55						4569
7197	051026	000017		.WORD	17						
7198	051030	011120		.WORD	MSG2						
7199	051032	000000		.WORD	0						
7200	051034	112764	000007	034446	MOV B	#7,WHY.DROPT(R4)				*,*(LUN)	4570
7201	051042	010400		MOV	R4,RO					LUN,*	4571
7202	051044	104451		TRAP	51						
7203	051046	000562		28\$: BR	36\$						4572
7204	051050	020527	000005	29\$: CMP	R5,#5					VALUE,*	4533
7205	051054	001157		BNE	36\$						
7206	051056	004767	166250	JSR	PC,ISOLATE						4577
7207	051062	032767	000001	131170	BIT	#1,ERRROUT					4579
7208	051070	001423		BEQ	30\$						
7209	051072	017700	163312	MOV	@ML.REG+42,RO						
7210	051076	006200		ASR	RO						
7211	051100	006200		ASR	RO						
7212	051102	006200		ASR	RO						
7213	051104	006200		ASR	RO						
7214	051106	006200		ASR	RO						
7215	051110	006200		ASR	RO						
7216	051112	042700	177700	BIC	#177700,RO						
7217	051116	010046		MOV	RO,-(SP)						
7218	051120	012746	006640	MOV	#FMT10B,-(SP)						
7219	051124	012746	000002	MOV	#2,-(SP)						
7220	051130	010600		MOV	SP,RO					SP,*	
7221	051132	104414		TRAP	14						
7222	051134	062706	000006	ADD	#6,SP						
7223	051140	017766	163246	000044	30\$: MOV	@ML.REG+44,44(SP)				*,OLDSEC	4582
7224	051146	017700	163236	MOV	@ML.REG+42,RO						4583
7225	051152	006200		ASR	RO						
7226	051154	006200		ASR	RO						
7227	051156	006200		ASR	RO						
7228	051160	006200		ASR	RO						
7229	051162	006200		ASR	RO						
7230	051164	006200		ASR	RO						
7231	051166	042700	177700	BIC	#177700,RO						
7232	051172	010066	000042	MOV	RO,42(SP)					*,OLDCHN	
7233	051176	012746	000001	MOV	#1,-(SP)						4585

Address	Instruction	Comment	Count	Address	Instruction	Comment	Count
7235		:MLX4					
7236		:			COMMAND INTEGRITY ROUTINE		
7237		:					
7238	051202	012746	046000		MOV #CHECK,-(SP)		
7239	051206	010446			MOV R4,-(SP)	: LUN,*	
7240	051210	010346			MOV R3,-(SP)	: WRDCNT,*	
7241	051212	012746	012670		MOV #WBUF,-(SP)		
7242	051216	011246			MOV (R2),-(SP)		
7243	051220	004767	175000		JSR PC,RETRY		
7244	051224	062706	000014		ADD #14,SP		
7245	051230	020027	000005		CMP R0,#5		
7246	051234	001051			BNE 33\$		
7247	051236	027766	163150	000044	CMP @ML.REG+44,44(SP)	: *,OLDSEC	4588
7248	051244	001034			BNE 32\$		
7249	051246	016600	000042		MOV 42(SP),R0	: OLDCHN,*	
7250	051252	017702	163132		MOV @ML.REG+42,R2		
7251	051256	006202			ASR R2		
7252	051260	006202			ASR R2		
7253	051262	006202			ASR R2		
7254	051264	006202			ASR R2		
7255	051266	006202			ASR R2		
7256	051270	006202			ASR R2		
7257	051272	042702	177700		BIC #177700,R2		
7258	051276	020200			CMP R2,R0		
7259	051300	001016			BNE 32\$		
7260	051302	032767	000001	130750	BIT #1,ERROUT	:	4592
7261	051310	001404			BEQ 31\$		
7262	051312	104456			TRAP 56		
7263	051314	000020			.WORD 20		
7264	051316	011202			.WORD MSG4		
7265	051320	000000			.WORD 0		
7266	051322	010446		31\$:	MOV R4,-(SP)	: LUN,*	4595
7267	051324	016746	163012		MOV BOARD,-(SP)		
7268	051330	004767	166066		JSR PC,UP.HARD.COUNT		
7269	051334	000426			BR 35\$:	4588
7270	051336	032767	000001	130714	32\$:	:	4600
7271	051344	001415			BIT #1,ERROUT	:	
7272	051346	104457			BEQ 34\$		
7273	051350	000021			TRAP 57		
7274	051352	011166			.WORD 21		
7275	051354	000000			.WORD MSG3		
7276	051356	000410			.WORD 0		
7277	051360	032767	000001	130672	33\$:	:	4603
7278	051366	001404			BR 34\$:	4609
7279	051370	104457			BIT #1,ERROUT	:	
7280	051372	000022			BEQ 34\$		
7281	051374	011166			TRAP 57		
7282	051376	000000			.WORD 22		
7283	051400	010446		34\$:	.WORD MSG3		
7284	051402	016746	162734		.WORD 0		
7285	051406	004767	166472		MOV R4,-(SP)	: LUN,*	4612
7286	051412	022626		35\$:	MOV BOARD,-(SP)		
7287	051414	062706	000026	36\$:	JSR PC,UP.SOFT.COUNT		
7288	051420	005204		37\$:	CMP (SP)+,(SP)+	:	4576
7289	051422	020466	000020	38\$:	ADD #26,SP	:	4373
					INC R4	: LUN	4366
					CMP R4,20(SP)	: LUN,*	

7291
7292
7293
7294 051426 002002
7295 051430 000167 175656
7296 051434 005726
7297 051436 005201
7298 051440 020127 000001
7299 051444 003002
7300 051446 000167 175616
7301 051452 062706 000022
7302 051456 000207
7303
7304
7305
7310
7311

```
      ;MLX4  
      ;  
      COMMAND INTEGRITY ROUTINE  
      BGE 39$  
      JMP 2$  
39$:  TST (SP)+  
      INC R1  
      CMP R1,#1  
      BGT 40$  
      JMP 1$  
40$:  ADD #22,SP  
      RTS PC  
  
      ; Routine Size: 587 words  
      ; Maximum stack depth per invocation: 33 words
```

27-Mar-1982 19:24:42 TOPS
27-Mar-1982 19:23:44 PA:<

4363
4362

4308

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (37)

```

7313 :MLX4
7314 :
7315 :
7316 : 4627 %sbttl 'DEFINITION OF OPTION 1'
7317 : 4628 routine OPT1 : novalue =
7318 : 4629 begin
7319 : 4630
7320 : 4631
7321 : 4632
7322 : 4633
7323 : 4634
7324 : 4635
7325 : 4636
7326 : 4637
7327 : 4638
7328 : 4639
7329 : 4640
7330 : 4641
7331 : 4642
7332 : 4643
7333 : 4644
7334 : 4645
7335 : 4646
7336 : 4647
7337 : 4648
7338 : 4649
7339 : 4650
7340 : 4651
7341 : 4652
7342 : 4653
7343 : 4654
7344 : 4655
7345 : 4656
7346 : 4657
7347 : 4658
7348 : 4659
7349 : 4660
7350 : 4661
7351 : 4662
7352 : 4663
7353 : 4664
7354 : 4665
7355 : 4666
7356 : 4667
7357 : 4668
7358 : 4669
7359 : 4670
7360 : 4671
7361 : 4672
7362 : 4673
7363 : 4674
7364 : 4675
7365 : 4676
7366 : 4677
7367 : 4678

DEFINITION OF OPTION 1

ROUTINE: 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)

Label
LOOP;

local
VALUE,
    
```

!* 1 * START OF ROUTINE

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (37)

```

7369 :MLX4
7370 :
7371 :
7372 :         4679         WRDCNT,
7373 :         4680         COMMAND,
7374 :         4681         PTR,
7375 :         4682         OLDSEC,
7376 :         4683         OLDCHN,
7377 :         4684         SECTOR,
7378 :         4685         DBL_VALUE;
7379 :         4686
7380 :         4687         PRINTB (SAY2, WRD34, RTN1);
7381 :         4688         !'RUNNING OPT1'
7382 :         4689
7383 :         4690         incr COMP_FLAG from 0 to 1 do
7384 :         4691         begin
7385 :         4692
7386 :         4693         incr LUN from 0 to (.LSUNIT - 1) do
7387 :         4694         begin
7388 :         4695         LOOP :
7389 :         4696         begin
7390 :         4697
7391 :         4698         if .DRIVE_STATUS [.LUN] eql ACTIVE
7392 :         4699         then
7393 :         4700         begin
7394 :         4701         L$LUN = .LUN;
7395 :         4702         WPTR = WBUFF;
7396 :         4703         RPTR = RBUFF;
7397 :         4704         SECTOR = LOWEST;
7398 :         4705
7399 :         4706         while .SECTOR lequ HIGHEST do
7400 :         4707         begin
7401 :         4708         GEN1 (.SECTOR, .COMP_FLAG);
7402 :         4709         WRDCNT = GET WRDCNT (.SECTOR, HIGHEST);
7403 :         4710         SET PTRS (.WRDCNT);
7404 :         4711         VALDE = write (.LUN, .WRDCNT, .WPTR, .SECTOR);
7405 :         4712
7406 :         4713         !+
7407 :         4714         !: SEE HOW SUCCESSFUL THE WRITE WAS:
7408 :         4715         !-
7409 :         4716
7410 :         4717         selectone .VALUE of
7411 :         4718         set
7412 :         4719
7413 :         4720         [1] :
7414 :         4721         begin
7415 :         4722
7416 :         4723         if RETRY (SIX, write, .LUN, .WRDCNT, .WPTR, .SECTOR) neg 0
7417 :         4724         then
7418 :         4725         begin
7419 :         4726         WHY DROPT [.LUN] = CODE_4;
7420 :         4727         ERRDF (101, MSG1, 0); !**** OPTION 1 ERROR 01 ****
7421 :         4728         DODU (.LUN);
7422 :         4729         leave LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
7423 :         4730         end;
    
```

!* 2 * START OF COMPLEMENT FLAG SELECTION LOOP

!* 3 * START OF LOGICAL UNIT SELECTION LOOP

!* 4 * START OF THE LOOP THAT COMPLETELY TESTS ONE UNIT

!* 5 * START OF TEST FOR AN ACTIVE UNIT

!* 6 * START OF SECTOR SELECTION LOOP

!* 6A * RETRY ALLOWED

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (37)

```

7425 :MLX4
7426 :
7427 :
7428 : 4731
7429 : 4732
7430 : 4733
7431 : 4734
7432 : 4735
7433 : 4736
7434 : 4737
7435 : 4738
7436 : 4739
7437 : 4740
7438 : 4741
7439 : 4742
7440 : 4743
7441 : 4744
7442 : 4745
7443 : 4746
7444 : 4747
7445 : 4748
7446 : 4749
7447 : 4750
7448 : 4751
7449 : 4752
7450 : 4753
7451 : 4754
7452 : 4755
7453 : 4756
7454 : 4757
7455 : 4758
7456 : 4759
7457 : 4760
7458 : 4761
7459 : 4762
7460 : 4763
7461 : 4764
7462 : 4765
7463 : 4766
7464 : 4767
7465 : 4768
7466 : 4769
7467 : 4770
7468 : 4771
7469 : 4772
7470 : 4773
7471 : 4774
7472 : 4775
7473 : 4776
7474 : 4777
7475 : 4778
7476 : 4779
7477 : 4780
7478 : 4781
7479 : 4782

DEFINITION OF OPTION 1

end;                !* 6A *

[2] :
begin                !* 6B * FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
WHY DROPT [.LUN] = CODE_5;
ERRDF (102, MSG1, 0); !**** OPTION 1 ERROR 02 ****
DODU (.LUN);
leave LOOP;          !JUMP JUST BEYOND END OF BLOCK * 4 *
end;                !* 6B *

[3] :
begin                !* 6C * FATAL DRIVE ERROR -- NO RETRY ALLOWED
ERRDF (103, MSG1, 0); !**** OPTION 1 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'

```

7481 :MLX4
 7482 :
 7483 :
 7484 :
 7485 :
 7486 :
 7487 :
 7488 :
 7489 :
 7490 :
 7491 :
 7492 :
 7493 :
 7494 :
 7495 :
 7496 :
 7497 :
 7498 :
 7499 :
 7500 :
 7501 :
 7502 :
 7503 :
 7504 :
 7505 :
 7506 :
 7507 :
 7508 :
 7509 :
 7510 :
 7511 :
 7512 :
 7513 :
 7514 :
 7515 :
 7516 :
 7517 :
 7518 :
 7519 :
 7520 :
 7521 :
 7522 :
 7523 :
 7524 :
 7525 :
 7526 :
 7527 :
 7528 :
 7529 :
 7530 :
 7531 :
 7532 :
 7533 :
 7534 :
 7535 :

DEFINITION OF OPTION 1

27-Mar-1982 19:24:42 PS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 P.:<NEALE>MLX4.BLI.5 (37)

```

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);
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 (106, MSG1, 0); !**** OPTION 1 ERROR 06 ****
DODU (.LUN);
leave LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
end; !* 6E *

[3] :
begin !* 6F * FATAL DRIVE ERROR -- NO RETRY ALLOWED
WHY DROPT [.LUN] = CODE_6;
ERRDF (107, MSG1, 0); !**** OPTION 1 ERROR 07 ****
DODU (.LUN);
leave LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
end; !* 6F *

[4] :
begin !* 6G * UNRECOVERABLE DATA ERROR
ISOLATE ();
ERRDF (108, MSG2, 0); !**** OPTION 1 ERROR 08 ****
WHY DROPT [.LUN] = CODE_7;
DODU (.LUN);
leave LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
end; !* 6G *
    
```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (37)

7537 :MLX4
 7538 :
 7539 :
 7540 :
 7541 :
 7542 :
 7543 :
 7544 :
 7545 :
 7546 :
 7547 :
 7548 :
 7549 :
 7550 :
 7551 :
 7552 :
 7553 :
 7554 :
 7555 :
 7556 :
 7557 :
 7558 :
 7559 :
 7560 :
 7561 :
 7562 :
 7563 :
 7564 :
 7565 :
 7566 :
 7567 :
 7568 :
 7569 :
 7570 :
 7571 :
 7572 :
 7573 :
 7574 :
 7575 :
 7576 :
 7577 :
 7578 :
 7579 :
 7580 :
 7581 :
 7582 :
 7583 :
 7584 :
 7585 :
 7586 :
 7587 :
 7588 :
 7589 :
 7590 :
 7591 :

DEFINITION OF OPTION 1

```
[5] :
begin                !* 6H * RECOVERABLE DATA ERROR
ISOLATE ();
if .ERROUT then PRINTB (FMT10B, .CHAN);
!' BIT 00'
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 (109, MSG4, 0);          !**** OPTION 1 ERROR 09 ****
            UP_HARD_COUNT (.LUN, .BOARD);
        end
    else
        begin
            if .ERROUT then ERRSOFT (110, MSG3, 0);          !**** OPTION 1 ERROR 10 ****
            UP_SOFT_COUNT (.LUN, .BOARD);
        end
    else
        begin
            if .ERROUT then ERRSOFT (111, MSG3, 0);          !**** OPTION 1 ERROR 11 ****
            UP_SOFT_COUNT (.LUN, .BOARD);
        end;
    end;
tes;                !* 6H *
WPTR = .WPTR + (.WRDCNT*2);
SECTOR = .SECTOR + (.WRDCNT/256);
end;                !* 6 * END OF SECTOR SELECTION LOOP
end;                !* 5 * END OF TEST FOR AN ACTIVE UNIT

!+
! Test to see if this uut's address space is
! to be read for soft errors. This test is
! is intended for DMT purposes.
!-
```

7593 :MLX4
 7594 :
 7595 :
 7596 :
 7597 :
 7598 :
 7599 :
 7600 :
 7601 :
 7602 :
 7603 :
 7604 :
 7605 :
 7606 :
 7607 :
 7608 :
 7609 :
 7610 :
 7611 :
 7612 :
 7613 :
 7614 :
 7615 :
 7616 :
 7617 :
 7618 :
 7619 :
 7620 :
 7621 :
 7622 :
 7623 :
 7624 :
 7625 :
 7626 :
 7627 :
 7628 :
 7629 :
 7630 :
 7631 :
 7632 :
 7633 :
 7634 :
 7635 :
 7636 :
 7637 :
 7638 :
 7639 :
 7640 :
 7641 :
 7642 :
 7643 :
 7644 : P
 7645 : P
 7646 :
 7647 :

DEFINITION OF OPTION 1

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (37)

```

    if .EFNS21
    then
    begin
        !Is the background pattern to be read

        : version czmlbb changed incr to incru
        incru SECTOR from LOWEST to HIGHEST do
            if read (.LUN, 256, RBUFF, .SECTOR) eql 5
            then
                begin
                    ISOLATE ();           !Find the failing bank and board no.
                    if .ERROUT then PRINTB (FMT10B, .CHAN);
                    ! Print where the error is
                    ! Save the contents of the ML error location
                    ! register so we can compare it to the new
                    ! contents of this register after the retry.
                    ! This is done to classify the error.
                    OLDSEC = .MLEL;
                    OLDCHN = .CHAN;
                    ! Do a classify retry call. If the same error
                    ! occurs then classify it as a hard error. If
                    ! a different error occurred or the error went away
                    ! then classify it as a soft error.
                    if RETRY (ONE, .COMMAND, .LUN, 256, .PTR, .OLDSEC) eql 5
                    then
                        ! The same error occurred so see if it is at the same
                        ! sector and channel number, if so then classify
                        ! it as a hard error else classify it as a soft
                        ! error.
                        if ((.MLEL eql .OLDSEC) and (.CHAN eql .OLDCHN))
                        then
                            begin
                                !Same error occurred 'hard'
                                if .ERROUT           !Print error if enabled
                                then
                                    begin
                                        ERRHRD (112,           !Error number
                                        MSG4,           !Error message
                                        0);           !Additional message routine
                                    end;
                            end;
                end;
    end;
    
```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (37)

```

7649 :MLX4
7650 :      DEFINITION OF OPTION 1
7651 :
7652 :      4939
7653 :      4940      UP_HARD_COUNT (.LUN, .BOARD);
7654 :      4941      end
7655 :      4942      else
7656 :      4943      begin          !Not the same error 'soft'
7657 :      4944
7658 :      4945      if .ERROUT      !Print error if enabled
7659 :      4946      then
7660 :      4947      begin
7661 :      4948      ERRSOFT (113,      !Error number
7662 :      4949      MSG3,      !Error message
7663 :      4950      0);      !Additional message routine
7664 :      4951      end;
7665 :      4952
7666 :      4953      UP_SOFT_COUNT (.LUN, .BOARD);
7667 :      4954      end
7668 :      4955
7669 :      4956      else          !Not the same error 'soft'
7670 :      4957      begin
7671 :      4958
7672 :      4959      if .ERROUT      !Print error if enabled
7673 :      4960      then
7674 :      4961      begin
7675 :      4962      ERRSOFT (114,      !Error number
7676 :      4963      MSG3,      !Error message
7677 :      4964      0);      !Additional message routine
7678 :      4965      end;
7679 :      4966
7680 :      4967      UP_SOFT_COUNT (.LUN, .BOARD);
7681 :      4968      end;
7682 :      4969
7683 :      4970      end;
7684 :      4971
7685 :      4972      end;
7686 :      4973
7687 :      4974      end;          !* 4 * END OF LOOP THAT COMPLETELY TESTS ONE UNIT
7688 :      4975      end;          !* 3 * END OF LOGICAL UNIT SELECTION LOOP
7689 :      4976
7690 :      4977      end;          !* 2 * END OF COMPLEMENT FLAG SELECTION LOOP
7691 :      4978
7692 :      4979      return;
7693 :      4980      end;          !* 1 * END OF ROUTINE
7697 :
7698 :
7702 051460 004167 133650      OPT1: .SBTTL OPT1 DEFINITION OF OPTION 1
7703 051464 162706 000020      JSR   R1,$SAVE5
      SUB   #20,SP
    
```


Address	Label	Instruction	Comments	Address	Instruction	Comments	Date/Time	Page
7761							27-Mar-1982 19:24:42	TOPS
7762	:MLX4						27-Mar-1982 19:23:44	PA:<
7763	:							
7764	051752	016646	000030		MOV	30(SP), -(SP)		: LUN,*
7765	051756	010246			MOV	R2, -(SP)		: WRDCNT,*
7766	051760	016746	160704		MOV	WPTR, -(SP)		
7767	051764	010446			MOV	R4, -(SP)		: SECTOR,*
7768	051766	004767	174232		JSR	PC, RETRY		
7769	051772	062706	000014		ADD	#14, SP		
7770	051776	005700			TST	R0		
7771	052000	001456			BEQ	9\$		
7772	052002	016601	000024		MOV	24(SP), R1		: LUN,*
7773	052006	112761	000004	034446	MOVB	#4, WHY.DROPT(R1)		4726
7774	052014	104455			TRAP	55		: 4727
7775	052016	000145			.WORD	145		
7776	052020	011070			.WORD	MSG1		
7777	052022	000000			.WORD	0		
7778	052024	016600	000024		MOV	24(SP), R0		: LUN,*
7779	052030	104451			TRAP	51		4728
7780	052032	000436			BR	8\$: 4729
7781	052034	020327	000002	6\$:	CMP	R3, #2		: VALUE,*
7782	052040	001015			BNE	7\$		4717
7783	052042	016601	000024		MOV	24(SP), R1		: LUN,*
7784	052046	112761	000005	034446	MOVB	#5, WHY.DROPT(R1)		4736
7785	052054	104455			TRAP	55		: 4737
7786	052056	000146			.WORD	146		
7787	052060	011070			.WORD	MSG1		
7788	052062	000000			.WORD	0		
7789	052064	016600	000024		MOV	24(SP), R0		: LUN,*
7790	052070	104451			TRAP	51		4738
7791	052072	000416			BR	8\$: 4739
7792	052074	020327	000003	7\$:	CMP	R3, #3		: VALUE,*
7793	052100	001016			BNE	9\$		4717
7794	052102	104455			TRAP	55		: 4744
7795	052104	000147			.WORD	147		
7796	052106	011070			.WORD	MSG1		
7797	052110	000000			.WORD	0		
7798	052112	016601	000024		MOV	24(SP), R1		: LUN,*
7799	052116	112761	000006	034446	MOVB	#6, WHY.DROPT(R1)		4745
7800	052124	010100			MOV	R1, R0		: LUN,*
7801	052126	104451			TRAP	51		4746
7802	052130	062706	000014	8\$:	ADD	#14, SP		: 4747
7803	052134	000543			BR	14\$		
7804	052136	004767	174024	9\$:	JSR	PC, CHOOSE		: 4751
7805	052142	010066	000034		MOV	R0, 34(SP)		: *,COMMAND
7806	052146	005001			CLR	R1		: 4753
7807	052150	020027	045612		CMP	R0, #READ		: COMMAND,*
7808	052154	001015			BNE	10\$		
7809	052156	005201			INC	R1		
7810	052160	016766	160506	000036	MOV	RPTR, 36(SP)		: *,PTR
7811	052166	016646	000024		MOV	24(SP), -(SP)		: LUN,*
7812	052172	010246			MOV	R2, -(SP)		: WRDCNT,*
7813	052174	016746	160472		MOV	RPTR, -(SP)		: 4757
7814	052200	010446			MOV	R4, -(SP)		: SECTOR,*
7815	052202	004767	173404		JSR	PC, READ		

Address	OpCode	Operand 1	Operand 2	Operand 3	Label	Instruction	Comments	Address	Time	Page
7817						BR 11\$		27-Mar-1982 19:24:42	TOPS	
7818						MOV WPTR,36(SP)	: *,PTR	27-Mar-1982 19:23:44	PA:<	4761
7819						MOV 24(SP),-(SP)	: LUN,*			4762
7820	052206	000413				MOV R2,-(SP)	: WRDCNT,*			
7821	052210	016766	160454	000036	10\$:	MOV WPTR,-(SP)	: SECTOR,*			
7822	052216	016646	000024			MOV R4,-(SP)	: *,VALUE			
7823	052222	010246				JSR PC,CHECK	:			
7824	052224	016746	160440			MOV R0,R3	: *,VALUE			
7825	052230	010446				BNE 15\$:			4769
7826	052232	004767	173542			ROR R1	:			4774
7827	052236	010003			11\$:	BLO 13\$:			
7828	052240	001102				JMP 28\$:			
7829	052242	006001				MOV WPTR,-(SP)	:			4778
7830	052244	103402				MOV RPTR,-(SP)	:			
7831	052246	000167	001016		12\$:	MOV R2,-(SP)	: WRDCNT,*			
7832	052252	016746	160412		13\$:	JSR PC,DOUBLE.CHECK	:			
7833	052256	016746	160410			ADD #6,SP	:			
7834	052262	010246				MOV R0,42(SP)	: *,DBL.VALUE			
7835	052264	004767	172036			BEQ 12\$:			
7836	052270	062706	000006			MOV 34(SP),-(SP)	: LUN,*			4781
7837	052274	010066	000042			JSR PC,SAYWHO	:			
7838	052300	001762				MOV #MSG5,(SP)	:			4782
7839	052302	016646	000034			MOV #SAY1,-(SP)	:			
7840	052306	004767	163212			MOV #2,-(SP)	: SP,*			
7841	052312	012716	011216			MOV SP,R0	: SP,*			
7842	052316	012746	007072			TRAP 14	:			
7843	052322	012746	000002			MOV 50(SP),(SP)	: DBL.VALUE,*			4783
7844	052326	010600				MOV 250(SP),-(SP)	: DBL.VALUE,*			
7845	052330	104414				MOV #FMT12A,-(SP)	:			
7846	052332	016616	000050			MOV #3,-(SP)	: SP,*			
7847	052336	017646	000050			MOV SP,R0	: SP,*			
7848	052342	012746	006710			TRAP 14	:			
7849	052346	012746	000003			ADD #10000,56(SP)	: *,DBL.VALUE			4785
7850	052352	010600				MOV 56(SP),(SP)	: DBL.VALUE,*			4786
7851	052354	104414				MOV 256(SP),-(SP)	: DBL.VALUE,*			
7852	052356	062766	010000	000056		MOV #FMT12B,-(SP)	:			
7853	052364	016616	000056			MOV #3,-(SP)	: SP,*			
7854	052370	017646	000056			MOV SP,R0	: SP,*			
7855	052374	012746	006756			TRAP 14	:			
7856	052400	012746	000003			ADD #10,WHY.DROPT(R1)	: LUN,*			4788
7857	052404	010600				TRAP 55	:			4789
7858	052406	104414				.WORD 150	:			
7859	052410	016601	000056			.WORD MSG1	:			
7860	052414	112761	000010	034446		.WORD 0	:			
7861	052422	104455				MOV 56(SP),R0	: LUN,*			4790
7862	052424	000150				TRAP 51	:			
7863	052426	011070				ADD #46,SP	:			4791
7864	052430	000000				BR 20\$: VALUE,*			4769
7865	052432	016600	000056			CMP R3,#1	:			
7866	052436	104451				BNE 16\$:			
7867	052440	062706	000046			MOV #6,-(SP)	:			4799
7868	052444	000522			14\$:					
7869	052446	020327	000001		15\$:					
7870	052452	001035								
7871	052454	012746	000006							

Address	OpCode	Operand 1	Operand 2	Operand 3	Label	Instruction	Comments	Date/Time	Page
7873								27-Mar-1982 19:24:42	TOPS
7874								27-Mar-1982 19:23:44	PA:<
7875									
7876	052466	016646	000046			MOV 46(SP),-(SP)	: COMMAND,*		
7877	052464	016646	000040			MOV 40(SP),-(SP)	: LUN,*		
7878	052470	010246				MOV R2,-(SP)	: WRDCNT,*		
7879	052472	016646	000056			MOV 56(SP),-(SP)	: PTR,*		
7880	052476	010446				MOV R4,-(SP)	: SECTOR,*		
7881	052500	004767	173520			JSR PC,RETRY			
7882	052504	062706	000014			ADD #14,SP			
7883	052510	005700				TST R0			
7884	052512	001655				BEQ 12\$			
7885	052514	016601	000034			MOV 34(SP),R1	: LUN,*		4802
7886	052520	112761	000004	034446		MOVB #4,WHY.DROPT(R1)			
7887	052526	104455				TRAP 55	:		4803
7888	052530	000151				.WORD 151			
7889	052532	011070				.WORD MSG1			
7890	052534	000000				.WORD 0			
7891	052536	016600	000034			MOV 34(SP),R0	: LUN,*		4804
7892	052542	104451				TRAP 51			
7893	052544	000460				BR 19\$:		4805
7894	052546	020327	000002	16\$:		CMP R3,#2	: VALUE,*		4769
7895	052552	001015				BNE 17\$			
7896	052554	016601	000034			MOV 34(SP),R1	: LUN,*		4812
7897	052560	112761	000005	034446		MOVB #5,WHY.DROPT(R1)			
7898	052566	104455				TRAP 55	:		4813
7899	052570	000152				.WORD 152			
7900	052572	011070				.WORD MSG1			
7901	052574	000000				.WORD 0			
7902	052576	016600	000034			MOV 34(SP),R0	: LUN,*		4814
7903	052602	104451				TRAP 51			
7904	052604	000440				BR 19\$:		4815
7905	052606	020327	000003	17\$:		CMP R3,#3	: VALUE,*		4769
7906	052612	001015				BPE 18\$			
7907	052614	016601	000034			MOV 34(SP),R1	: LUN,*		4820
7908	052620	112761	000006	034446		MOVB #6,WHY.DROPT(R1)			
7909	052626	104455				TRAP 55	:		4821
7910	052630	000153				.WORD 153			
7911	052632	011070				.WORD MSG1			
7912	052634	000000				.WORD 0			
7913	052636	016600	000034			MOV 34(SP),R0	: LUN,*		4822
7914	052642	104451				TRAP 51			
7915	052644	000420				BR 19\$:		4823
7916	052646	020327	000004	18\$:		CMP R3,#4	: VALUE,*		4769
7917	052652	001021				BNE 21\$			
7918	052654	004767	164452			JSR PC,ISOLATE	:		4828
7919	052660	104455				TRAP 55	:		4829
7920	052662	000154				.WORD 154			
7921	052664	011120				.WORD MSG2			
7922	052666	000000				.WORD 0			
7923	052670	016601	000034			MOV 34(SP),R1	: LUN,*		4830
7924	052674	112761	000007	034446		MOVB #7,WHY.DROPT(R1)			
7925	052702	010100				MOV R1,R0	: LUN,*		4831
7926	052704	104451				TRAP 51	:		
7927	052706	062706	000024	19\$:		ADD #24,SP	:		4832

Address	Instruction	Operand 1	Operand 2	Label	Comments	Date/Time	Page
7929							
7930							
7931							
7932	052712	000167	001070			27-Mar-1982 19:24:42	TOPS
7933	052716	020327	000005			27-Mar-1982 19:23:44	PA:<
7934	052722	001162					
7935	052724	004767	164402				
7936	052730	032767	000001	127322			4837
7937	052736	001423					4839
7938	052740	017701	161444				
7939	052744	006201					
7940	052746	006201					
7941	052750	006201					
7942	052752	006201					
7943	052754	006201					
7944	052756	006201					
7945	052760	042701	177700				
7946	052764	010146					
7947	052766	012746	006640				
7948	052772	012746	000002				
7949	052776	010600					
7950	053000	104414					
7951	053002	062706	000006				
7952	053006	017766	161400	000050	22\$:		4842
7953	053014	017701	161370				4843
7954	053020	006201					
7955	053022	006201					
7956	053024	006201					
7957	053026	006201					
7958	053030	006201					
7959	053032	006201					
7960	053034	042701	177700				
7961	053040	010166	000040				
7962	053044	012746	000001				
7963	053050	016646	000046				4845
7964	053054	016646	000040				
7965	053060	010246					
7966	053062	016646	000056				
7967	053066	010446					
7968	053070	004767	173130				
7969	053074	062706	000014				
7970	053100	020027	000005				
7971	053104	001052					
7972	053106	027766	161300	000050			4848
7973	053114	001035					
7974	053116	016600	000040				
7975	053122	017701	161262				
7976	053126	006201					
7977	053130	006201					
7978	053132	006201					
7979	053134	006201					
7980	053136	006201					
7981	053140	006201					
7982	053142	042701	177700				
7983	053146	020100					

7985				:MLX4				27-Mar-1982 19:24:42	TOPS
7986				:	DEFINITION OF OPTION 1			27-Mar-1982 19:23:44	PA:<
7987				:					
7988	053150	001017			BNE	24\$			
7989	053152	032767	000001	127100	BIT	#1,ERROUT	:		4852
7990	053160	001404			BEQ	23\$			
7991	053162	104456			TRAP	56			
7992	053164	000155			.WORD	155			
7993	053166	011202			.WORD	MSG4			
7994	053170	000000			.WORD	0			
7995	053172	016646	000034	23\$:	MOV	34(SP),-(SP)	:	LUN,*	4854
7996	053176	016746	161140		MOV	BOARD,-(SP)			
7997	053202	004767	164214		JSR	PC,UP.HARD.COUNT			
7998	053206	000427			BR	27\$:		4848
7999	053210	032767	000001	127042	24\$:	BIT	#1,ERROUT	:	4859
8000	053216	001415			BEQ	26\$			
8001	053220	104457			TRAP	57			
8002	053222	000156			.WORD	156			
8003	053224	011166			.WORD	MSG3			
8004	053226	000000			.WORD	0			
8005	053230	000410			BR	26\$:		4861
8006	053232	032767	000001	127020	25\$:	BIT	#1,ERROUT	:	4867
8007	053240	001404			BEQ	26\$			
8008	053242	104457			TRAP	57			
8009	053244	000157			.WORD	157			
8010	053246	011166			.WORD	MSG3			
8011	053250	000000			.WORD	0			
8012	053252	016646	000034	26\$:	MOV	34(SP),-(SP)	:	LUN,*	4869
8013	053256	016746	161060		MOV	BOARD,-(SP)			
8014	053262	004767	164616		JSR	PC,UP.SOFT.COUNT			
8015	053266	022626			27\$:	CMP	(SP)+,(SP)+	:	4836
8016	053270	010200			28\$:	MOV	R2,R0	:	4875
8017	053272	006300			ASL	R0			
8018	053274	066700	157370		ADD	WPTR,R0			
8019	053300	010067	157364		MOV	R0,WPTR			
8020	053304	010216			MOV	R2,(SP)	:	WRDCNT,*	4876
8021	053306	012746	000400		MOV	#400,-(SP)			
8022	053312	004767	131652		JSR	PC,BLSDIV			
8023	053316	060400			ADD	R4,R0	:	SECTOR,*	
8024	053320	010004			MOV	R0,R4	:	* ,SECTOR	
8025	053322	062706	000026		ADD	#26,SP	:		4707
8026	053326	000167	176316		JMP	5\$:		4706
8027	053332	032767	000001	126722	29\$:	BIT	#1,EFNS21	:	4887
8028	053340	001002			BNE	30\$			
8029	053342	000167	000440		JMP	40\$			
8030	053346	016600	000010	30\$:	MOV	10(SP),R0	:	LUN,*	4894
8031	053352	006300			ASL	R0			
8032	053354	016001	034500		MOV	TOP.SECT(R0),R1			
8033	053360	016005	034460		MOV	LOW.SECT(R0),R5	:	* ,SECTOR	
8034	053364	000167	000406		JMP	39\$			
8035	053370	016646	000010	31\$:	MOV	10(SP),-(SP)	:	LUN,*	4896
8036	053374	012746	000400		MOV	#400,-(SP)			
8037	053400	012746	022670		MOV	#RBUF,-(SP)			
8038	053404	010546			MOV	R5,-(SP)	:	SECTOR,*	
8039	053406	004767	172200		JSR	PC,READ			

8041											
8042				:MLX4							27-Mar-1982 19:24:42 TOPS
8043				:	DEFINITION OF OPTION 1						27-Mar-1982 19:23:44 PA:<
8044	053412	062706	000010		ADD #10,SP						
8045	053416	020027	000005		CMP R0,#5						
8046	053422	001164			BNE 38\$						
8047	053424	004767	163702		JSR PC,ISOLATE						4899
8048	053430	032767	000001	126622	BIT #1,ERROUT	:					4901
8049	053436	001423			BEQ 32\$:					
8050	053440	017700	160744		MOV @ML.REG+42,R0						
8051	053444	006200			ASR R0						
8052	053446	006200			ASR R0						
8053	053450	006200			ASR R0						
8054	053452	006200			ASR R0						
8055	053454	006200			ASR R0						
8056	053456	006200			ASR R0						
8057	053460	042700	177700		BIC #177700,R0						
8058	053464	010046			MOV R0,-(SP)						
8059	053466	012746	006640		MOV #FMT10B,-(SP)						
8060	053472	012746	000002		MOV #2,-(SP)						
8061	053476	010600			MOV SP,R0	:	SP,*				
8062	053500	104414			TRAP 14						
8063	053502	062706	000006		ADD #6,SP						
8064	053506	017766	160700	000024 32\$:	MOV @ML.REG+44,24(SP)	:	*,OLDSEC				4910
8065	053514	017700	160670		MOV @ML.REG+42,R0	:					4911
8066	053520	006200			ASR R0						
8067	053522	006200			ASR R0						
8068	053524	006200			ASR R0						
8069	053526	006200			ASR R0						
8070	053530	006200			ASR R0						
8071	053532	006200			ASR R0						
8072	053534	042700	177700		BIC #177700,R0						
8073	053540	010066	000014		MOV R0,14(SP)	:	*,OLDCHN				
8074	053544	012746	000001		MOV #1,-(SP)	:					4919
8075	053550	016646	000022		MOV 22(SP),-(SP)	:	COMMAND,*				
8076	053554	016646	000014		MOV 14(SP),-(SP)	:	LUN,*				
8077	053560	012746	000400		MOV #400,-(SP)	:					
8078	053564	016646	000032		MOV 32(SP),-(SP)	:	PTR,*				
8079	053570	016646	000036		MOV 36(SP),-(SP)	:	OLDSEC,*				
8080	053574	004767	172424		JSR PC,RETRY						
8081	053600	062706	000014		ADD #14,SP						
8082	053604	020027	000005		CMP R0,#5						
8083	053610	001052			BNE 35\$						
8084	053612	027766	160574	000024	CMP @ML.REG+44,24(SP)	:	*,OLDSEC				4928
8085	053620	001035			BNE 34\$						
8086	053622	016646	000014		MOV 14(SP),-(SP)	:	OLDCHN,*				
8087	053626	017700	160556		MOV @ML.REG+42,R0						
8088	053632	006200			ASR R0						
8089	053634	006200			ASR R0						
8090	053636	006200			ASR R0						
8091	053640	006200			ASR R0						
8092	053642	006200			ASR R0						
8093	053644	006200			ASR R0						
8094	053646	042700	177700		BIC #177700,R0						
8095	053652	020026			CMP R0,(SP)+						

Address	OpCode	Operand1	Operand2	Label	Instruction	Comments	Date/Time	Page
8097								
8098							27-Mar-1982 19:24:42	TOPS
8099							27-Mar-1982 19:23:44	PA:<
8100	053654	001017			BNE 34\$			
8101	053656	032767	000001	126374	BIT #1,ERROUT	:		4932
8102	053664	001404			BEQ 33\$:		
8103	053666	104456			TRAP 56	:		4937
8104	053670	000160			.WORD 160			
8105	053672	011202			.WORD MSG4			
8106	053674	000000			.WORD 0			
8107	053676	016646	000010	33\$:	MOV 10(SP),-(SP)	: LUN,*		4940
8108	053702	016746			MOV BOARD,-(SP)			
8109	053706	004767			JSR PC,UP.HARD.COUNT			
8110	053712	000427			BR 37\$:		4928
8111	053714	032767	000001	126336	34\$:	BIT #1,ERROUT	:	4945
8112	053722	001415			BEQ 36\$:		
8113	053724	104457			TRAP 57	:		4950
8114	053726	000161			.WORD 161			
8115	053730	011166			.WORD MSG3			
8116	053732	000000			.WORD 0			
8117	053734	000410			BR 36\$:		4953
8118	053736	032767	000001	126314	35\$:	BIT #1,ERROUT	:	4959
8119	053744	001404			BEQ 36\$:		
8120	053746	104457			TRAP 57	:		4964
8121	053750	000162			.WORD 162			
8122	053752	011166			.WORD MSG3			
8123	053754	000000			.WORD 0			
8124	053756	016646	000010	36\$:	MOV 10(SP),-(SP)	: LUN,*		4967
8125	053762	016746			MOV BOARD,-(SP)			
8126	053766	004767			JSR PC,UP.SOFT.COUNT			
8127	053772	022626			37\$:	CMP (SP)+,(SP)+	:	4898
8128	053774	005205			38\$:	INC R5	: SECTOR	4894
8129	053776	020501			39\$:	CMP R5,R1	: SECTOR,*	
8130	054000	101002			BHI 40\$			
8131	054002	000167	177362		JMP 31\$			
8132	054006	005266	000010	40\$:	INC 10(SP)	: LUN		4693
8133	054012	026666	000010	000012	41\$:	CMP 10(SP),12(SP)	: LUN,*	
8134	054020	002002			BGE 42\$			
8135	054022	000167	175510		JMP 2\$			
8136	054026	005266	000026	42\$:	INC 26(SP)	: COMP.FLAG		4690
8137	054032	026627	000026	000001	CMP 26(SP),#1	: COMP.FLAG,*		
8138	054040	003002			BGT 43\$			
8139	054042	000167	175452		JMP 1\$			
8140	054046	062706	000030	43\$:	ADD #30,SP	:		4628
8141	054052	000207			RTS PC			

: Routine Size: 638 words
: Maximum stack depth per invocation: 37 words

8142
8143
8144
8149
8150

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (38)

```

3152 :MLX4
8153 :
8154 :
8155 : 4981 %sbttl 'DEFINITION OF OPTION 2'
8156 : 4982 routine OPT2 : novalue =
8157 : 4983 begin
8158 : 4984
8159 : 4985
8160 : 4986 ++
8161 : 4987 ROUTINE: OPT2
8162 : 4988 PURPOSE: TO CHECK ON DATA RELIABILITY USING THE PATTERNS FROM
8163 : 4989 THE PATTERN TABLE.
8164 : 4990
8165 : 4991 THE CODE FOR 'OPT2' IN BRIEF:
8166 : 4992
8167 : 4993 BEGIN 1 (START OF ROUTINE)
8168 : 4994 SAY THE ROUTINE IS RUNNING
8169 : 4995 CHOOSE A MAXIMUM PATTERN NUMBER
8170 : 4996 INCR COUNT FROM 1 TO (2*MAX)
8171 : 4997 : BEGIN 2 (START OF PATTERN SELECTION LOOP)
8172 : 4998 : GENERATE THE PATTERN
8173 : 4999 : INCR LUN FROM 0 TO LAST
8174 : 5000 : : BEGIN 3 (START OF LOGICAL UNIT SELECTION LOOP)
8175 : 5001 : : TESTLOOP:
8176 : 5002 : : : BEGIN 4 (START OF LOOP THAT COMPLETELY TESTS 1 UNIT)
8177 : 5003 : : : IF UNIT IS ACTIVE
8178 : 5004 : : : THEN
8179 : 5005 : : : : BEGIN 5 (START OF TEST FOR AN ACTIVE UNIT)
8180 : 5006 : : : : INITIALIZE THE WRITE AND READ BUFFER POINTERS
8181 : 5007 : : : : SECTOR = LOWEST
8182 : 5008 : : : : WHILE SECTOR LEQ HIGHEST DO
8183 : 5009 : : : : : BEGIN 6 (START OF SECTOR SELECTION LOOP)
8184 : 5010 : : : : : GET WIDCNT
8185 : 5011 : : : : : SET UP BUFFER POINTERS BEFORE TRANSFER
8186 : 5012 : : : : : WRITE
8187 : 5013 : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
8188 : 5014 : : : : : CHOOSE WHETHER TO WRITE CHECK OR READ
8189 : 5015 : : : : : DO THE WRITE CHECK OR READ
8190 : 5016 : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
8191 : 5017 : : : : : CHANGE BUFFER POINTERS AFTER TRANSFER
8192 : 5018 : : : : : UPDATE SECTOR NUMBER BY NUMBER OF SECTORS IN PREVIOUS TRANSFER
8193 : 5019 : : : : : END 6 (END OF SECTOR SELECTION LOOP)
8194 : 5020 : : : : : END 5 (END OF TEST FOR AN ACTIVE UNIT)
8195 : 5021 : : : : : END 4 (END OF TESTLOOP)
8196 : 5022 : : : : : END 3 (END OF LOGICAL UNIT SELECTION LOOP)
8197 : 5023 : : IF NOT THE QUICK VERIFY PASS THEN 'LOOP READ' (DESCRIBED BELOW)
8198 : 5024 : : END 2 (END OF PATTERN SELECTION LOOP)
8199 : 5025 RETURN
8200 : 5026 END 1 (END OF ROUTINE)
8201 : 5027
8202 : 5028
8203 : 5029 Label
8204 : 5030 LOOP
8205 : 5031 LOOP2:
8206 : 5032
    
```

!* 1 * START OF ROUTINE

27-Mar-1982 19:24:42
27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
PA:<NEALE>MLX4.BLI.5 (38)

8208 :MLX4
8209 :
8210 :
8211 :
8212 :
8213 :
8214 :
8215 :
8216 :
8217 :
8218 :
8219 :
8220 :
8221 :
8222 :
8223 :
8224 :
8225 :
8226 :
8227 :
8228 :
8229 :
8230 :
8231 :
8232 :
8233 :
8234 :
8235 :
8236 :
8237 :
8238 :
8239 :
8240 :
8241 :
8242 :
8243 :
8244 :
8245 :
8246 :
8247 :
8248 :
8249 :
8250 :
8251 :
8252 :
8253 :
8254 :
8255 :
8256 :
8257 :
8258 :
8259 :
8260 :
8261 :
8262 :

DEFINITION OF OPTION 2

```

local
WRDCNT,
VALUE,
TEMP,
MAXPAT,
OLDSEC,
OLDCHN,
SECTOR,
PTR,
COMMAND,
DBL_VALUE;

PRINTB (SAY2, WRD34, RTN2);
!'RUNNING OPT2'
PATTERN = 0;

if .QUICK neq 0 then MAXPAT = 1 else MAXPAT = NUM_PATS;

incr COUNT from 1 to (.MAXPAT*2) do
begin
SELPAT ();
!* 2 * START OF PATTERN SELECTION LOOP

if .PATTERN gtr 0
then
PRINTB (FMT1A, PHR9, .PATTERN)
!'PATTERN NUMBER XX'
else
begin
TEMP = -(.PATTERN);
PRINTB (FMT1B, PHR9, .TEMP);
!' PATTERN NUMBER - XX'
end;

GEN2 (.PATTERN);

incr LUN from 0 to (.LSUNIT - 1) do
begin
LOOP :
begin
!* 3 * START OF LOGICAL UNIT SELECTION LOOP
!* 4 * START OF THE LOOP THAT COMPLETELY TESTS 1 UNIT

if .DRIVE_STATUS [LUN] eql ACTIVE
then
begin
!* 5 * START OF TEST FOR AN ACTIVE UNIT
LSLUN = .LUN;
WPTR = WBIFF;
RPTR = RBUFF;
SECTOR = LOWEST;

while .SECTOR lequ HIGHEST do
begin
!* 6 * START OF SECTOR SELECTION LOOP
WRDCNT = GET WRDCNT (.SECTOR, HIGHEST);
SET_PTRS (.WRDCNT);

```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (38)

8264 :MLX4
 8265 :
 8266 :
 8267 :
 8268 :
 8269 :
 8270 :
 8271 :
 8272 :
 8273 :
 8274 :
 8275 :
 8276 :
 8277 :
 8278 :
 8279 :
 8280 :
 8281 :
 8282 :
 8283 :
 8284 :
 8285 :
 8286 :
 8287 :
 8288 :
 8289 :
 8290 :
 8291 :
 8292 :
 8293 :
 8294 :
 8295 :
 8296 :
 8297 :
 8298 :
 8299 :
 8300 :
 8301 :
 8302 :
 8303 :
 8304 :
 8305 :
 8306 :
 8307 :
 8308 :
 8309 :
 8310 :
 8311 :
 8312 :
 8313 :
 8314 :
 8315 :
 8316 :
 8317 :
 8318 :

DEFINITION OF OPTION 2

```

VALUE = write (.LUN, .WRDCNT, .WPTR, .SECTOR);

!+
!- SEE HOW SUCCESSFUL THE WRITE WAS:

selectone .VALUE of          !SEE 'SYSERR' FOR DEFINITION
set                          !OF ERROR # CONTAINED IN 'VALUE'

[1] :
begin                        !* 6A * RETRY ALLOWED
    if RETRY (SIX, write, .LUN, .WRDCNT, .WPTR, .SECTOR) neq 0
    then                      !THE RETRY FAILED -- SYSTEM FATAL ERROR
        begin
            WHY DROPT [.LUN] = CODE_4;
            ERRDF (201, MSG1, 0); !**** OPTION 2 ERROR 01 ****
            DODU (.LUN);
            leave LOOP;        !JUMP JUST BEYOND END OF BLOCK * 4 *
        end;
    end;                      !* 6A *

[2] :
begin                        !* 6B * FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
    WHY DROPT [.LUN] = CODE_5;
    ERRDF (202, MSG1, 0);     !**** OPTION 2 ERROR 02 ****
    DODU (.LUN);
    leave LOOP;              !JUMP JUST BEYOND END OF BLOCK * 4 *
end;                          !* 6B *

[3] :
begin                        !* 6C * FATAL DRIVE ERROR -- NO RETRY ALLOWED
    ERRDF (203, MSG1, 0);     !**** OPTION 2 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
    
```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (38)

```
8320 :MLX4
8321 :
8322 :
8323 : 5137
8324 : 5138
8325 : 5139
8326 : 5140
8327 : 5141
8328 : 5142
8329 : 5143
8330 : 5144
8331 : 5145
8332 : 5146
8333 : 5147
8334 : 5148
8335 : 5149
8336 : 5150
8337 : 5151
8338 : 5152
8339 : 5153
8340 : 5154
8341 : 5155
8342 : 5156
8343 : 5157
8344 : 5158
8345 : 5159
8346 : 5160
8347 : 5161
8348 : 5162
8349 : 5163
8350 : 5164
8351 : 5165
8352 : 5166
8353 : 5167
8354 : 5168
8355 : 5169
8356 : 5170
8357 : 5171
8358 : 5172
8359 : 5173
8360 : 5174
8361 : 5175
8362 : 5176
8363 : 5177
8364 : 5178
8365 : 5179
8366 : 5180
8367 : 5181
8368 : 5182
8369 : 5183
8370 : 5184
8371 : 5185
8372 : 5186
8373 : 5187
8374 : 5188

DEFINITION OF OPTION 2

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 (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);
```

8376 :MLX4
 8377 :
 8378 :
 8379 :
 8380 :
 8381 :
 8382 :
 8383 :
 8384 :
 8385 :
 8386 :
 8387 :
 8388 :
 8389 :
 8390 :
 8391 :
 8392 :
 8393 :
 8394 :
 8395 :
 8396 :
 8397 :
 8398 :
 8399 :
 8400 :
 8401 :
 8402 :
 8403 :
 8404 :
 8405 :
 8406 :
 8407 :
 8408 :
 8409 :
 8410 :
 8411 :
 8412 :
 8413 :
 8414 :
 8415 :
 8416 :
 8417 :
 8418 :
 8419 :
 8420 :
 8421 :
 8422 :
 8423 :
 8424 :
 8425 :
 8426 :
 8427 :
 8428 :
 8429 :
 8430 :

DEFINITION OF OPTION 2

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (38)

```

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;
DODD (.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;
DODD (.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 (209, MSG4, 0);          !**** OPTION 2 ERROR 09 ****
            UP_HARD_COUNT (.LUN, .BOARD);
        end
    else
        begin
            if .ERROUT then ERRSOFT (210, MSG3, 0);          !**** OPTION 2 ERROR 10 ****
            UP_SOFT_COUNT (.LUN, .BOARD);
        end
    end

else
    begin

```

```

8432 :MLX4
8433 :
8434 :
8435 : 5241 if .ERROUT then ERRSDFT (211, MSG3, 0); !**** OPTION 2 ERROR 11 ****
8436 : 5242
8437 : 5243 UP_SOFT_COUNT (.LUN, .BOARD);
8438 : 5244 end;
8439 : 5245
8440 : 5246
8441 : 5247 tes; !* 6H *
8442 : 5248
8443 : 5249 WPTR = .WPTR + (.WRDCNT*2);
8444 : 5250 SECTOR = .SECTOR + (.WRDCNT/256);
8445 : 5251 end; !* 6 * END OF SECTOR SELECTION LOOP
8446 : 5252
8447 : 5253 end; !* 5 * END OF TEST FOR AN ACTIVE UNIT
8448 : 5254
8449 : 5255 end; !* 4 * END OF TESTLOOP
8450 : 5256 end; !* 3 * END OF LOGICAL UNIT SELECTION LOOP
8451 : 5257
8452 : 5258 if .QUICK eql 0
8453 : 5259 then
8454 : 5260 begin !* 11 * START OF LOOP READING SECTION
8455 : 5261
8456 : 5262
8457 : 5263 !++ THIS IS THE 'LOOP READ' SECTION WHICH WAS MENTIONED IN
8458 : 5264 THE DOCUMENTATION AT THE BEGINNING OF THIS ROUTINE. IT
8459 : 5265 IS NOT EXECUTED DURING THE QUICK VERIFY PASS, BUT IT IS
8460 : 5266 FOR EVERY OTHER PASS THROUGH OPT2.
8461 : 5267
8462 : 5268 THE CODE IN BRIEF:
8463 : 5269
8464 : 5270 BEGIN 11 (START OF LOOP READING SECTION)
8465 : 5271 INCR LUN FROM 0 TO LAST
8466 : 5272 : BEGIN 12 (START OF LOGICAL UNIT SELECTION LOOP)
8467 : 5273 : TESTLOOP2:
8468 : 5274 : : BEGIN 13 (START OF LOOP THAT COMPLETELY TESTS 1 UNIT)
8469 : 5275 : : IF UNIT IS ACTIVE
8470 : 5276 : : THEN
8471 : 5277 : : : BEGIN 14 (START OF TEST FOR AN ACTIVE UNIT)
8472 : 5278 : : : INITIALIZE THE WRITE AND READ BUFFER POINTERS
8473 : 5279 : : : SECTOR = LOWEST
8474 : 5280 : : : WHILE SECTOR LEQ HIGHEST DO
8475 : 5281 : : : : BEGIN 15 (START OF SECTOR SELECTION LOOP)
8476 : 5282 : : : : GET WRDCNT
8477 : 5283 : : : : SET UP BUFFER POINTERS BEFORE TRANSFER
8478 : 5284 : : : : CHOOSE WHETHER TO WRITE CHECK OR READ
8479 : 5285 : : : : INCR KOUNT FROM 1 TO TIMES
8480 : 5286 : : : : : BEGIN 16 (START OF COUNTING LOOP FOR LOOP READING)
8481 : 5287 : : : : : DO THE WRITE CHECK OR READ
8482 : 5288 : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP2)
8483 : 5289 : : : : : END 16 (END OF COUNTING LOOP FOR LOOP READING)
8484 : 5290 : : : : : CHANGE BUFFER POINTERS AFTER TRANSFER
8485 : 5291 : : : : : UPDATE SECTOR NUMBER BY # SECTORS IN PREVIOUS TRANSFER
8486 : 5292 : : : : : END 15 (END OF SECTOR SELECTION LOOP)
    
```

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (38)

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (38)

```

8488 :MLX4
8489 :
8490 :
8491 : 5293          : : : END 14 (END OF TEST FOR AN ACTIVE UNIT)
8492 : 5294          : : : END 13 (END OF TESTLOOP2)
8493 : 5295          : : : END 12 (END OF LOGICAL UNIT SELECTION LOOP)
8494 : 5296          : : : END 11 (END OF LOOP READING SECTION)
8495 : 5297          : : :
8496 : 5298          : : :
8497 : 5299          : : :
8498 : 5300          : : :
8499 : 5301          : : :
8500 : 5302          : : :
8501 : 5303          : : :
8502 : 5304          : : :
8503 : 5305          : : :
8504 : 5306          : : :
8505 : 5307          : : :
8506 : 5308          : : :
8507 : 5309          : : :
8508 : 5310          : : :
8509 : 5311          : : :
8510 : 5312          : : :
8511 : 5313          : : :
8512 : 5314          : : :
8513 : 5315          : : :
8514 : 5316          : : :
8515 : 5317          : : :
8516 : 5318          : : :
8517 : 5319          : : :
8518 : 5320          : : :
8519 : 5321          : : :
8520 : 5322          : : :
8521 : 5323          : : :
8522 : 5324          : : :
8523 : 5325          : : :
8524 : 5326          : : :
8525 : 5327          : : :
8526 : 5328          : : :
8527 : 5329          : : :
8528 : 5330          : : :
8529 : 5331          : : :
8530 : 5332          : : :
8531 : 5333          : : :
8532 : 5334          : : :
8533 : 5335          : : :
8534 : 5336          : : :
8535 : 5337          : : :
8536 : 5338          : : :
8537 : 5339          : : :
8538 : 5340          : : :
8539 : 5341          : : :
8540 : 5342          : : :
8541 : 5343          : : :
8542 : 5344          : : :

DEFINITION OF OPTION 2

incr LUN from 0 to (.LSUNIT - 1) do
begin
LOOP2 :
begin
if .DRIVE_STATUS [.LUN] eql ACTIVE
then
begin
LSLUN = .LUN;
WPTR = WBUFF;
RPTR = RBUFF;
SECTOR = LOWEST;
while .SECTOR lequ HIGHEST do
begin
WRDCNT = GET WRDCNT (.SECTOR, HIGHEST);
SET PTRS (.WRDCNT);
COMMAND = CHOOSE ();
incr KOUNT from 1 to TIMES_TO_LOOP do
begin
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
set
[0] :
if .COMMAND eql read
then
begin
! * 12 * START OF LOGICAL UNIT SELECTION LOOP
! * 13 * START OF THE 2ND LOOP THAT COMPLETELY TESTS 1 UNIT
! * 14 * START OF LOOP THAT TESTS AN ACTIVE UNIT
! * 15 * START OF SECTOR SELECTION LOOP
! * 16 * START OF COUNTING LOOP FOR LOOP READING
!SEE 'SYSERR' FOR DEFINITION
!OF ERROR # CONTAINED IN 'VALUE'

```

8544 :MLX4
8545 :
8546 :
8547 :
8548 :
8549 :
8550 :
8551 :
8552 :
8553 :
8554 :
8555 :
8556 :
8557 :
8558 :
8559 :
8560 :
8561 :
8562 :
8563 :
8564 :
8565 :
8566 :
8567 :
8568 :
8569 :
8570 :
8571 :
8572 :
8573 :
8574 :
8575 :
8576 :
8577 :
8578 :
8579 :
8580 :
8581 :
8582 :
8583 :
8584 :
8585 :
8586 :
8587 :
8588 :
8589 :
8590 :
8591 :
8592 :
8593 :
8594 :
8595 :
8596 :
8597 :
8598 :

5345
5346
5347
5348
5349
5350
5351
5352
5353
5354
5355
5356
5357
5358
5359
5360
5361
5362
5363
5364
5365
5366
5367
5368
5369
5370
5371
5372
5373
5374
5375
5376
5377
5378
5379
5380
5381
5382
5383
5384
5385
5386
5387
5388
5389
5390
5391
5392
5393
5394
5395
5396

DEFINITION OF OPTION 2

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (38)

```

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 ****
  DODU (.LUN);
  leave LOOP2;          !JUMP JUST BEYOND END OF BLOCK * 13 *
end;
end;      !* 16A *

[2] :
begin      !* 16B *      FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
WHY DROPT [.LUN] = CODE_5;
ERRDF (214, MSG1, 0);   !**** OPTION 2 ERROR 14 ****
DODU (.LUN);
leave LOOP2;           !JUMP JUST BEYOND END OF BLOCK * 13 *
end;      !* 16B *

[3] :
begin      !* 16C *      FATAL DRIVE ERROR -- NO RETRY ALLOWED
ERRDF (215, MSG1, 0);   !**** OPTION 2 ERROR 15 ****
WHY DROPT [.LUN] = CODE_6;
DODU (.LUN);
leave LOOP2;           !JUMP JUST BEYOND END OF BLOCK * 13 *
end;      !* 16C *

[4] :
begin      !* 16D *      UNRECOVERABLE DATA ERROR

```

8600 :MLX4
 8601 :
 8602 :
 8603 :
 8604 :
 8605 :
 8606 :
 8607 :
 8608 :
 8609 :
 8610 :
 8611 :
 8612 :
 8613 :
 8614 :
 8615 :
 8616 :
 8617 :
 8618 :
 8619 :
 8620 :
 8621 :
 8622 :
 8623 :
 8624 :
 8625 :
 8626 :
 8627 :
 8628 :
 8629 :
 8630 :
 8631 :
 8632 :
 8633 :
 8634 :
 8635 :
 8636 :
 8637 :
 8638 :
 8639 :
 8640 :
 8641 :
 8642 :
 8643 :
 8644 :
 8645 :
 8646 :
 8647 :
 8648 :
 8649 :
 8650 :
 8651 :
 8652 :
 8653 :
 8654 :

DEFINITION OF OPTION 2

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (38)

```

ISOLATE ();
ERRDF (216, MSG2, 0);          !**** OPTION 2 ERROR 16 ****
WHY DROPT (.LUN) = CODE_7;
DODD (.LUN);
Leave LOOP2:                    !JUMP JUST BEYOND END OF BLOCK * 13 *
end;                            !* 16D *

[5] :
begin                            !* 16E *          RECOVERABLE DATA ERROR
ISOLATE ();

if .ERROUT then PRINTB (FMT10B, .CHAN);    !'   BIT 00'

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 (217, MSG4, 0);

                !**** OPTION 2 ERROR 17 ****
            UP_HARD_COUNT (.LUN, .BOARD);
        end
    else
        begin
            if .ERROUT then ERRSOFT (218, MSG3, 0);

                !**** OPTION 2 ERROR 18 ****
            UP_SOFT_COUNT (.LUN, .BOARD);
        end
    else
        begin
            if .ERROUT then ERRSOFT (219, MSG3, 0); !**** OPTION 2 ERROR 19 ****

            UP_SOFT_COUNT (.LUN, .BOARD);
        end;
    end;                            !* 16E *
tes;

end;                            !* 16 *   END OF COUNTING LOOP FOR LOOP READING

WPTR = .WPTR + (.WRDCNT*2);
SECTOR = .SECTOR + (.WRDCNT/256);
  
```


27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (38)

8656 :MLX4

DEFINITION OF OPTION 2

```

8657 :
8658 :
8659 :          5449          end:          !* 15 * END OF SECTOR SELECTION LOOP
8660 :          5450
8661 :          5451          end:          !* 14 * END OF TEST FOR AN ACTIVE UNIT
8662 :          5452
8663 :          5453
8664 :          5454          !+
8665 :          5455          ! Test to see if this uut's address space is
8666 :          5456          ! to be read for soft errors. This test is
8667 :          5457          ! is intended for DMT purposes.
8668 :          5458          :-
8669 :          5459          if .EFNS21          !Is the background pattern to be read
8670 :          5460          then
8671 :          5461          begin
8672 :          5462
8673 :          5463          !
8674 :          5464          ! version czmlbb changed incr to incru
8675 :          5465          !
8676 :          5466          incru SECTOR from LOWEST to HIGHEST do
8677 :          5467
8678 :          5468          if read (.LUN, 256, RBUFF, .SECTOR) eql 5
8679 :          5469          then
8680 :          5470          begin
8681 :          5471          ISOLATE ();          !Find the failing bank and board no.
8682 :          5472
8683 :          5473          if .ERROUT then PRINTB (FMT10B, .CHAN);
8684 :          5474
8685 :          5475          ! Print where the error is
8686 :          5476          !
8687 :          5477          ! Save the contents of the ML error location
8688 :          5478          ! register so we can compare it to the new
8689 :          5479          ! contents of this register after the retry.
8690 :          5480          ! This is done to classify the error.
8691 :          5481
8692 :          5482          OLDSEC = .MLEL;
8693 :          5483          OLDCHN = .CHAN;
8694 :          5484
8695 :          5485          ! Do a classify retry call. If the same error
8696 :          5486          ! occurs then classify it as a hard error. If
8697 :          5487          ! a different error occurred or the error went away
8698 :          5488          ! then classify it as a soft error.
8699 :          5489
8700 :          5490          !
8701 :          5491          if RETRY (ONE, .COMMAND, .LUN, 256, .PTR, .OLDSEC) eql 5
8702 :          5492          then
8703 :          5493
8704 :          5494          ! The same error occurred so see if it is at the same
8705 :          5495          ! sector and channel number, if so then classify
8706 :          5496          ! it as a hard error else classiy it as a soft
8707 :          5497          ! error.
8708 :          5498          !
8709 :          5499
8710 :          5500          if ((.MLEL eql .OLDSEC) and (.CHAN eql .OLDCHN))
    
```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (38)

8712 :MLX4

DEFINITION OF OPTION 2

```
8713 :  
8714 :  
8715 : 5501 then  
8716 : 5502 begin !Same error occurred 'hard'  
8717 : 5503 if .ERROUT !Print error if enabled  
8718 : 5504 then  
8719 : 5505 begin  
8720 : 5506 ERRHRD (220, !Error number  
8721 : P 5507 MSG4, !Error message  
8722 : P 5508 0); !Additional message routine  
8723 : 5509 end;  
8724 : 5510 UP_HARD_COUNT (.LUN, .BOARD);  
8725 : 5511 end  
8726 : 5512 else  
8727 : 5513 begin !Not the same error 'soft'  
8728 : 5514 if .ERROUT !Print error if enabled  
8729 : 5515 then  
8730 : 5516 begin  
8731 : 5517 ERRSOFT (221, !Error number  
8732 : 5518 MSG3, !Error message  
8733 : P 5519 0); !Additional message routine  
8734 : P 5520 end;  
8735 : P 5521 UP_SOFT_COUNT (.LUN, .BOARD);  
8736 : 5522 end  
8737 : 5523 else  
8738 : 5524 begin !Not the same error 'soft'  
8739 : 5525 if .ERROUT !Print error if enabled  
8740 : 5526 then  
8741 : 5527 begin  
8742 : 5528 ERRSOFT (222, !Error number  
8743 : 5529 MSG3, !Error message  
8744 : 5530 0); !Additional message routine  
8745 : 5531 end;  
8746 : 5532 UP_SOFT_COUNT (.LUN, .BOARD);  
8747 : 5533 end;  
8748 : P 5534 end;  
8749 : P 5535 end;  
8750 : 5536 end;  
8751 : 5537 end;  
8752 : 5538 end;  
8753 : 5539 end;  
8754 : 5540 end;  
8755 : 5541 end;  
8756 : 5542 end;  
8757 : 5543 end;  
8758 : 5544 end;  
8759 : 5545 end;  
8760 : 5546 end; !* 13 * END OF TESTLOOP2  
8761 : 5547 end; !* 12 * END OF LOGICAL UNIT SELECTION LOOP  
8762 : 5548 end; !* 11 * END OF LOOP READING SECTION  
8763 : 5549 end; !* 2 * END OF PATTERN SELECTION LOOP  
8764 : 5550  
8765 : 5551  
8766 : 5552
```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (38)

8768 :MLX4
 8769 :
 8770 :
 8771 : 5553 return;
 8772 : 5554 end;

!* 1 * END OF ROUTINE

```

8777 .SBTTL OPT2 DEFINITION OF OPTION 2
8781 054054 004167 131254 OPT2: JSR R1,$SAVE5 ; 4982
8782 054060 162706 000026 SUB #26,SP ;
8783 054064 012746 007634 MOV #RTN2,-(SP) ; 5045
8784 054070 012746 007344 MOV #WRD34,-(SP)
8785 054074 012746 007100 MOV #SAY2,-(SP)
8786 054100 012746 000003 MOV #3,-(SP)
8787 054104 010600 MOV SP,RO ; SP,*
8788 054106 104414 TRAP 14
8789 054110 005067 156562 CLR PATTERN ; 5047
8790 054114 005767 156554 TST QUICK ; 5049
8791 054120 001403 BEQ 1$
8792 054122 012705 000001 MOV #1,R5 ; *,MAXPAT
8793 054126 000402 BR 2$
8794 054130 012705 000012 1$: MOV #12,R5 ; *,MAXPAT
8795 054134 010566 000032 2$: MOV R5,32(SP) ; MAXPAT,* 5051
8796 054140 006366 000032 ASL 32(SP)
8797 054144 005066 000030 CLR 30(SP) ; COUNT
8798 054150 000167 003650 JMP 74$
8799 054154 004767 164706 3$: JSR PC,SELPAT ; 5053
8800 054160 005767 156512 TST PATTERN ; 5055
8801 054164 003413 BLE 4$
8802 054166 016746 156504 MOV PATTERN,-(SP) ; 5057
8803 054172 012746 010100 MOV #PHR9,-(SP)
8804 054176 012746 006126 MOV #FMT1A,-(SP)
8805 054202 012746 000003 MOV #3,-(SP)
8806 054206 010600 MOV SP,RO ; SP,*
8807 054210 104414 TRAP 14
8808 054212 000417 BR 5$
8809 054214 016766 156456 000034 4$: MOV PATTERN,34(SP) ; *,TEMP 5055
8810 054222 005466 000034 NEG 34(SP) ; TEMP 5061
8811 054226 016646 000034 MOV 34(SP),-(SP) ; TEMP,* 5062
8812 054232 012746 010100 MOV #PHR9,-(SP)
8813 054236 012746 006142 MOV #FMT1B,-(SP)
8814 054242 012746 000003 MOV #3,-(SP)
8815 054246 010600 MOV SP,RO ; SP,*
8816 054250 104414 TRAP 14
8817 054252 016716 156420 5$: MOV PATTERN,(SP) ; 5066
8818 054256 004767 165150 JSR PC,GEN2
8819 054262 016766 125524 000034 MOV L$UNIT,34(SP) ; 5068
8820 054270 005005 CLR R5 ; LUN
8821 054272 000167 001462 JMP 33$
8822 054276 010500 6$: MOV R5,RO ; LUN,* 5073
    
```

Address	OpCode	Operand 1	Operand 2	Label	Definition	Comments	Page
8824							
8825							
8826							
8827	054300	006200			ASR R0		
8828	054302	006200			ASR R0		
8829	054304	006200			ASR R0		
8830	054305	062700	034442		ADD #DRIVE.STATUS,R0		
8831	054312	010046			MOV R0,-(SP)		
8832	054314	010546			MOV R5,-(SP)	; LUN,*	
8833	054316	042716	177770		BIC #177770,(SP)		
8834	054322	012746	000001		MOV #1,-(SP)		
8835	054326	005046			CLR -(SP)		
8836	054330	004767	130022		JSR PC,BL\$GT2		
8837	054334	062706	000010		ADD #10,SP		
8838	054340	005300			DEC R0		
8839	054342	001127			BNE 11\$		
8840	054344	010567	125524		MOV R5,L\$LUN	; LUN,*	5076
8841	054350	012767	012670	156312	MOV #WBUFF,WPTR	:	5077
8842	054356	012767	022670	156306	MOV #RBUFF,RPTR	:	5078
8843	054364	010501			MOV R5,R1	; LUN,*	5079
8844	054366	006301			ASL R1		
8845	054370	016104	034460		MOV LOW.SECT(R1),R4	; *,SECTOR	
8846	054374	020461	034500	7\$:	CMP R4, TOP.SECT(R1)	; SECTOR,*	5081
8847	054400	101110			BHI 11\$		
8848	054402	010446			MOV R4,-(SP)	; SECTOR,*	5083
8849	054404	016146	034500		MOV TOP.SECT(R1),-(SP)		
8850	054410	004767	172562		JSR PC,GET.WRDCNT		
8851	054414	010002			MOV R0,R2	; *,WRDCNT	
8852	054416	010216			MOV R2,(SP)	; WRDCNT,*	5084
8853	054420	004767	172506		JSR PC,SET.PTRS		
8854	054424	010516			MOV R5,(SP)	; LUN,*	5085
8855	054426	010246			MOV R2,-(SP)	; WRDCNT,*	
8856	054430	016746	156234		MOV WPTR,-(SP)		
8857	054434	010446			MOV R4,-(SP)	; SECTOR,*	
8858	054436	004767	170762		JSR PC,WRITE		
8859	054442	010003			MOV R0,R3	; *,VALUE	
8860	054444	020327	000001		CMP R3,#1	; VALUE,*	5091
8861	054450	001031			BNE 8\$		
8862	054452	012746	000006		MOV #6,-(SP)	:	5097
8863	054456	012746	045424		MOV #WRITE,-(SP)		
8864	054462	010546			MOV R5,-(SP)	; LUN,*	
8865	054464	010246			MOV R2,-(SP)	; WRDCNT,*	
8866	054466	016746	156176		MOV WPTR,-(SP)		
8867	054472	010446			MOV R4,-(SP)	; SECTOR,*	
8868	054474	004767	171524		JSR PC,RETRY		
8869	054500	062706	000014		ADD #14,SP		
8870	054504	005700			TST R0		
8871	054506	001446			BEQ 12\$		
8872	054510	112765	000004	034446	MOVB #4,WHY.DROPT(R5)	; *,*(LUN)	5100
8873	054516	104455			TRAP 55	:	5101
8874	054520	000311			.WORD 311		
8875	054522	011070			.WORD MSG1		
8876	054524	000000			.WORD 0		
8877	054526	010500			MOV R5,R0	; LUN,*	5102
8878	054530	104451			TRAP 51		

Address	OpCode	Operand 1	Operand 2	Label	Instruction	Comments	Seq	
8880								
8881								
8882								
8883	054532	000431			BR 10\$		5103	
8884	054534	020327	000002	8\$:	CMP R3,#2	: VALUE,*	5091	
8885	054540	001012			BNE 9\$			
8886	054542	112765	000005	034446	MOVB #5,WHY.DROPT(R5)	: *,*(LUN)	5110	
8887	054550	104455			TRAP 55	:	5111	
8888	054552	000312			.WORD 312			
8889	054554	011070			.WORD MSG1			
8890	054556	000000			.WORD 0			
8891	054560	010500			MOV R5,RO	: LUN,*	5112	
8892	054562	104451			TRAP 51			
8893	054564	000414			BR 10\$		5113	
8894	054566	020327	000003	9\$:	CMP R3,#3	: VALUE,*	5091	
8895	054572	001014			BNE 12\$			
8896	054574	104455			TRAP 55	:	5118	
8897	054576	000313			.WORD 313			
8898	054600	011070			.WORD MSG1			
8899	054602	000000			.WORD 0			
8900	054604	112765	000006	034446	MOVB #6,WHY.DROPT(R5)	: *,*(LUN)	5119	
8901	054612	010500			MOV R5,RO	: LUN,*	5120	
8902	054614	104451			TRAP 51			
8903	054616	062706	000012	10\$:	ADD #12,SP	:	5121	
8904	054622	000537		11\$:	BR 15\$			
8905	054624	004767	171336	12\$:	JSR PC,CHOOSE	:	5125	
8906	054630	010066	000034		MOV RO,34(SP)	: *,COMMAND		
8907	054634	005066	000040		CLR 40(SP)	:	5127	
8908	054640	020027	045612		CMP RO,#READ	: COMMAND,*		
8909	054644	001015			BNE 13\$			
8910	054646	005266	000040		INC 40(SP)			
8911	054652	016766	156014	000036	MOV RPTR,36(SP)	: *,PTR	5130	
8912	054660	010546			MOV R5,-(SP)	: LUN,*	5131	
8913	054662	010246			MOV R2,-(SP)	: WRDCNT,*		
8914	054664	016746	156002		MOV RPTR,-(SP)			
8915	054670	010446			MOV R4,-(SP)	: SECTOR,*		
8916	054672	004767	170714		JSR PC,READ			
8917	054676	000412			BR 14\$			
8918	054700	016766	155764	000036	13\$:	MOV WPTR,36(SP)	: *,PTR	5135
8919	054706	010546			MOV R5,-(SP)	: LUN,*	5136	
8920	054710	010246			MOV R2,-(SP)	: WRDCNT,*		
8921	054712	016746	155752		MOV WPTR,-(SP)			
8922	054716	010446			MOV R4,-(SP)	: SECTOR,*		
8923	054720	004767	171054		JSR PC,CHECK			
8924	054724	010003		14\$:	MOV RO,R3	: *,VALUE		
8925	054726	001076			BNE 16\$:	5143	
8926	054730	032766	000001	000050	BIT #1,50(SP)	:	5148	
8927	054736	001513			BEQ 17\$			
8928	054740	016746	155724		MOV WPTR,-(SP)	:	5152	
8929	054744	016746	155722		MOV RPTR,-(SP)			
8930	054750	010246			MOV R2,-(SP)	: WRDCNT,*		
8931	054752	004767	167350		JSR PC,DOUBLE.CHECK			
8932	054756	062706	000006		ADD #6,SP			
8933	054762	010066	000060		MOV RO,60(SP)	: *,DBL.VALUE		
8934	054766	001477			BEQ 17\$			

Address	Instruction	Comment	Label	Definition	Value	
8992			:MLX4	DEFINITION OF OPTION 2		
8993			:			
8994						
8995	055226	112765	000005	034446	MOVB #5,WHY.DROPT(R5) ; *,*(LUN) 5186	
8996	055234	104455			TRAP 55 ; 5187	
8997	055236	000316			.WORD 316	
8998	055240	011070			.WORD MSG1	
8999	055242	000000			.WORD 0	
9000	055244	010500			MOV R5,R0 ; LUN,* 5188	
9001	055246	104451			TRAP 51	
9002	055250	000433			BR 22\$	
9003	055252	020327	000003	20\$:	CMP R3,#3 ; VALUE,* 5189	
9004	055256	001012			BNE 21\$; 5143	
9005	055260	104455			TRAP 55 ; 5194	
9006	055262	000317			.WORD 317	
9007	055264	011070			.WORD MSG1	
9008	055266	000000			.WORD 0	
9009	055270	112765	000006	034446	MOVB #6,WHY.DROPT(R5) ; *,*(LUN) 5195	
9010	055276	010500			MOV R5,R0 ; LUN,* 5196	
9011	055300	104451			TRAP 51	
9012	055302	000416			BR 22\$	
9013	055304	020327	000004	21\$:	CMP R3,#4 ; VALUE,* 5197	
9014	055310	001017			BNE 24\$; 5143	
9015	055312	004767	162014		JSR PC,ISOLATE ; 5202	
9016	055316	104455			TRAP 55 ; 5203	
9017	055320	000320			.WORD 320	
9018	055322	011120			.WORD MSG2	
9019	055324	000000			.WORD 0	
9020	055326	112765	000007	034446	MOVB #7,WHY.DROPT(R5) ; *,*(LUN) 5204	
9021	055334	010500			MOV R5,R0 ; LUN,* 5205	
9022	055336	104451			TRAP 51	
9023	055340	062706	000022	22\$:	ADD #22,SP ; 5206	
9024	055344	000167	000406	23\$:	JMP 32\$	
9025	055350	020327	000005	24\$:	CMP R3,#5 ; VALUE,* 5143	
9026	055354	001157			BNE 31\$	
9027	055356	004767	161750		JSR PC,ISOLATE ; 5211	
9028	055362	032767	000001	124670	BIT #1,ERROUT ; 5213	
9029	055370	001423			BEQ 25\$	
9030	055372	017700	157012		MOV @ML.REG+42,R0	
9031	055376	006200			ASR R0	
9032	055400	006200			ASR R0	
9033	055402	006200			ASR R0	
9034	055404	006200			ASR R0	
9035	055406	006200			ASR R0	
9036	055410	006200			ASR R0	
9037	055412	042700	177700		BIC #177700,R0	
9038	055416	010046			MOV R0,-(SP)	
9039	055420	012746	006640		MOV #FMT10B,-(SP)	
9040	055424	012746	000002		MOV #2,-(SP)	
9041	055430	010600			MOV SP,R0 ; SP,*	
9042	055432	104414			TRAP 14	
9043	055434	062706	000006		ADD #6,SP	
9044	055440	017766	156746	000052	25\$:	MOV @ML.REG+44,52(SP) ; *,OLDSEC 5216
9045	055446	017700	156736		MOV @ML.REG+42,R0 ; 5217	
9046	055452	006200			ASR R0	

Address	Label	OpCode	Operand 1	Operand 2	Comment	Page
9048						
9049						
9050						
9051	055454	006200	ASR	RO		
9052	055456	006200	ASR	RO		
9053	055460	006200	ASR	RO		
9054	055462	006200	ASR	RO		
9055	055464	006200	ASR	RO		
9056	055466	042700	BIC	#177700,RO		
9057	055472	010066	MOV	RO,54(SP)	: *,OLDCHN	
9058	055476	012746	MOV	#1,-(SP)	:	5219
9059	055502	016646	MOV	46(SP),-(SP)	: COMMAND,*	
9060	055506	010546	MOV	R5,-(SP)	: LUN,*	
9061	055510	010246	MOV	R2,-(SP)	: WRDCNT,*	
9062	055512	016646	MOV	56(SP),-(SP)	: PTR,*	
9063	055516	010446	MOV	R4,-(SP)	: SECTOR,*	
9064	055520	004767	JSR	PC,RETRY		
9065	055524	062706	ADD	#14,SP		
9066	055530	020027	CMP	RO,#5		
9067	055534	001051	BNE	28\$		
9068	055536	027766	CMP	@ML.REG+44,52(SP)	: *,OLDSEC	5222
9069	055544	001034	BNE	27\$		
9070	055546	016646	MOV	54(SP),-(SP)	: OLDCHN,*	
9071	055552	017700	MOV	@ML.REG+42,RO		
9072	055556	006200	ASR	RO		
9073	055560	006200	ASR	RO		
9074	055562	006200	ASR	RO		
9075	055564	006200	ASR	RO		
9076	055566	006200	ASR	RO		
9077	055570	006200	ASR	RO		
9078	055572	042700	BIC	#177700,RO		
9079	055576	020026	CMP	RO,(SP)+		
9080	055600	001016	BNE	27\$		
9081	055602	032767	BIT	#1,ERROUT	:	5226
9082	055610	001404	BEQ	26\$		
9083	055612	104456	TRAP	56		
9084	055614	000321	.WORD	321		
9085	055616	011202	.WORD	MSG4		
9086	055620	000000	.WORD	0		
9087	055622	010546	MOV	R5,-(SP)	: LUN,*	5228
9088	055624	016746	MOV	BOARD,-(SP)		
9089	055630	004767	JSR	PC,UP.HARD.COUNT		
9090	055634	000426	BR	30\$:	5222
9091	055636	032767	BIT	#1,ERROUT	:	5233
9092	055644	001415	BEQ	29\$		
9093	055646	104457	TRAP	57		
9094	055650	000322	.WORD	322		
9095	055652	011166	.WORD	MSG3		
9096	055654	000000	.WORD	0		
9097	055656	000410	BR	29\$:	5235
9098	055660	032767	BIT	#1,ERROUT	:	5241
9099	055666	001404	BEQ	29\$		
9100	055670	104457	TRAP	57		
9101	055672	000323	.WORD	323		
9102	055674	011166	.WORD	MSG3		

Address	Label	OpCode	Operand	Comment	Address	Label	OpCode	Operand	Comment
9104					27-Mar-1982 19:24:42	TOPS			
9105					27-Mar-1982 19:23:44	PA:<			
9106									
9107	055676	000000							
9108	055700	010546							
9109	055702	016746	156434						
9110	055706	004767	162172						
9111	055712	022626							5210
9112	055714	010200							5249
9113	055716	006300							
9114	055720	066700	154744						
9115	055724	010067	154740						
9116	055730	010216							
9117	055732	012746	000400						
9118	055736	004767	127226						
9119	055742	060400							
9120	055744	010004							
9121	055746	062706	000024						5082
9122	055752	000167	176416						5081
9123	055756	005205							5068
9124	055760	020566	000034						
9125	055764	002002							
9126	055766	000167	176304						
9127	055772	005767	154676						
9128	055776	001402							
9129	056000	000167	002014						
9130	056004	016766	124002	000034					5299
9131	056012	005005							
9132	056014	000167	001766						
9133	056020	010500							
9134	056022	006200							
9135	056024	006200							
9136	056026	006200							
9137	056030	062700	034442						
9138	056034	010046							
9139	056036	010546							
9140	056040	042716	177770						
9141	056044	012746	000001						
9142	056050	005046							
9143	056052	004767	126300						
9144	056056	062706	000010						
9145	056062	005300							
9146	056064	001402							
9147	056066	000167	001246						
9148	056072	010567	123776						
9149	056076	012767	012670	154564					5307
9150	056104	012767	022670	154560					5308
9151	056112	010501							5309
9152	056114	006301							5310
9153	056116	016104	034460						
9154	056122	020461	034500						
9155	056126	101357							
9156	056130	010446							
9157	056132	016146	034500						
9158	056136	004767	171034						

Address	Code	Operand 1	Operand 2	Label	Comments	Address	Code	Operand 1	Operand 2	Label	Comments
9160						9160					
9161						9161					
9162						9162					
9163	056142	010002			MOV R0,R2	9163					TOPS
9164	056144	010216			MOV R2,(SP)	9164					PA:<
9165	056146	004767	170760		JSR PC,SET_PTRS	9165					5315
9166	056152	004767	170010		JSR PC,CHOOSE	9166					5316
9167	056156	010066	000026		MOV R0,26(SP)	9167					
9168	056162	012766	000001	000032	MOV #1,32(SP)	9168					5318
9169	056170	005066	000024		CLR 24(SP)	9169					5321
9170	056174	026627	000026	045612	CMP 26(SP),#READ	9170					
9171	056202	001015			BNE 41\$	9171					
9172	056204	005266	000024		INC 24(SP)	9172					
9173	056210	016766	154456	000030	MOV RPTR,30(SP)	9173					5324
9174	056216	010546			MOV R5,-(SP)	9174					5325
9175	056220	010246			MOV R2,-(SP)	9175					
9176	056222	016746	154444		MOV RPTR,-(SP)	9176					
9177	056226	010446			MOV R4,-(SP)	9177					
9178	056230	004767	167356		JSR PC,READ	9178					
9179	056234	000412			BR 42\$	9179					
9180	056236	016766	154426	000030	41\$: MOV WPTR,30(SP)	9180					5329
9181	056244	010546			MOV R5,-(SP)	9181					5330
9182	056246	010246			MOV R2,-(SP)	9182					
9183	056250	016746	154414		MOV WPTR,-(SP)	9183					
9184	056254	010446			MOV R4,-(SP)	9184					
9185	056256	004767	167516		JSR PC,CHECK	9185					
9186	056262	010003			42\$: MOV R0,R3	9186					
9187	056264	001076			BNE 43\$	9187					5337
9188	056266	032766	000001	000034	BIT #1,34(SP)	9188					5342
9189	056274	001513			BEQ 44\$	9189					
9190	056276	016746	154366		MOV WPTR,-(SP)	9190					5346
9191	056302	016746	154364		MOV RPTR,-(SP)	9191					
9192	056306	010246			MOV R2,-(SP)	9192					
9193	056310	004767	166012		JSR PC,DOUBLE.CHECK	9193					
9194	056314	062706	000006		ADD #6,SP	9194					
9195	056320	010066	000052		MOV R0,52(SP)	9195					
9196	056324	001477			BEQ 44\$	9196					
9197	056326	010546			MOV R5,-(SP)	9197					5349
9198	056330	004767	157170		JSR PC,SAYWHO	9198					
9199	056334	012716	011216		MOV #MSG5,(SP)	9199					5350
9200	056340	012746	007072		MOV #SAY1,-(SP)	9200					
9201	056344	012746	000002		MOV #2,-(SP)	9201					
9202	056350	010600			MOV SP,R0	9202					
9203	056352	104414			TRAP 14	9203					
9204	056354	016616	000060		MOV 60(SP),(SP)	9204					5352
9205	056360	017646	000060		MOV @60(SP),-(SP)	9205					
9206	056364	012746	006710		MOV #FMT12A,-(SP)	9206					
9207	056370	012746	000003		MOV #3,-(SP)	9207					
9208	056374	010600			MOV SP,R0	9208					
9209	056376	104414			TRAP 14	9209					
9210	056400	062766	010000	000066	ADD #10000,66(SP)	9210					5354
9211	056406	016616	000066		MOV 66(SP),(SP)	9211					5355
9212	056412	017646	000066		MOV @66(SP),-(SP)	9212					
9213	056416	012746	006756		MOV #FMT12B,-(SP)	9213					
9214	056422	012746	000003		MOV #3,-(SP)	9214					

Address	Hex	Dec	Label	Op	Opnd	Comment	Seq
9216							
9217							
9218							
9219	056426	010600		MOV	SP,R0	: SP,*	
9220	056430	104414		TRAP	14		
9221	056432	112765	000010	MOV	#10,WHY.DROPT(R5)	: *,*(LUN)	5357
9222	056440	104455		TRAP	55	:	5358
9223	056442	000324		.WORD	324		
9224	056444	011070		.WORD	MSG1		
9225	056446	000000		.WORD	0		
9226	056450	010500		MOV	R5,R0	: LUN,*	5359
9227	056452	104451		TRAP	51		
9228	056454	062706	000036	ADD	#36,SP	:	5360
9229	056460	000510		BR	50\$		
9230	056462	020327	000001	43\$: CMP	R3,#1	: VALUE,*	5337
9231	056466	001033		BNE	46\$		
9232	056470	012746	000006	MOV	#6,-(SP)	:	5368
9233	056474	016646	000040	MOV	40(SP),-(SP)	: COMMAND,*	
9234	056500	010546		MOV	R5,-(SP)	: LUN,*	
9235	056502	010246		MOV	R2,-(SP)	: WRDCNT,*	
9236	056504	016646	000050	MOV	50(SP),-(SP)	: PTR,*	
9237	056510	010446		MOV	R4,-(SP)	: SECTOR,*	
9238	056512	004767	167506	JSR	PC,RETRY		
9239	056516	062706	000014	ADD	#14,SP		
9240	056522	005700		TST	R0		
9241	056524	001002		44\$: BNE	45\$		
9242	056526	000167	000520	JMP	58\$		
9243	056532	112765	000004	45\$: MOV	#4,WHY.DROPT(R5)	: *,*(LUN)	5371
9244	056540	104455		TRAP	55	:	5372
9245	056542	000325		.WORD	325		
9246	056544	011070		.WORD	MSG1		
9247	056546	000000		.WORD	0		
9248	056550	010500		MOV	R5,R0	: LUN,*	5373
9249	056552	104451		TRAP	51		
9250	056554	000450		BR	49\$		5374
9251	056556	020327	000002	46\$: CMP	R3,#2	: VALUE,*	5337
9252	056562	001012		BNE	47\$		
9253	056564	112765	000005	MOV	#5,WHY.DROPT(R5)	: *,*(LUN)	5381
9254	056572	104455		TRAP	55	:	5382
9255	056574	000326		.WORD	326		
9256	056576	011070		.WORD	MSG1		
9257	056600	000000		.WORD	0		
9258	056602	010500		MOV	R5,R0	: LUN,*	5383
9259	056604	104451		TRAP	51		
9260	056606	000433		BR	49\$		5384
9261	056610	020327	000003	47\$: CMP	R3,#3	: VALUE,*	5337
9262	056614	001012		BNE	48\$		
9263	056616	104455		TRAP	55	:	5389
9264	056620	000327		.WORD	327		
9265	056622	011070		.WORD	MSG1		
9266	056624	000000		.WORD	0		
9267	056626	112765	000006	MOV	#6,WHY.DROPT(R5)	: *,*(LUN)	5390
9268	056634	010500		MOV	R5,R0	: LUN,*	5391
9269	056636	104451		TRAP	51		
9270	056640	000416		BR	49\$:	5392

Address	OpCode	Operand1	Operand2	Operand3	Label	Instruction	Comment	Value
9272								
9273								
9274								
9275	056642	020327	000004		48\$:	CMP R3,#4	: VALUE,*	5337
9276	056646	001017				BNE 51\$		
9277	056650	004767	160456			JSR PC,ISOLATE		5397
9278	056654	104455				TRAP 55		5398
9279	056656	000330				.WORD 330		
9280	056660	011120				.WORD MSG2		
9281	056662	000000				.WORD 0		
9282	056664	112765	000007	034446		MOVB #7,WHY.DROPT(R5)	: *,*(LUN)	5399
9283	056672	010500				MOV R5,R0	: LUN,*	5400
9284	056674	104451				TRAP 51		
9285	056676	062706	000014		49\$:	ADD #14,SP		5401
9286	056702	000167	001076		50\$:	JMP 71\$		
9287	056706	020327	000005		51\$:	CMP R3,#5	: VALUE,*	5337
9288	056712	001157				BNE 58\$		
9289	056714	004767	160412			JSR PC,ISOLATE		5406
9290	056720	032767	000001	123332		BIT #1,ERROUT		5408
9291	056726	001423				BEQ 52\$		
9292	056730	017700	155454			MOV @ML.REG+42,R0		
9293	056734	006200				ASR R0		
9294	056736	006200				ASR R0		
9295	056740	006200				ASR R0		
9296	056742	006200				ASR R0		
9297	056744	006200				ASR R0		
9298	056746	006200				ASR R0		
9299	056750	042700	177700			BIC #177700,R0		
9300	056754	010046				MOV R0,-(SP)		
9301	056756	012746	006640			MOV #FMT10B,-(SP)		
9302	056762	012746	000002			MOV #2,-(SP)		
9303	056766	010600				MOV SP,R0	: SP,*	
9304	056770	104414				TRAP 14		
9305	056772	062706	000006			ADD #6,SP		
9306	056776	017766	155410	000044	52\$:	MOV @ML.REG+44,44(SP)	: *,OLDSEC	5410
9307	057004	017700	155400			MOV @ML.REG+42,R0		5411
9308	057010	006200				ASR R0		
9309	057012	006200				ASR R0		
9310	057014	006200				ASR R0		
9311	057016	006200				ASR R0		
9312	057020	006200				ASR R0		
9313	057022	006200				ASR R0		
9314	057024	042700	177700			BIC #177700,R0		
9315	057030	010066	000046			MOV R0,46(SP)	: *,OLDCHN	
9316	057034	012746	000001			MOV #1,-(SP)		5413
9317	057040	016646	000040			MOV 40(SP),-(SP)	: COMMAND,*	
9318	057044	010546				MOV R5,-(SP)	: LUN,*	
9319	057046	010246				MOV R2,-(SP)	: WRDCNT,*	
9320	057050	016646	000050			MOV 50(SP),-(SP)	: PTR,*	
9321	057054	010446				MOV R4,-(SP)	: SECTOR,*	
9322	057056	004767	167142			JSR PC,RETRY		
9323	057062	062706	000014			ADD #14,SP		
9324	057066	020027	000005			CMP R0,#5		
9325	057072	001051				BNE 55\$		
9326	057074	027766	155312	000044		CMP @ML.REG+44,44(SP)	: *,OLDSEC	5416

Address	OpCode	Operand 1	Operand 2	Label	Comment	Definition	Register	Value
9384					:MLX4			
9385					:	DEFINITION OF OPTION 2		
9386								
9387	057326	010004				MOV R0,R4	R0	
9388	057330	062706	000006			ADD #6,SP	R0	5313
9389	057334	000167	176562			JMP 39\$	R0	5312
9390	057340	032767	000001	122714	60\$:	BIT #1,EFN\$21	R0	5459
9391	057346	001002				BNE 61\$	R0	
9392	057350	000167	000430			JMP 71\$	R0	
9393	057354	010500			61\$:	MOV R5,R0	R5	5466
9394	057356	006300				ASL R0	R0	
9395	057360	016066	034500	000526		MOV TOP.SECT(R0),26(SP)	R0	
9396	057366	016001	034460			MOV LOW.SECT(R0),R1	R0	
9397	057372	000577				BR 70\$	R0	
9398	057374	010546			62\$:	MOV R5,-(SP)	R5	5468
9399	057376	012746	000400			MOV #400,-(SP)	R5	
9400	057402	012746	022670			MOV #RBUF,-(SP)	R5	
9401	057406	010146				MOV R1,-(SP)	R1	
9402	057410	004767	166176			JSR PC,READ	R0	
9403	057414	062706	000010			ADD #10,SP	R0	
9404	057420	020027	000005			CMP R0,#5	R0	
9405	057424	001161				BNE 69\$	R0	
9406	057426	004767	157700			JSR PC,ISOLATE	R0	5471
9407	057432	032767	000001	122620		BIT #1,ERROUT	R0	5473
9408	057440	001423				BEQ 63\$	R0	
9409	057442	017700	154742			MOV @ML.REG+42,R0	R0	
9410	057446	006200				ASR R0	R0	
9411	057450	006200				ASR R0	R0	
9412	057452	006200				ASR R0	R0	
9413	057454	006200				ASR R0	R0	
9414	057456	006200				ASR R0	R0	
9415	057460	006200				ASR R0	R0	
9416	057462	042700	177700			BIC #177700,R0	R0	
9417	057466	010046				MOV R0,-(SP)	R0	
9418	057470	012746	006640			MOV #FMT10B,-(SP)	R0	
9419	057474	012746	000002			MOV #2,-(SP)	R0	
9420	057500	010600				MOV SP,R0	R0	
9421	057502	104414				TRAP 14	R0	
9422	057504	062706	000006			ADD #6,SP	R0	
9423	057510	017766	154676	000030	63\$:	MOV @ML.REG+44,30(SP)	R0	5482
9424	057516	017700	154666			MOV @ML.REG+42,R0	R0	5483
9425	057522	006200				ASR R0	R0	
9426	057524	006200				ASR R0	R0	
9427	057526	006200				ASR R0	R0	
9428	057530	006200				ASR R0	R0	
9429	057532	006200				ASR R0	R0	
9430	057534	006200				ASR R0	R0	
9431	057536	042700	177700			BIC #177700,R0	R0	
9432	057542	010066	000032			MOV R0,32(SP)	R0	
9433	057546	012746	000001			MOV #1,-(SP)	R0	5491
9434	057552	016646	000024			MOV 24(SP),-(SP)	R0	
9435	057556	010546				MOV R5,-(SP)	R5	
9436	057560	012746	000400			MOV #400,-(SP)	R5	
9437	057564	016646	000034			MOV 34(SP),-(SP)	R5	
9438	057570	016646	000042			MOV 42(SP),-(SP)	R5	

27-Mar-1982 19:24:42 TOPS
27-Mar-1982 19:23:44 PA:<

9496
9497
9498
9499 060020 062706 000010
9500 060024 005266 000030
9501 060030 026666 000030 000032
9502 060036 003002
9503 060040 000167 174110
9504 060044 062706 000036
9505 060050 000207
9506
9507
9508
9513
9514

:MLX4
:

DEFINITION OF OPTION 2

73\$: ADD #10,SP
74\$: INC 30(SP)
CMP 30(SP),32(SP)
BGT 75\$
JMP 3\$
75\$: ADD #36,SP
RTS PC

27-Mar-1982 19:24:42 TOPS
27-Mar-1982 19:23:44 PA:<
5052
5051
4982

: Routine Size: 1023 words
: Maximum stack depth per invocation: 43 words

9516 :MLX4
 9517 :
 9518 :
 9519 :
 9520 :
 9521 :
 9522 :
 9523 :
 9524 :
 9525 :
 9526 :
 9527 :
 9528 :
 9529 :
 9530 :
 9531 :
 9532 :
 9533 :
 9534 :
 9535 :
 9536 :
 9537 :
 9538 :
 9539 :
 9540 :
 9541 :
 9542 :
 9543 :
 9544 :
 9545 :
 9546 :
 9547 :
 9548 :
 9549 :
 9550 :
 9551 :
 9552 :
 9553 :
 9554 :
 9555 :
 9556 :
 9557 :
 9558 :
 9559 :
 9560 :
 9561 :
 9562 :
 9563 :
 9564 :
 9565 :
 9566 :
 9567 :
 9568 :
 9569 :
 9570 :

```

DEFINITION OF OPTION 3
5555 %sbttl 'DEFINITION OF OPTION 3'
5556 routine OPT3 : novalue =
5557     begin
5558
5559     ++
5560     ROUTINE:      OPT3
5561
5562     PURPOSE:      TO DO A UNIQUE DATA CHECK ON ALL AVAILABLE UNITS.
5563
5564     THE CODE FOR 'OPT3' IN BRIEF:
5565
5566     BEGIN 1 (START OF ROUTINE)
5567     SAY ROUTINE IS RUNNING
5568     INCR COMPLEMENT FLAG FROM 0 TO 1
5569     : BEGIN 2 (START OF COMPLEMENT FLAG SELECTION LOOP)
5570     : GENERATE THE PATTERN
5571     : INCR LUN FROM 0 TO LAST
5572     : : BEGIN 3 (START OF LOGICAL UNIT SELECTION LOOP)
5573     : : TESTLOOP:
5574     : : : BEGIN 4 (START OF LOOP THAT COMPLETELY TESTS 1 UNIT)
5575     : : : IF UNIT IS ACTIVE
5576     : : : THEN
5577     : : : : BEGIN 5 (START OF TEST FOR AN ACTIVE UNIT)
5578     : : : : INITIALIZE THE WRITE AND READ BUFFER POINTERS
5579     : : : : SECTOR = LOWEST
5580     : : : : WHILE SECTOR LEQ HIGHEST DO
5581     : : : : : BEGIN 6 (START OF SECTOR SELECTION LOOP)
5582     : : : : : GET WRDCNT
5583     : : : : : SET UP BUFFER POINTERS BEFORE TRANSFER
5584     : : : : : WRITE
5585     : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
5586     : : : : : CHOOSE WHETHER TO WRITE CHECK OR READ
5587     : : : : : DO THE WRITE CHECK OR READ
5588     : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
5589     : : : : : CHANGE BUFFER POINTERS AFTER TRANSFER
5590     : : : : : UPDATE SECTOR NUMBER BY NUMBER OF SECTORS IN PREVIOUS TRANSFER
5591     : : : : : END 6 (END OF SECTOR SELECTION LOOP)
5592     : : : : : END 5 (END OF TEST FOR AN ACTIVE UNIT)
5593     : : : : : END 4 (END OF TESTLOOP)
5594     : : : : : END 3 (END OF LOGICAL UNIT SELECTION LOOP)
5595     : : : : : END 2 (END OF COMPLEMENT FLAG SELECTION LOOP)
5596     RETURN
5597     END 1 (END OF ROUTINE)
5598
5599
5600     label
5601     LOOP;
5602
5603     local
5604     WRDCNT,
5605     VALUE,
5606     OLDSEC,
  
```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (39)

!* 1 * START OF ROUTINE

```

9572 :MLX4
9573 :
9574 :
9575 : 5607 OLDCHN,
9576 : 5608 SECTOR,
9577 : 5609 PTR,
9578 : 5610 COMMAND,
9579 : 5611 DBL_VALUE;
9580 :
9581 : 5613 PRINTB (SAY2, WRD34, RTN3);
9582 : 5614 !'RUNNING OPT3'
9583 : 5615
9584 : 5616 incr COMP_FLAG from 0 to 1 do
9585 : 5617 begin !* 2 * START OF COMPLEMENT FLAG SELECTION LOOP
9586 : 5618 GEN3 (.COMP_FLAG);
9587 : 5619
9588 : 5620 incr LUN from 0 to (.LSUNIT - 1) do
9589 : 5621 LOOP : begin !* 3 * START OF LOGICAL UNIT SELECTION LOOP
9590 : 5622 !* 4 * START OF THE LOOP THAT COMPLETELY TESTS 1 UNIT
9591 : 5623
9592 : 5624 if .DRIVE_STATUS [.LUN] eql ACTIVE
9593 : 5625 then !* 5 * START OF TEST FOR AN ACTIVE UNIT
9594 : 5626
9595 : 5627 begin
9596 : 5628 L$LUN = .LUN;
9597 : 5629 WPTR = W$BUFF;
9598 : 5630 RPTR = R$BUFF;
9599 : 5631 SECTOR = LOWEST;
9600 : 5632
9601 : 5633 while .SECTOR lequ HIGHEST do !* 6 * START OF SECTOR SELECTION LOOP
9602 : 5634 begin
9603 : 5635 WRDCNT = GET WRDCNT (.SECTOR, HIGHEST);
9604 : 5636 SET PTRS (.WRDCNT);
9605 : 5637 VALUE = write (.LUN, .WRDCNT, .WPTR, .SECTOR);
9606 : 5638
9607 : 5639 !+
9608 : 5640 ! SEE HOW SUCCESSFUL THE WRITE WAS:
9609 : 5641 !-
9610 : 5642
9611 : 5643 selectone .VALUE of !SEE 'SYSERR' FOR DEFINITION
9612 : 5644 set !OF ERROR # CONTAINED IN 'VALUE'
9613 : 5645
9614 : 5646 [1] :
9615 : 5647 begin !* 6A * RETRY ALLOWED
9616 : 5648
9617 : 5649 if RETRY (SIX, write, .LUN, .WRDCNT, .WPTR, .SECTOR) neq 0
9618 : 5650 then !THE RETRY FAILED -- SYSTEM FATAL ERROR
9619 : 5651 begin
9620 : 5652 WHY_DROPT [.LUN] = CODE_4;
9621 : 5653 ERRDF (301, MSG1, 0); !**** OPTION 3 ERROR 01 ****
9622 : 5654 DODU (.LUN);
9623 : 5655 leave LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
9624 : 5656 end;
9625 : 5657
9626 : 5658 end; !* 6A *

```

27-Mar-1982 19:24:42
27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
PA:<NEALE>MLX4.BLI.5 (39)

9628 :MLX4

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (39)

DEFINITION OF OPTION 3

```

9629 :
9630 :
9631 : 5659
9632 : 5660
9633 : 5661
9634 : 5662
9635 : 5663
9636 : 5664
9637 : 5665
9638 : 5666
9639 : 5667
9640 : 5668
9641 : 5669
9642 : 5670
9643 : 5671
9644 : 5672
9645 : 5673
9646 : 5674
9647 : 5675
9648 : 5676
9649 : 5677
9650 : 5678
9651 : 5679
9652 : 5680
9653 : 5681
9654 : 5682
9655 : 5683
9656 : 5684
9657 : 5685
9658 : 5686
9659 : 5687
9660 : 5688
9661 : 5689
9662 : 5690
9663 : 5691
9664 : 5692
9665 : 5693
9666 : 5694
9667 : 5695
9668 : 5696
9669 : 5697
9670 : 5698
9671 : 5699
9672 : 5700
9673 : 5701
9674 : 5702
9675 : 5703
9676 : 5704
9677 : 5705
9678 : 5706
9679 : 5707
9680 : 5708
9681 : 5709
9682 : 5710

[2] :
begin
WHY DROPT [.LUN] = CODE_5;
ERRDF (302, MSG1, 0);
DODU (.LUN);
leave LOOP;
end;

[3] :
begin
ERRDF (303, MSG1, 0);
WHY DROPT [.LUN] = CODE_6;
DODD (.LUN);
leave LOOP;
end;

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
set

[0] :
if .COMMAND eql read
then
begin
if (DBL_VALUE = DOUBLE_CHECK (.WPTR, .RPTR, .WRDCNT)) neq 0
then
begin
SAYWHO (.LUN);
PRINTB (SAY1, MSG5);
PRINTB (FMT12A, .DBL_VALUE, .DBL_VALUE);
end;
end;
end;

```

9684 :MLX4
9685 :
9686 :
9687 :
9688 :
9689 :
9690 :
9691 :
9692 :
9693 :
9694 :
9695 :
9696 :
9697 :
9698 :
9699 :
9700 :
9701 :
9702 :
9703 :
9704 :
9705 :
9706 :
9707 :
9708 :
9709 :
9710 :
9711 :
9712 :
9713 :
9714 :
9715 :
9716 :
9717 :
9718 :
9719 :
9720 :
9721 :
9722 :
9723 :
9724 :
9725 :
9726 :
9727 :
9728 :
9729 :
9730 :
9731 :
9732 :
9733 :
9734 :
9735 :
9736 :
9737 :
9738 :

DEFINITION OF OPTION 3

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (39)

```

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 (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 *

[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 ();
ERRDF (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 ERROR

```

```

9740 :MLX4
9741 :
9742 :      DEFINITION OF OPTION 3
9743 :      5763
9744 :      5764
9745 :      5765
9746 :      5766
9747 :      5767
9748 :      5768
9749 :      5769
9750 :      5770
9751 :      5771
9752 :      5772
9753 :      5773
9754 :      5774
9755 :      5775
9756 :      5776
9757 :      5777
9758 :      5778
9759 :      5779
9760 :      5780
9761 :      5781
9762 :      5782
9763 :      5783
9764 :      5784
9765 :      5785
9766 :      5786
9767 :      5787
9768 :      5788
9769 :      5789
9770 :      5790
9771 :      5791
9772 :      5792
9773 :      5793
9774 :      5794
9775 :      5795
9776 :      5796
9777 :      5797
9778 :      5798
9779 :      5799
9780 :      5800
9781 :      5801
9782 :      5802
9783 :      5803
9784 :      5804
9785 :      5805
9786 :      5806
9787 :      5807
9788 :      5808
9789 :      5809
9790 :      5810
9791 :      5811
9792 :      5812
9793 :      5813
9794 :      5814

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 ****
      UP_SOFT_COUNT (.LUN, .BOARD);
    end;
  end;
tes;
!* 6H *
WPTR = .WPTR + (.WRDCNT*2);
SECTOR = .SECTOR + (.WRDCNT/256);
end;
!* 6 * END OF SECTOR SELECTION LOOP
end;
!* 5 * END OF TEST FOR AN ACTIVE UNIT

+
Test to see if this uut's address space is
to be read for soft errors. This test is
is intended for DMT purposes.
-

if .EFNS21
then
!*Is the background pattern to be read

```

27-Mar-1982 19:24:42
27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
PA:<NEALE>MLX4.BLI.5 (39)

9796 :MLX4

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (39)

DEFINITION OF OPTION 3

```

9797 :
9798 :
9799 :           begin
9800 :
9801 :           :
9802 :           : version czmlbb changed incr to incru
9803 :           :
9804 :           : incru SECTOR from LOWEST to HIGHEST do
9805 :           :
9806 :           :   if read (.LUN, 256, RBUFF, .SECTOR) eql 5
9807 :           :   then
9808 :           :     begin
9809 :           :       ISOLATE ();           !Find the failing bank and board no.
9810 :           :
9811 :           :       if .ERROUT then PRINTB (FMT10B, .CHAN);
9812 :           :
9813 :           :       ! Print where the error is
9814 :           :
9815 :           :       ! Save the contents of the ML error location
9816 :           :       ! register so we can compare it to the new
9817 :           :       ! contents of this register after the retry.
9818 :           :       ! This is done to classify the error.
9819 :           :
9820 :           :       OLDSEC = .MLEL;
9821 :           :       OLDCHN = .CHAN;
9822 :           :
9823 :           :       ! Do a classify retry call. If the same error
9824 :           :       ! occurs then classify it as a hard error. If
9825 :           :       ! a different error occurred or the error went away
9826 :           :       ! then classify it as a soft error.
9827 :           :
9828 :           :
9829 :           :       if RETRY (ONE, .COMMAND, .LUN, 256, .PTR, .OLDSEC) eql 5
9830 :           :       then
9831 :           :
9832 :           :       ! The same error occurred so see if it is at the same
9833 :           :       ! sector and channel number, if so then classify
9834 :           :       ! it as a hard error else classiy it as a soft
9835 :           :       ! error.
9836 :           :
9837 :           :
9838 :           :       if ((.MLEL eql .OLDSEC) and (.CHAN eql .OLDCHN))
9839 :           :       then
9840 :           :         begin           !Same error occurred 'hard'
9841 :           :
9842 :           :         if .ERROUT       !Print error if enabled
9843 :           :         then
9844 :           :           begin
9845 :           :             ERRHRD (312,           !Error number
9846 :           :             MSG4,           !Error message
9847 :           :             0);           !Additional message routine
9848 :           :           end;
9849 :           :
9850 :           :           UP_HARD_COUNT (.LUN, .BOARD);
    
```

P 5861
 P 5862

9852 :MLX4

DEFINITION OF OPTION 3

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (39)

```

9853 :
9854 :
9855 :           5867           end
9856 :           5868           else
9857 :           5869           begin           !Not the same error 'soft'
9858 :           5870
9859 :           5871           if .ERROUT           !Print error if enabled
9860 :           5872           then
9861 :           5873           begin
9862 :           P 5874           ERRSOFT (313,           !Error number
9863 :           P 5875           MSG3,           !Error message
9864 :           5876           0);           !Additional message routine
9865 :           5877           end;
9866 :           5878           UP_SOFT_COUNT (.LUN, .BOARD);
9867 :           5879           end
9868 :           5880
9869 :           5881           else           !Not the same error 'soft'
9870 :           5882           begin
9871 :           5883           if .ERROUT           !Print error if enabled
9872 :           5884           then
9873 :           5885           begin
9874 :           5886           ERRSOFT (314,           !Error number
9875 :           P 5887           MSG3,           !Error message
9876 :           P 5888           0);           !Additional message routine
9877 :           P 5889           end;
9878 :           5890           UP_SOFT_COUNT (.LUN, .BOARD);
9879 :           5891           end;
9880 :           5892           end;
9881 :           5893           end;
9882 :           5894           end;
9883 :           5895           end;
9884 :           5896           end;
9885 :           5897           end;
9886 :           5898           end;
9887 :           5899           end;
9888 :           5900           end;           !* 4 * END OF TESTLOOP
9889 :           5901           end;           !* 3 * END OF LOGICAL UNIT SELECTION LOOP
9890 :           5902           end;           !* 2 * END OF COMPLEMENT FLAG SELECTION LOOP
9891 :           5903           end;
9892 :           5904           return;
9893 :           5905           end;           !* 1 * END OF ROUTINE
9894 :           5906
9898 :
9899 :

```

9903	060052	004167	125256	OPT3:	.SBTTL	OPT3 DEFINITION OF OPTION 3		5556
9904	060056	162706	000020		JSR	R1,\$SAVE5	:	
9905	060062	012746	007642		SUB	#20,SP	:	5613
9906	060066	012746	007344		MOV	#RTN3,-(SP)	:	
					MOV	#WRD34,-(SP)	:	

Address	OpCode	Operand1	Operand2	Label	Instruction	Comments	Address
9908							
9909							
9910							
9911	060072	012746	007100		MOV #SAY2,-(SP)		
9912	060076	012746	000003		MOV #3,-(SP)		
9913	060102	010600			MOV SP,R0	: SP,*	
9914	060104	104414			TRAP 14		
9915	060106	005066	000026		CLR 26(SP)	: COMP.FLAG	5616
9916	060112	016646	000026	1\$:	MOV 26(SP),-(SP)	: COMP.FLAG,*	5618
9917	060116	004767	161344		JSR PC,GEN3		
9918	060122	016766	121664	000014	MOV L\$UNIT,14(SP)		5620
9919	060130	005066	000012		CLR 12(SP)	: LUN	
9920	060134	000167	002240		JMP 41\$		
9921	060140	016600	000012	2\$:	MOV 12(SP),R0	: LUN,*	5625
9922	060144	006200			ASR R0		
9923	060146	006200			ASR R0		
9924	060150	006200			ASR R0		
9925	060152	062700	034442		ADD #DRIVE.STATUS,R0		
9926	060156	010046			MOV R0,-(SP)		
9927	060160	016646	000014		MOV 14(SP),-(SP)	: LUN,*	
9928	060164	042716	177770		BIC #177770,(SP)		
9929	060170	012746	000001		MOV #1,-(SP)		
9930	060174	005046			CLR -(SP)		
9931	060176	004767	124154		JSR PC,BL\$GT2		
9932	060202	062706	000010		ADD #10,SP		
9933	060206	005300			DEC R0		
9934	060210	001402			BEQ 4\$		
9935	060212	000167	001502	3\$:	JMP 29\$		
9936	060216	016667	000012	121650	MOV 12(SP),L\$LUN	: LUN,*	5628
9937	060224	012767	012670	152436	MOV #L\$BUFF,WPTR	:	5629
9938	060232	012767	022670	152432	MOV #R\$BUFF,RPTR	:	5630
9939	060240	016605	000012		MOV 12(SP),R5	: LUN,*	5631
9940	060244	006305			ASL R5		
9941	060246	016503	034460		MOV LOW.SECT(R5),R3	: *,SECTOR	
9942	060252	020365	034500	5\$:	CMP R3,TOP.SECT(R5)	: SECTOR,*	5633
9943	060256	101355			BHI 3\$		
9944	060260	010346			MOV R3,-(SP)	: SECTOR,*	5635
9945	060262	016546	034500		MOV TOP.SECT(R5),-(SP)		
9946	060266	004767	166704		JSR PC,GET.WRDCNT		
9947	060272	010002			MOV R0,R2	: *,WRDCNT	
9948	060274	010216			MOV R2,(SP)	: WRDCNT,*	5636
9949	060276	004767	166630		JSR PC,SET.PTRS		
9950	060302	016616	000016		MOV 16(SP),(SP)	: LUN,*	5637
9951	060306	010246			MOV R2,-(SP)	: WRDCNT,*	
9952	060310	016746	152354		MOV WPTR,-(SP)		
9953	060314	010346			MOV R3,-(SP)	: SECTOR,*	
9954	060316	004767	165102		JSR PC,WRITE		
9955	060322	010004			MOV R0,R4	: *,VALUE	
9956	060324	020427	000001		CMP R4,#1	: VALUE,*	5643
9957	060330	001035			BNE 6\$		
9958	060332	012746	000006		MOV #6,-(SP)		5649
9959	060336	012746	045424		MOV #WRITE,-(SP)		
9960	060342	016646	000030		MOV 30(SP),-(SP)	: LUN,*	
9961	060346	010246			MOV R2,-(SP)	: WRDCNT,*	
9962	060350	016746	152314		MOV WPTR,-(SP)		

Address	OpCode	Op1	Op2	Op3	Label	Instruction	Comments	Page
9964								
9965								
9966								
9967	060354	010346				MOV R3,-(SP)	: SECTOR,*	
9968	060356	004767	165642			JSR PC,RETRY		
9969	060362	062706	000014			ADD #14,SP		
9970	060366	005700				TST R0		
9971	060370	001456				BEQ 9\$		
9972	060372	016601	000024			MOV 24(SP),R1	: LUN,*	5652
9973	060376	112761	000004	034446		MOVB #4,WHY.DROPT(R1)		
9974	060404	104455				TRAP 55	:	5653
9975	060406	000455				.WORD 455		
9976	060410	011070				.WORD MSG1		
9977	060412	000000				.WORD 0		
9978	060414	016600	000024			MOV 24(SP),R0	: LUN,*	5654
9979	060420	104451				TRAP 51		
9980	060422	000436				BR 8\$:	5655
9981	060424	020427	000002	6\$:		CMP R4,#2	: VALUE,*	5643
9982	060430	001015				BNE 7\$		
9983	060432	016601	000024			MOV 24(SP),R1	: LUN,*	5662
9984	060436	112761	000005	034446		MOVB #5,WHY.DROPT(R1)		
9985	060444	104455				TRAP 55	:	5663
9986	060446	000456				.WORD 456		
9987	060450	011070				.WORD MSG1		
9988	060452	000000				.WORD 0		
9989	060454	016600	000024			MOV 24(SP),R0	: LUN,*	5664
9990	060460	104451				TRAP 51		
9991	060462	000416				BR 8\$:	5665
9992	060464	020427	000003	7\$:		CMP R4,#3	: VALUE,*	5643
9993	060470	001016				BNE 9\$		
9994	060472	104455				TRAP 55	:	5670
9995	060474	000457				.WORD 457		
9996	060476	011070				.WORD MSG1		
9997	060500	000000				.WORD 0		
9998	060502	016601	000024			MOV 24(SP),R1	: LUN,*	5671
9999	060506	112761	000006	034446		MOVB #6,WHY.DROPT(R1)		
10000	060514	010100				MOV R1,R0	: LUN,*	5672
10001	060516	104451				TRAP 51		
10002	060520	062706	000012	8\$:		ADD #12,SP	:	5673
10003	060524	000543				BR 14\$		
10004	060526	004767	165434	9\$:		JSR PC,CHOOSE	:	5677
10005	060532	010066	000040			MOV R0,40(SP)	: *,COMMAND	
10006	060536	005001				CLR R1	:	5679
10007	060540	020027	045612			CMP R0,#READ	: COMMAND,*	
10008	060544	001015				BNE 10\$		
10009	060546	005201				INC R1		
10010	060550	016766	152116	000036		MOV RPTR,36(SP)	: *,PTR	5682
10011	060556	016646	000024			MOV 24(SP),-(SP)	: LUN,*	5683
10012	060562	010246				MOV R2,-(SP)	: WRDCNT,*	
10013	060564	016746	152102			MOV RPTR,-(SP)		
10014	060570	010346				MOV R3,-(SP)	: SECTOR,*	
10015	060572	004767	165014			JSR PC,READ		
10016	060576	000413				BR 11\$		
10017	060600	016766	152064	000036	10\$:	MOV WPTR,36(SP)	: *,PTR	5687
10018	060606	016646	000024			MOV 24(SP),-(SP)	: LUN,*	5688

Address	Label	OpCode	OpData	OpComment	Symbol	Value
10188						
10189						
10190						
10191	061550	104456		TRAP	56	
10192	061552	000465		.WORD	465	
10193	061554	011202		.WORD	MSG4	
10194	061556	000000		.WORD	0	
10195	061560	016646	000034	MOV	34(SP),-(SP)	: LUN,* 5780
10196	061564	016746	152552	MOV	BOARD,-(SP)	
10197	061570	004767	155626	JSR	PC,UP.HARD.COUNT	
10198	061574	000427		BR	27\$: 5774
10199	061576	032767	000001	BIT	#1,ERROUT	: 5785
10200	061604	001415		BEQ	26\$	
10201	061606	104457		TRAP	57	
10202	061610	000466		.WORD	466	
10203	061612	011166		.WORD	MSG3	
10204	061614	000000		.WORD	0	
10205	061616	000410		BR	26\$: 5787
10206	061620	032767	000001	BIT	#1,ERROUT	: 5793
10207	061626	001404		BEQ	26\$	
10208	061630	104457		TRAP	57	
10209	061632	000467		.WORD	467	
10210	061634	011166		.WORD	MSG3	
10211	061636	000000		.WORD	0	
10212	061640	016646	000034	MOV	34(SP),-(SP)	: LUN,* 5795
10213	061644	016746	152472	MOV	BOARD,-(SP)	
10214	061650	004767	156230	JSR	PC,UP.SOFT.COUNT	
10215	061654	022626		CMP	(SP)+,(SP)+	: 5762
10216	061656	010200		MOV	R2,R0	: WRDCNT,* 5801
10217	061660	006300		ASL	R0	
10218	061662	066700	151002	ADD	WPTR,R0	
10219	061666	010067	150776	MOV	R0,WPTR	
10220	061672	010216		MOV	R2,(SP)	: WRDCNT,* 5802
10221	061674	012746	000400	MOV	#400,-(SP)	
10222	061700	004767	123264	JSR	PC,BL\$DIV	
10223	061704	060300		ADD	R3,R0	: SECTOR,*
10224	061706	010003		MOV	R0,R3	: *,SECTOR
10225	061710	062706	000024	ADD	#24,SP	: 5634
10226	061714	000167	176332	JMP	5\$: 5633
10227	061720	032767	000001	BIT	#1,EFNS21	: 5813
10228	061726	001002		BNE	30\$	
10229	061730	000167	000440	JMP	40\$	
10230	061734	016600	000012	MOV	12(SP),R0	: LUN,* 5820
10231	061740	006300		ASL	R0	
10232	061742	016001	034500	MOV	TOP.SECT(R0),R1	
10233	061746	016005	034460	MOV	LOW.SECT(R0),R5	: *,SECTOR
10234	061752	000167	000406	JMP	39\$	
10235	061756	016646	000012	MOV	12(SP),-(SP)	: LUN,* 5822
10236	061762	012746	000400	MOV	#400,-(SP)	
10237	061766	012746	022670	MOV	#RBUF,-(SP)	
10238	061772	010546		MOV	R5,-(SP)	: SECTOR,*
10239	061774	004767	163612	JSR	PC,READ	
10240	062000	062706	000010	ADD	#10,SP	
10241	062004	020027	000005	CMP	R0,#5	
10242	062010	001164		BNE	38\$	

Address	Op	Op1	Op2	Op3	Label	Instruction	Comments	Seq
10244								
10245					:MLX4			
10246						DEFINITION OF OPTION 3		
10247	062012	004767	155314			JSR PC,ISOLATE	:	5825
10248	062016	032767	000001	120234		BIT #1,ERROUT	:	5827
10249	062024	001423				BEQ 32\$		
10250	062026	017700	152356			MOV @ML.REG+42,RO		
10251	062032	006200				ASR RO		
10252	062034	006200				ASR RO		
10253	062036	006200				ASR RO		
10254	062040	006200				ASR RO		
10255	062042	006200				ASR RO		
10256	062044	006200				ASR RO		
10257	062046	042700	177700			BIC #177700,RO		
10258	062052	010046				MOV RO,-(SP)		
10259	062054	012746	006640			MOV #FMT10B,-(SP)		
10260	062060	012746	000002			MOV #2,-(SP)		
10261	062064	010600				MOV SP,RO	: SP,*	
10262	062066	104414				TRAP 1,		
10263	062070	062706	000006			ADD #6,SP		
10264	062074	017766	152312	000022	32\$:	MOV @ML.REG+44,22(SP)	: *,OLDSEC	5836
10265	062102	017700	152302			MOV @ML.REG+42,RO	:	5837
10266	062106	006200				ASR RO		
10267	062110	006200				ASR RO		
10268	062112	006200				ASR RO		
10269	062114	006200				ASR RO		
10270	062116	006200				ASR RO		
10271	062120	006200				ASR RO		
10272	062122	042700	177700			BIC #177700,RO		
10273	062126	010066	000016			MOV RO,16(SP)	: *,OLDCHN	
10274	062132	012746	000001			MOV #1,-(SP)	:	5845
10275	062136	016646	000030			MOV 30(SP),-(SP)	: COMMAND,*	
10276	062142	016646	000016			MOV 16(SP),-(SP)	: LUN,*	
10277	062146	012746	000400			MOV #400,-(SP)		
10278	062152	016646	000034			MOV 34(SP),-(SP)	: PTR,*	
10279	062156	016646	000034			MOV 34(SP),-(SP)	: OLDSEC,*	
10280	062162	004767	164036			JSR PC,RETRY		
10281	062166	062706	000014			ADD #14,SP		
10282	062172	070027	000005			CMP RO,#5		
10283	062176	001052				BNE 35\$		
10284	062200	027766	152206	000022		CMP @ML.REG+44,22(SP)	: *,OLDSEC	5854
10285	062206	001035				BNE 34\$		
10286	062210	016646	000016			MOV 16(SP),-(SP)	: OLDCHN,*	
10287	062214	017700	152170			MOV @ML.REG+42,RO		
10288	062220	006200				ASR RO		
10289	062222	006200				ASR RO		
10290	062224	006200				ASR RO		
10291	062226	006200				ASR RO		
10292	062230	006200				ASR RO		
10293	062232	006200				ASR RO		
10294	062234	042700	177700			BIC #177700,RO		
10295	062240	020026				CMP RO,(SP)+		
10296	062242	001017				BNE 34\$		
10297	062244	032767	J00001	120006		BIT #1,ERROUT	:	5858
10298	062252	001404				BEQ 33\$		

10353 :MLX4

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (40)

DEFINITION OF OPTION 4

```

10354 :
10355 :
10356 : 5907 %sbttl 'DEFINITION OF OPTION 4'
10357 : 5908 routine OPT4 : novalue =
10358 : 5909 begin
10359 : 5910
10360 : 5911 !* 1 * START OF ROUTINE
10361 : 5912 ++
10362 : 5913 ROUTINE: OPT4
10363 : 5914 PURPOSE: TO LOOK FOR INTERACTIONS BETWEEN SECTORS
10364 : 5915 USING A MARCH TEST.
10365 : 5916
10366 : 5917 THE CODE IN BRIEF:
10367 : 5918
10368 : 5919 BEGIN 1 (START OF ROUTINE)
10369 : 5920 SAY ROUTINE IS RUNNING
10370 : 5921 WORD COUNT = 256
10371 : 5922 GENERATE A BUFFER OF DATA
10372 : 5923 GENERATE A BUFFER OF COMP
10373 : 5924 INITIALIZE POINTERS TO 4 BUFFERS FOR WRITE/READ DATA/COMP
10374 : 5925 INCR LUN FROM 0 TO LAST
10375 : 5926 : BEGIN 2 (START OF LOGICAL UNIT SELECTION LOOP)
10376 : 5927 : TESTLOOP:
10377 : 5928 : : BEGIN 3 (START OF LOOP THAT COMPLETELY TESTS 1 UNIT)
10378 : 5929 : : IF UNIT IS ACTIVE
10379 : 5930 : : THEN
10380 : 5931 : : : BEGIN 4 (START OF TEST FOR AN ACTIVE UNIT)
10381 : 5932 : : : INCR SECTOR FROM LOWEST TO HIGHEST
10382 : 5933 : : : : BEGIN 4A (START OF 1ST SECTOR SELECTION LOOP)
10383 : 5934 : : : : WRITE 'DATA'
10384 : 5935 : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
10385 : 5936 : : : : END 4A (END OF 1ST SECTOR SELECTION LOOP)
10386 : 5937 : : : : CHOOSE WHETHER TO WRITE CHECK OR READ 'DATA'
10387 : 5938 : : : : INCR SECTOR FROM LOWEST TO HIGHEST
10388 : 5939 : : : : : BEGIN 4B (START OF 2ND SECTOR SELECTION LOOP)
10389 : 5940 : : : : : DO THE WRITE CHECK OR READ OF 'DATA'
10390 : 5941 : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
10391 : 5942 : : : : : WRITE 'COMP'
10392 : 5943 : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
10393 : 5944 : : : : : END 4B (END OF 2ND SECTOR SELECTION LOOP)
10394 : 5945 : : : : : CHOOSE WHETHER TO WRITE CHECK OR READ 'COMP'
10395 : 5946 : : : : : DECR SECTOR FROM HIGHEST TO LOWEST
10396 : 5947 : : : : : : BEGIN 4C (START OF 3RD SECTOR SELECTION LOOP)
10397 : 5948 : : : : : : DO THE WRITE CHECK OR READ OF 'COMP'
10398 : 5949 : : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
10399 : 5950 : : : : : : WRITE DATA
10400 : 5951 : : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
10401 : 5952 : : : : : : END 4C (END OF 3RD SECTOR SELECTION LOOP)
10402 : 5953 : : : : : : CHOOSE WHETHER TO WRITE CHECK OR READ 'DATA'
10403 : 5954 : : : : : : INCR SECTOR FROM LOWEST TO HIGHEST
10404 : 5955 : : : : : : : BEGIN 4D (START OF 4TH SECTOR SELECTION LOOP)
10405 : 5956 : : : : : : : DO THE WRITE CHECK OR READ OF 'DATA'
10406 : 5957 : : : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
10407 : 5958 : : : : : : : END 4D (END OF 4TH SECTOR SELECTION LOOP)
    
```


27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (40)

10409 :MLX4
 10410 :
 10411 :
 10412 :
 10413 :
 10414 :
 10415 :
 10416 :
 10417 :
 10418 :
 10419 :
 10420 :
 10421 :
 10422 :
 10423 :
 10424 :
 10425 :
 10426 :
 10427 :
 10428 :
 10429 :
 10430 :
 10431 :
 10432 :
 10433 :
 10434 :
 10435 :
 10436 :
 10437 :
 10438 :
 10439 :
 10440 :
 10441 :
 10442 :
 10443 :
 10444 :
 10445 :
 10446 :
 10447 :
 10448 :
 10449 :
 10450 :
 10451 :
 10452 :
 10453 :
 10454 :
 10455 :
 10456 :
 10457 :
 10458 :
 10459 :
 10460 :
 10461 :
 10462 :
 10463 :

DEFINITION OF OPTION 4

```

5959 : : : : END 4 (END OF TEST FOR AN ACTIVE UNIT)
5960 : : : : END 3 (END OF TESTLOOP)
5961 : : : : END 2 (END OF LOGICAL UNIT SELECTION LOOP)
5962 : RETURN
5963 : END 1 (END OF ROUTINE)
  
```

Label

LOOP;

Local

```

VALUE,
OLDSEC,
OLDCHN,
PTR,
COMMAND,
DBL VALJE,
WDPTR,
RDPTR,
WCPTR,
RCPTR;
  
```

PRINTB (SAY2, WRD34, RTN4);

!;RUNNING OPT4'

GEN4 (.MARPAT, WDBUFF);

GEN4 ((-.MARPAT), WCBUFF);

WDPTR = WDBUFF;

WCPTR = WCBUFF;

RDPTR = RDBUFF;

RCPTR = RCBUFF;

```

!POINTER TO 256-WORD WRITE DATA BUFFER
!POINTER TO 256-WORD WRITE COMP BUFFER
!POINTER TO 256-WORD READ DATA BUFFER
!POINTER TO 256-WORD READ COMP BUFFER
  
```

incr LUN from 0 to (.LSUNIT - 1) do

begin

!* 2 * START OF LOGICAL UNIT SELECTION LOOP

LOOP :

begin

!* 3 * START OF THE LOOP THAT COMPLETELY TESTS 1 UNIT

if .DRIVE_STATUS [.LUN] eql ACTIVE

then

!* 4 * START OF TEST FOR AN ACTIVE UNIT

begin

LSLUN = .LUN;

PRINTB (SAY1, WRD35);

!'MARCHING'

PRINTB (FMT2, WRD24);

!' UP'

version czmlbb changed incr to incru

incru SECTOR from LOWEST to HIGHEST do

begin

VALUE = write (.LUN, 256, .WDPTR, .SECTOR);

!* 4A * START OF 1ST SECTOR SELECTION LOOP

27-Mar-1982 19:24:42
27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
PA:<NEALE>MLX4.BLI.5 (40)

10521 :MLX4

DEFINITION OF OPTION 4

```

10522 :
10523 :
10524 :      6063      begin
10525 :      6064      PTR = .RDPTR;
10526 :      6065      VALUE = read (.LUN, 256, .PTR, .SECTOR);
10527 :      6066      end
10528 :      6067      else
10529 :      6068      begin
10530 :      6069      PTR = .WDPTR;
10531 :      6070      VALUE = CHECK (.LUN, 256, .PTR, .SECTOR);
10532 :      6071      end;
10533 :      6072
10534 :      6073      !+
10535 :      6074      ! SEE HOW SUCCESSFUL THE OPERATION WAS:
10536 :      6075      !-
10537 :      6076
10538 :      6077      selectone .VALUE of
10539 :      6078      set
10540 :      6079
10541 :      6080      [0] :
10542 :      6081
10543 :      6082      if .COMMAND eql read
10544 :      6083      then
10545 :      6084      begin
10546 :      6085
10547 :      6086      if (DBL_VALUE = DOUBLE_CHECK (.WDPTR, .RDPTR, 256)) neq 0
10548 :      6087      then
10549 :      6088      begin
10550 :      6089      SAYWHO (.LUN);
10551 :      6090      PRINTB (SAY1, MSG5);
10552 :      6091      !'ECC LOGIC FAILED TO DETECT DATA ERROR'
10553 :      6092      PRINTB (FMT12A, .DBL_VALUE, .DBL_VALUE);
10554 :      6093      !'GOOD DATA: XXXXXX AT LOCATION YYYYYY'
10555 :      6094      DBL_VALUE = .DBL_VALUE + BUFSIZ*2;
10556 :      6095      PRINTB (FMT12B, .DBL_VALUE, .DBL_VALUE);
10557 :      6096      !'BAD DATA: P P P P P AT LOCATION Q Q Q Q Q'
10558 :      6097      WHY DROPT [.LUN] = CODE_8;
10559 :      6098      ERRDF (404, MSG1, 0); !**** OPTION 4 ERROR 04 ****
10560 :      6099      DODU (.LUN);
10561 :      6100      leave LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
10562 :      6101      end;
10563 :      6102
10564 :      6103      end;
10565 :      6104
10566 :      6105      [1] :
10567 :      6106      begin
10568 :      6107      !* 4B1 *      RETRY ALLOWED
10569 :      6108      if RETRY (SIX, .COMMAND, .LUN, 256, .PTR, .SECTOR) neq 0
10570 :      6109      then
10571 :      6110      !THE RETRY FAILED -- SYSTEM FATAL ERROR
10572 :      6111      begin
10573 :      6112      WHY DROPT [.LUN] = CODE_4;
10574 :      6113      ERRDF (405, MSG1, 0); !**** OPTION 4 ERROR 05 ****
10575 :      6114      DODU (.LUN);
10575 :      6114      leave LOOP; !JUMP JUST BEYOND END OF BLOCK * 3 *

```


27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (40)

```

10689 :MLX4
10690 :
10691 :
10692 :         6219
10693 :         6220
10694 :         6221
10695 :         6222
10696 :         6223
10697 :         6224
10698 :         6225
10699 :         6226
10700 :         6227
10701 :         6228
10702 :         6229
10703 :         6230
10704 :         6231
10705 :         6232
10706 :         6233
10707 :         6234
10708 :         6235
10709 :         6236
10710 :         6237
10711 :         6238
10712 :         6239
10713 :         6240
10714 :         6241
10715 :         6242
10716 :         6243
10717 :         6244
10718 :         6245
10719 :         6246
10720 :         6247
10721 :         6248
10722 :         6249
10723 :         6250
10724 :         6251
10725 :         6252
10726 :         6253
10727 :         6254
10728 :         6255
10729 :         6256
10730 :         6257
10731 :         6258
10732 :         6259
10733 :         6260
10734 :         6261
10735 :         6262
10736 :         6263
10737 :         6264
10738 :         6265
10739 :         6266
10740 :         6267
10741 :         6268
10742 :         6269
10743 :         6270

DEFINITION OF OPTION 4

DODU (.LUN);
Leave LOOP;
end;
!* 488 *

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
set
!*SEE 'SYSERR' FOR DEFINITION
!*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: 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 (415, MSG1, 0);
!**** OPTION 4 ERROR 15 ****

```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (40)

10745 :MLX4
 10746 :
 10747 :
 10748 :
 10749 :
 10750 :
 10751 :
 10752 :
 10753 :
 10754 :
 10755 :
 10756 :
 10757 :
 10758 :
 10759 :
 10760 :
 10761 :
 10762 :
 10763 :
 10764 :
 10765 :
 10766 :
 10767 :
 10768 :
 10769 :
 10770 :
 10771 :
 10772 :
 10773 :
 10774 :
 10775 :
 10776 :
 10777 :
 10778 :
 10779 :
 10780 :
 10781 :
 10782 :
 10783 :
 10784 :
 10785 :
 10786 :
 10787 :
 10788 :
 10789 :
 10790 :
 10791 :
 10792 :
 10793 :
 10794 :
 10795 :
 10796 :
 10797 :
 10798 :
 10799 :

DEFINITION OF OPTION 4

```

        DODU (.LUN);
        leave LOOP;
    end;
    end;
[1] :
    begin
        !* 4C1 *          RETRY ALLOWED
        if RETRY (SIX, .COMMAND, .LUN, 256, .PTR, .SECTOR) neq 0
        then
            !THE RETRY FAILED -- SYSTEM FATAL ERROR
            begin
                WHY DROPT [.LUN] = CODE_4;
                ERRDF (416, MSG1, 0);
                !**** OPTION 4 ERROR 16 ****
                DODU (.LUN);
                leave LOOP;
                !JUMP JUST BEYOND END OF BLOCK * 3 *
            end;
        end;
        !* 4C1 *
    end;
[2] :
    begin
        !* 4C2 *          FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
        WHY DROPT [.LUN] = CODE_5;
        ERRDF (417, MSG1, 0);
        !**** OPTION 4 ERROR 17 ****
        DODU (.LUN);
        leave LOOP;
        !JUMP JUST BEYOND END OF BLOCK * 3 *
    end;
    !* 4C2 *
[3] :
    begin
        !* 4C3 *          FATAL DRIVE ERROR -- NO RETRY ALLOWED
        ERRDF (418, MSG1, 0);
        !**** OPTION 4 ERROR 18 ****
        WHY DROPT [.LUN] = CODE_6;
        DODD (.LUN);
        leave LOOP;
        !JUMP JUST BEYOND END OF BLOCK * 3 *
    end;
    !* 4C3 *
[4] :
    begin
        !* 4C4 *          UNRECOVERABLE DATA ERROR
        ISOLATE ();
        ERRDF (419, MSG2, 0);
        !**** OPTION 4 ERROR 19 ****
        WHY DROPT [.LUN] = CODE_7;
        DODD (.LUN);
        leave LOOP;
        !JUMP JUST BEYOND END OF BLOCK * 3 *
    end;
    !* 4C4 *
[5] :
    begin
        !* 4C5 *          RECOVERABLE DATA ERROR
        ISOLATE ();
        if .ERROUT then PRINTB (FMT10B, .CHAN);
        !' BIT 00'
    end;

```


27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (40)

```

10857 :MLX4
10858 :
10859 :
10860 : 6375          end:
10861 : 6376
10862 : 6377          end:          !* 4C6 *
10863 : 6378
10864 : 6379          [2] :
10865 : 6380          begin          !* 4C7 *          FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
10866 : 6381          WHY DROPT [.LUN] = CODE_5;
10867 : 6382          ERRDF (424, MSG1, 0); !**** OPTION 4 ERROR 24 ****
10868 : 6383          DODU (.LUN);
10869 : 6384          leave LOOP;          !JUMP JUST BEYOND END OF BLOCK * 3 *
10870 : 6385          end:          !* 4C7 *
10871 : 6386
10872 : 6387          [3] :
10873 : 6388          begin          !* 4C8 *          FATAL DRIVE ERROR -- NO RETRY ALLOWED
10874 : 6389          ERRDF (425, MSG1, 0); !**** OPTION 4 ERROR 25 ****
10875 : 6390          WHY DROPT [.LUN] = CODE_6;
10876 : 6391          DODU (.LUN);
10877 : 6392          leave LOOP;          !JUMP JUST BEYOND END OF BLOCK * 3 *
10878 : 6393          end:          !* 4C8 *
10879 : 6394          tes:
10880 : 6395
10881 : 6396          end:          !* 4C * END OF 3RD SECTOR SELECTION LOOP
10882 : 6397
10883 : 6398          PRINTB (FMT2, WRD24);
10884 : 6399          !' UP'
10885 : 6400          COMMAND = CHOOSE ();
10886 : 6401
10887 : 6402          !
10888 : 6403          ! version czmlbb changed incr to incru
10889 : 6404          !
10890 : 6405          incru SECTOR from LOWEST to HIGHEST do
10891 : 6406          begin          !* 4D * START OF 4TH SECTOR SELECTION LOOP
10892 : 6407
10893 : 6408          if .COMMAND eql read
10894 : 6409          then
10895 : 6410          begin
10896 : 6411          PTR = .RDPTR;
10897 : 6412          VALUE = read (.LUN, 256, .PTR, .SECTOR);
10898 : 6413          end
10899 : 6414          else
10900 : 6415          begin
10901 : 6416          PTR = .WDPTR;
10902 : 6417          VALUE = CHECK (.LUN, 256, .PTR, .SECTOR);
10903 : 6418          end;
10904 : 6419
10905 : 6420          !+
10906 : 6421          ! SEE HOW SUCCESSFUL THE OPERATION WAS:
10907 : 6422          !-
10908 : 6423
10909 : 6424          selectone .VALUE of          !SEE 'SYSERR' FOR DEFINITION
10910 : 6425          set          !OF ERROR # CONTAINED IN 'VALUE'
10911 : 6426
    
```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (40)

10913 :MLX4
 10914 :
 10915 :
 10916 :
 10917 :
 10918 :
 10919 :
 10920 :
 10921 :
 10922 :
 10923 :
 10924 :
 10925 :
 10926 :
 10927 :
 10928 :
 10929 :
 10930 :
 10931 :
 10932 :
 10933 :
 10934 :
 10935 :
 10936 :
 10937 :
 10938 :
 10939 :
 10940 :
 10941 :
 10942 :
 10943 :
 10944 :
 10945 :
 10946 :
 10947 :
 10948 :
 10949 :
 10950 :
 10951 :
 10952 :
 10953 :
 10954 :
 10955 :
 10956 :
 10957 :
 10958 :
 10959 :
 10960 :
 10961 :
 10962 :
 10963 :
 10964 :
 10965 :
 10966 :
 10967 :

DEFINITION OF OPTION 4

```
[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: P P P P P AT LOCATION Q Q Q Q Q'
              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;

[1] :
      begin
          !* 4D1 *          RETRY ALLOWED
          if RETRY (SIX, .COMMAND, .LUN, 256, .PTR, .SECTOR) neq 0
          then
          !THE RETRY FAILED -- SYSTEM FATAL ERROR
          begin
              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;
      end;
      !* 4D1 *

[2] :
      begin
          !* 4D2 *          FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
          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);
```

```

10969 :MLX4
10970 :
10971 :
10972 : 6479
10973 : 6480
10974 : 6481
10975 : 6482
10976 : 6483
10977 : 6484
10978 : 6485
10979 : 6486
10980 : 6487
10981 : 6488
10982 : 6489
10983 : 6490
10984 : 6491
10985 : 6492
10986 : 6493
10987 : 6494
10988 : 6495
10989 : 6496
10990 : 6497
10991 : 6498
10992 : 6499
10993 : 6500
10994 : 6501
10995 : 6502
10996 : 6503
10997 : 6504
10998 : 6505
10999 : 6506
11000 : 6507
11001 : 6508
11002 : 6509
11003 : 6510
11004 : 6511
11005 : 6512
11006 : 6513
11007 : 6514
11008 : 6515
11009 : 6516
11010 : 6517
11011 : 6518
11012 : 6519
11013 : 6520
11014 : 6521
11015 : 6522
11016 : 6523
11017 : 6524
11018 : 6525
11019 : 6526
11020 : 6527
11021 : 6528
11022 : 6529
11023 : 6530

DEFINITION OF OPTION 4

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;
DODD (.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 QQ'
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);
end

else
begin
if .ERROUT then ERRSOFT (433, MSG3, 0); !**** OPTION 4 ERROR 33 ****
UP_SOFT_COUNT (.LUN, .BOARD);
end;

end; !* 4D5 *
tes;

```

```

11025 :MLX4
11026 :
11027 :
11028 :      6531          end;
11029 :      6532
11030 :      6533          end;
11031 :      6534
11032 :      6535
11033 :      6536          !+
11034 :      6537          ! Test to see if this uut's address space is
11035 :      6538          ! to be read for soft errors. This test is
11036 :      6539          ! is intended for DMT purposes.
11037 :      6540          !-
11038 :      6541          if .EFNS21          !Is the background pattern to be read
11039 :      6542          then
11040 :      6543          begin
11041 :      6544
11042 :      6545          !
11043 :      6546          ! version czmlbb changed incr to incru
11044 :      6547          !
11045 :      6548          incru SECTOR from LOWEST to HIGHEST do
11046 :      6549
11047 :      6550          if read (.LUN, 256, RBUF, .SECTOR) eql 5
11048 :      6551          then
11049 :      6552          begin
11050 :      6553          ISOLATE ();          !Find the failing bank and board no.
11051 :      6554
11052 :      6555          if .ERROUT then PRINTB (FMT10B, .CHAN);
11053 :      6556
11054 :      6557          ! Print where the error is
11055 :      6558          !
11056 :      6559          ! Save the contents of the ML error location
11057 :      6560          ! register so we can compare it to the new
11058 :      6561          ! contents of this register after the retry.
11059 :      6562          ! This is done to classify the error.
11060 :      6563
11061 :      6564          OLDSEC = .MLEL;
11062 :      6565          OLDCHN = .CHAN;
11063 :      6566          !
11064 :      6567          ! Do a classify retry call. If the same error
11065 :      6568          ! occurs then classify it as a hard error. If
11066 :      6569          ! a different error occurred or the error went away
11067 :      6570          ! then classify it as a soft error.
11068 :      6571          !
11069 :      6572          !
11070 :      6573          if RETRY (ONE, .COMMAND, .LUN, 256, .PTR, .OLDSEC) eql 5
11071 :      6574          then
11072 :      6575          !
11073 :      6576          ! The same error occurred so see if it is at the same
11074 :      6577          ! sector and channel number, if so then classify
11075 :      6578          ! it as a hard error else classiy it as a soft
11076 :      6579          ! error.
11077 :      6580          !
11078 :      6581          !
11079 :      6582          if ((.MLEL eql .OLDSEC) and (.CHAN eql .OLDCHN))
    
```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (40)

```

11081 :MLX4
11082 :
11083 :
11084 : 6583
11085 : 6584
11086 : 6585
11087 : 6586
11088 : 6587
11089 : 6588
11090 : P 6589
11091 : P 6590
11092 : 6591
11093 : 6592
11094 : 6593
11095 : 6594
11096 : 6595
11097 : 6596
11098 : 6597
11099 : 6598
11100 : 6599
11101 : 6600
11102 : 6601
11103 : P 6602
11104 : P 6603
11105 : 6604
11106 : 6605
11107 : 6606
11108 : 6607
11109 : 6608
11110 : 6609
11111 : 6610
11112 : 6611
11113 : 6612
11114 : 6613
11115 : 6614
11116 : 6615
11117 : P 6616
11118 : P 6617
11119 : 6618
11120 : 6619
11121 : 6620
11122 : 6621
11123 : 6622
11124 : 6623
11125 : 6624
11126 : 6625
11127 : 6626
11128 : 6627
11129 : 6628
11130 : 6629
11131 : 6630
11132 : 6631
11133 : 6632

DEFINITION OF OPTION 4

then
begin
!Same error occurred 'hard'
if .ERROUT
!Print error if enabled
then
begin
ERRHRD (434,
!Error number
MSG4,
!Error message
0);
!Additional message routine
end;
UP_HARD_COUNT (.LUN, .BOARD);
end
else
begin
!Not the same error 'soft'
if .ERROUT
!Print error if enabled
then
begin
ERRSOFT (435,
!Error number
MSG3,
!Error message
0);
!Additional message routine
end;
UP_SOFT_COUNT (.LUN, .BOARD);
end
else
begin
!Not the same error 'soft'
if .ERROUT
!Print error if enabled
then
begin
ERRSOFT (436,
!Error number
MSG3,
!Error message
0);
!Additional message routine
end;
UP_SOFT_COUNT (.LUN, .BOARD);
end;
end;
end;
end;
return;
end;

!* 3 * END OF LOOP THAT COMPLETELY TESTS 1 UNIT
!* 2 * END OF LOGICAL UNIT SELECTION LOOP
!* 1 * END OF ROUTINE
    
```

Address	Hex	Hex	Hex	Label	Instruction	Comment	Address
11141							
11142							
11146	062444	004167	122664	OPT4:	.SBTTL OPT4 DEFINITION OF OPTION 4		5908
11147	062450	162706	000030		JSR R1,\$SAVE5	:	
11148	062454	012746	007650		SUB #30,SP	:	5981
11149	062460	012746	007344		MOV #RTN4,-(SP)	:	
11150	062464	012746	007100		MOV #WRD34,-(SP)	:	
11151	062470	012746	000003		MOV #SAY2,-(SP)	:	
11152	062474	010600			MOV #3,-(SP)	:	
11153	062476	104414			MOV SP,R0	: SP,*	
11154	062500	016716	117542		TRAP 14	:	5983
11155	062504	012746	012670		MOV MARPAT,(SP)	:	
11156	062510	004767	157050		MOV #WDBUFF,-(SP)	:	
11157	062514	016716	117526		JSR PC,GEN4	:	5984
11158	062520	005416			MOV MARPAT,(SP)	:	
11159	062522	012746	013670		NEG (SP)	:	
11160	062526	004767	157032		MOV #WCBUFF,-(SP)	:	
11161	062532	012766	012670	000016	JSR PC,GEN4	: *,WDPTR	5985
11162	062540	012766	013670	000026	MOV #WDBUFF,16(SP)	: *,WCPTR	5986
11163	062546	012766	022670	000024	MOV #WCBUFF,26(SP)	: *,RDPTR	5987
11164	062554	012766	023670	000034	MOV #RDBUFF,24(SP)	: *,RCPTR	5988
11165	062562	016766	117224	000042	MOV #RCBUFF,34(SP)	: *	5990
11166	062570	005066	000014		MOV L\$UNIT,42(SP)	: LUN	
11167	062574	000167	005362		CLR 14(SP)	: LUN,*	5995
11168	062600	016600	000014	1\$:	JMP 95\$:	
11169	062604	006200			MOV 14(SP),R0	:	
11170	062606	006200			ASR R0	:	
11171	062610	006200			ASR R0	:	
11172	062612	062700	034442		ASR R0	:	
11173	062616	010046			ADD #DRIVE.STATUS,R0	:	
11174	062620	016646	000016		MOV R0,-(SP)	: LUN,*	
11175	062624	042716	177770		MOV 16(SP),-(SP)	:	
11176	062630	012746	000001		BIC #177770,(SP)	:	
11177	062634	005046			MOV #1,-(SP)	:	
11178	062636	004767	121514		CLR -(SP)	:	
11179	062642	062706	000010		JSR PC,BL\$GT2	:	
11180	062646	005300			ADD #10,SP	:	
11181	062650	001402			DEC R0	:	
11182	062652	000167	004630		BEQ 2\$:	
11183	062656	016667	000014	117210 2\$:	JMP 83\$: LUN,*	5998
11184	062664	012746	007354		MOV 14(SP),L\$LUN	: LUN,*	5999
11185	062670	012746	007072		MOV #WRD35,-(SP)	:	
11186	062674	012746	000002		MOV #SAY1,-(SP)	:	
11187	062700	010600			MOV #2,-(SP)	:	
11188	062702	104414			MOV SP,R0	: SP,*	
11189	062704	012716	00332		TRAP 14	:	6001
11190	062710	012746	006160		MOV #WRD24,(SP)	:	
11191	062714	012746	000002		MOV #FMT2,-(SP)	:	
					MOV #2,-(SP)	:	

Address	OpCode	Operand 1	Operand 2	Operand 3	Label	Instruction	Comments	Time	Page
11193								27-Mar-1982 19:24:42	TOPS
11194								27-Mar-1982 19:23:44	PA:<
11195									
11196	062720	010600				MOV SP,R0	: SP,*		
11197	062722	104414				TRAP 14			
11198	062724	016600	000026			MOV 26(SP),R0	: LUN,*		6007
11199	062730	006300				ASL R0			
11200	062732	01276E	034460	000052		MOV #LOW.SECT,52(SP)			
11201	062740	06006C	000052			ADD R0,52(SP)			
11202	062744	012766	034500	000050		MOV #TOP.SECT,50(SP)			
11203	062752	060066	000050			ADD R0,50(SP)			
11204	062756	017601	000050			MOV @50(SP),R1			
11205	062762	017604	000052			MOV @52(SP),R4	: *,SECTOR		
11206	062766	000516				BR 8\$			
11207	062770	016646	000026		3\$:	MOV 26(SP),-(SP)	: LUN,*		6009
11208	062774	012746	000400			MOV #400,-(SP)			
11209	063000	016646	000034			MOV 34(SP),-(SP)	: WDPTR,*		
11210	063004	01044C				MOV R4,-(SP)	: SECTOR,*		
11211	063006	004767	162412			JSR PC,WRITE			
11212	063012	010005				MOV R0,R5	: *,VALUE		
11213	063014	020527	000001			CMP R5,#1	: VALUE,*		6015
11214	063020	001036				BNE 4\$			
11215	063022	012746	000006			MOV #6,-(SP)	:		6021
11216	063026	012746	045424			MOV #WRITE,-(SP)			
11217	063032	016646	000042			MOV 42(SP),-(SP)	: LUN,*		
11218	063036	012746	000400			MOV #400,-(SP)			
11219	063042	016646	000050			MOV 50(SP),-(SP)	: WDPTR,*		
11220	063046	010446				MOV R4,-(SP)	: SECTOR,*		
11221	063050	004767	163150			JSR PC,RETRY			
11222	063054	062706	000014			ADD #14,SP			
11223	063060	005700				TST R0			
11224	063062	001455				BEQ 7\$			
11225	063064	016600	000036			MOV 36(SP),R0	: LUN,*		6024
11226	063070	112760	000004	034446		MOVB #4,WHY.DROPT(R0)			
11227	063076	104455				TRAP 55	:		6025
11228	063100	000621				.WORD 621			
11229	063102	011070				.WORD MSG1			
11230	063104	000000				.WORD 0			
11231	063106	016600	000036			MOV 36(SP),R0	: LUN,*		6026
11232	063112	104451				TRAP 51	:		
11233	063114	000435				BR 6\$:		6027
11234	063116	020527	000002		4\$:	CMP R5,#2	: VALUE,*		6015
11235	063122	001015				BNE 5\$			
11236	063124	016600	000036			MOV 36(SP),R0	: LUN,*		6034
11237	063130	112760	000005	034446		MOVB #5,WHY.DROPT(R0)			
11238	063136	104455				TRAP 55	:		6035
11239	063140	000622				.WORD 622			
11240	063142	011070				.WORD MSG1			
11241	063144	000000				.WORD 0			
11242	063146	016600	000036			MOV 36(SP),R0	: LUN,*		6036
11243	063152	104451				TRAP 51	:		
11244	063154	000415				BR 6\$:		6037
11245	063156	020527	000003		5\$:	CMP R5,#3	: VALUE,*		6015
11246	063162	001015				BNE 7\$			
11247	063164	104455				TRAP 55	:		6042

27-Mar-1982 19:24:42 TOPS
27-Mar-1982 19:23:44 PA:<

Address	OpCode	Operand1	Operand2	Label	Instruction	Comments	Address
11249							
11250							
11251							
11252	063166	000623			.WORD 623		
11253	063170	011070			.WORD MSG1		
11254	063172	000000			.WORD 0		
11255	063174	016600	000036		MOV 36(SP),R0	: LUN,*	6043
11256	063200	112760	000006	034446	MOV #6,WHY.DROPT(R0)		
11257	063206	104451			TRAP 51		6044
11258	063210	062706	000022	6\$:	ADD #22,SP		6045
11259	063214	000565			BR 14\$		
11260	063216	062706	000010	7\$:	ADD #10,SP		6008
11261	063222	005204			INC R4	: SECTOR	6007
11262	063224	020401		8\$:	CMP R4,R1	: SECTOR,*	
11263	063226	101660			BLOS 3\$		
11264	063230	012716	007332		MOV #WRD24,(SP)		6051
11265	063234	012746	006160		MOV #FMT2,-(SP)		
11266	063240	012746	000002		MOV #2,-(SP)		
11267	063244	010600			MOV SP,R0	: SP,*	
11268	063246	104414			TRAP 14		
11269	063250	004767	162712		JSR PC,CHOOSE		6053
11270	063254	010002			MOV R0,R2	: *,COMMAND	
11271	063256	017666	000054	000046	MOV @54(SP),46(SP)		6058
11272	063264	017604	000056		MOV @56(SP),R4	: *,SECTOR	
11273	063270	000167	001350		JMP 33\$		
11274	063274	005001		9\$:	CLR R1		6061
11275	063276	020227	045612		CMP R2,#READ	: COMMAND *	
11276	063302	001014			BNE 10\$		
11277	063304	005201			INC R1		
11278	063306	016603	000042		MOV 42(SP),R3	: RDPTR,PTR	6064
11279	063312	016646	000032		MOV 32(SP),-(SP)	: LUN,*	6065
11280	063316	012746	000400		MOV #400,-(SP)		
11281	063322	010346			MOV R3,-(SP)	: PTR,*	
11282	063324	010446			MOV R4,-(SP)	: SECTOR,*	
11283	063326	004767	162260		JSR PC,READ		
11284	063332	000412			BR 11\$		
11285	063334	016603	000034	10\$:	MOV 34(SP),R3	: WDPTR,PTR	6069
11286	063340	016646	000032		MOV 32(SP),-(SP)	: LUN,*	6070
11287	063344	012746	000400		MOV #400,-(SP)		
11288	063350	010346			MOV R3,-(SP)	: PTR,*	
11289	063352	010446			MOV R4,-(SP)	: SECTOR,*	
11290	063354	004767	162420		JSR PC,CHECK		
11291	063360	010005		11\$:	MOV R0,R5	: *,VALUE	
11292	063362	001103			BNE 15\$		6077
11293	063364	006001			ROR R1		6082
11294	063366	103402			BLO 13\$		
11295	063370	000167	001012	12\$:	JMP 28\$		
11296	063374	016646	000044	13\$:	MOV 44(SP),-(SP)	: WDPTR,*	6086
11297	063400	016646	000054		MOV 54(SP),-(SP)	: RDPTR,*	
11298	063404	012746	000400		MOV #400,-(SP)		
11299	063410	004767	160712		JSR PC,DOUBLE.CHECK		
11300	063414	062706	000006		ADD #6,SP		
11301	063420	010066	000060		MOV R0,60(SP)	: *,DBL.VALUE	
11302	063424	001761			BEQ 12\$		
11303	063426	016646	000042		MOV 42(SP),-(SP)	: LUN,*	6089

Address	OpCode	Operand 1	Operand 2	Operand 3	Instruction	Comment	Address	Time	Page
11305								27-Mar-1982 19:24:42	TOPS
11306								27-Mar-1982 19:23:44	PA:<
11307									
11308	063432	004767	152066		JSR	PC,SAYWHD			
11309	063436	012716	011216		MOV	#MSG5,(SP)	:		6090
11310	063442	012746	007072		MOV	#SAY1,-(SP)			
11311	063446	012746	000002		MOV	#2,-(SP)			
11312	063452	010600			MOV	SP,R0	: SP,*		
11313	063454	104414			TRAP	14			
11314	063456	016616	000066		MOV	66(SP),(SP)	: DBL.VALUE,*		6092
11315	063462	017646	000066		MOV	@66(SP),-(SP)	: DBL.VALUE,*		
11316	063466	012746	006710		MOV	#FMT12A,-(SP)			
11317	063472	012746	000003		MOV	#3,-(SP)			
11318	063476	010600			MOV	SP,R0	: SP,*		
11319	063500	104414			TRAP	14			
11320	063502	062766	010000	000074	ADD	#10000,74(SP)	: *,DBL.VALUE		6094
11321	063510	016616	000074		MOV	74(SP),(SP)	: DBL.VALUE,*		6095
11322	063514	017646	000074		MOV	@74(SP),-(SP)	: DBL.VALUE,*		
11323	063520	012746	006756		MOV	#FMT12B,-(SP)			
11324	063524	012746	000003		MOV	#3,-(SP)			
11325	063530	010600			MOV	SP,R0	: SP,*		
11326	063532	104414			TRAP	14			
11327	063534	016601	000064		MOV	64(SP),R1	: LUN,*		6097
11328	063540	112761	000010	034446	MOVB	#10,WHY.DROPT(R1)			
11329	063546	104455			TRAP	55	:		6098
11330	063550	000624			.WORD	624			
11331	063552	011070			.WORD	MSG1			
11332	063554	000000			.WORD	0			
11333	063556	016600	000064		MOV	64(SP),R0	: LUN,*		6099
11334	063562	104451			TRAP	51			
11335	063564	062706	000050		ADD	#50,SP	:		6100
11336	063570	000520		14\$:	BR	20\$			
11337	063572	020527	000001	15\$:	CMP	R5,#1	: VALUE,*		6077
11338	063576	001034			BNE	16\$			
11339	063600	012746	000006		MOV	#6,-(SP)	:		6108
11340	063604	010246			MOV	R2,-(SP)	: COMMAND,*		
11341	063606	016646	000046		MOV	46(SP),-(SP)	: LUN,*		
11342	063612	012746	000400		MOV	#400,-(SP)			
11343	063616	010346			MOV	R3,-(SP)	: PTR,*		
11344	063620	010446			MOV	R4,-(SP)	: SECTOR,*		
11345	063622	004767	162376		JSR	PC,RETRY			
11346	063626	062706	000014		ADD	#14,SP			
11347	063632	005700			TST	R0			
11348	063634	001655			BEQ	12\$			
11349	063636	016601	000042		MOV	42(SP),R1	: LUN,*		6111
11350	063642	112761	000004	034446	MOVB	#4,WHY.DROPT(R1)			
11351	063650	104455			TRAP	55	:		6112
11352	063652	000625			.WORD	625			
11353	063654	011070			.WORD	MSG1			
11354	063656	000000			.WORD	0			
11355	063660	016600	000042		MOV	42(SP),R0	: LUN,*		6113
11356	063664	104451			TRAP	51			
11357	063666	000457			BR	19\$:		6114
11358	063670	020527	000002	16\$:	CMP	R5,#2	: VALUE,*		6077
11359	063674	001015			BNE	17\$			

27-Mar-1982 19:24:42 TOPS
27-Mar-1982 19:23:44 PA:<

Address	OpCode	Operand1	Operand2	Operand3	Label	Instruction	Comment	Seq
11361					:MLX4		DEFINITION OF OPTION 4	
11362					:			
11363								
11364	063676	016601	000042			MOV 42(SP),R1	: LUN,*	6121
11365	063702	112761	000005	034446		MOVB #5,WHY.DROPT(R1)		
11366	063710	104455				TRAP 55	:	6122
11367	063712	000626				.WORD 626		
11368	063714	011070				.WORD MSG1		
11369	063716	000000				.WORD 0		
11370	063720	016600	000042			MOV 42(SP),R0	: LUN,*	6123
11371	063724	104451				TRAP 51		
11372	063726	000437				BR 19\$:	6124
11373	063730	020527	000003	17\$:		CMP R5,#3	: VALUE,*	6077
11374	063734	001014				BNE 18\$:	
11375	063736	104455				TRAP 55	:	6129
11376	063740	000627				.WORD 627		
11377	063742	011070				.WORD MSG1		
11378	063744	000000				.WORD 0		
11379	063746	016601	000042			MOV 42(SP),R1	: LUN,*	6130
11380	063752	112761	000006	034446		MOVB #6,WHY.DROPT(R1)		
11381	063760	010100				MOV R1,R0	: LUN,*	6131
11382	063762	104451				TRAP 51		
11383	063764	000420				BR 19\$:	6132
11384	063766	020527	000004	18\$:		CMP R5,#4	: VALUE,*	6077
11385	063772	001021				BNE 21\$:	
11386	063774	004767	153332			JSR PC,ISOLATE	:	6137
11387	064000	104455				TRAP 55	:	6138
11388	064002	000630				.WORD 630		
11389	064004	011120				.WORD MSG2		
11390	064006	000000				.WORD 0		
11391	064010	016601	000042			MOV 42(SP),R1	: LUN,*	6139
11392	064014	112761	000007	034446		MOVB #7,WHY.DROPT(R1)		
11393	064022	010100				MOV R1,R0	: LUN,*	6140
11394	064024	104451				TRAP 51		
11395	064026	062706	000026	19\$:		ADD #26,SP	:	6141
11396	064032	000167	004120	20\$:		JMP 94\$		
11397	064036	020527	000005	21\$:		CMP R5,#5	: VALUE,*	6077
11398	064042	001161				BNE 28\$:	
11399	064044	004767	153262			JSR PC,ISOLATE	:	6146
11400	064050	032767	000001	116202		BIT #1,ERROUT	:	6148
11401	064056	001423				BEQ 22\$		
11402	064060	017701	150324			MOV @ML.REG+42,R1		
11403	064064	006201				ASR R1		
11404	064066	006201				ASR R1		
11405	064070	006201				ASR R1		
11406	064072	006201				ASR R1		
11407	064074	006201				ASR R1		
11408	064076	006201				ASR R1		
11409	064100	042701	177700			BIC #177700,R1		
11410	064104	010146				MOV R1,-(SP)		
11411	064106	012746	006640			MOV #FMT10B,-(SP)		
11412	064112	012746	000002			MOV #2,-(SP)		
11413	064116	010600				MOV SP,R0	: SP,*	
11414	064120	104414				TRAP 14		
11415	064122	062706	000006			ADD #6,SP		

Address	Hex	Hex	Hex	Label	Op	Instruction	Comments	Date/Time	Page
11417									
11418									
11419									
11420	064126	017766	150260	000046	22\$:	MOV @ML.REG+44,46(SP)	: *,OLDSEC	27-Mar-1982 19:24:42	TOPS
11421	064134	017701	150250			MOV @ML.REG+42,R1	:	27-Mar-1982 19:23.44	PA:<
11422	064140	006201				ASR R1			
11423	064142	006201				ASR R1			
11424	064144	006201				ASR R1			
11425	064146	006201				ASR R1			
11426	064150	006201				ASR R1			
11427	064152	006201				ASR R1			
11428	064154	042701	177700			BIC #177700,R1			
11429	064160	010166	000050			MOV R1,50(SP)	: *,OLDCHN		
11430	064164	012746	000001			MOV #1,-(SP)	:		6154
11431	064170	010246				MOV R2,-(SP)	: COMMAND,*		
11432	064172	016646	000046			MOV 46(SP),-(SP)	: LUN,*		
11433	064176	012746	000400			MOV #400,-(SP)			
11434	064202	010346				MOV R3,-(SP)	: PTR,*		
11435	064204	010446				MOV R4,-(SP)	: SECTOR,*		
11436	064206	004767	162012			JSR PC,RETRY			
11437	064212	062706	000014			ADD #14,SP			
11438	064216	020027	000005			CMP R0,#5			
11439	064222	001052				BNE 25\$			
11440	064224	027766	150162	000046		CMP @ML.REG+44,46(SP)	: *,OLDSEC		6157
11441	064232	001035				BNE 24\$			
11442	064234	016600	000050			MOV 50(SP),R0	: OLDCHN,*		
11443	064240	017701	150144			MOV @ML.REG+42,R1			
11444	064244	006201				ASR R1			
11445	064246	006201				ASR R1			
11446	064250	006201				ASR R1			
11447	064252	006201				ASR R1			
11448	064254	006201				ASR R1			
11449	064256	006201				ASR R1			
11450	064260	042701	177700			BIC #177700,R1			
11451	064264	020100				CMP R1,R0			
11452	064266	001017				BNE 24\$			
11453	064270	032767	000001	115762		BIT #1,ERROUT	:		6161
11454	064276	001404				BEQ 23\$			
11455	064300	104456				TRAP 56			
11456	064302	000631				.WORD 631			
11457	064304	011202				.WORD MSG4			
11458	064306	000000				.WORD 0			
11459	064310	016646	000042		23\$:	MOV 42(SP),-(SP)	: LUN,*		6163
11460	064314	016746	150022			MOV BOARD,-(SP)			
11461	064320	004767	153076			JSR PC,UP.HARD.COUNT			
11462	064324	000427				BR 27\$:		6157
11463	064326	032767	000001	115724	24\$:	BIT #1,ERROUT	:		6168
11464	064334	001415				BEQ 26\$			
11465	064336	104457				TRAP 57			
11466	064340	000632				.WORD 632			
11467	064342	011166				.WORD MSG3			
11468	064344	000000				.WORD 0			
11469	064346	000410				BR 26\$:		6170
11470	064350	032767	000001	115702	25\$:	BIT #1,ERROUT	:		6176
11471	064356	001404				BEQ 26\$			

Address	OpCode	Operand1	Operand2	Label	Instruction	Comments	Address
11473					TRAP	57	
11474					.WORD	633	
11475					.WORD	MSG3	
11476	064360	104457			.WORD	0	
11477	064362	000633			MOV	42(SP), -(SP)	; LUN,* 6178
11478	064364	011166			MOV	BOARD, -(SP)	
11479	064366	000000			JSR	PC, UP.SOFT.COUNT	
11480	064370	016646	000042	26\$:	CMP	(SP)+, (SP)+	; 6145
11481	064374	016746	147742		MOV	42(SP), (SP)	; LUN,* 6184
11482	064400	004767	153500		MOV	#400, -(SP)	
11483	064404	022626		27\$:	MOV	56(SP), -(SP)	; WCPTR,*
11484	064406	016616	000042	28\$:	MOV	R4, -(SP)	; SECTOR,*
11485	064412	012746	000400		JSR	PC, WRITE	
11486	064416	016646	000056		MOV	R0, R5	; *, VALUE
11487	064422	010446			CMP	R5, #1	; VALUE,* 6190
11488	064424	004767	160774		BNE	29\$	
11489	064430	010005			MOV	#6, -(SP)	
11490	064432	020527	000001		MOV	#WRITE, -(SP)	
11491	064436	001036			MOV	54(SP), -(SP)	; LUN,*
11492	064440	012746	000006		MOV	#400, -(SP)	
11493	064444	012746	045424		MOV	72(SP), -(SP)	; WCPTR,*
11494	064450	016646	000054		MOV	R4, -(SP)	; SECTOR,*
11495	064454	012746	000400		JSR	PC, RETRY	
11496	064460	016646	000072		ADD	#14, SP	
11497	064464	010446			TST	R0	
11498	064466	004767	161532		BEQ	32\$	
11499	064472	062706	000014		MOV	50(SP), R1	; LUN,* 6199
11500	064476	005700			MOVB	#4, WHY.DROPT(R1)	
11501	064500	001456			TRAP	55	; 6200
11502	064502	016601	000050		.WORD	634	
11503	064506	112761	000004	034446	.WORD	MSG1	
11504	064514	104455			.WORD	0	
11505	064516	000634			MOV	50(SP), R0	; LUN,* 6201
11506	064520	011070			TRAP	51	
11507	064522	000000			BR	31\$; 6202
11508	064524	016600	000050		CMP	R5, #2	; VALUE,* 6190
11509	064530	104451			BNE	30\$	
11510	064532	000436			MOV	50(SP), R1	; LUN,* 6209
11511	064534	020527	000002	29\$:	MOVB	#5, WHY.DROPT(R1)	
11512	064540	001015			TRAP	55	; 6210
11513	064542	016601	000050		.WORD	635	
11514	064546	112761	000005	034446	.WORD	MSG1	
11515	064554	104455			.WORD	0	
11516	064556	000635			MOV	50(SP), R0	; LUN,* 6211
11517	064560	011070			TRAP	51	
11518	064562	000000			BR	31\$; 6212
11519	064564	016600	000050		CMP	R5, #3	; VALUE,* 6190
11520	064570	104451			BNE	32\$	
11521	064572	000416			TRAP	55	; 6217
11522	064574	020527	000003	30\$:	.WORD	636	
11523	064600	001016			.WORD	MSG1	
11524	064602	104455			.WORD	0	
11525	064604	000636					
11526	064606	011070					
11527	064610	000000					

Address	Hex	Hex	Hex	Label	Instruction	Comment	Seq
11585				:MLX4			
11586				:			
11587					DEFINITION OF OPTION 4		
11588	065064	012716	011216		MOV #MSG5,(SP)		6262
11589	065070	012746	007072		MOV #SAY1,-(SP)		
11590	065074	012746	000002		MOV #2,-(SP)		
11591	065100	010600			MOV SP,R0	: SP,*	
11592	065102	104414			TRAP 14		
11593	065104	016616	000072		MOV 72(SP),(SP)	: DBL.VALUE,*	6264
11594	065110	017646	000072		MOV @72(SP),-(SP)	: DBL.VALUE,*	
11595	065114	012746	006710		MOV #FMT12A,-(SP)		
11596	065120	012746	000003		MOV #3,-(SP)		
11597	065124	010600			MOV SP,R0	: SP,*	
11598	065126	104414			TRAP 14		
11599	065130	062766	010000 000100		ADD #10000,100(SP)	: *,DBL.VALUE	6266
11600	065136	016616	000100		MOV 100(SP),(SP)	: DBL.VALUE,*	6267
11601	065142	017646	000100		MOV @100(SP),-(SP)	: DBL.VALUE,*	
11602	065146	012746	006756		MOV #FMT12B,-(SP)		
11603	065152	012746	000003		MOV #3,-(SP)		
11604	065156	010600			MOV SP,R0	: SP,*	
11605	065160	104414			TRAP 14		
11606	065162	016601	000070		MOV 70(SP),R1	: LUN,*	6269
11607	065166	112761	000010 034446		MOVB #10,WHY.DROPT(R1)		
11608	065174	104455			TRAP 55		6270
11609	065176	000637			.WORD 637		
11610	065200	011070			.WORD MSG1		
11611	065202	000000			.WORD 0		
11612	065204	016600	000070		MOV 70(SP),R0	: LUN,*	6271
11613	065210	104451			TRAP 51		
11614	065212	062706	000054		ADD #54,SP		6272
11615	065216	000520		40\$:	BR 46\$		
11616	065220	020527	000001	41\$:	CMP R5,#1	: VALUE,*	6249
11617	065224	001034			BNE 42\$		
11618	065226	012746	000006		MOV #6,-(SP)		6280
11619	065232	010246			MOV R2,-(SP)	: COMMAND,*	
11620	065234	016646	000052		MOV 52(SP),-(SP)	: LUN,*	
11621	065240	012746	000400		MOV #400,-(SP)		
11622	065244	010346			MOV R3,-(SP)	: PTR,*	
11623	065246	010446			MOV R4,-(SP)	: SECTOR,*	
11624	065250	004767	160750		JSR PC,RETRY		
11625	065254	062706	000014		ADD #14,SP		
11626	065260	005700			TST R0		
11627	065262	001655			BEQ 38\$		
11628	065264	016601	000046		MOV 46(SP),R1	: LUN,*	6283
11629	065270	112761	000004 034446		MOVB #4,WHY.DROPT(R1)		
11630	065276	104455			TRAP 55		6284
11631	065300	000640			.WORD 640		
11632	065302	011070			.WORD MSG1		
11633	065304	000000			.WORD 0		
11634	065306	016600	000046		MOV 46(SP),R0	: LUN,*	6285
11635	065312	104451			TRAP 51		
11636	065314	000457			BR 45\$		6286
11637	065316	020527	000002	42\$:	CMP R5,#2	: VALUE,*	6249
11638	065322	001015			BNE 43\$		
11639	065324	016601	000046		MOV 46(SP),R1	: LUN,*	6293

Address	OpCode	Operand1	Operand2	Label	Instruction	Comment	Address	Time	Page
11641								27-Mar-1982 19:24:42	TOPS
11642								27-Mar-1982 19:23:44	PA:<
11643									
11644	065330	112761	000005	034446	MOVB	#5,WHY.DROPT(R1)			
11645	065336	104455			TRAP	55	:		6294
11646	065340	000641			.WORD	641			
11647	065342	011070			.WORD	MSG1			
11648	065344	000000			.WORD	0			
11649	065346	016600	000046		MOV	46(SP),R0	: LUN,*		6295
11650	065352	104451			TRAP	51			
11651	065354	000437			BR	45\$:		6296
11652	065356	020527	000003	43\$:	CMP	R5,#3	: VALUE,*		6249
11653	065362	001014			BNE	44\$			
11654	065364	104455			TRAP	55	:		6301
11655	065366	000642			.WORD	642			
11656	065370	011070			.WORD	MSG1			
11657	065372	000000			.WORD	0			
11658	065374	016601	000046		MOV	46(SP),R1	: LUN,*		6302
11659	065400	112761	000006	034446	MOVB	#6,WHY.DROPT(R1)			
11660	065406	010100			MOV	R1,R0	: LUN,*		6303
11661	065410	104451			TRAP	51			
11662	065412	000420			BR	45\$:		6304
11663	065414	020527	000004	44\$:	CMP	R5,#4	: VALUE,*		6249
11664	065420	001021			BNE	47\$			
11665	065422	004767	151704		JSR	PC,ISOLATE	:		6309
11666	065426	104455			TRAP	55	:		6310
11667	065430	000643			.WORD	643			
11668	065432	011120			.WORD	MSG2			
11669	065434	000000			.WORD	0			
11670	065436	016601	000046		MOV	46(SP),R1	: LUN,*		6311
11671	065442	112761	000007	034446	MOVB	#7,WHY.DROPT(R1)			
11672	065450	010100			MOV	R1,R0	: LUN,*		6312
11673	065452	104451			TRAP	51			
11674	065454	062706	000032	45\$:	ADD	#32,SP	:		6313
11675	065460	000167	002472	46\$:	JMP	94\$			
11676	065464	020527	000005	47\$:	CMP	R5,#5	: VALUE,*		6249
11677	065470	001161			BNE	54\$			
11678	065472	004767	151634		JSR	PC,ISOLATE	:		6318
11679	065476	032767	000001	114554	BIT	#1,ERRRUT	:		6320
11680	065504	001423			BEQ	48\$			
11681	065506	017701	146676		MOV	@ML.REG+42,R1			
11682	065512	006201			ASR	R1			
11683	065514	006201			ASR	R1			
11684	065516	006201			ASR	R1			
11685	065520	006201			ASR	R1			
11686	065522	006201			ASR	R1			
11687	065524	006201			ASR	R1			
11688	065526	042701	177700		BIC	#177700,R1			
11689	065532	010146			MOV	R1,-(SP)			
11690	065534	012746	006640		MOV	#FMT10B,-(SP)			
11691	065540	012746	000002		MOV	#2,-(SP)			
11692	065544	010600			MOV	SP,R0	: SP,*		
11693	065546	104414			TRAP	14			
11694	065550	062706	000006		ADD	#6,SP			
11695	065554	017766	146632	000052	48\$:	MOV	@ML.REG+44,52(SP)	: *,OLDSEC	6323

11697											
11698											
11699											
11700	065562	017701	146622								
11701	065566	006201									
11702	065570	006201									
11703	065572	006201									
11704	065574	006201									
11705	065576	006201									
11706	065600	006201									
11707	065602	042701	177700								
11708	065606	010166	000054								
11709	065612	012746	000001								
11710	065616	010246									
11711	065620	016646	000052								
11712	065624	012746	0000400								
11713	065630	010346									
11714	065632	010446									
11715	065634	004767	160364								
11716	065640	062706	000014								
11717	065644	020027	000005								
11718	065650	001052									
11719	065652	027766	146534	000052							
11720	065660	001035									
11721	065662	016600	000054								
11722	065666	017701	146516								
11723	065672	006201									
11724	065674	006201									
11725	065676	006201									
11726	065700	006201									
11727	065702	006201									
11728	065704	006201									
11729	065706	042701	177700								
11730	065712	020100									
11731	065714	001017									
11732	065716	032767	000001	114334							
11733	065724	001404									
11734	065726	104456									
11735	065730	000644									
11736	065732	011202									
11737	065734	000000									
11738	065736	016646	000046		49\$:						
11739	065742	016746	146374								
11740	065746	004767	151450								
11741	065752	000427									
11742	065754	032767	000001	114276	50\$:						
11743	065762	001415									
11744	065764	104457									
11745	065766	000645									
11746	065770	011166									
11747	065772	000000									
11748	065774	000410									
11749	065776	032767	000001	114254	51\$:						
11750	066004	001404									
11751	066006	104457									

27-Mar-1982 19:24:42 TOPS
27-Mar-1982 19:23:44 PA:<

6324

6326

6329

6333

6335

6329

6340

6342

6348

27-Mar-1982 19:24:42 TOPS
27-Mar-1982 19:23:44 PA:<

Address	Instruction	Operand 1	Operand 2	Label	Definition	Address
11753					DEFINITION OF OPTION 4	
11754						
11755						
11756	066010	000646			.WORD 646	
11757	066012	011166			.WORD MSG3	
11758	066014	000000			.WORD 0	
11759	066016	016646	000046	52\$:	MOV 46(SP),-(SP) ; LUN,*	6350
11760	066022	016746	146314		MOV BOARD,-(SP)	
11761	066026	004767	152052		JSR PC,UP.SOFT.COUNT	
11762	066032	022626		53\$:	CMP (SP)+,(SP)+ ;	6317
11763	066034	016616	000046	54\$:	MOV 46(SP),(SP) ; LUN,*	6356
11764	066040	012746	000400		MOV #400,-(SP)	
11765	066044	016646	000052		MOV 52(SP),-(SP) ; WDPTR,*	
11766	066050	010446			MOV R4,-(SP) ; SECTOR,*	
11767	066052	004767	157346		JSR PC,WRITE	
11768	066056	010005			MOV R0,R5 ; *,VALUE	
11769	066060	020527	000001		CMP R5,#1 ; VALUE,*	6362
11770	066064	001036			BNE 55\$	
11771	066066	012746	000006		MOV #6,-(SP) ;	6368
11772	066072	012746	045424		MOV #WRITE,-(SP)	
11773	066076	016646	000060		MOV 60(SP),-(SP) ; LUN,*	
11774	066102	012746	000400		MOV #400,-(SP)	
11775	066106	016646	000066		MOV 66(SP),-(SP) ; WDPTR,*	
11776	066112	010446			MOV R4,-(SP) ; SECTOR,*	
11777	066114	004767	160104		JSR PC,RETRY	
11778	066120	062706	000014		ADD #14,SP	
11779	066124	005700			TST R0	
11780	066126	001456			BEQ 58\$	
11781	066130	016601	000054		MOV 54(SP),R1 ; LUN,*	6371
11782	066134	112761	000004 034446		MOVB #4,WHY.DROPT(R1)	
11783	066142	104455			TRAP 55 ;	6372
11784	066144	000647			.WORD 647	
11785	066146	011070			.WORD MSG1	
11786	066150	000000			.WORD 0	
11787	066152	016600	000054		MOV 54(SP),R0 ; LUN,*	6373
11788	066156	104451			TRAP 51 ;	6374
11789	066160	000436			BR 57\$; VALUE,*	6362
11790	066162	020527	000002	55\$:	CMP R5,#2 ; VALUE,*	6362
11791	066166	001015			BNE 56\$	
11792	066170	016601	000054		MOV 54(SP),R1 ; LUN,*	6381
11793	066174	112761	000005 034446		MOVB #5,WHY.DROPT(R1)	
11794	066202	104455			TRAP 55 ;	6382
11795	066204	000650			.WORD 650	
11796	066206	011070			.WORD MSG1	
11797	066210	000000			.WORD 0	
11798	066212	016600	000054		MOV 54(SP),R0 ; LUN,*	6383
11799	066216	104451			TRAP 51 ;	6384
11800	066220	000416			BR 57\$; VALUE,*	6362
11801	066222	020527	000003	56\$:	CMP R5,#3 ; VALUE,*	6389
11802	066226	001016			BNE 58\$	
11803	066230	104455			TRAP 55 ;	6389
11804	066232	000651			.WORD 651	
11805	066234	011070			.WORD MSG1	
11806	066236	000000			.WORD 0	
11807	066240	016601	000054		MOV 54(SP),R1 ; LUN,*	6390

Address	OpCode	Operand1	Operand2	Operand3	Label	Instruction	Comments	Date	Time	Page
11809								27-Mar-1982	19:24:42	TOPS
11810								27-Mar-1982	19:23:44	PA:<
11811										
11812	066244	112761	000006	034446		MOVB #6,WHY.DROPT(R1)				
11813	066252	010100				MOV R1,R0	: LUN,*			6391
11814	066254	104451				TRAP 51				
11815	066256	062706	000040		57\$:	ADD #40,SP	:			6392
11816	066262	000570				BR 66\$				
11817	066264	062706	000016		58\$:	ADD #16,SP	:			
11818	066270	005304				DEC R4	: SECTOR			6231
11819	066272	020466	000052		59\$:	CMP R4,52(SP)	: SECTOR,*			6230
11820	066276	002402				BLT 60\$				
11821	066300	000167	176416			JMP 35\$				
11822	066304	012716	007332		60\$:	MOV #WRD24,(SP)	:			6398
11823	066310	012746	006160			MOV #FMT2,-(SP)				
11824	066314	012746	000002			MOV #2,-(SP)				
11825	066320	010600				MOV SP,R0	: SP,*			
11826	066322	104414				TRAP 14				
11827	066324	004767	157636			JSR PC,CHOOSE	:			6400
11828	066330	010002				MOV R0,R2	: *,COMMAND			
11829	066332	017666	000064	000064		MOV @64(SP),64(SP)	:			6405
11830	066340	017604	000066			MOV @66(SP),R4	: *,SECTOR			
11831	066344	000167	001120			JMP 81\$				
11832	066350	005001			61\$:	CLR R1	:			6408
11833	066352	020227	045612			CMP R2,#READ	: COMMAND,*			
11834	066356	001014				BNE 62\$				
11835	066360	005201				INC R1				
11836	066362	016603	000052			MOV 52(SP),R3	: RDPTR,PTR			6411
11837	066366	016646	000042			MOV 42(SP),-(SP)	: LUN,*			6412
11838	066372	012746	000400			MOV #400,-(SP)				
11839	066376	010346				MOV R3,-(SP)	: PTR,*			
11840	066400	010446				MOV R4,-(SP)	: SECTOR,*			
11841	066402	004767	157204			JSR PC,READ				
11842	066406	000412				BR 63\$				
11843	066410	016603	000044		62\$:	MOV 44(SP),R3	: WDPTR,PTR			6416
11844	066414	016646	000042			MOV 42(SP),-(SP)	: LUN,*			6417
11845	066420	012746	000400			MOV #400,-(SP)				
11846	066424	010346				MOV R3,-(SP)	: PTR,*			
11847	066426	010446				MOV R4,-(SP)	: SECTOR,*			
11848	066430	004767	157344			JSR PC,CHECK				
11849	066434	010005			63\$:	MOV R0,R5	: *,VALUE			
11850	066436	001103				BNE 67\$:			6424
11851	066440	006001				ROR R1	:			6429
11852	066442	103402				BLO 65\$				
11853	066444	000167	001012		64\$:	JMP 80\$				
11854	066450	016646	000054		65\$:	MOV 54(SP),-(SP)	: WDPTR,*			6433
11855	066454	016646	000064			MOV 64(SP),-(SP)	: RDPTR,*			
11856	066460	012746	000400			MOV #400,-(SP)				
11857	066464	004767	155636			JSR PC,DOUBLE.CHECK				
11858	066470	062706	000006			ADD #6,SP				
11859	066474	010066	000070			MOV R0,70(SP)	: *,DBL.VALUE			
11860	066500	001761				BEO 64\$				
11861	066502	016646	000052			MOV 52(SP),-(SP)	: LUN,*			6436
11862	066506	004767	147012			JSR PC,SAYWHO				
11863	066512	012716	011216			MOV #MSG5,(SP)	:			6437

Address	OpCode	Operand 1	Operand 2	Operand 3	Instruction	Comments	Value
11865							
11866							
11867							
11868	066516	012746	007072		MOV #SAY1,-(SP)		
11869	066522	012746	000002		MOV #2,-(SP)		
11870	066526	010600			MOV SP,R0	: SP,*	
11871	066530	104414			TRAP 14		
11872	066532	016616	000076		MOV 76(SP),(SP)	: DBL.VALUE,*	6439
11873	066536	017646	000076		MOV @76(SP),-(SP)	: DBL.VALUE,*	
11874	066542	012746	006710		MOV #FMT12A,-(SP)		
11875	066546	012746	000003		MOV #3,-(SP)		
11876	066552	010600			MOV SP,R0	: SP,*	
11877	066554	104414			TRAP 14		
11878	066556	062766	010000	000104	ADD #10000,104(SP)	: *,DBL.VALUE	6441
11879	066564	016616	000104		MOV 104(SP),(SP)	: DBL.VALUE,*	6442
11880	066570	017646	000104		MOV @104(SP),-(SP)	: DBL.VALUE,*	
11881	066574	012746	006756		MOV #FMT12B,-(SP)		
11882	066600	012746	000003		MOV #3,-(SP)		
11883	066604	010600			MOV SP,R0	: SP,*	
11884	066606	104414			TRAP 14		
11885	066610	016601	000074		MOV 74(SP),R1	: LUN,*	6444
11886	066614	112761	000010	034446	MOVB #10,WHY.DROPT(R1)		
11887	066622	104455			TRAP 55	:	6445
11888	066624	000652			.WORD 652		
11889	066626	011070			.WORD MSG1		
11890	066630	000000			.WORD 0		
11891	066632	016600	000074		MOV 74(SP),R0	: LUN,*	6446
11892	066636	104451			TRAP 51		
11893	066640	062706	000060		ADD #60,SP	:	6447
11894	066644	000520			BR 72\$		
11895	066646	020527	000001		CMP R5,#1	: VALUE,*	6424
11896	066652	001034			BNE 68\$		
11897	066654	012746	000006		MOV #6,-(SP)	:	6455
11898	066660	010246			MOV R2,-(SP)	: COMMAND,*	
11899	066662	016646	000056		MOV 56(SP),-(SP)	: LUN,*	
11900	066666	012746	000400		MOV #400,-(SP)		
11901	066672	010346			MOV R3,-(SP)	: PTR,*	
11902	066674	010446			MOV R4,-(SP)	: SECTOR,*	
11903	066676	004767	157322		JSR PC,RETRY		
11904	066702	062706	000014		ADD #14,SP		
11905	066706	005700			TST R0		
11906	066710	001655			BEQ 64\$		
11907	066712	016601	000052		MOV 52(SP),R1	: LUN,*	6458
11908	066716	112761	000004	034446	MOVB #4,WHY.DROPT(R1)		
11909	066724	104455			TRAP 55	:	6459
11910	066726	000653			.WORD 653		
11911	066730	011070			.WORD MSG1		
11912	066732	000000			.WORD 0		
11913	066734	016600	000052		MOV 52(SP),R0	: LUN,*	6460
11914	066740	104451			TRAP 51		
11915	066742	000457			BR 71\$:	6461
11916	066744	020527	000002		CMP R5,#2	: VALUE,*	6424
11917	066750	001015			BNE 69\$		
11918	066752	016601	000052		MOV 52(SP),R1	: LUN,*	6468
11919	066756	112761	000005	034446	MOVB #5,WHY.DROPT(R1)		

:MLX4
:

DEFINITION OF OPTION 4

66\$:
67\$:

68\$:

Address	Hex	Hex	Label	Instruction	Comment	Address	Page
11921			:MLX4				
11922			:		DEFINITION OF OPTION 4		
11923							
11924	066764	104455		TRAP	55		6469
11925	066766	000654		.WORD	654		
11926	066770	011070		.WORD	MSG1		
11927	066772	000000		.WORD	0		
11928	066774	016600	000052	MOV	52(SP),R0	: LUN,*	6470
11929	067000	104451		TRAP	51		
11930	067002	000437		BR	71\$		6471
11931	067004	020527	000003	CMP	R5,#3	: VALUE,*	6424
11932	067010	001014		BNE	70\$		
11933	067012	104455		TRAP	55		6476
11934	067014	000655		.WORD	655		
11935	067016	011070		.WORD	MSG1		
11936	067020	000000		.WORD	0		
11937	067022	016601	000052	MOV	52(SP),R1	: LUN,*	6477
11938	067026	112761	000006	MOVB	#6,WHY.DROPT(R1)		
11939	067034	010100		MOV	R1,R0	: LUN,*	6478
11940	067036	104451		TRAP	51		
11941	067040	000420		BR	71\$		6479
11942	067042	020527	000004	CMP	R5,#4	: VALUE,*	6424
11943	067046	001021		BNE	73\$		
11944	067050	004767	150256	JSR	PC,ISOLATE		6484
11945	067054	104455		TRAP	55		6485
11946	067056	000656		.WORD	656		
11947	067060	011120		.WORD	MSG2		
11948	067062	000000		.WORD	0		
11949	067064	016601	000052	MOV	52(SP),R1	: LUN,*	6486
11950	067070	112761	000007	MOVB	#7,WHY.DROPT(R1)		
11951	067076	010100		MOV	R1,R0	: LUN,*	6487
11952	067100	104451		TRAP	51		
11953	067102	062706	000036	ADD	#36,SP		6488
11954	067106	000167	001044	JMP	94\$		
11955	067112	020527	000005	CMP	R5,#5	: VALUE,*	6424
11956	067116	001161		BNE	80\$		
11957	067120	004767	150206	JSR	PC,ISOLATE		6493
11958	067124	032767	000001	BIT	#1,ERROUT		6495
11959	067132	001423		BEQ	74\$		
11960	067134	017701	145250	MOV	@ML.REG+42,R1		
11961	067140	006201		ASR	R1		
11962	067142	006201		ASR	R1		
11963	067144	006201		ASR	R1		
11964	067146	006201		ASR	R1		
11965	067150	006201		ASR	R1		
11966	067152	006201		ASR	R1		
11967	067154	042701	177700	BIC	#177700,R1		
11968	067160	010146		MOV	R1,-(SP)		
11969	067162	012746	006640	MOV	#FMT10B,-(SP)		
11970	067166	012746	000002	MOV	#2,-(SP)		
11971	067172	010600		MOV	SP,R0	: SP,*	
11972	067174	104414		TRAP	14		
11973	067176	062706	000006	ADD	#6,SP		
11974	067202	017766	145204	MOV	@ML.REG+44,56(SP)	: *,OLDSEC	6498
11975	067210	017701	145174	MOV	@ML.REG+42,R1	:	6499

				:MLX4	DEFINITION OF OPTION 4		
11977				:			
11978				:			
11979				:			
11980	067214	006201			ASR R1		
11981	067216	006201			ASR R1		
11982	067220	006201			ASR R1		
11983	067222	006201			ASR R1		
11984	067224	006201			ASR R1		
11985	067226	006201			ASR R1		
11986	067230	042701	177700		BIC #177700,R1		
11987	067234	010166	000060		MOV R1,60(SP)	: *,OLDCHN	
11988	067240	012746	000001		MOV #1,-(SP)	: :	6501
11989	067244	010246			MOV R2,-(SP)	: : COMMAND,*	
11990	067246	016646	000056		MOV 56(SP),-(SP)	: LUN,*	
11991	067252	012746	000400		MOV #400,-(SP)	:	
11992	067256	010346			MOV R3,-(SP)	: PTR,*	
11993	067260	010446			MOV R4,-(SP)	: SECTOR,*	
11994	067262	004767	156736		JSR PC,RETRY		
11995	067266	062706	000014		ADD #14,SP		
11996	067272	020027	000005		CMP R0,#5		
11997	067276	001052			BNE 77\$		
11998	067300	027766	145106	000056	CMP @ML.REG+44,56(SP)	: *,OLDSEC	6504
11999	067306	001035			BNE 76\$		
12000	067310	016600	000060		MOV 60(SP),R0	: OLDCHN,*	
12001	067314	017701	145070		MOV @ML.REG+42,R1		
12002	067320	006201			ASR R1		
12003	067322	006201			ASR R1		
12004	067324	006201			ASR R1		
12005	067326	006201			ASR R1		
12006	067330	006201			ASR R1		
12007	067332	006201			ASR R1		
12008	067334	042701	177700		BIC #177700,R1		
12009	067340	020100			CMP R1,R0		
12010	067342	001017			BNE 76\$		
12011	067344	032767	000001	112706	BIT #1,ERROUT	:	6508
12012	067352	001404			BEQ 75\$		
12013	067354	104456			TRAP 56		
12014	067356	000657			.WORD 657		
12015	067360	011202			.WORD MSG4		
12016	067362	000000			.WORD 0		
12017	067364	016646	000052	75\$:	MOV 52(SP),-(SP)	: LUN,*	6510
12018	067370	016746	144746		MOV BOARD,-(SP)		
12019	067374	004767	150022		JSR PC,UP.HARD.COUNT		
12020	067400	000427			BR 79\$: :	6504
12021	067402	032767	000001	112650	BIT #1,ERROUT	: :	6515
12022	067410	001415			BEQ 78\$		
12023	067412	104457			TRAP 57		
12024	067414	000660			.WORD 660		
12025	067416	011166			.WORD MSG3		
12026	067420	000000			.WORD 0		
12027	067422	000410			BR 78\$: :	6517
12028	067424	032767	000001	112626	BIT #1,ERROUT	: :	6523
12029	067432	001404			BEQ 78\$		
12030	067434	104457			TRAP 57		
12031	067436	000661			.WORD 661		

Address	OpCode	Op1	Op2	Op3	Op4	Label	Instruction	Comments	Page
12033							:MLX4		
12034							:	DEFINITION OF OPTION 4	
12035									
12036	067440	011166					.WORD	MSG3	
12037	067442	000000					.WORD	0	
12038	067444	016646	000052			78\$:	MOV	52(SP),-(SP)	: LUN,* 6525
12039	067450	016746	144666				MOV	BOARD,-(SP)	
12040	067454	004767	150424				JSR	PC,UP.SOFT.COUNT	
12041	067460	022626				79\$:	CMP	(SP)+,(SP)+	: 6492
12042	067462	062706	000010			80\$:	ADD	#10,SP	: 6406
12043	067466	005204					INC	R4	: SECTOR 6405
12044	067470	020466	000064			81\$:	CMP	R4,64(SP)	: SECTOR,*
12045	067474	101002					BHI	82\$	
12046	067476	000167	176646				JMP	61\$	
12047	067502	062706	000026			82\$:	ADD	#26,SP	: 5997
12048	067506	032767	000001	112546		83\$:	BIT	#1,EFNS21	: 6541
12049	067514	001002					BNE	84\$	
12050	067516	000167	000434				JMP	94\$	
12051	067522	016600	000014			84\$:	MOV	14(SP),RO	: LUN,* 6548
12052	067526	006300					ASL	RO	
12053	067530	016001	034500				MOV	TOP.SECT(RO),R1	
12054	067534	016004	034460				MOV	LOW.SECT(RO),R4	: *,SECTOR
12055	067540	000167	000402				JMP	93\$	
12056	067544	016646	000014			85\$:	MOV	14(SP),-(SP)	: LUN,* 6550
12057	067550	012746	000400				MOV	#400,-(SP)	
12058	067554	012746	022670				MOV	#RBUF,-(SP)	
12059	067560	010446					MOV	R4,-(SP)	: SECTOR,*
12060	067562	004767	156024				JSR	PC,READ	
12061	067566	062706	000010				ADD	#10,SP	
12062	067572	020027	000005				CMP	RO,#5	
12063	067576	001162					BNE	92\$	
12064	067600	004767	147526				JSR	PC,ISOLATE	: 6553
12065	067604	032767	000001	112446			BIT	#1,ERROUT	: 6555
12066	067612	001423					BEQ	86\$	
12067	067614	017700	144570				MOV	@ML.REG+42,RO	
12068	067620	006200					ASR	RO	
12069	067622	006200					ASR	RO	
12070	067624	006200					ASR	RO	
12071	067626	006200					ASR	RO	
12072	067630	006200					ASR	RO	
12073	067632	006200					ASR	RO	
12074	067634	042700	177700				BIC	#177700,RO	
12075	067640	010046					MOV	RO,-(SP)	
12076	067642	012746	006640				MOV	#FMT10B,-(SP)	
12077	067646	012746	000002				MOV	#2,-(SP)	
12078	067652	010600					MOV	SP,RO	: SP,*
12079	067654	104414					TRAP	14	
12080	067656	062706	000006				ADD	#6,SP	
12081	067662	017766	144524	000020		86\$:	MOV	@ML.REG+44,20(SP)	: *,OLDSEC 6564
12082	067670	017700	144514				MOV	@ML.REG+42,RO	: 6565
12083	067674	006200					ASR	RO	
12084	067676	006200					ASR	RO	
12085	067700	006200					ASR	RO	
12086	067702	006200					ASR	RO	
12087	067704	006200					ASR	RO	

```

12089      :MLX4
12090      :
12091
12092 067706 006200      ASR      RO
12093 067710 042700 177700 BIC      #177700,RO
12094 067714 010066 000022 MOV      RO,22(SP)      : *,OLDCHN
12095 067720 012746 000001 MOV      #1,-(SP)      :
12096 067724 010246      MOV      R2,-(SP)      : COMMAND,*
12097 067726 016646 000020 MOV      20(SP),-(SP)  : LUN,*
12098 067732 012746 000400 MOV      #400,-(SP)   :
12099 067736 010346      MOV      R3,-(SP)      : PTR,*
12100 067740 016646 000032 MOV      32(SP),-(SP)  : OLDSEC,*
12101 067744 004767 156254 JSR      PC,RETRY
12102 067750 062706 000014 ADD      #14,SP
12103 067754 020027 000005 CMP      RO,#5
12104 067760 001052      BNE      89$
12105 067762 027766 144424 000020 CMP      @ML.REG+44,20(SP) : *,OLDSEC
12106 067770 001035      BNE      88$
12107 067772 016646 000022 MOV      22(SP),-(SP)  : OLDCHN,*
12108 067776 017700 144406 MOV      @ML.REG+42,RO
12109 070002 006200      ASR      RO
12110 070004 006200      ASR      RO
12111 070006 006200      ASR      RO
12112 070010 006200      ASR      RO
12113 070012 006200      ASR      RO
12114 070014 006200      ASR      RO
12115 070016 042700 177700 BIC      #177700,RO
12116 070022 020026      CMP      RO,(SP)+
12117 070024 001017      BNE      88$
12118 070026 032767 000001 112224 BIT      #1,ERROUT      :
12119 070034 001404      BEQ      87$
12120 070036 104456      TRAP    56
12121 070040 000662      .WORD  662
12122 070042 011202      .WORD  MSG4
12123 070044 000000      .WORD  0
12124 070046 016646 000014 87$: MOV      14(SP),-(SP)   : LUN,*
12125 070052 016746 144264      MOV      BOARD,-(SP)
12126 070056 004767 147340      JSR      PC,UP.HARD.COUNT
12127 070062 000427      BR      91$
12128 070064 032767 000001 112166 88$: BIT      #1,ERROUT      :
12129 070072 001415      BEQ      90$
12130 070074 104457      TRAP    57
12131 070076 000663      .WORD  663
12132 070100 011166      .WORD  MSG3
12133 070102 000000      .WORD  0
12134 070104 000410      BR      90$
12135 070106 032767 000001 112144 89$: BIT      #1,ERROUT      :
12136 070114 001404      BEQ      90$
12137 070116 104457      TRAP    57
12138 070120 000664      .WORD  664
12139 070122 011166      .WORD  MSG3
12140 070124 000000      .WORD  0
12141 070126 016646 000014 90$: MOV      14(SP),-(SP)   : LUN,*
12142 070132 016746 144204      MOV      BOARD,-(SP)
12143 070136 004767 147742      JSR      PC,UP.SOFT.COUNT

```

```
12145          ;MLX4
12146          .
12147          .
12148 070142 022626          91$:  CMP      (SP)+,(SP)+
12149 070144 005204          92$:  INC      R4
12150 070146 020401          93$:  CMP      R4,R1
12151 070150 101002          BHI     94$
12152 070152 000167 177366  JMP     85$
12153 070156 005266 000014 94$:  INC      14(SP)
12154 070162 026666 000014 000042 95$:  CMP      14(SP),42(SP)
12155 070170 002002          BGE     96$
12156 070172 000167 172402  JMP     1$
12157 070176 062706 000044 96$:  ADD     #44,SP
12158 070202 000207          RTS     PC
12159
12160          ; Routine Size: 1456 words
12161          ; Maximum stack depth per invocation: 48 words
12166
12167
```

27-Mar-1982 19:24:42 TOPS
27-Mar-1982 19:23:44 PA:<

6552
6548

5990

5908

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (41)

```

12169 :MLX4
12170 :
12171 :
12172 :      6633 %sbttl 'SELECTING A RANDOM WORD COUNT'
12173 :      6634 routine RNDWC =
12174 :      6635     begin
12175 :      6636
12176 :      6637     ++
12177 :      6638     ROUTINE:      RNDWC
12178 :      6639
12179 :      6640     PURPOSE:     TO SELECT A RANDOM WORD COUNT WITHIN THE RANGE
12180 :      6641     1 TO BUFSIZ.
12181 :      6642
12182 :      6643     RESULT:      THE VALUE RETURNED WILL BE USED BY THE CALLER
12183 :      6644     AS ITS 'WRDCNT'
12184 :      6645     --
12185 :      6646
12186 :      6647     local
12187 :      6648     WRDCNT;
12188 :      6649
12189 :      6650     RN ();
12190 :      6651     RANDOM = .RANDOM and %o'077777';           !IGNORE SIGN BIT
12191 :      6652     WRDCNT = ((.RANDOM mod BUFSIZ) + 1);       !FORCE THE RANGE
12192 :      6653     return .WRDCNT;
12193 :      6654     end;
12197 :
12198 :      .SBTTL RNDWC SELECTING A RANDOM WORD COUNT
12202 070204 004767 115160 RNDWC: JSR PC,RN ;
12203 070210 042767 100000 115246 BIC #100000,RANDOM ;
12204 070216 016746 115242 MOV RANDOM,-(SP) ;
12205 070222 012746 004000 MOV #4000,-(SP) ;
12206 070226 004767 114750 JSR PC,BL$MOD ;
12207 070232 005200 INC R0 ;
12208 070234 022626 CMP (SP)+,(SP)+ ;
12209 070236 000207 RTS PC ;
12210 :
12211 :      ; Routine Size: 14 words
12212 :      ; Maximum stack depth per invocation: 2 words
12217 :
12218 :
    
```

6650
 6651
 6652
 6634

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (42)

```

12220 :MLX4
12221 :
12222 : SELECTING A RANDOM SECTOR
12223 : 6655 %sbttl 'SELECTING A RANDOM SECTOR'
12224 : 6656 routine RNDSEC (LUN) =
12225 : 6657 begin
12226 : 6658
12227 : 6659 :++
12228 : 6660 : ROUTINE: RNDSEC(LUN)
12229 : 6661 :
12230 : 6662 : PURPOSE: TO SELECT A RANDOM SECTOR NUMBER WITHIN THE RANGE
12231 : 6663 : OF TESTABLE SECTORS (LOWEST TO HIGHEST)
12232 : 6664 :
12233 : 6665 : ARGUMENT: LUN = THE CURRENT LOGICAL UNIT
12234 : 6666 : NOTE: 'LUN' IS REQUIRED TO CALCULATE LOWEST/HIGHEST
12235 : 6667 : FOR THE PARTICULAR LOGICAL UNIT.
12236 : 6668 :
12237 : 6669 : RESULT: THE VALUE RETURNED WILL BE USED BY THE CALLER
12238 : 6670 : AS ITS 'SECTOR'.
12239 : 6671 :--
12240 : 6672
12241 : 73 local
12242 : 6674 SECTOR,
12243 : 6675 SIZE;
12244 : 6676
12245 : 6677 :
12246 : 6678 : version czmlbb changed eql to eqlu
12247 : 6679 :
12248 : 6680 : if LOWEST eqlu HIGHEST
12249 : 6681 : then
12250 : 6682 : SECTOR = LOWEST
12251 : 6683 : else
12252 : 6684 : begin
12253 : 6685 : RN ();
12254 : 6686 : RANDOM = .RANDOM and %o'077777'; !IGNORE THE SIGN BIT
12255 : 6687 :
12256 : 6688 : version czmlbb
12257 : 6689 :
12258 : 6690 : 'Highest - Lowest + 1' when testing an ML-11B with
12259 : 6691 : 16 array modules results in a negative number which
12260 : 6692 : causes the Bliss operator 'mod' to hang the CPU.
12261 : 6693 : Due to this malfunction the '+ 1' has been deleted
12262 : 6694 : which will avoid this malfunction
12263 : 6695 :
12264 : 6696 : SIZE = HIGHEST - LOWEST; !FIND THE SECTOR RANGE
12265 : 6697 : SECTOR = (LOWEST + (.RANDOM mod .SIZE)); !FORCE RANGE
12266 : 6698 : end;
12267 : 6699 :
12268 : 6700 : return .SECTOR;
12269 : 6701 : end;
12270 :
12271 :
12272 :
12273 :
12274 : .SBTTL RNDSEC SELECTING A RANDOM SECTOR
  
```

12276				:MLX4						27-Mar-1982 19:24:42	TOPS
12277				:		SELECTING A RANDOM SECTOR				27-Mar-1982 19:23:44	PA:<
12278				:							
12282	070240	004167	115032	RNDSEC:	JSR	R1,\$SAVE3	:				6656
12283	070244	016601	000012		MOV	12(SP),R1	:	LUN,*			6680
12284	070250	006301			ASL	R1	:				
12285	070252	012703	034460		MOV	#LOW,SECT,R3	:				
12286	070256	060103			ADD	R1,R3	:				
12287	070260	021361	C34500		CMP	(R3),TOP.SECT(R1)	:				
12288	070264	001002			BNE	1\$:				
12289	070266	011302			MOV	(R3),R2	:	*,SECTOR			6682
12290	070270	000420			BR	2\$:				6680
12291	070272	004767	115072	1*:	JSR	PC,RN	:				6685
12292	070276	042767	100000	115160	BIC	#100000,RANDOM	:				6686
12293	070304	016101	034500		MOV	TOP.SECT(R1),R1	:	*,SIZE			6696
12294	070310	161301			SUB	(R3),R1	:	*,SIZE			
12295	070312	016746	115146		MOV	RANDOM,-(SP)	:				6697
12296	070316	010146			MOV	R1,-(SP)	:	SIZE,*			
12297	070320	004767	114656		JSR	PC,BL\$MOD	:				
12298	070324	061300			ADD	(R3),R0	:				
12299	070326	010002			MOV	R0,R2	:	*,SECTOR			
12300	070330	022626			CMP	(SP)+,(SP)+	:				6684
12301	070332	010200		2\$:	MOV	R2,R0	:	SECTOR,*			6657
12302	070334	000207			RTS	PC	:				6656
12303							:				
12304							:				
12305							:				
12310							:				
12311							:				

; Routine Size: 31 words
 ; Maximum stack depth per invocation: 6 words

```

12313 :MLX4
12314 :
12315 :
12316 : 6702 %sbttl 'SELECTING A RANDOM UNIT'
12317 : 6703 routine RNDU =
12318 : 6704 begin !* 1 *
12319 : 6705
12320 : 6706 !++
12321 : 6707 ROUTINE: RNDU
12322 : 6708
12323 : 6709 PURPOSE: TO SELECT A RANDOM LOGICAL UNIT NUMBER WITHIN
12324 : 6710 THE RANGE OF TESTABLE UNITS (0 TO .LSUNIT-1)
12325 : 6711 !--
12326 : 6712
12327 : 6713 local
12328 : 6714 LUN;
12329 : 6715
12330 : 6716 if .LSUNIT eql 1
12331 : 6717 then
12332 : 6718 LUN = 0
12333 : 6719 else
12334 : 6720 begin !* 2 *
12335 : 6721 RN ();
12336 : 6722 RANDOM = .RANDOM and %o'077777'; !IGNORE THE SIGN BIT
12337 : 6723 RANDOM = (.RANDOM mod .LSUNIT); !FORCE THE RANGE
12338 : 6724
12339 : 6725 !+
12340 : 6726 MAKE SURE THE DRIVE IS ACTIVE. IF IT ISN'T,
12341 : 6727 THEN FIND THE NEXT AVAILABLE ACTIVE DRIVE:
12342 : 6728 !-
12343 : 6729
12344 : 6730 incr COUNT from 0 to (.LSUNIT - 1) do
12345 : 6731
12346 : 6732 if .DRIVE_STATUS [.RANDOM] eql ACTIVE
12347 : 6733 then
12348 : 6734 begin
12349 : 6735 LUN = .RANDOM;
12350 : 6736 exitloop;
12351 : 6737 end
12352 : 6738 else
12353 : 6739 RANDOM = ((.RANDOM + 1) mod .LSUNIT);
12354 : 6740
12355 : 6741 end; !* 2 *
12356 : 6742
12357 : 6743 return .LUN;
12358 : 6744 end; !* 1 *

```

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (43)

12362
 12363
 12367 070336 004167 114752 RNDU: .SBTTL RNDU SELECTING A RANDOM UNIT
 JSR R1,\$SAVE4 ;

```

12369      :MLX4
12370      :
12371      :
12372 070342 026727 111444 000001      CMP     L$UNIT,#1      :
12373 070350 001002      BNE     1$           :
12374 070352 005004      CLR     R4           ; LUN
12375 070354 000467      BR      6$           :
12376 070356 004767 115006      JSR     PC,RN        :
12377 070362 042767 100000 115074      BIC     #10000,RANDOM :
12378 070370 016746 115070      MOV     RANDOM,-(SP) :
12379 070374 016746 111412      MOV     L$UNIT,-(SP) :
12380 070400 004767 114576      JSR     PC,BL$MOD    :
12381 070404 010067 115054      MOV     R0,RANDOM   :
12382 070410 016703 111376      MOV     L$UNIT,R3   :
12383 070414 005001      CLR     R1           ; COUNT
12384 070416 000443      BR      4$           :
12385 070420 016702 115040      2$:  MOV     RANDOM,R2 :
12386 070424 006202      ASR     R2           :
12387 070426 006202      ASR     R2           :
12388 070430 006202      ASR     R2           :
12389 070432 062702 034442      ADD     #DRIVE.STATUS,R2
12390 070436 010246      MOV     R2,-(SP)
12391 070440 016746 115020      MOV     RANDOM,-(SP)
12392 070444 042716 177770      BIC     #177770,(SP)
12393 070450 012746 000001      MOV     #1,-(SP)
12394 070454 005046      CLR     -(SP)
12395 070456 004767 113674      JSR     PC,BL$GT2
12396 070462 062706 000010      ADD     #10,SP
12397 070466 005300      DEC     R0
12398 070470 001003      BNE     3$           :
12399 070472 016704 114766      MOV     RANDOM,R4   ; *.LUN
12400 070476 000415      BR      5$           :
12401 070500 016746 114760      3$:  MOV     RANDOM,-(SP) :
12402 070504 005216      INC     (SP)         :
12403 070506 016746 111300      MOV     L$UNIT,-(SP) :
12404 070512 004767 114464      JSR     PC,BL$MOD    :
12405 070516 010067 114742      MOV     R0,RANDOM   :
12406 070522 022626      CMP     (SP)+,(SP)+ :
12407 070524 005201      INC     R1           ; COUNT
12408 070526 020103      4$:  CMP     R1,R3       ; COUNT,*
12409 070530 002733      BLT     2$           :
12410 070532 022626      5$:  CMP     (SP)+,(SP)+ :
12411 070534 010400      6$:  MOV     R4,R0       ; LUN,*
12412 070536 000207      RTS     PC           :
12413
12414      ; Routine Size: 65 words
12415      ; Maximum stack depth per invocation: 11 words
12420
12421

```

```

27-Mar-1982 19:24:42 TOPS
27-Mar-1982 19:23:44 PA:<
:
6716
6718
6716
6721
6722
6723
6730
6732
6735
6736
6739
6730
6720
6704
6703

```

12423 :MLX4
 12424 :
 12425 :
 12426 :
 12427 :
 12428 :
 12429 :
 12430 :
 12431 :
 12432 :
 12433 :
 12434 :
 12435 :
 12436 :
 12437 :
 12438 :
 12439 :
 12440 :
 12441 :
 12442 :
 12443 :
 12444 :
 12445 :
 12446 :
 12447 :
 12448 :
 12449 :
 12450 :
 12451 :
 12452 :
 12453 :
 12454 :
 12455 :
 12456 :
 12457 :
 12458 :
 12459 :
 12460 :
 12461 :
 12462 :
 12463 :
 12464 :
 12465 :
 12466 :
 12467 :
 12468 :
 12469 :
 12470 :
 12471 :
 12472 :
 12473 :
 12474 :
 12475 :
 12476 :
 12477 :

TESTING RANDOM DATA

```

6745 %sbttl 'TESTING RANDOM DATA'
6746 routine RAND1 (REPEAT) : novalue =
6747     begin
6748
6749     ++
6750     ROUTINE:      RAND1(REPEAT)
6751
6752     PURPOSE:     TO TEST USING RANDOM DATA
6753
6754     ARGUMENT:    REPEAT = NUMBER OF TIMES TO EXECUTE THIS ROUTINE
6755                 BEFORE RETURNING TO THE CALLER (OPT5).
6756
6757     NOTE:        THIS TEST CODE FOLLOWS THE SAME FLOW AS OPT3,
6758                 BUT USES A RANDOM DATA PATTERN.
6759
6760     THE CODE FOR 'RAND1' IN BRIEF:
6761
6762     BEGIN 1 (START OF ROUTINE)
6763     SAY ROUTINE IS RUNNING
6764     INCR COUNT FROM 1 TO REPEAT
6765     : BEGIN 2 (START OF REPEAT LOOP FOR THE ROUTINE)
6766     : GENERATE THE RANDOM PATTERN
6767     : INCR LUN FROM 0 TO LAST
6768     : : BEGIN 3 (START OF LOGICAL UNIT SELECTION LOOP)
6769     : : TESTLOOP:
6770     : : : BEGIN 4 (START OF LOOP THAT COMPLETELY TESTS 1 UNIT)
6771     : : : IF UNIT IS ACTIVE
6772     : : : THEN
6773     : : : : BEGIN 5 (START OF TEST FOR AN ACTIVE UNIT)
6774     : : : : INITIALIZE THE WRITE AND READ BUFFER POINTERS
6775     : : : : SECTOR = LOWEST
6776     : : : : WHILE SECTOR LEQ HIGHEST DO
6777     : : : : : BEGIN 6 (START OF SECTOR SELECTION LOOP)
6778     : : : : : GET WRDCNT
6779     : : : : : SET-UP BUFFER POINTERS BEFORE TRANSFER
6780     : : : : : WRITE
6781     : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
6782     : : : : : CHOOSE WHETHER TO WRITE CHECK OR READ
6783     : : : : : DO THE WRITE CHECK OR READ
6784     : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
6785     : : : : : CHANGE BUFFER POINTERS AFTER TRANSFER
6786     : : : : : UPDATE SECTOR NUMBER BY NUMBER OF SECTORS IN PREVIOUS TRANSFER
6787     : : : : : END 6 (END OF SECTOR SELECTION LOOP)
6788     : : : : : END 5 (END OF TEST FOR AN ACTIVE UNIT)
6789     : : : : : END 4 (END OF TESTLOOP)
6790     : : : : : END 3 (END OF LOGICAL UNIT SELECTION LOOP)
6791     : : : : : END 2 (END OF REPEAT LOOP FOR THIS ROUTINE)
6792     RETURN
6793     END 1 (END OF ROUTINE)
6794
6795
6796     Label
  
```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (44)

12479 :MLX4
 12480 :
 12481 :
 12482 :
 12483 :
 12484 :
 12485 :
 12486 :
 12487 :
 12488 :
 12489 :
 12490 :
 12491 :
 12492 :
 12493 :
 12494 :
 12495 :
 12496 :
 12497 :
 12498 :
 12499 :
 12500 :
 12501 :
 12502 :
 12503 :
 12504 :
 12505 :
 12506 :
 12507 :
 12508 :
 12509 :
 12510 :
 12511 :
 12512 :
 12513 :
 12514 :
 12515 :
 12516 :
 12517 :
 12518 :
 12519 :
 12520 :
 12521 :
 12522 :
 12523 :
 12524 :
 12525 :
 12526 :
 12527 :
 12528 :
 12529 :
 12530 :
 12531 :
 12532 :
 12533 :

```

TESTING RANDOM DATA

LOOP:
  local
  WRDCNT,
  VALUE,
  OLDSEC,
  OLDCHN,
  SECTOR,
  PTR,
  COMMAND,
  DBL_VALUE;

PRINTB (SAY1, RTNSA);
!'RAND1'

incr COUNT from 1 to .REPEAT do
  begin
  GENS ();
  incr LUN from 0 to (.LSUNIT - 1) do
    begin
    LOOP :
      begin
      if .DRIVE_STATUS [.LUN] eq1 ACTIVE
      then
        begin
        L$LUN = .LUN;
        WPTR = W$WFF;
        RPTR = R$WFF;
        SECTOR = LOWEST;

        while .SECTOR lequ HIGHEST do
          begin
          WRDCNT = GET WRDCNT (.SECTOR, HIGHEST);
          SET PTRS (.WRDCNT);
          VALUE = write (.LUN, .WRDCNT, .WPTR, .SECTOR);

          !+
          !- SEE HOW SUCCESSFUL THE WRITE WAS:

          selectone .VALUE of
            set
            [1] :
            begin
            if RETRY (SIX, write, .LUN, .WRDCNT, .WPTR, .SECTOR) neq 0
            then
              begin
              WHY_DROPT [.LUN] = CODE_4;
  
```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (44)

!* 2 * START OF REPEAT LOOP FOR THIS ROUTINE
 !FIRST 3 WORDS OF W\$WFF ARE THE SEEDS
 !* 3 * START OF LOGICAL UNIT SELECTION LOOP
 !* 4 * START OF THE LOOP THAT COMPLETELY TESTS 1 UNIT
 !* 5 * START OF TEST FOR AN ACTIVE UNIT
 !* 6 * START OF SECTOR SELECTION LOOP
 !SEE 'SYSERR' FOR DEFINITION
 !OF ERROR # CONTAINED IN 'VALUE'
 !* 6A * RETRY ALLOWED
 !THE RETRY FAILED -- SYSTEM FATAL ERROR

```

12535 :MLX4
12536 :          TESTING RANDOM DATA
12537 :
12538 :          6849          ERRDF (5101, MSG1, 0); !**** OPTION 5, RAND1 ERROR 01 ****
12539 :          6850          DODU (.LUN);
12540 :          6851          leave LOOP;          !JUMP JUST BEYOND END OF BLOCK * 4 *
12541 :          6852          end;
12542 :          6853
12543 :          6854          end;          !* 6A *
12544 :          6855
12545 :          6856          [2] :
12546 :          6857          begin          !* 6B * FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
12547 :          6858          WHY DROPT [.LUN] = CODE_5;
12548 :          6859          ERRDF (5102, MSG1, 0);          !**** OPTION 5, RAND1 ERROR 02 ****
12549 :          6860          DODU (.LUN);
12550 :          6861          leave LOOP;          !JUMP JUST BEYOND END OF BLOCK * 4 *
12551 :          6862          end;          !* 6B *
12552 :          6863
12553 :          6864          [3] :
12554 :          6865          begin          !* 6C * FATAL DRIVE ERROR -- NO RETRY ALLOWED
12555 :          6866          ERRDF (5103, MSG1, 0);          !**** OPTION 5, RAND1 ERROR 03 ****
12556 :          6867          WHY DROPT [.LUN] = CODE_6;
12557 :          6868          DODU (.LUN);
12558 :          6869          leave LOOP;          !JUMP JUST BEYOND END OF BLOCK * 4 *
12559 :          6870          end;          !* 6C *
12560 :          6871          tes;
12561 :          6872
12562 :          6873          COMMAND = CHOOSE ();
12563 :          6874
12564 :          6875          if .COMMAND eql read
12565 :          6876          then
12566 :          6877          begin
12567 :          6878          PTR = .RPTR;
12568 :          6879          VALUE = read (.LUN, .WRDCNT, .RPTR, .SECTOR);
12569 :          6880          end
12570 :          6881          else
12571 :          6882          begin
12572 :          6883          PTR = .WPTR;
12573 :          6884          VALUE = CHECK (.LUN, .WRDCNT, .WPTR, .SECTOR);
12574 :          6885          end;
12575 :          6886
12576 :          6887          !+
12577 :          6888          ! SEE HOW SUCCESSFUL THE OPERATION WAS:
12578 :          6889          !-
12579 :          6890
12580 :          6891          selectone .VALUE of          !SEE 'SYSERR' FOR DEFINITION
12581 :          6892          set          !OF ERROR # CONTAINED IN 'VALUE'
12582 :          6893
12583 :          6894          [0] :
12584 :          6895
12585 :          6896          if .COMMAND eql read
12586 :          6897          then
12587 :          6898          begin
12588 :          6899
12589 :          6900          if (DBL_VALUE = DOUBLE_CHECK (.WPTR, .RPTR, .WRDCNT)) neq 0
  
```


27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (44)

12591 :MLX4
 12592 :
 12593 :
 12594 : 5901
 12595 : 6702
 12596 : 6903
 12597 : 6904
 12598 : 6905
 12599 : 6906
 12600 : 6907
 12601 : 6908
 12602 : 6909
 12603 : 6910
 12604 : 6911
 12605 : 6912
 12606 : 6913
 12607 : 6914
 12608 : 6915
 12609 : 6916
 12610 : 6917
 12611 : 6918
 12612 : 6919
 12613 : 6920
 12614 : 6921
 12615 : 6922
 12616 : 6923
 12617 : 6924
 12618 : 6925
 12619 : 6926
 12620 : 6927
 12621 : 6928
 12622 : 6929
 12623 : 6930
 12624 : 6931
 12625 : 6932
 12626 : 6933
 12627 : 6934
 12628 : 6935
 12629 : 6936
 12630 : 6937
 12631 : 6938
 12632 : 6939
 12633 : 6940
 12634 : 6941
 12635 : 6942
 12636 : 6943
 12637 : 6944
 12638 : 6945
 12639 : 6946
 12640 : 6947
 12641 : 6948
 12642 : 6949
 12643 : 6950
 12644 : 6951
 12645 : 6952

TESTING RANDOM DATA

```

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
then
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;

```

12647 :MLX4
 12648 :
 12649 :
 12650 :
 12651 :
 12652 :
 12653 :
 12654 :
 12655 :
 12656 :
 12657 :
 12658 :
 12659 :
 12660 :
 12661 :
 12662 :
 12663 :
 12664 :
 12665 :
 12666 :
 12667 :
 12668 :
 12669 :
 12670 :
 12671 :
 12672 :
 12673 :
 12674 :
 12675 :
 12676 :
 12677 :
 12678 :
 12679 :
 12680 :
 12681 :
 12682 :
 12683 :
 12684 :
 12685 :
 12686 :
 12687 :
 12688 :
 12689 :
 12690 :
 12691 :
 12692 :
 12693 :
 12694 :
 12695 :
 12696 :
 12697 :
 12698 :
 12699 :
 12700 :
 12701 :

TESTING RANDOM DATA

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (44)

```

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;

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 ERRSOFT (5110, MSG3, 0);

      !**** OPTION 5, RAND1 ERROR 10 ****
      UP_SOFT_COUNT (.LUN, .BOARD);
    end
  else
    begin
      if .ERROUT then ERRSOFT (5111, MSG3, 0);

      !**** OPTION 5, RAND1 ERROR 11 ****
      UP_SOFT_COUNT (.LUN, .BOARD);
    end;
  end;
end;                 !* 6H *
tes;

WPTR = .WPTR + (.WRDCNT*2);
SECTOR = .SECTOR + (.WRDCNT/256);
end;                 !* 6 * END OF SECTOR SELECTION LOOP

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

```

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (44)

12703 :MLX4
 12704 :
 12705 :
 12706 :
 12707 :
 12708 :
 12709 :
 12710 :
 12711 :
 12712 :
 12713 :
 12714 :
 12715 :
 12716 :
 12717 :
 12718 :
 12719 :
 12720 :
 12721 :
 12722 :
 12723 :
 12724 :
 12725 :
 12726 :
 12727 :
 12728 :
 12729 :
 12730 :
 12731 :
 12732 :
 12733 :
 12734 :
 12735 :
 12736 :
 12737 :
 12738 :
 12739 :
 12740 :
 12741 :
 12742 :
 12743 :
 12744 :
 12745 :
 12746 :
 12747 :
 12748 :
 12749 :
 12750 :
 12751 :
 12752 :
 12753 :
 12754 :
 12755 :
 12756 :
 12757 :

TESTING RANDOM DATA

7005
 7006
 7007
 7008
 7009
 7010
 7011
 7012
 7013
 7014
 7015
 7016
 7017
 7018
 7019
 7020
 7021
 7022
 7023
 7024
 7025
 7026
 7027
 7028
 7029
 7030
 7031
 7032
 7033
 7034
 7035
 7036
 7037
 7038
 7039
 7040
 7041
 7042
 7043
 7044
 7045
 7046
 7047
 7048
 7049
 7050
 7051
 7052
 7053
 7054
 7055
 7056

```

: +
: Test to see if this uut's address space is
: to be read for soft errors. This test is
: is intended for DMT purposes.
: -
if .EFN$21 !Is the background pattern to be read
then
  begin
: version czmlbb changed incr to incru
  incru SECTOR from LOWEST to HIGHEST do
    if read (.LUN, 256, RBUFF, .SECTOR) eql 5
    then
      begin
        ISOLATE (); !Find the failing bank and board no.
        if .ERROUT then PRINTB (FMT10B, .CHAN);
        ! Print where the error is
        ! Save the contents of the ML error location
        ! register so we can compare it to the new
        ! contents of this register after the retry.
        ! This is done to classify the error.
        OLDSEC = .MLEL;
        OLDCHN = .CHAN;
        ! Do a classify retry call. If the same error
        ! occurs then classify it as a hard error. If
        ! a different error occurred or the error went away
        ! then classify it as a soft error.
        if RETRY (ONE, .COMMAND, .LUN, 256, .PTR, .OLDSEC) eql 5
        then
          ! The same error occurred so see if it is at the same
          ! sector and channel number, if so then classify
          ! it as a hard error else classiy it as a soft
          ! error.
          if ((.MLEL eql .OLDSEC) and (.CHAN eql .OLDCHN))
          then
            begin !Same error occurred 'hard'
  
```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (44)

```
12759 :MLX4
12760 : TESTING RANDOM DATA
12761 :
12762 : 7057
12763 : 7058
12764 : 7059
12765 : P 7060
12766 : P 7061
12767 : 7062
12768 : 7063
12769 : 7064
12770 : 7065
12771 : 7066
12772 : 7067
12773 : 7068
12774 : 7069
12775 : 7070
12776 : 7071
12777 : 7072
12778 : P 7073
12779 : P 7074
12780 : 7075
12781 : 7076
12782 : 7077
12783 : 7078
12784 : 7079
12785 : 7080
12786 : 7081
12787 : 7082
12788 : 7083
12789 : 7084
12790 : 7085
12791 : 7086
12792 : P 7087
12793 : P 7088
12794 : 7089
12795 : 7090
12796 : 7091
12797 : 7092
12798 : 7093
12799 : 7094
12800 : 7095
12801 : 7096
12802 : 7097
12803 : 7098
12804 : 7099
12805 : 7100
12806 : 7101
12807 : 7102
12808 : 7103
12809 : 7104
12810 : 7105

if .ERROUT !Print error if enabled
then
begin
ERRHRD (5112, !Error number
MSG4, !Error message
0); !Additional message routine
end;

UP_HARD_COUNT (.LUN, .BOARD);
end
else
begin !Not the same error 'soft'
if .ERROUT !Print error if enabled
then
begin
ERRSOFT (5113, !Error number
MSG3, !Error message
0); !Additional message routine
end;

UP_SOFT_COUNT (.LUN, .BOARD);
end
else !Not the same error 'soft'
begin
if .ERROUT !Print error if enabled
then
begin
ERRSOFT (5114, !Error number
MSG3, !Error message
0); !Additional message routine
end;

UP_SOFT_COUNT (.LUN, .BOARD);
end;
end;

end;
end;
end;
return;
end;

!* 4 * END OF LOOP THAT COMPLETELY TESTS 1 UNIT
!* 3 * END OF LOGICAL UNIT SELECTION LOOP
!* 2 * END OF REPEAT LOOP FOR THIS ROUTINE
!* 1 * END OF ROUTINE
```

```
12815      :MLX4
12816      :
12817
12818
12819
12823 070540 004167 114570      RAND1: .SBTTL RAND1 TESTING RANDOM DATA
12824 070544 162706 000020      JSR R1,$SAVE5
12825 070550 012746 007664      SUB #20,SP
12826 070554 012746 007072      MOV #RTN5A,-(SP)
12827 070560 012746 000002      MOV #SAY1,-(SP)
12828 070564 010600      MOV #2,-(SP)
12829 070566 104414      MOV SP,R0
12830 070570 005066 000024      TRAP 14
12831 070574 000167 002276      CLR 24(SP)
12832 070600 004767 151014      JMP 42$
12833 070604 016766 111202      JSR PC,GEN5
12834 070612 005066 000006      MOV L$UNIT,10(SP)
12835 070616 000167 002240      CLR 6(SP)
12836 070622 016600 000006      JMP 41$
12837 070626 006200      MOV 6(SP),R0
12838 070630 006200      ASR R0
12839 070632 006200      ASR R0
12840 070634 062700 034442      ASR R0
12841 070640 010046      ADD #DRIVE.STATUS,R0
12842 070642 016646 000010      MOV R0,-(SP)
12843 070646 042716 177770      MOV 10(SP),-(SP)
12844 070652 012746 000001      BIC #177770,(SP)
12845 070656 005046      MOV #1,-(SP)
12846 070660 004767 113472      CLR -(SP)
12847 070664 062706 000010      JSR PC,BL$GT2
12848 070670 005300      ADD #10,SP
12849 070672 001402      DEC R0
12850 070674 000167 001502      BEQ 4$
12851 070700 016667 000006      JMP 29$
12852 070706 012767 012670      MOV 6(SP),L$LUN
12853 070714 012767 022670      MOV #L$BUFF,W$PTR
12854 070722 016605 000006      MOV #R$BUFF,R$PTR
12855 070726 006305      MOV 6(SP),R5
12856 070730 016503 034460      ASL R5
12857 070734 020365 034500      MOV LOW.SECT(R5),R3
12858 070740 101355      CMP R3,TOP.SECT(R5)
12859 070742 010346      BHI 3$
12860 070744 016546 034500      MOV R3,-(SP)
12861 070750 004767 156222      MOV TOP.SECT(R5),-(SP)
12862 070754 010002      JSR PC,GET.WRDCNT
12863 070756 010216      MOV R0,R2
12864 070760 004767 156146      MOV R2,(SP)
12865 070764 016616 000012      JSR PC,SET.PTRS
12866 070770 010246      MOV 12(SP),(SP)
12867 070772 016746 141672      MOV R2,-(SP)
12868 070776 010346      MOV W$PTR,-(SP)
12869 071000 004767 154420      MOV R3,-(SP)
12870      :MLX4
12871      :
12872
12873 071004 010004      JSR PC,WRITE
12874 071006 020427 000001      MOV R0,R4
12874 071006 020427 000001      CMP R4,#1
```

27-Mar-1982 19:24:42 TOPS
27-Mar-1982 19:23:44 PA:<

6746

6809

6812

6814

6816

6821

6824

6825

6826

6827

6829

6831

6832

6833

27-Mar-1982 19:24:42 TOPS
27-Mar-1982 19:23:44 PA:<

6839

12875	071012	001035		BNE	6\$				
12876	071014	012746	000006	MOV	#6,-(SP)	:		6845	
12877	071020	012746	045424	MOV	#WRITE,-(SP)	:			
12878	071024	016646	000024	MOV	24(SP),-(SP)	:	LUN,*		
12879	071030	010246		MOV	R2,-(SP)	:	WRDCNT,*		
12880	071032	016746	141632	MOV	WPTR,-(SP)	:			
12881	071036	010346		MOV	R3,-(SP)	:	SECTOR,*		
12882	071040	004767	155160	JSR	PC,RETRY	:			
12883	071044	062706	000014	ADD	#14,SP	:			
12884	071050	005700		TST	R0	:			
12885	071052	001456		BEQ	9\$:			
12886	071054	016601	000020	MOV	20(SP),R1	:	LUN,*	6848	
12887	071060	112761	000004	MOVB	#4,WHY.DROPT(R1)	:		6849	
12888	071066	104455		TRAP	55	:			
12889	071070	011755		.WORD	11755	:			
12890	071072	011070		.WORD	MSG1	:			
12891	071074	000000		.WORD	0	:			
12892	071076	016600	000020	MOV	20(SP),R0	:	LUN,*	6850	
12893	071102	104451		TRAP	51	:			
12894	071104	000436		BR	8\$:		6851	
12895	071106	020427	000002	6\$: CMP	R4,#2	:	VALUE,*	6839	
12896	071112	001015		BNE	7\$:			
12897	071114	016601	000020	MOV	20(SP),R1	:	LUN,*	6858	
12898	071120	112761	000005	MOVB	#5,WHY.DROPT(R1)	:		6859	
12899	071126	104455		TRAP	55	:			
12900	071130	011756		.WORD	11756	:			
12901	071132	011070		.WORD	MSG1	:			
12902	071134	000000		.WORD	0	:			
12903	071136	016600	000020	MOV	20(SP),R0	:	LUN,*	6860	
12904	071142	104451		TRAP	51	:			
12905	071144	000416		BR	8\$:		6861	
12906	071146	020427	000003	7\$: CMP	R4,#3	:	VALUE,*	6839	
12907	071152	001016		BNE	9\$:			
12908	071154	104455		TRAP	55	:		6866	
12909	071156	011757		.WORD	11757	:			
12910	071160	011070		.WORD	MSG1	:			
12911	071162	000000		.WORD	0	:			
12912	071164	016601	000020	MOV	20(SP),R1	:	LUN,*	6867	
12913	071170	112761	000006	MOVB	#6,WHY.DROPT(R1)	:		6868	
12914	071176	010100		MOV	R1,R0	:	LUN,*		
12915	071200	104451		TRAP	51	:		6869	
12916	071202	062706	000012	8\$: ADD	#12,SP	:			
12917	071206	000543		BR	14\$:			
12918	071210	004767	154752	9\$: JSR	PC,CHOOSE	:		6873	
12919	071214	010066	000034	MOV	R0,34(SP)	:	*,COMMAND		
12920	071220	005001		CLR	R1	:		6875	
12921	071222	020027	045612	CMP	R0,#READ	:	COMMAND,*		
12922	071226	001015		BNE	10\$:			
12923	071230	005201		INC	R1	:			
12924	071232	016766	141434	000032	MOV	RPTR,32(SP)	:	*,PTR	6878
12925				:MLX4				27-Mar-1982 19:24:42	TOPS
12926				:	TESTING RANDOM DATA			27-Mar-1982 19:23:44	PA:<
12927									
12928	071240	016646	000020	MOV	20(SP),-(SP)	:	LUN,*	6879	
12929	071244	010246		MOV	R2,-(SP)	:	WRDCNT,*		
12930	071246	016746	141420	MOV	RPTR,-(SP)	:			
12931	071252	010346		MOV	R3,-(SP)	:	SECTOR,*		

12932	071254	004767	154332		JSR	PC,READ			
12933	071260	000413			BR	11\$			
12934	071262	016766	141402	000032	10\$:	MOV	WPT?,32(SP)	: *,PTR	6883
12935	071270	016646	000020			MOV	20(SP),-(SP)	: LUN,*	6884
12936	071274	010246				MOV	R2, -(SP)	: WRDCNT,*	
12937	071276	016746	141366			MOV	WPTR, -(SP)		
12938	071302	010346				MOV	R3, -(SP)	: SECTOR,*	
12939	071304	004767	154470			JSR	PC,CHECK		
12940	071310	010004			11\$:	MOV	R0,R4	: *,VALUE	
12941	071312	001107				BNE	15\$:	6891
12942	071314	006001				ROR	R1	:	6896
12943	071316	103402				BLO	13\$:	
12944	071320	000167	001014		12\$:	JMP	28\$:	
12945	071324	016746	141340		13\$:	MOV	WPTR, -(SP)	:	6900
12946	071330	016746	141336			MOV	RPTR, -(SP)	:	
12947	071334	010246				MOV	R2, -(SP)	: WRDCNT,*	
12948	071336	004767	152764			JSR	PC,DOUBLE.CHECK		
12949	071342	062706	000006			ADD	#6,SP		
12950	071346	010066	000036			MOV	R0,36(SP)	: *,DBL.VALUE	
12951	071352	001762				BEQ	12\$		
12952	071354	016646	000030			MOV	30(SP),-(SP)	: LUN,*	6903
12953	071360	004767	144140			JSR	PC,SAYWHO		
12954	071364	012716	011216			MOV	#MSG5,(SP)	:	6904
12955	071370	012746	007072			MOV	#SAY1, -(SP)		
12956	071374	012746	000002			MOV	#2, -(SP)		
12957	071400	010600				MOV	SP,R0	: SP,*	
12958	071402	104414				TRAP	14		
12959	071404	016616	000044			MOV	44(SP), (SP)	: DBL.VALUE,*	6905
12960	071410	017646	000044			MOV	@44(SP), -(SP)	: DBL.VALUE,*	
12961	071414	012746	006710			MOV	#FMT12A, -(SP)		
12962	071420	012746	000003			MOV	#3, -(SP)		
12963	071424	010600				MOV	SP,R0	: SP,*	
12964	071426	104414				TRAP	14		
12965	071430	062766	010000	000052		ADD	#10000,52(SP)	: *,DBL.VALUE	6907
12966	071436	016616	000052			MOV	52(SP), (SP)	: DBL.VALUE,*	6908
12967	071442	017646	000052			MOV	@52(SP), -(SP)	: DBL.VALUE,*	
12968	071446	012746	006756			MOV	#FMT12B, -(SP)		
12969	071452	012746	000003			MOV	#3, -(SP)		
12970	071456	010600				MOV	SP,R0	: SP,*	
12971	071460	104414				TRAP	14		
12972	071462	016601	000052			MOV	52(SP),R1	: LUN,*	6910
12973	071466	112761	000010	034446		MOVB	#10,WHY.DROPT(R1)		
12974	071474	104455				TRAP	55	:	6911
12975	071476	011760				.WORD	11760		
12976	071500	011070				.WORD	MSG1		
12977	071502	000000				.WORD	0		
12978	071504	016600	000052			MOV	52(SP),R0	: LUN,*	6912
12979	071510	104451				TRAP	51		
12980					:MLX4				
12981					:				
12982						TESTING RANDOM DATA			
12983	071512	062706	000044			ADD	#44,SP	:	6913
12984	071516	000521			14\$:	BR	20\$		
12985	071520	020427	000001		15\$:	CMP	R4,#1	: VALUE,*	6891
12986	071524	001035				BNE	16\$:	
12987	071526	012746	000006			MOV	#6, -(SP)	:	6921
12988	071532	016646	000046			MOV	46(SP), -(SP)	: COMMAND,*	

27-Mar-1982 19:24:42 TOPS
 27-Mar-1982 19:23:44 PA:<

12989	071536	016646	000034		MOV	34(SP),-(SP)	:	LUN,*	
12990	071542	010246			MOV	R2,-(SP)	:	WRDCNT,*	
12991	071544	016646	000052		MOV	52(SP),-(SP)	:	PTR,*	
12992	071550	010346			MOV	R3,-(SP)	:	SECTOR,*	
12993	071552	004767	154446		JSR	PC,RETRY	:		
12994	071556	062706	000014		ADD	#14,SP	:		
12995	071562	005700			TST	R0	:		
12996	071564	001655			BEQ	12\$:		
12997	071566	016601	000030		MOV	30(SP),R1	:	LUN,*	6924
12998	071572	112761	000004	034446	MOVB	#4,WHY.DROPT(R1)	:		6925
12999	071600	104455			TRAP	55	:		
13000	071602	011761			.WORD	11761	:		
13001	071604	011070			.WORD	MSG1	:		
13002	071606	000000			.WORD	0	:		
13003	071610	016600	000030		MOV	30(SP),R0	:	LUN,*	6926
13004	071614	104451			TRAP	51	:		
13005	071616	000457			BR	19\$:		6927
13006	071620	020427	000002	16\$:	CMP	R4,#2	:	VALUE,*	6891
13007	071624	001015			BNE	17\$:		
13008	071626	016601	000030		MOV	30(SP),R1	:	LUN,*	6934
13009	071632	112761	000005	034446	MOVB	#5,WHY.DROPT(R1)	:		6935
13010	071640	104455			TRAP	55	:		
13011	071642	011762			.WORD	11762	:		
13012	071644	011070			.WORD	MSG1	:		
13013	071646	000000			.WORD	0	:		
13014	071650	016600	000030		MOV	30(SP),R0	:	LUN,*	6936
13015	071654	104451			TRAP	51	:		
13016	071656	000437			BR	19\$:		6937
13017	071660	020427	000003	17\$:	CMP	R4,#3	:	VALUE,*	6891
13018	071664	001014			BNE	18\$:		
13019	071666	104455			TRAP	55	:		6942
13020	071670	011763			.WORD	11763	:		
13021	071672	011070			.WORD	MSG1	:		
13022	071674	000000			.WORD	0	:		
13023	071676	016601	000030		MOV	30(SP),R1	:	LUN,*	6943
13024	071702	112761	000006	034446	MOVB	#6,WHY.DROPT(R1)	:		6944
13025	071710	010100			MOV	R1,R0	:	LUN,*	
13026	071712	104451			TRAP	51	:		6945
13027	071714	000420			BR	19\$:		6891
13028	071716	020427	000004	18\$:	CMP	R4,#4	:	VALUE,*	
13029	071722	001021			BNE	21\$:		
13030	071724	004767	145402		JSR	PC,ISOLATE	:		6950
13031	071730	104455			TRAP	55	:		6951
13032	071732	011764			.WORD	11764	:		
13033	071734	011120			.WORD	MSG2	:		
13034	071736	000000			.WORD	0	:		
13035									
13036									
13037									
13038	071740	016601	000030		MOV	30(SP),R1	:	LUN,*	6952
13039	071744	112761	000007	034446	MOVB	#7,WHY.DROPT(R1)	:		6953
13040	071752	010100			MOV	R1,R0	:	LUN,*	
13041	071754	104451			TRAP	51	:		6954
13042	071756	062706	000022	19\$:	ADD	#22,SP	:		
13043	071762	000167	001070	20\$:	JMP	40\$:		
13044	071766	020427	000005	21\$:	CMP	R4,#5	:	VALUE,*	6891
13045	071772	001162			BNE	28\$:		

27-Mar-1982 19:24:42 TOPS
 27-Mar-1982 19:23:44 PA:<

13046	071774	004767	145332		JSR	PC,ISOLATE	:		6959
13047	072000	032767	000001	110252	BIT	#1,ERROUT	:		6961
13048	072006	001423			BEQ	22\$:		
13049	072010	017701	142374		MOV	@ML.REG+42,R1	:		
13050	072014	006201			ASR	R1	:		
13051	072016	006201			ASR	R1	:		
13052	072020	006201			ASR	R1	:		
13053	072022	006201			ASR	R1	:		
13054	072024	006201			ASR	R1	:		
13055	072026	006201			ASR	R1	:		
13056	072030	042701	177700		BIC	#177700,R1	:		
13057	072034	010146			MOV	R1,-(SP)	:		
13058	072036	012746	006640		MOV	#FMT10B,-(SP)	:		
13059	072042	012746	000002		MOV	#2,-(SP)	:		
13060	072046	010600			MOV	SP,R0	:	SP,*	
13061	072050	104414			TRAP	14	:		
13062	072052	062706	000006		ADD	#6,SP	:		
13063	072056	017766	142330	000040 22\$:	MOV	@ML.REG+44,40(SP)	:	*,OLDSEC	6964
13064	072064	017701	142320		MOV	@ML.REG+42,R1	:		6965
13065	072070	006201			ASR	R1	:		
13066	072072	006201			ASR	R1	:		
13067	072074	006201			ASR	R1	:		
13068	072076	006201			ASR	R1	:		
13069	072100	006201			ASR	R1	:		
13070	072102	006201			ASR	R1	:		
13071	072104	042701	177700		BIC	#177700,R1	:		
13072	072110	010166	000034		MOV	R1,34(SP)	:	*,OLDCHN	
13073	072114	012746	000001		MOV	#1,-(SP)	:		6967
13074	072120	016646	000046		MOV	46(SP),-(SP)	:	COMMAND,*	
13075	072124	016646	000034		MOV	34(SP),-(SP)	:	LUN,*	
13076	072130	010246			MOV	R2,-(SP)	:	WRDCNT,*	
13077	072132	016646	000052		MOV	52(SP),-(SP)	:	PTR,*	
13078	072136	010346			MOV	R3,-(SP)	:	SECTOR,*	
13079	072140	004767	154060		JSR	PC,RETRY	:		
13080	072144	062706	000014		ADD	#14,SP	:		
13081	072150	020027	000005		CMP	R0,#5	:		
13082	072154	001052			BNE	25\$:		
13083	072156	027766	142230	000040	CMP	@ML.REG+44,40(SP)	:	*,OLDSEC	6970
13084	072164	001035			BNE	24\$:		
13085	072166	016600	000034		MOV	34(SP),R0	:	OLDCHN,*	
13086	072172	017701	142212		MOV	@ML.REG+42,R1	:		
13087	072176	006201			ASR	R1	:		
13088	072200	006201			ASR	R1	:		
13089	072202	006201			ASR	R1	:		
13090									
13091					:MLX4				
13092					:	TESTING RANDOM DATA			
13093	072204	006201			ASR	R1	:		
13094	072206	006201			ASR	R1	:		
13095	072210	006201			ASR	R1	:		
13096	072212	042701	177700		BIC	#177700,R1	:		
13097	072216	020100			CMP	R1,R0	:		
13098	072220	001017			BNE	24\$:		
13099	072222	032767	000001	110030	BIT	#1,ERROUT	:		6974
13100	072230	001404			BEQ	23\$:		
13101	072232	104456			TRAP	56	:		
13102	072234	011765			.WORD	11765	:		

13103	072236	011202			.WORD	MSG4			
13104	072240	000000			.WORD	0			
13105	072242	016646	000030	23\$:	MOV	30(SP),-(SP)	:	LUN,*	6977
13106	072246	016746	142070		MOV	BOARD,-(SP)			
13107	072252	004767	145144		JSR	PC,UP.HARD.COUNT			
13108	072256	000427			BR	27\$:		6970
13109	072260	032767	000001	107772	24\$:	BIT	#1,ERROUT	:	6982
13110	072266	001415			BEQ	26\$			
13111	072270	104457			TRAP	57			
13112	072272	011766			.WORD	11766			
13113	072274	011166			.WORD	MSG3			
13114	072276	000000			.WORD	0			
13115	072300	000410			BR	26\$:		6985
13116	072302	032767	000001	107750	25\$:	BIT	#1,ERROUT	:	6971
13117	072310	001404			BEQ	26\$			
13118	072312	104457			TRAP	57			
13119	072314	011767			.WORD	11767			
13120	072316	011166			.WORD	MSG3			
13121	072320	C00000			.WORD	0			
13122	072322	016646	000030	26\$:	MOV	30(SP),-(SP)	:	LUN,*	6994
13123	072326	016746	142010		MOV	BOARD,-(SP)			
13124	072332	004767	145546		JSR	PC,UP.SOFT.COUNT			
13125	072336	022626		27\$:	CMP	(SP)+,(SP)+	:		6958
13126	072340	010200		28\$:	MOV	R2,R0	:	WRDCNT,*	7000
13127	072342	006300			ASL	R0			
13128	072344	066700	140320		ADD	WPTR,R0			
13129	072350	010067	140314		MOV	R0,WPTR			
13130	072354	010216			MOV	R2,(SP)	:	WRDCNT,*	7001
13131	072356	012746	000400		MOV	#400,-(SP)			
13132	072362	004767	112602		JSR	PC,BL\$DIV			
13133	072366	060300			ADD	R3,R0	:	SECTOR,*	
13134	072370	010003			MOV	R0,R3	:	*,SECTOR	
13135	072372	062706	000024		ADD	#24,SP	:		6830
13136	072376	000167	176332		JMP	5\$:		6829
13137	072402	032767	000001	107652	29\$:	BIT	#1,EFNS21	:	7012
13138	072410	001002			BNE	30\$			
13139	072412	000167	000440		JMP	40\$			
13140	072416	016600	000006	30\$:	MOV	6(SP),R0	:	LUN,*	7019
13141	072422	006300			ASL	R0			
13142	072424	016001	034500		MOV	TOP.SECT(R0),R1			
13143	072430	016005	034460		MOV	LOW.SECT(R0),R5	:	*,SECTOR	
13144	072434	000167	000406		JMP	39\$			
13145					:MLX4				
13146					:	TESTING RANDOM DATA			
13147									
13148	072440	016646	000006	31\$:	MOV	6(SP),-(SP)	:	LUN,*	7021
13149	072444	012746	000400		MOV	#400,-(SP)			
13150	072450	012746	022670		MOV	#RBUF,-(SP)			
13151	072454	010546			MOV	R5,-(SP)	:	SECTOR,*	
13152	072456	004767	153130		JSR	PC,READ			
13153	072462	062706	000010		ADD	#10,SP			
13154	072466	020027	000005		CMP	R0,#5			
13155	072472	001164			BNE	38\$			
13156	072474	004767	144632		JSR	PC,ISOLATE	:		7024
13157	072500	032767	000001	107552	BIT	#1,ERROUT	:		7026
13158	072506	001423			BEQ	32\$			
13159	072510	017700	141674		MOV	@ML.REG+42,R0			

27-Mar-1982 19:24:42 TOPS
27-Mar-1982 19:23:44 PA:<

13160	072514	006200			ASR	R0			
13161	072516	006200			ASR	R0			
13162	072520	006200			ASR	R0			
13163	072522	006200			ASR	R0			
13164	072524	006200			ASR	R0			
13165	072526	006200			ASR	R0			
13166	072530	042700	177700		BIC	#177700,R0			
13167	072534	010046			MOV	R0,-(SP)			
13168	072536	012746	006640		MOV	#FMT10B,-(SP)			
13169	072542	012746	000002		MOV	#2,-(SP)			
13170	072546	010600			MOV	SP,R0		; SP,*	
13171	072550	104414			TRAP	14			
13172	072552	062706	000006		ADD	#6,SP			
13173	072556	017766	141630	000016 32\$:	MOV	@ML.REG+44,16(SP)		; *,OLDSEC	7035
13174	072564	017700	141620		MOV	@ML.REG+42,R0		:	7036
13175	072570	006200			ASR	R0			
13176	072572	006200			ASR	R0			
13177	072574	006200			ASR	R0			
13178	072576	006200			ASR	R0			
13179	072600	006200			ASR	R0			
13180	072602	006200			ASR	R0			
13181	072604	042700	177700		BIC	#177700,R0			
13182	072610	010066	000012		MOV	R0,12(SP)		; *,OLDCHN	
13183	072614	012746	000001		MOV	#1,-(SP)		:	7044
13184	072620	016646	000024		MOV	24(SP),-(SP)		; COMMAND,*	
13185	072624	016646	000012		MOV	12(SP),-(SP)		; LUN,*	
13186	072630	012746	000400		MOV	#400,-(SP)			
13187	072634	016646	000030		MOV	30(SP),-(SP)		; PTR,*	
13188	072640	016646	000030		MOV	30(SP),-(SP)		; OLDSEC,*	
13189	072644	004767	153354		JSR	PC,RETRY			
13190	072650	062706	000014		ADD	#14,SP			
13191	072654	020027	000005		CMP	R0,#5			
13192	072660	001052			BNE	35\$			
13193	072662	027766	141524	000016	CMP	@ML.REG+44,16(SP)		; *,OLDSEC	7053
13194	072670	001035			BNE	34\$			
13195	072672	016646	000012		MOV	12(SP),-(SP)		; OLDCHN,*	
13196	072676	017700	141506		MOV	@ML.REG+42,R0			
13197	072702	006200			ASR	R0			
13198	072704	006200			ASR	R0			
13199	072706	006200			ASR	R0			
13200									
13201				:MLX4					
13202				:		TESTING RANDOM DATA			
13203	072710	006200			ASR	R0			
13204	072712	006200			ASR	R0			
13205	072714	006200			ASR	R0			
13206	072716	042700	177700		BIC	#177700,R0			
13207	072722	020026			CMP	R0,(SP)+			
13208	072724	001017			BNE	34\$			
13209	072726	032767	000001	107324	BIT	#1,ERROUT		:	7057
13210	072734	001404			BEQ	33\$			
13211	072736	104456			TRAP	56		:	7062
13212	072740	011770			.WORD	11770			
13213	072742	011202			.WORD	MSG4			
13214	072744	000000			.WORD	0			
13215	072746	016646	000006	33\$:	MOV	6(SP),-(SP)		; LUN,*	7065
13216	072752	016746	141364		MOV	BOARD,-(SP)			

27-Mar-1982 19:24:42 TOPS
 27-Mar-1982 19:23:44 PA:<

13217	072756	004767	144440		JSR	PC,UP,HARD,COUNT		
13218	072762	000427			BR	37\$:	7053
13219	072764	032767	000001	107266	34\$:	BIT	#1,ERROUT	7070
13220	072772	001415			BEQ	36\$:	
13221	072774	104457			TRAP	57	:	7075
13222	072776	011771			.WORD	11771		
13223	073000	011166			.WORD	MSG3		
13224	073002	000000			.WORD	0		
13225	073004	000410			BR	36\$:	7078
13226	073006	032767	000001	107244	35\$:	BIT	#1,ERROUT	7084
13227	073014	001404			BEQ	36\$:	
13228	073016	104457			TRAP	57	:	7089
13229	073020	011772			.WORD	11772		
13230	073022	011166			.WORD	MSG3		
13231	073024	000000			.WORD	0		
13232	073026	016646	000006		36\$:	MOV	6(SP),-(SP)	: LUN,*
13233	073032	016746	141304			MOV	BOARD,-(SP)	
13234	073036	004767	145042			JSR	PC,UP,SOFT,COUNT	
13235	073042	022626			37\$:	CMP	(SP)+,(SP)+	: 7023
13236	073044	005205			38\$:	INC	R5	: SECTOR
13237	073046	020501			39\$:	CMP	R5,R1	: SECTOR,*
13238	073050	101002				BHI	40\$	
13239	073052	000167	177362			JMP	31\$	
13240	073056	005266	000006		40\$:	INC	6(SP)	: LUN
13241	073062	026666	000006	000010	41\$:	CMP	6(SP),10(SP)	: LUN,*
13242	073070	002002				BGE	42\$	
13243	073072	000167	175524			JMP	2\$	
13244	073076	005266	000024		42\$:	INC	24(SP)	: COUNT
13245	073102	026666	000024	000044		CMP	24(SP),44(SP)	: COUNT,REPEAT
13246	073110	003002				BGT	43\$	
13247	073112	000167	175462			JMP	1\$	
13248	073116	062706	000026		43\$:	ADD	#26,SP	: 6746
13249	073122	000207				RTS	PC	
13250								
13251								
13252								
13260								
13261								

: Routine Size: 634 words
 : Maximum stack depth per invocation: 35 words

```

13263 :MLX4
13264 :
13265 :
13266 : 7106 %sbttl 'TESTING RANDOM DATA & WORD COUNTS'
13267 : 7107 routine RAND2 (REPEAT) : novalue =
13268 : 7108 begin
13269 : 7109
13270 : 7110 !* 1 * START OF ROUTINE
13271 : 7111 ++
13272 : 7112 ROUTINE: RAND2(REPEAT)
13273 : 7113 PURPOSE: TO TEST ALL UNITS IN A SEQUENTIAL FASHION, BUT
13274 : 7114 USING RANDOM WORD COUNTS
13275 : 7115
13276 : 7116 ARGUMENT: REPEAT = NUMBER OF TIMES TO EXECUTE THIS ROUTINE
13277 : 7117 BEFORE RETURNING TO THE CALLER (OPT5).
13278 : 7118
13279 : 7119 NOTE: SINCE ONLY THE WORD COUNTS AND THE DATA ARE RANDOM,
13280 : 7120 THE OTHER TRANSFER VALUES MUST BE SET UP BY THIS
13281 : 7121 ROUTINE. THESE VALUES INCLUDE:
13282 : 7122 THE LUN, SECTOR AND BUFFER POINTERS.
13283 : 7123
13284 : 7124 THE CODE FOR 'RAND2' IN BRIEF:
13285 : 7125
13286 : 7126 BEGIN 1 (START OF ROUTINE)
13287 : 7127 SAY ROUTINE IS RUNNING
13288 : 7128 INCR COUNT FROM 1 TO REPEAT
13289 : 7129 : BEGIN 2 (START OF REPEAT LOOP FOR THE ROUTINE)
13290 : 7130 : INCR LUN FROM 0 TO LAST
13291 : 7131 : : BEGIN 3 (START OF LOGICAL UNIT SELECTION LOOP)
13292 : 7132 : : GENERATE THE RANDOM PATTERN
13293 : 7133 : : TESTLOOP:
13294 : 7134 : : : BEGIN 4 (START OF LOOP THAT COMPLETELY TESTS 1 UNIT)
13295 : 7135 : : : IF UNIT IS ACTIVE
13296 : 7136 : : : THEN
13297 : 7137 : : : : BEGIN 5 (START OF TEST FOR AN ACTIVE UNIT)
13298 : 7138 : : : : SECTOR = LOWEST
13299 : 7139 : : : : INITIALIZE THE WRITE AND READ BUFFER POINTERS
13300 : 7140 : : : : WHILE SECTOR LEQ HIGHEST DO
13301 : 7141 : : : : : BEGIN 6 (START OF A PASS THROUGH ALL SECTORS)
13302 : 7142 : : : : : CHOOSE A RANDOM WORD COUNT
13303 : 7143 : : : : : SET UP BUFFER POINTERS BEFORE TRANSFER
13304 : 7144 : : : : : CALCULATE NEXT STARTING SECTOR (BASED ON WORD COUNT)
13305 : 7145 : : : : : IF NEXT STARTING SECTOR GTR HIGHEST
13306 : 7146 : : : : : THEN ADJUST THE WORD COUNT & NEXT SECTOR SO THEY FIT
13307 : 7147 : : : : : : WITHIN THE TESTABLE SECTOR LIMITS
13308 : 7148 : : : : : WRITE
13309 : 7149 : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
13310 : 7150 : : : : : CHOOSE WHETHER TO WRITE CHECK OR READ
13311 : 7151 : : : : : DO THE WRITE CHECK OR READ
13312 : 7152 : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE TESTLOOP)
13313 : 7153 : : : : : SECTOR = THE CALCULATED NEXT STARTING SECTOR
13314 : 7154 : : : : : CHANGE BUFFER POINTERS AFTER TRANSFER
13315 : 7155 : : : : : END 6 (END OF A PASS THROUGH ALL SECTORS)
13316 : 7156 : : : : : END 5 (END OF TEST FOR AN ACTIVE UNIT)
13317 : 7157 : : : : : END 4 (END TESTLOOP)
  
```

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (45)

```

13319 :MLX4
13320 :
13321 :
13322 : 7158 : : END 3 (END OF LOGICAL UNIT SELECTION LOOP)
13323 : 7159 : : END 2 (END OF REPEAT LOOP FOR THIS ROUTINE)
13324 : 7160 : RETURN
13325 : 7161 : END 1 (END OF ROUTINE)
13326 : 7162 : --
13327 : 7163 :
13328 : 7164 : local
13329 : 7165 : WRDCNT,
13330 : 7166 : SECTOR,
13331 : 7167 : NEXT_SECT,
13332 : 7168 : VALUE,
13333 : 7169 : OLDSECT,
13334 : 7170 : OLDCHN,
13335 : 7171 : PTR,
13336 : 7172 : COMMAND,
13337 : 7173 : DBL_VALUE;
13338 : 7174 :
13339 : 7175 : label
13340 : 7176 : LOOP;
13341 : 7177 :
13342 : 7178 : PRINTB (SAY1, RTN5B);
13343 : 7179 : !'RAND2'
13344 : 7180 :
13345 : 7181 : incr COUNT from 1 to .REPEAT do
13346 : 7182 : begin
13347 : 7183 :
13348 : 7184 :     incr LUN from 0 to (.LSUNIT - 1) do
13349 : 7185 :     begin
13350 : 7186 :     GENS ();
13351 : 7187 : LOOP :
13352 : 7188 :     begin
13353 : 7189 :
13354 : 7190 :     if .DRIVE_STATUS [.LUN] eql ACTIVE
13355 : 7191 :     then
13356 : 7192 :     begin
13357 : 7193 :     L$LUN = .LUN;
13358 : 7194 :     SECTOR = LOWEST;
13359 : 7195 :     WPTR = WBUFF;
13360 : 7196 :     RPTR = RBUFF;
13361 : 7197 :
13362 : 7198 :     while .SECTOR leqa HIGHEST do
13363 : 7199 :     begin
13364 : 7200 :     WRDCNT = RNDWC ();
13365 : 7201 :     SET PTRS (.WRDCNT);
13366 : 7202 :     NEXT_SECT = .SECTOR + (.WRDCNT/256);
13367 : 7203 :
13368 : 7204 :     if .SECTOR eql .NEXT_SECT then NEXT_SECT = .NEXT_SECT + 1;
13369 : 7205 :
13370 : 7206 :     if .NEXT_SECT gtra HIGHEST
13371 : 7207 :     then
13372 : 7208 :     begin
13373 : 7209 :     WRDCNT = 256*(HIGHEST - .SECTOR + 1);
  
```

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (45)

!* 2 * START OF REPEAT LOOP FOR THIS ROUTINE

!* 3 * START OF LOGICAL UNIT SELECTION LOOP
 !RANDOM DATA PATTERN

!* 4 * START OF THE LOOP FOR COMPLETELY TESTING 1 UNIT

!* 5 * START OF TEST FOR AN ACTIVE UNIT

!* 6 * START OF A PASS THROUGH ALL SECTORS
 !EXPECT VALUE BETWEEN 1 AND BUFSIZ

13375 :MLX4
 13376 :
 13377 :
 13378 : 7210
 13379 : 7211
 13380 : 7212
 13381 : 7213
 13382 : 7214
 13383 : 7215
 13384 : 7216
 13385 : 7217
 13386 : 7218
 13387 : 7219
 13388 : 7220
 13389 : 7221
 13390 : 7222
 13391 : 7223
 13392 : 7224
 13393 : 7225
 13394 : 7226
 13395 : 7227
 13396 : 7228
 13397 : 7229
 13398 : 7230
 13399 : 7231
 13400 : 7232
 13401 : 7233
 13402 : 7234
 13403 : 7235
 13404 : 7236
 13405 : 7237
 13406 : 7238
 13407 : 7239
 13408 : 7240
 13409 : 7241
 13410 : 7242
 13411 : 7243
 13412 : 7244
 13413 : 7245
 13414 : 7246
 13415 : 7247
 13416 : 7248
 13417 : 7249
 13418 : 7250
 13419 : 7251
 13420 : 7252
 13421 : 7253
 13422 : 7254
 13423 : 7255
 13424 : 7256
 13425 : 7257
 13426 : 7258
 13427 : 7259
 13428 : 7260
 13429 : 7261

TESTING RANDOM DATA & WORD COUNTS

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (45)

```

NEXT_SECT = HIGHEST + 1;
end;

VALUE = write (.LUN, .WRDCNT, .WPTR, .SECTOR);

!+ SEE HOW SUCCESSFUL THE WRITE WAS:
!-

selectone .VALUE of          !SEE 'SYSFRR' FOR DEFINITION
  set                        !OF ERROR # CONTAINED IN 'VALUE'

  [1] :
    begin                    !* 6A * RETRY ALLOWED
      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;

      end;                    !* 6A *

  [2] :
    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] :
    begin                    !* 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 eqf read
then
  begin
    PTR = .RPTR;
    VALUE = read (.LUN, .WRDCNT, .RPTR, .SECTOR);
  end
else

```

13431 :MLX4
 13432 :
 13433 :
 13434 :
 13435 :
 13436 :
 13437 :
 13438 :
 13439 :
 13440 :
 13441 :
 13442 :
 13443 :
 13444 :
 13445 :
 13446 :
 13447 :
 13448 :
 13449 :
 13450 :
 13451 :
 13452 :
 13453 :
 13454 :
 13455 :
 13456 :
 13457 :
 13458 :
 13459 :
 13460 :
 13461 :
 13462 :
 13463 :
 13464 :
 13465 :
 13466 :
 13467 :
 13468 :
 13469 :
 13470 :
 13471 :
 13472 :
 13473 :
 13474 :
 13475 :
 13476 :
 13477 :
 13478 :
 13479 :
 13480 :
 13481 :
 13482 :
 13483 :
 13484 :
 13485 :

TESTING RANDOM DATA & WORD COUNTS

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (45)

```

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 (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

```


13487 :MLX4
 13488 :
 13489 :
 13490 :
 13491 :
 13492 :
 13493 :
 13494 :
 13495 :
 13496 :
 13497 :
 13498 :
 13499 :
 13500 :
 13501 :
 13502 :
 13503 :
 13504 :
 13505 :
 13506 :
 13507 :
 13508 :
 13509 :
 13510 :
 13511 :
 13512 :
 13513 :
 13514 :
 13515 :
 13516 :
 13517 :
 13518 :
 13519 :
 13520 :
 13521 :
 13522 :
 13523 :
 13524 :
 13525 :
 13526 :
 13527 :
 13528 :
 13529 :
 13530 :
 13531 :
 13532 :
 13533 :
 13534 :
 13535 :
 13536 :
 13537 :
 13538 :
 13539 :
 13540 :
 13541 :

TESTING RANDOM DATA & WORD COUNTS

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (45)

```

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;
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 (5209, MSG4, 0);

                                !**** OPTION 5, RAND2 ERROR 09 ****
            UP_HARD_COUNT (.LUN, .BOARD);
        end
    else
        begin
            if .ERROUT then ERRSOFT (5210, MSG3, 0);

                                !**** OPTION 5, RAND2 ERROR 10 ****
            UP_SOFT_COUNT (.LUN, .BOARD);
        end
    end
end

```

```

13543 :MLX4
13544 : TESTING RANDOM DATA & WORD COUNTS
13545 :
13546 : 7366 end
13547 : 7367
13548 : 7368 else
13549 : 7369 begin
13550 : 7370
13551 : 7371 if .ERROUT then ERRSOFT (5211, MSG3, 0);
13552 : 7372
13553 : 7373 !**** OPTION 5, RAND2 ERROR 11 ****
13554 : 7374 UP_SOFT_COUNT (.LUN, .BOARD);
13555 : 7375 end;
13556 : 7376
13557 : 7377 end; !* 6H *
13558 : 7378 tes;
13559 : 7379
13560 : 7380 SECTOR = .NEXT_SECT;
13561 : 7381 WPTR = .WPTR + 2;
13562 : 7382 end; !* 6 * END OF A PASS THROUGH ALL SECTORS
13563 : 7383
13564 : 7384 end; !* 5 * END OF TEST FOR AN ACTIVE UNIT
13565 : 7385
13566 : 7386 !+
13567 : 7387 ! Test to see if this ut's address space is
13568 : 7388 ! to be read for soft errors. This test is
13569 : 7389 ! is intended for DMT purposes.
13570 : 7390 !-
13571 : 7391
13572 : 7392 if .EFNS21 !Is the background pattern to be read
13573 : 7393 then
13574 : 7394 begin
13575 : 7395
13576 : 7396 !
13577 : 7397 ! version czmlbb changed incr to incru
13578 : 7398 !
13579 : 7399 incru SECTOR from LOWEST to HIGHEST do
13580 : 7400
13581 : 7401 if read (.LUN, 256, RBUFF, .SECTOR) eql 5
13582 : 7402 then
13583 : 7403 begin
13584 : 7404 ISOLATE (); !Find the failing bank and board no.
13585 : 7405
13586 : 7406 if .ERROUT then PRINTB (FMT10B, .CHAN);
13587 : 7407
13588 : 7408 ! Print where the error is
13589 : 7409 !
13590 : 7410 ! Save the contents of the ML error location
13591 : 7411 ! register so we can compare it to the new
13592 : 7412 ! contents of this register after the retry.
13593 : 7413 ! This is done to classify the error.
13594 : 7414 !
13595 : 7415 OLDSEC = .MLEL;
13596 : 7416 OLDCHN = .CHAN;
13597 : 7417 !
  
```

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (45)

13599 :MLX4
 13600 :
 13601 :
 13602 :
 13603 :
 13604 :
 13605 :
 13606 :
 13607 :
 13608 :
 13609 :
 13610 :
 13611 :
 13612 :
 13613 :
 13614 :
 13615 :
 13616 :
 13617 :
 13618 :
 13619 :
 13620 :
 13621 :
 13622 :
 13623 :
 13624 : P 7440
 13625 : P 7441
 13626 : 7442
 13627 : 7443
 13628 : 7444
 13629 : 7445
 13630 : 7446
 13631 : 7447
 13632 : 7448
 13633 : 7449
 13634 : 7450
 13635 : 7451
 13636 : 7452
 13637 : P 7453
 13638 : P 7454
 13639 : 7455
 13640 : 7456
 13641 : 7457
 13642 : 7458
 13643 : 7459
 13644 : 7460
 13645 : 7461
 13646 : 7462
 13647 : 7463
 13648 : 7464
 13649 : 7465
 13650 : 7466
 13651 : P 7467
 13652 : P 7468
 13653 : 7469

TESTING RANDOM DATA & WORD COUNTS

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (45)

```

: Do a classify retry call. If the same error
: occurs then classify it as a hard error. If
: a different error occurred or the error went away
: then classify it as a soft error.
:
if RETRY (ONE, .COMMAND, .LUN, 256, .PTR, .OLDSEC) eql 5
then
:
: The same error occurred so see if it is at the same
: sector and channel number, if so then classify
: it as a hard error else classiy it as a soft
: error.
:
  if ((.MLEL eql .OLDSEC) and (.CHAN eql .OLDCHN))
  then
    begin
      !Same error occurred 'hard'
      if .ERROUT !Print error if enabled
      then
        begin
          ERRHRD (5212, !Error number
            MSG4, !Error message
            0); !Additional message routine
        end;
      UP_HARD_COUNT (.LUN, .BOARD);
    end
  else
    begin
      !Not the same error 'soft'
      if .ERROUT !Print error if enabled
      then
        begin
          ERRSOFT (5213, !Error number
            MSG3, !Error message
            0); !Additional message routine
        end;
      UP_SOFT_COUNT (.LUN, .BOARD);
    end
  else
    begin
      !Not the same error 'soft'
      if .ERROUT !Print error if enabled
      then
        begin
          ERRSOFT (5214, !Error number
            MSG3, !Error message
            0); !Additional message routine
        end;
    end
  end

```

```

13655 :MLX4
13656 : TESTING RANDOM DATA & WORD COUNTS
13657 :
13658 : 7470 end:
13659 : 7471
13660 : 7472 UP_SOFT_COUNT (.LUN, .BOARD);
13661 : 7473 end:
13662 : 7474
13663 : 7475 end:
13664 : 7476
13665 : 7477 end:
13666 : 7478
13667 : 7479 end:
13668 : 7480 end:
13669 : 7481
13670 : 7482 end:
13671 : 7483
13672 : 7484 return:
13673 : 7485 end:
13677 :
13678 :
  
```

```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (45)

!* 4 * END OF LOOP THAT COMPLETELY TESTS 1 UNIT
!* 3 * END OF LOGICAL UNIT SELECTION LOOP

!* 2 * END OF REPEAT LOOP FOR THIS ROUTINE

!* 1 * END OF ROUTINE
  
```

```

13678 .SBTTL RAND2 TESTING RANDOM DATA & WORD COUNTS
13682 073124 004167 112204 RAND2: JSR R1,$SAVE5 ; 7107
13683 073130 162706 000022 SUB #22,SP ;
13684 073134 012746 007672 MOV #RTN5B,-(SP) ; 7178
13685 073140 012746 007072 MOV #SAY1,-(SP) ;
13686 073144 012746 000002 MOV #2,-(SP) ;
13687 073150 010600 MOV SP,R0 ; SP,*
13688 073152 104414 TRAP 14 ;
13689 073154 005066 000022 CLR 22(SP) ; COUNT 7181
13690 073160 000167 002232 JMP 44$ ;
13691 073164 016766 106622 000026 1$: MOV L$UNIT,26(SP) ; 7184
13692 073172 005001 CLR R1 ; LUN
13693 073174 000167 002204 JMP 43$ ;
13694 073200 004767 146414 2$: JSR PC,GEN5 ; 7186
13695 073204 010100 MOV R1,R0 ; LUN,* 7190
13696 073206 006200 ASR R0
13697 073210 006200 ASR R0
13698 073212 006200 ASR R0
13699 073214 062700 034442 ADD #DRIVE.STATUS,R0
13700 073220 010046 MOV R0,-(SP)
13701 073222 010146 MOV R1,-(SP) ; LUN,*
13702 073224 042716 177770 BIC #177770,(SP)
13703 073230 012746 000001 MOV #1,-(SP)
13704 073234 005046 CLR -(SP)
13705 073236 004767 111114 JSR PC,BL$GT2
13706 073242 062706 000010 ADD #10,SP
13707 073246 005300 DEC R0
13708 073250 001402 BEQ 4$
13709 073252 000167 001460 3$: JMP 31$
  
```

Address	OpCode	Operand1	Operand2	Operand3	Operand4	Label	Instruction	Comments	Address	Count	Address	Count
13711						:MLX4						
13712						:						
13713							TESTING RANDOM DATA & WORD COUNTS					
13714	073256	010167	106612			4\$:	MOV R1,LSLUN	: LUN,*				7193
13715	073262	010100					MOV R1,R0	: LUN,*				7194
13716	073264	006300					ASL R0					
13717	073266	016003	034460				MOV LOW.SECT(R0),R3	: *,SECTOR				
13718	073272	012767	012670	137370			MOV #WBUF,WPTR	:				7195
13719	073300	012767	022670	137364			MOV #RBUF,RPTR	:				7196
13720	073306	012766	034500	000024			MOV #TOP.SECT,24(SP)	:				7198
13721	073314	060066	000024				ADD R0,24(SP)					
13722	073320	020376	000024			5\$:	CMP R3,@24(SP)	: SECTOR,*				
13723	073324	101352					BHI 3\$					
13724	073326	004767	174652				JSR PC,RNDWC	:				7200
13725	073332	010002					MOV R0,R2	: *,WRDCNT				
13726	073334	010246					MOV R2,-(SP)	: WRDCNT,*				7201
13727	073336	004767	153570				JSR PC,SET.PTRS					
13728	073342	010216					MOV R2,(SP)	: WRDCNT,*				7202
13729	073344	012746	000400				MOV #400,-(SP)					
13730	073350	004767	111614				JSR PC,BLSDIV					
13731	073354	060300					ADD R3,R0	: SECTOR,*				
13732	073356	010066	000012				MOV R0,12(SP)	: *,NEXT.SECT				
13733	073362	020300					CMP R3,R0	: SECTOR,NEXT.SECT				7204
13734	073364	001002					BNE 6\$					
13735	073366	005266	000012				INC 12(SP)	: NEXT.SECT				
13736	073372	026676	000012	000030		6\$:	CMP 12(SP),@30(SP)	: NEXT.SECT,*				7206
13737	073400	101415					BLOS 7\$					
13738	073402	017600	000030				MOV @30(SP),R0	:				7209
13739	073406	160300					SUB R3,R0	: SECTOR,*				
13740	073410	000300					SWAB R0	:				7208
13741	073412	105000					CLRB R0					
13742	073414	010002					MOV R0,R2	: *,WRDCNT				7209
13743	073416	062702	000400				ADD #400,R2	: *,WRDCNT				
13744	073422	017666	000030	000012			MOV @30(SP),12(SP)	: *,NEXT.SECT				7210
13745	073430	005266	000012				INC 12(SP)	: NEXT.SECT				
13746	073434	010116				7\$:	MOV R1,(SP)	: LUN,*				7213
13747	073436	010246					MOV R2,-(SP)	: WRDCNT,*				
13748	073440	016746	137224				MOV WPTR,-(SP)					
13749	073444	010346					MOV R3,-(SP)	: SECTOR,*				
13750	073446	004767	151752				JSR PC,WRITE					
13751	073452	010004					MOV R0,R4	: *,VALUE				
13752	073454	020427	000001				CMP R4,#1	: VALUE,*				7219
13753	073460	001031					BNE 8\$					
13754	073462	012746	000006				MOV #6,-(SP)	:				7225
13755	073466	012746	045424				MOV #WRITE,-(SP)					
13756	073472	010146					MOV R1,-(SP)	: LUN,*				
13757	073474	010246					MOV R2,-(SP)	: WRDCNT,*				
13758	073476	016746	137166				MOV WPTR,-(SP)					
13759	073502	010346					MOV R3,-(SP)	: SECTOR,*				
13760	073504	004767	152514				JSR PC,RETRY					
13761	073510	062706	000014				ADD #14,SP					
13762	073514	005700					TST R0					
13763	073516	001446					BEQ 11\$					
13764	073520	112761	000004	034446			MOVB #4,WHY.DROPT(R1)	: *,*(LUN)				7228
13765	073526	104455					TRAP 55	:				7229

Address	Instruction	Operand 1	Operand 2	Label	Comment	Register	Value	Address	Instruction	Operand 1	Operand 2	Label	Comment	Register	Value
13767					:MLX4										
13768					:				TESTING RANDOM DATA & WORD COUNTS						
13769															
13770	073530	012121							.WORD	12121					
13771	073532	011070							.WORD	MSG1					
13772	073534	000000							.WORD	0					
13773	073536	010100							MOV	R1,R0	:	LUN,*			7230
13774	073540	104451							TRAP	51					
13775	073542	000431							BR	10\$:				7231
13776	073544	020427	000002		8\$:				CMP	R4,#2	:	VALUE,*			7219
13777	073550	001012							BNE	9\$					
13778	073552	112761	000005	034446					MOVB	#5,WHY.DROPT(R1)	:	*,*(LUN)			7238
13779	073560	104455							TRAP	55	:				7239
13780	073562	012122							.WORD	12122					
13781	073564	011070							.WORD	MSG1					
13782	073566	000000							.WORD	0					
13783	073570	010100							MOV	R1,R0	:	LUN,*			7240
13784	073572	104451							TRAP	51					
13785	073574	000414							BR	10\$:				7241
13786	073576	020427	000003		9\$:				CMP	R4,#3	:	VALUE,*			7219
13787	073602	001014							BNE	11\$					
13788	073604	104455							TRAP	55	:				7246
13789	073606	012123							.WORD	12123					
13790	073610	011070							.WORD	MSG1					
13791	073612	000000							.WORD	0					
13792	073614	112761	000006	034446					MOVB	#6,WHY.DROPT(R1)	:	*,*(LUN)			7247
13793	073622	010100							MOV	R1,R0	:	LUN,*			7248
13794	073624	104451							TRAP	51					
13795	073626	062706	000012		10\$:				ADD	#12,SP	:				7249
13796	073632	000535							BR	16\$					
13797	073634	004767	152326		11\$:				JSR	PC,CHOOSE	:				7253
13798	073640	010066	000022						MOV	R0,22(SP)	:	*,COMMAND			
13799	073644	005005							CLR	R5	:				7255
13800	073646	020027	045612						CMP	R0,#READ	:	COMMAND,*			
13801	073652	001014							BNE	12\$					
13802	073654	005205							INC	R5					
13803	073656	016766	137010	000024					MOV	RPTR,24(SP)	:	*,PTR			7258
13804	073664	010146							MOV	R1,-(SP)	:	LUN,*			7259
13805	073666	010246							MOV	R2,-(SP)	:	WRDCNT,*			
13806	073670	016746	136776						MOV	RPTR,-(SP)	:				
13807	073674	010346							MOV	R3,-(SP)	:	SECTOR,*			
13808	073676	004767	151710						JSR	PC,READ					
13809	073702	000412							BR	13\$					
13810	073704	016766	136760	000024	12\$:				MOV	WPTR,24(SP)	:	*,PTR			7263
13811	073712	010146							MOV	R1,-(SP)	:	LUN,*			7264
13812	073714	010246							MOV	R2,-(SP)	:	WRDCNT,*			
13813	073716	016746	136746						MOV	WPTR,-(SP)	:				
13814	073722	010346							MOV	R3,-(SP)	:	SECTOR,*			
13815	073724	004767	152050						JSR	PC,CHECK					
13816	073730	010004			13\$:				MOV	R0,R4	:	*,VALUE			
13817	073732	001074							BNE	17\$:				7271
13818	073734	006005							ROR	R5	:				7276
13819	073736	103402							BLO	15\$					
13820	073740	000167	000750		14\$:				JMP	30\$					
13821	073744	016746	136720		15\$:				MOV	WPTR,-(SP)	:				7280

Address	Hex	Hex	Hex	Hex	Instruction	Comments	Count
13823							
13824							
13825							
13826	073750	016746	136716		MOV RPTR,-(SP)		
13827	073754	010246			MOV R2,-(SP)	: WRDCNT,*	
13828	073756	004767	150344		JSR PC,DOUBLE.CHECK		
13829	073762	062706	000006		ADD #6,SP		
13830	073766	010066	000040		MOV R0,40(SP)	: *,DBL.VALUE	
13831	073772	001762			BEQ 14\$		
13832	073774	010146			MOV R1,-(SP)	: LUN,*	7283
13833	073776	004767	141522		JSR PC,SAYWHO		
13834	074002	012716	011216		MOV #MSG5,(SP)	:	7284
13835	074006	012746	007072		MOV #SAY1,-(SP)		
13836	074012	012746	000002		MOV #2,-(SP)		
13837	074016	010600			MOV SP,R0	: SP,*	
13838	074020	104414			TRAP 14		
13839	074022	016616	000046		MOV 46(SP),(SP)	: DBL.VALUE,*	7285
13840	074026	017646	000046		MOV 46(SP),-(SP)	: DBL.VALUE,*	
13841	074032	012746	006710		MOV #FMT12A,-(SP)		
13842	074036	012746	000003		MOV #3,-(SP)		
13843	074042	010600			MOV SP,R0	: SP,*	
13844	074044	104414			TRAP 14		
13845	074046	062766	010000 000054		ADD #10000,54(SP)	: *,DBL.VALUE	7287
13846	074054	016616	000054		MOV 54(SP),(SP)	: DBL.VALUE,*	7288
13847	074060	017646	000054		MOV 54(SP),-(SP)	: DBL.VALUE,*	
13848	074064	012746	006756		MOV #FMT12B,-(SP)		
13849	074070	012746	000003		MOV #3,-(SP)		
13850	074074	010600			MOV SP,R0	: SP,*	
13851	074076	104414			TRAP 14		
13852	074100	112761	000010 034446		MOVB #10,WHY.DROPT(R1)	: *,*(LUN)	7290
13853	074106	104455			TRAP 55	:	7291
13854	074110	012124			.WORD 12124		
13855	074112	011070			.WORD MSG1		
13856	074114	000000			.WORD 0		
13857	074116	010100			MOV R1,R0	: LUN,*	7292
13858	074120	104451			TRAP 51		
13859	074122	062706	000044		ADD #44,SP	:	7293
13860	074126	000506			BR 22\$		
13861	074130	020427	000001		CMP R4,#1	: VALUE,*	7271
13862	074134	001031			BNE 18\$		
13863	074136	012746	000006		MOV #6,-(SP)	:	7301
13864	074142	016646	000034		MOV 34(SP),-(SP)	: COMMAND,*	
13865	074146	010146			MOV R1,-(SP)	: LUN,*	
13866	074150	010246			MOV R2,-(SP)	: WRDCNT,*	
13867	074152	016646	000044		MOV 44(SP),-(SP)	: PTR,*	
13868	074156	010346			MOV R3,-(SP)	: SECTOR,*	
13869	074160	004767	152040		JSR PC,RETRY		
13870	074164	062706	000014		ADD #14,SP		
13871	074170	005700			TST R0		
13872	074172	001662			BEQ 14\$		
13873	074174	112761	000004 034446		MOVB #4,WHY.DROPT(R1)	: *,*(LUN)	7304
13874	074202	104455			TRAP 55	:	7305
13875	074204	012125			.WORD 12125		
13876	074206	011070			.WORD MSG1		
13877	074210	000000			.WORD 0		

				27-Mar-1982 19:24:42	TOPS
				27-Mar-1982 19:23:44	PA:<
13879			:MLX4		
13880			:	TESTING RANDOM DATA & WORD COUNTS	
13881					
13882	074212	010100		MOV R1,RO	; LUN,*
13883	074214	104451		TRAP 51	
13884	074216	000450		BR 21\$	
13885	074220	020427	000002	CMP R4,#2	; VALUE,*
13886	074224	001012		BNE 19\$	
13887	074226	112761	000005	MOV #5,WHY.DROPT(R1)	; *,*(LUN)
13888	074234	104455		TRAP 55	
13889	074236	012126		.WORD 12126	
13890	074240	011070		.WORD MSG1	
13891	074242	000000		.WORD 0	
13892	074244	010100		MOV R1,RO	; LUN
13893	074246	104451		TRAP 51	
13894	074250	000433		BR 21\$	
13895	074252	020427	000003	CMP R4,#3	; VALUE,*
13896	074256	001012		BNE 20\$	
13897	074260	104455		TRAP 55	
13898	074262	012127		.WORD 12127	
13899	074264	011070		.WORD MSG1	
13900	074266	000000		.WORD 0	
13901	074270	112761	000006	MOV #6,WHY.DROPT(R1)	; *,*(LUN)
13902	074276	010100		MOV R1,RO	; LUN,*
13903	074300	104451		TRAP 51	
13904	074302	000416		BR 21\$	
13905	074304	020427	000004	CMP R4,#4	; VALUE,*
13906	074310	001017		BNE 23\$	
13907	074312	004767	143014	JSR PC,ISOLATE	
13908	074316	104455		TRAP 55	
13909	074320	012130		.WORD 12130	
13910	074322	011120		.WORD MSG2	
13911	074324	000000		.WORD 0	
13912	074326	112761	000007	MOV #7,WHY.DROPT(R1)	; *,*(LUN)
13913	074334	010100		MOV R1,RO	; LUN,*
13914	074336	104451		TRAP 51	
13915	074340	062706	000022	ADD #22,SP	
13916	074344	000167	001032	JMP 42\$	
13917	074350	020427	000005	CMP R4,#5	; VALUE,*
13918	074354	001157		BNE 30\$	
13919	074356	004767	142750	JSR PC,ISOLATE	
13920	074362	032767	000001	BIT #1,ERRRUT	
13921	074370	001423		BEQ 24\$	
13922	074372	017705	140012	MOV @ML.REG+42,R5	
13923	074376	006205		ASR R5	
13924	074400	006205		ASR R5	
13925	074402	006205		ASR R5	
13926	074404	006205		ASR R5	
13927	074406	006205		ASR R5	
13928	074410	006205		ASR R5	
13929	074412	042705	177700	BIC #177700,R5	
13930	074416	010546		MOV R5,-(SP)	
13931	074420	012746	006640	MOV #FMT10B,-(SP)	
13932	074424	012746	000002	MOV #2,-(SP)	
13933	074430	010600		MOV SP,RO	; SP,*

Address	OpCode	Operand 1	Operand 2	Operand 3	Operand 4	Instruction	Comment	Date/Time	Page
13935						TRAP	14	27-Mar-1982 19:24:42	TOPS
13936						ADD	#6,SP	27-Mar-1982 19:23:44	PA:<
13937						MOV	2ML.REG+44,36(SP)		
13938	074432	104414				MOV	2ML.REG+42,R5		7344
13939	074434	062706	000006	000036	24\$:	ASR	R5		7345
13940	074440	017766	137746			ASR	R5		
13941	074446	017705	137736			ASR	R5		
13942	074452	006205				ASR	R5		
13943	074454	006205				ASR	R5		
13944	074456	006205				ASR	R5		
13945	074460	006205				ASR	R5		
13946	074462	006205				ASR	R5		
13947	074464	006205				ASR	R5		
13948	074466	042705	177700			BIC	#177700,R5		
13949	074472	010566	000042			MOV	R5,42(SP)		7347
13950	074476	012746	000001			MOV	#1,-(SP)		
13951	074502	016646	000034			MOV	34(SP),-(SP)		
13952	074506	010146				MOV	R1,-(SP)		
13953	074510	010246				MOV	R2,-(SP)		
13954	074512	016646	000044			MOV	44(SP),-(SP)		
13955	074516	010346				MOV	R3,-(SP)		
13956	074520	004767	151500			JSR	PC,RETRY		
13957	074524	062706	000014			ADD	#14,SP		
13958	074530	020027	000005			CMP	R0,#5		
13959	074534	001051				BNE	27\$		
13960	074536	027766	137650	000036		CMP	2ML.REG+44,36(SP)		7350
13961	074544	001034				BNE	26\$		
13962	074546	016600	000042			MOV	42(SP),R0		
13963	074552	017705	137632			MOV	2ML.REG+42,R5		
13964	074556	006205				ASR	R5		
13965	074560	006205				ASR	R5		
13966	074562	006205				ASR	R5		
13967	074564	006205				ASR	R5		
13968	074566	006205				ASR	R5		
13969	074570	006205				ASR	R5		
13970	074572	042705	177700			BIC	#177700,R5		
13971	074576	020500				CMP	R5,R0		
13972	074600	001016				BNE	26\$		
13973	074602	032767	000001	105450		BIT	#1,ERROUT		7354
13974	074610	001404				BEQ	25\$		
13975	074612	104456				TRAP	56		
13976	074614	012131				.WORD	12131		
13977	074616	011202				.WORD	MSG4		
13978	074620	000000				.WORD	0		
13979	074622	010146			25\$:	MOV	R1,-(SP)		7357
13980	074624	016746	137512			MOV	BOARD,-(SP)		
13981	074630	004767	142566			JSR	PC,UP.HARD.COUNT		
13982	074634	000426				BR	29\$		7350
13983	074636	032767	000001	105414	26\$:	BIT	#1,ERROUT		7362
13984	074644	001415				BEQ	28\$		
13985	074646	104457				TRAP	57		
13986	074650	012132				.WORD	12132		
13987	074652	011166				.WORD	MSG3		
13988	074654	000000				.WORD	0		
13989	074656	000410				BR	28\$		7365

Address	Hex	Hex	Hex	Hex	Hex	Op	Comment	Count	Date/Time	Page
13991										
13992									27-Mar-1982 19:24:42	TOPS
13993							TESTING RANDOM DATA & WORD COUNTS		27-Mar-1982 19:23:44	PA:<
13994	074660	032767	000001	105372	27\$:	BIT	#1,ERROUT			7371
13995	074666	001404				BEQ	28\$			
13996	074670	104457				TRAP	57			
13997	074672	012133				.WORD	12133			
13998	074674	011166				.WORD	MSG3			
13999	074676	000000				.WORD	0			
14000	074700	010146			28\$:	MOV	R1,-(SP)		: LUN,*	7374
14001	074702	016746	137434			MOV	BOARD,-(SP)			
14002	074706	004767	143172			JSR	PC,UP.SOFT.COUNT			
14003	074712	022626			29\$:	CMP	(SP)+,(SP)+			7338
14004	074714	016603	000030		30\$:	MOV	30(SP),R3		: NEXT.SECT,SECTOR	7380
14005	074720	062767	000002	135742		ADD	#2,WPTR			7381
14006	074726	062706	000022			ADD	#22,SP			7199
14007	074732	000167	176362			JMP	5\$			7198
14008	074736	032767	000001	105316	31\$:	BIT	#1,EFNS21			7392
14009	074744	001002				BNE	32\$			
14010	074746	000167	000430			JMP	42\$			
14011	074752	010100			32\$:	MOV	R1,R0		: LUN,*	7399
14012	074754	006300				ASL	R0			
14013	074756	016066	034500	000024		MOV	TOP.SECT(R0),24(SP)			
14014	074764	016005	034460			MOV	LOW.SECT(R0),R5		: *,SECTOR	
14015	074770	000577				BR	41\$			
14016	074772	010146			33\$:	MOV	R1,-(SP)		: LUN,*	7401
14017	074774	012746	000400			MOV	#400,-(SP)			
14018	075000	012746	022670			MOV	#RBUF,-(SP)			
14019	075004	010546				MOV	R5,-(SP)		: SECTOR,*	
14020	075006	004767	150600			JSR	PC,READ			
14021	075012	062706	000010			ADD	#10,SP			
14022	075016	020027	000005			CMP	R0,#5			
14023	075022	001161				BNE	40\$			
14024	075024	004767	142302			JSR	PC,ISOLATE			7404
14025	075030	032767	000001	105222		BIT	#1,ERROUT			7406
14026	075036	001423				BEQ	34\$			
14027	075040	017700	137344			MOV	@ML.REG+42,R0			
14028	075044	006200				ASR	R0			
14029	075046	006200				ASR	R0			
14030	075050	006200				ASR	R0			
14031	075052	006200				ASR	R0			
14032	075054	006200				ASR	R0			
14033	075056	006200				ASR	R0			
14034	075060	042700	177700			BIC	#177700,R0			
14035	075064	010046				MOV	R0,-(SP)			
14036	075066	012746	006640			MOV	#FMT10B,-(SP)			
14037	075072	012746	000002			MOV	#2,-(SP)			
14038	075076	010600				MOV	SP,R0		: SP,*	
14039	075100	104414				TRAP	14			
14040	075102	062706	000000			ADD	#6,SP			
14041	075106	017766	137300	000014	34\$:	MOV	@ML.REG+44,14(SP)		: *,OLDSEC	7415
14042	075114	017700	137270			MOV	@ML.REG+42,R0			7416
14043	075120	006200				ASR	R0			
14044	075122	006200				ASR	R0			
14045	075124	006200				ASR	R0			

Address	Instruction	Operand 1	Operand 2	Operand 3	Operand 4	Comments	Count
14047							
14048							
14049							
14050	ASR	RO					
14051	ASR	RO					
14052	ASR	RO					
14053	BIC	#177700,RO	177700				
14054	MOV	RO,20(SP)	000020			: *,OLDCHN	
14055	MOV	#1,-(SP)	000001				7424
14056	MOV	12(SP),-(SP)	000012			: COMMAND,*	
14057	MOV	R1,-(SP)				: LUN,*	
14058	MOV	#400,-(SP)	000400				
14059	MOV	22(SP),-(SP)	000022			: PTR,*	
14060	MOV	26(SP),-(SP)	000026			: OLDSEC,*	
14061	JSR	PC,RETRY	151026				
14062	ADD	#14,SP	000014				
14063	CMP	RO,#5	000005				
14064	BNE	37\$					
14065	CMP	@ML.REG+44,14(SP)	137176	000014		: *,OLDSEC	7433
14066	BNE	36\$					
14067	MOV	20(SP),-(SP)	000020			: OLDCHN,*	
14068	MOV	@ML.REG+42,RO	137160				
14069	ASR	RO					
14070	ASR	RO					
14071	ASR	RO					
14072	ASR	RO					
14073	ASR	RO					
14074	ASR	RO					
14075	BIC	#177700,RO	177700				
14076	CMP	RO,(SP)+					
14077	BNE	36\$					
14078	BIT	#1,ERROUT	000001	104776			7437
14079	BEQ	35\$					
14080	TRAP	56					7442
14081	.WORD	12134					
14082	.WORD	MSG4					
14083	.WORD	0					
14084	MOV	R1,-(SP)			35\$:	: LUN,*	7445
14085	MOV	BOARD,-(SP)	137040				
14086	JSR	PC,UP.HARD.COUNT	142114				
14087	BR	39\$					7433
14088	BIT	#1,ERROUT	000001	104742	36\$:		7450
14089	BEQ	38\$					
14090	TRAP	57					7455
14091	.WORD	12135					
14092	.WORD	MSG3					
14093	.WORD	0					
14094	BR	38\$					7458
14095	BIT	#1,ERROUT	000001	104720	37\$:		7464
14096	BEQ	38\$					
14097	TRAP	57					7469
14098	.WORD	12136					
14099	.WORD	MSG3					
14100	.WORD	0					
14101	MOV	R1,-(SP)			38\$:	: LUN,*	7472

```
14103      :MLX4
14104      :
14105
14106 075354 016746 136762      MOV      BOARD,-(SP)
14107 075360 004767 142520      JSR      PC,UP.SOFT.COUNT
14108 075364 022626      39$:    CMP      (SP)+,(SP)+          :
14109 075366 005205      40$:    INC      R5                : SECTOR
14110 075370 020566 000024      41$:    CMP      R5,24(SP)       : SECTOR,*
14111 075374 101002      BHI     42$
14112 075376 000167 177370      JMP     33$
14113 075402 005201      42$:    INC      R1                : LUN
14114 075404 020166 000026      43$:    CMP      R1,26(SP)       : LUN,*
14115 075410 002002      BGE     44$
14116 075412 000167 175562      JMP     2$
14117 075416 005266 000022      44$:    INC      22(SP)          : COUNT
14118 075422 026666 000022 000046      CMP     22(SP),46(SP)         : COUNT,REPEAT
14119 075430 003002      BGT     45$
14120 075432 000167 175526      JMP     1$
14121 075436 062706 000030      45$:    ADD     #30,SP          :
14122 075442 000207      RTS     PC
14123
14124      : Routine Size: 616 words
14125      : Maximum stack depth per invocation: 36 words
14130
14131
```

```

14133 :MLX4
14134 : TESTING RANDOM SECTORS, DATA, WORD COUNTS
14135 :
14136 : 7486 %sbttl 'TESTING RANDOM SECTORS, DATA, WORD COUNTS'
14137 : 7487 routine RAND3 (REPEAT) : novalue =
14138 : 7488 begin !* 1 * START OF ROUTINE
14139 : 7489
14140 : 7490 ++
14141 : 7491 ROUTINE: RAND3(REPEAT)
14142 : 7492
14143 : 7493 PURPOSE: TO CHECK RANDOM SECTORS
14144 : 7494
14145 : 7495 ARGUMENT: REPEAT = NUMBER OF TIMES TO EXECUTE ENTIRE ROUTINE
14146 : 7496
14147 : 7497 THE CODE FOR 'RAND3' IN BRIEF:
14148 : 7498
14149 : 7499 BEGIN 1 (START OF ROUTINE)
14150 : 7500 SAY ROUTINE IS RUNNING
14151 : 7501 INCR COUNT FROM 1 TO REPEAT
14152 : 7502 : BEGIN 2 (START OF REPEAT LOOP FOR THIS ROUTINE)
14153 : 7503 : GENERATE THE RANDOM PATTERN
14154 : 7504 : INCR LUN FROM 0 TO LAST
14155 : 7505 : : BEGIN 3 (START OF LOGICAL UNIT SELECTION LOOP)
14156 : 7506 : : INITIALIZE BUFFER POINTERS
14157 : 7507 : : TESTLOOP:
14158 : 7508 : : : BEGIN 4 (START OF LOOP THAT COMPLETELY TESTS 1 UNIT)
14159 : 7509 : : : IF UNIT IS ACTIVE
14160 : 7510 : : : THEN
14161 : 7511 : : : : BEGIN 5 (START OF TEST FOR AN ACTIVE UNIT)
14162 : 7512 : : : : : TIMES = (HIGHEST - LOWEST)/2 + 1
14163 : 7513 : : : : : INCR KOUNT FROM 1 TO TIMES
14164 : 7514 : : : : : BEGIN 6 (START OF COUNTING LOOP FOR SECTOR SELECTION)
14165 : 7515 : : : : : CHOOSE A RANDOM WORD COUNT
14166 : 7516 : : : : : SET UP BUFFER POINTERS BEFORE TRANSFER
14167 : 7517 : : : : : CHOOSE A RANDOM SECTOR
14168 : 7518 : : : : : CALCULATE WHERE TRANSFER WILL END (BASED ON WORD COUNT)
14169 : 7519 : : : : : IF CALCULATED VALUE GTR HIGHEST
14170 : 7520 : : : : : : THEN ADJUST THE WORD COUNT SO IT FITS
14171 : 7521 : : : : : : : WITHIN THE TESTABLE SECTOR LIMITS
14172 : 7522 : : : : : WRITE
14173 : 7523 : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE LABEL)
14174 : 7524 : : : : : CHOOSE WHETHER TO WRITE CHECK OR READ
14175 : 7525 : : : : : DO THE WRITE CHECK OR READ
14176 : 7526 : : : : : LOOK FOR ERRORS (IF DROP UNIT, LEAVE LABEL)
14177 : 7527 : : : : : CHANGE BUFFER POINTERS AFTER TRANSFER
14178 : 7528 : : : : : END 6 (END OF COUNTING LOOP FOR SECTOR SELECTION)
14179 : 7529 : : : : : END 5 (END OF TEST FOR AN ACTIVE UNIT)
14180 : 7530 : : : : : END 4 (END OF TESTLOOP)
14181 : 7531 : : : : : END 3 (END OF LOGICAL UNIT SELECTION LOOP)
14182 : 7532 : : : : : END 2 (END OF REPEAT LOOP FOR THIS ROUTINE)
14183 : 7533 RETURN
14184 : 7534 END 1 (END OF ROUTINE)
14185 : 7535
14186 : 7536
14187 : 7537
  
```

Local

```

14189 :MLX4
14190 : TESTING RANDOM SECTORS, DATA, WORD COUNTS
14191 :
14192 : 7538 WRDCNT,
14193 : 7539 SECTOR,
14194 : 7540 TIMES,
14195 : 7541 VALUE,
14196 : 7542 OLDSEC,
14197 : 7543 OLDCHN,
14198 : 7544 PTR,
14199 : 7545 COMMAND,
14200 : 7546 DBL_VALUE;
14201 : 7547
14202 : Label
14203 : LOOP;
14204 :
14205 : PRINTB (SAY1, RTN5C);
14206 : !'RAND3'
14207 : 7553
14208 : 7554 incr COUNT from 1 to .REPEAT do
14209 : 7555 begin !* 2 * START OF REPEAT LOOP FOR THIS ROUTINE
14210 : 7556
14211 : 7557 incr LUN from 0 to (.LSUNIT - 1) do
14212 : 7558 begin !* 3 * START OF LOGICAL UNIT SELECTION LOOP
14213 : 7559 GENS ();
14214 : 7560 LOOP :
14215 : 7561 begin !* 4 * START OF THE LOOP THAT COMPLETELY TESTS 1 UNIT
14216 : 7562
14217 : 7563 if .DRIVE_STATUS [.LUN] eql ACTIVE
14218 : 7564 then
14219 : 7565 begin !* 5 * START OF TEST FOR AN ACTIVE UNIT
14220 : 7566 L$LUN = .LUN;
14221 : 7567 WPTR = WBUFF;
14222 : 7568 RPTR = RBUFF;
14223 : 7569 TIMES = (HIGHEST - LOWEST)/2 + 1;
14224 : 7570
14225 : 7571 incr KOUNT from 1 to .TIMES do
14226 : 7572 begin !* 6 * START OF COUNTING LOOP FOR SECTOR SELECTION
14227 : 7573 WRDCNT = RNDWC (); !VALUE BETWEEN 1 AND BUFSIZ
14228 : 7574 SET PTRS (.WRDCNT);
14229 : 7575 SECTOR = RNDSEC (.LUN); !VALUE BETWEEN LOWEST AND HIGHEST
14230 : 7576
14231 : 7577 if (.SECTOR + .WRDCNT/256) gtra HIGHEST then WRDCNT = 256*(HIGHEST - .SECTOR + 1);
14232 : 7578
14233 : 7579 VALUE = write (.LUN, .WRDCNT, .WPTR, .SECTOR);
14234 : 7580
14235 : 7581 !+
14236 : 7582 ! SEE HOW SUCCESSFUL THE WRITE WAS:
14237 : 7583 !-
14238 : 7584
14239 : 7585 selectone .VALUE of !SEE 'SYSERR' FOR DEFINITION
14240 : 7586 set !OF ERROR # CONTAINED IN 'VALUE'
14241 : 7587
14242 : 7588 [1] :
14243 : 7589 begin !* 6A * RETRY ALLOWED
  
```

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (46)

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (46)

14245 :MLX4
 14246 :
 14247 :
 14248 :
 14249 :
 14250 :
 14251 :
 14252 :
 14253 :
 14254 :
 14255 :
 14256 :
 14257 :
 14258 :
 14259 :
 14260 :
 14261 :
 14262 :
 14263 :
 14264 :
 14265 :
 14266 :
 14267 :
 14268 :
 14269 :
 14270 :
 14271 :
 14272 :
 14273 :
 14274 :
 14275 :
 14276 :
 14277 :
 14278 :
 14279 :
 14280 :
 14281 :
 14282 :
 14283 :
 14284 :
 14285 :
 14286 :
 14287 :
 14288 :
 14289 :
 14290 :
 14291 :
 14292 :
 14293 :
 14294 :
 14295 :
 14296 :
 14297 :
 14298 :
 14299 :

7590
 7591
 7592
 7593
 7594
 7595
 7596
 7597
 7598
 7599
 7600
 7601
 7602
 7603
 7604
 7605
 7606
 7607
 7608
 7609
 7610
 7611
 7612
 7613
 7614
 7615
 7616
 7617
 7618
 7619
 7620
 7621
 7622
 7623
 7624
 7625
 7626
 7627
 7628
 7629
 7630
 7631
 7632
 7633
 7634
 7635
 7636
 7637
 7638
 7639
 7640
 7641

TESTING RANDOM SECTORS, DATA, WORD COUNTS

```

if RETRY (SIX, write, .LUN, .WRDCNT, .WPTR, .SECTOR) neq 0
then
  !THE RETRY FAILED -- SYSTEM FATAL ERROR
  begin
    WHY DROPT [.LUN] = CODE_4;
    ERRDF (5301, MSG1, 0); !**** OPTION 5, RAND3 ERROR 01 ****
    DODU (.LUN);
    leave LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
  end;
end; !* 6A *

[2] :
begin !* 6B * FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
  WHY DROPT [.LUN] = CODE_5;
  ERRDF (5302, MSG1, 0); !**** OPTION 5, RAND3 ERROR 02 ****
  DODU (.LUN);
  leave LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
end; !* 6B *

[3] :
begin !* 6C * FATAL DRIVE ERROR -- NO RETRY ALLOWED
  ERRDF (5303, MSG1, 0); !**** OPTION 5, RAND3 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] :
```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (46)

```
14301 :MLX4
14302 : TESTING RANDOM SECTORS, DATA, WORD COUNTS
14303 :
14304 : 7642 if .COMMAND eql read
14305 : 7643 then
14306 : 7644 begin
14307 : 7645
14308 : 7646 if (DBL_VALUE = DOUBLE_CHECK (.WPTR, .RPTR, .WRDCNT)) neq 0
14309 : 7647 then
14310 : 7648 begin
14311 : 7649 SAYWHO (.LUN);
14312 : 7650 PRINTB (SAY1, MSG5); !'ECC LOGIC FAILED TO DETECT DATA ERROR'
14313 : 7651 PRINTB (FMT12A, .DBL_VALUE, .DBL_VALUE);
14314 : 7652 !'GOOD DATA: XXXXXX AT LOCATION YYYYYY'
14315 : 7653 DBL_VALUE = .DBL_VALUE + BUFSIZ*2;
14316 : 7654 PRINTB (FMT12B, .DBL_VALUE, .DBL_VALUE);
14317 : 7655 !'BAD DATA: PPPPPP AT LOCATION QQQQQQ'
14318 : 7656 WHY DROPT [.LUN] = CODE_8;
14319 : 7657 ERRDF (5304, MSG1, 0); !**** OPTION 5, RAND3 ERROR 04 ****
14320 : 7658 DODU (.LUN);
14321 : 7659 leave LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
14322 : 7660 end;
14323 : 7661
14324 : 7662 end;
14325 : 7663
14326 : 7664 [1] :
14327 : 7665 begin !* 6D * RETRY ALLOWED
14328 : 7666
14329 : 7667 if RETRY (SIX, .COMMAND, .LUN, .WRDCNT, .PTR, .SECTOR) neq 0
14330 : 7668 then !THE RETRY FAILED -- SYSTEM FATAL ERROR
14331 : 7669 begin
14332 : 7670 WHY DROPT [.LUN] = CODE_4;
14333 : 7671 ERRDF (5305, MSG1, 0); !**** OPTION 5, RAND3 ERROR 05 ****
14334 : 7672 DODU (.LUN);
14335 : 7673 leave LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
14336 : 7674 end;
14337 : 7675
14338 : 7676 end; !* 6D *
14339 : 7677
14340 : 7678 [2] :
14341 : 7679 begin !* 6E * FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
14342 : 7680 WHY DROPT [.LUN] = CODE_5;
14343 : 7681 ERRDF (5306, MSG1, 0); !**** OPTION 5, RAND3 ERROR 06 ****
14344 : 7682 DODU (.LUN);
14345 : 7683 leave LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
14346 : 7684 end;
14347 : 7685
14348 : 7686 [3] :
14349 : 7687 begin !* 6F * FATAL DRIVE ERROR -- NO RETRY ALLOWED
14350 : 7688 ERRDF (5307, MSG1, 0); !**** OPTION 5, RAND3 ERROR 07 ****
14351 : 7689 WHY DROPT [.LUN] = CODE_6;
14352 : 7690 DODU (.LUN);
14353 : 7691 leave LOOP; !JUMP JUST BEYOND END OF BLOCK * 4 *
14354 : 7692 end; !* 6F *
14355 : 7693
```


27-Mar-1982 19:24.42 TOPS-20 Bliss-16 V2(212)
27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (46)

14357 :MLX4
14358 :
14359 :
14360 :
14361 :
14362 :
14363 :
14364 :
14365 :
14366 :
14367 :
14368 :
14369 :
14370 :
14371 :
14372 :
14373 :
14374 :
14375 :
14376 :
14377 :
14378 :
14379 :
14380 :
14381 :
14382 :
14383 :
14384 :
14385 :
14386 :
14387 :
14388 :
14389 :
14390 :
14391 :
14392 :
14393 :
14394 :
14395 :
14396 :
14397 :
14398 :
14399 :
14400 :
14401 :
14402 :
14403 :
14404 :
14405 :
14406 :
14407 :
14408 :
14409 :
14410 :
14411 :

7694
7695
7696
7697
7698
7699
7700
7701
7702
7703
7704
7705
7706
7707
7708
7709
7710
7711
7712
7713
7714
7715
7716
7717
7718
7719
7720
7721
7722
7723
7724
7725
7726
7727
7728
7729
7730
7731
7732
7733
7734
7735
7736
7737
7738
7739
7740
7741
7742
7743
7744
7745

TESTING RANDOM SECTORS, DATA, WORD COUNTS

```
[4] :  
begin                !* 6G * UNRECOVERABLE DATA ERROR  
ISOLATE ();  
ERRDF (5308, MSG2, 0);      !**** OPTION 5, RAND3 ERROR 08 ****  
WHY DROPT [.LUN] = CODE_7;  
DODC (.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 eql .OLDCHN))  
    then  
        begin  
            if .ERROUT then ERRHRD (5309, MSG4, 0);  
  
                !**** OPTION 5, RAND3 ERROR 09 ****  
            UP_HARD_COUNT (.LUN, .BOARD);  
        end  
    else  
        begin  
            if .ERROUT then ERRSOFT (5310, MSG3, 0);  
  
                !**** OPTION 5, RAND3 ERROR 10 ****  
            UP_SOFT_COUNT (.LUN, .BOARD);  
        end  
    else  
        begin  
            if .ERROUT then ERRSOFT (5311, MSG3, 0);  
  
                !**** OPTION 5, RAND3 ERROR 11 ****  
            UP_SOFT_COUNT (.LUN, .BOARD);  
        end  
    end;  
end;  
tes;                  !* 6H *
```

```

14413 :MLX4
14414 :
14415 :
14416 : 7746
14417 : 7747
14418 : 7748
14419 : 7749
14420 : 7750
14421 : 7751
14422 : 7752
14423 : 7753
14424 : 7754
14425 : 7755
14426 : 7756
14427 : 7757
14428 : 7758
14429 : 7759
14430 : 7760
14431 : 7761
14432 : 7762
14433 : 7763
14434 : 7764
14435 : 7765
14436 : 7766
14437 : 7767
14438 : 7768
14439 : 7769
14440 : 7770
14441 : 7771
14442 : 7772
14443 : 7773
14444 : 7774
14445 : 7775
14446 : 7776
14447 : 7777
14448 : 7778
14449 : 7779
14450 : 7780
14451 : 7781
14452 : 7782
14453 : 7783
14454 : 7784
14455 : 7785
14456 : 7786
14457 : 7787
14458 : 7788
14459 : 7789
14460 : 7790
14461 : 7791
14462 : 7792
14463 : 7793
14464 : 7794
14465 : 7795
14466 : 7796
14467 : 7797

TESTING RANDOM SECTORS, DATA, WORD COUNTS

WPTR = .WPTR + 2;
end;
!* 6 * END OF COUNTING LOOP FOR SECTOR SELECTION

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

!+
! Test to see if this uut's address space is
! to be read for soft errors. This test is
! is intended for DMT purposes.
!-

if .EFNS21
then
begin
!Is the background pattern to be read

version czmlbb changed incr to incru
incru SECTOR from LOWEST to HIGHEST do
if read (.LUN, 256, RBUFF, .SECTOR) eql 5
then
begin
ISOLATE (); !Find the failing bank and board no.
if .ERROUT then PRINTB (FMT10B, .CHAN);
! Print where the error is
! Save the contents of the ML error location
! register so we can compare it to the new
! contents of this register after the retry.
! This is done to classify the error.
OLDSEC = .MLEL;
OLDCHN = .CHAN;
! Do a classify retry call. If the same error
! occurs then classify it as a hard error. If
! a different error occurred or the error went away
! then classify it as a soft error.

if RETRY (ONE, .COMMAND, .LUN, 256, .PTR, .OLDSEC) eql 5
then
! The same error occurred so see if it is at the same
! sector and channel number, if so then classify
! it as a hard error else classiy it as a soft
! error.

```

```

14469 :MLX4
14470 :
14471 :
14472 : 7798
14473 : 7799
14474 : 7800
14475 : 7801
14476 : 7802
14477 : 7803
14478 : 7804
14479 : P 7805
14480 : P 7806
14481 : 7807
14482 : 7808
14483 : 7809
14484 : 7810
14485 : 7811
14486 : 7812
14487 : 7813
14488 : 7814
14489 : 7815
14490 : 7816
14491 : 7817
14492 : P 7818
14493 : P 7819
14494 : 7820
14495 : 7821
14496 : 7822
14497 : 7823
14498 : 7824
14499 : 7825
14500 : 7826
14501 : 7827
14502 : 7828
14503 : 7829
14504 : 7830
14505 : 7831
14506 : P 7832
14507 : P 7833
14508 : 7834
14509 : 7835
14510 : 7836
14511 : 7837
14512 : 7838
14513 : 7839
14514 : 7840
14515 : 7841
14516 : 7842
14517 : 7843
14518 : 7844
14519 : 7845
14520 : 7846
14521 : 7847
14522 : 7848
14523 : 7849

TESTING RANDOM SECTORS, DATA, WORD COUNTS

if ((.MLEL eql .OLDSEC) and (.CHAN eql .OLDCHN))
then
begin
!Same error occurred 'hard'
if .ERROUT !Print error if enabled
then
begin
ERRHRD (5312, !Error number
MSG4, !Error message
0); !Additional message routine
end;
UP_HARD_COUNT (.LUN, .BOARD);
end
else
begin
!Not the same error 'soft'
if .ERROUT !Print error if enabled
then
begin
ERRSOFT (5313, !Error number
MSG3, !Error message
0); !Additional message routine
end;
UP_SOFT_COUNT (.LUN, .BOARD);
end
else
begin
!Not the same error 'soft'
if .ERROUT !Print error if enabled
then
begin
ERRSOFT (5314, !Error number
MSG3, !Error message
0); !Additional message routine
end;
UP_SOFT_COUNT (.LUN, .BOARD);
end;
end;
end;
end;
end;
end;
end;
end;
end;
return;

!* 4 * END OF LOOP THAT COMPLETELY TESTS 1 UNIT
!* 3 * END OF LOGICAL UNIT SELECTION LOOP
!* 2 * END OF REPEAT LOOP FOR THIS ROUTINE
  
```

14525 :MLX4
 14526 :
 14527 :
 14528 : 7850 end;

TESTING RANDOM SECTORS, DATA, WORD COUNTS

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (46)

!* 1 * END OF ROUTINE

Address	Hex	Hex	Hex	Label	Instruction	Comments	Count
14533					.SBTTL	RAND3 TESTING RANDOM SECTORS, DATA, WORD COUNTS	
14537	075444	004167	107664	RAND3:	JSR R1,\$SAVE5		7487
14538	075450	162706	000024		SUB #24,SP		
14539	075454	012746	007700		MOV #RTN5C,-(SP)		7551
14540	075460	012746	007072		MOV #SAY1,-(SP)		
14541	075464	012746	000002		MOV #2,-(SP)		
14542	075470	010600			MOV SP,R0	: SP,*	
14543	075472	104414			TRAP 14		
14544	075474	005066	000030		CLR 30(SP)	: COUNT	7554
14545	075500	000167	002254		JMP 44\$		
14546	075504	016766	104302	000012	1\$: MOV L\$UNIT,12(SP)		7557
14547	075512	005003			CLR R3	: LUN	
14548	075514	000167	002226		JMP 43\$		
14549	075520	004767	144074	2\$:	JSR PC,GEN5		7559
14550	075524	010300			MOV R3,R0	: LUN,*	7563
14551	075526	006200			ASR R0		
14552	075530	006200			ASR R0		
14553	075532	006200			ASR R0		
14554	075534	062700	034442		ADD #DRIVE.STATUS,R0		
14555	075540	010046			MOV R0,-(SP)		
14556	075542	010346			MOV R3,-(SP)	: LUN,*	
14557	075544	042716	177770		BIC #177770,(SP)		
14558	075550	012746	000001		MOV #1,-(SP)		
14559	075554	005046			CLR -(SP)		
14560	075556	004767	106574		JSR PC,BL\$GT2		
14561	075562	062706	000010		ADD #10,SP		
14562	075566	005300			DEC R0		
14563	075570	001402			BEQ 3\$		
14564	075572	000167	001502		JMP 31\$		
14565	075576	010367	104272	3\$:	MOV R3,L\$LUN	: LUN,*	7566
14566	075602	012767	012670	135060	MOV #W\$BUFF,W\$PTR		7567
14567	075610	012767	022670	135054	MOV #R\$BUFF,R\$PTR		7568
14568	075616	010300			MOV R3,R0	: LUN,*	7569
14569	075620	006300			ASL R0		
14570	075622	012766	034500	000010	MOV #TOP.SECT,10(SP)		
14571	075630	060066	000010		ADD R0,10(SP)		
14572	075634	017646	000010		MOV @10(SP),-(SP)		
14573	075640	166016	034460		SUB LOW.SECT(R0),(SP)		
14574	075644	012746	000002		MOV #2,-(SP)		
14575	075650	004767	107314		JSR PC,BL\$DIV		
14576	075654	010066	000020		MOV R0,20(SP)	: *,TIMES	
14577	075660	005266	000020		INC 20(SP)	: TIMES	
14578	075664	005066	000012		CLR 12(SP)	: KOUNT	7571
14579	075670	000167	001362		JMP 29\$		

Address	OpCode	Operand 1	Operand 2	Operand 3	Instruction	Comments	Time	Page
14581					:MLX4		27-Mar-1982 19:24:42	TOPS
14582					:	TESTING RANDOM SECTORS, DATA, WORD COUNTS	27-Mar-1982 19:23:44	PA:<
14583					:			7573
14584	075674	004767	172304		4\$: JSR PC,RNDWC			
14585	075700	010002			MOV R0,R2	: *,WRDCNT		
14586	075702	010246			MOV R2,-(SP)	: WRDCNT,*		7574
14587	075704	004767	151222		JSR PC,SET_PTRS			
14588	075710	010316			MOV R3,(SP)	: LUN,*		7575
14589	075712	004767	172322		JSR PC,RNDSEC			
14590	075716	010001			MOV R0,R1	: *,SECTOR		
14591	075720	010246			MOV R2,-(SP)	: WRDCNT,*		7577
14592	075722	012746	000400		MOV #400,-(SP)			
14593	075726	004767	107236		JSR PC,BL\$DIV			
14594	075732	022626			CMP (SP)+,(SP)+			
14595	075734	060100			ADD R1,R0	: SECTOR,*		
14596	075736	020076	000016		CMP R0,@16(SP)			
14597	075742	101410			BLOS 5\$			
14598	075744	017600	000016		MOV @16(SP),R0			
14599	075750	160100			SUB R1,R0	: SECTCR,*		
14600	075752	000300			SWAB R0			
14601	075754	105000			CLRB R0			
14602	075756	010002			MOV R0,R2	: *,WRDCNT		
14603	075760	062702	000400		ADD #400,R2	: *,WRDCNT		
14604	075764	010316			5\$: MOV R3,(SP)	: LUN,*		7579
14605	075766	010246			MOV R2,-(SP)	: WRDCNT,*		
14606	075770	016746	134674		MOV WPTR,-(SP)			
14607	075774	010146			MOV R1,-(SP)	: SECTOR,*		
14608	075776	004767	147422		JSR PC,WRITE			
14609	076002	010004			MOV R0,R4	: *,VALUE		
14610	076004	020427	000001		CMP R4,#1	: VALUE,*		7585
14611	076010	001031			BNE 6\$			
14612	076012	012746	000006		MOV #6,-(SP)			7591
14613	076016	012746	045424		MOV #WRITE,-(SP)			
14614	076022	010346			MOV R3,-(SP)	: LUN,*		
14615	076024	010246			MOV R2,-(SP)	: WRDCNT,*		
14616	076026	016746	134636		MOV WPTR,-(SP)			
14617	076032	010146			MOV R1,-(SP)	: SECTOR,*		
14618	076034	004767	150164		JSR PC,RETRY			
14619	076040	062706	000014		ADD #14,SP			
14620	076044	005700			TST R0			
14621	076046	001446			BEQ 9\$			
14622	076050	112763	000004	034446	MOVB #4,WHY.DROPT(R3)	: *,*(LUN)		7594
14623	076056	104455			TRAP 55	:		7595
14624	076060	012265			.WORD 12265			
14625	076062	011070			.WORD MSG1			
14626	076064	000000			.WORD 0			
14627	076066	010300			MOV R3,R0	: LUN,*		7596
14628	076070	104451			TRAP 51			
14629	076072	000431			BR 8\$: VALUE,*		7597
14630	076074	020427	000002		6\$: CMP R4,#2	: VALUE,*		7585
14631	076100	001012			BNE 7\$			
14632	076102	112763	000005	034446	MOVB #5,WHY.DROPT(R3)	: *,*(LUN)		7604
14633	076110	104455			TRAP 55	:		7605
14634	076112	012266			.WORD 12266			
14635	076114	011070			.WORD MSG1			

Address	OpCode	Operand1	Operand2	Operand3	Label	Instruction	Comments	Address	Time	Page
14637									27-Mar-1982 19:24:42	TOPS
14638									27-Mar-1982 19:23:44	PA:
14639										
14640	076116	000000				.WORD 0				
14641	076120	010300				MOV R3,RO	: LUN,*			7606
14642	076122	104451				TRAP 51				
14643	076124	080414				BR 8\$				7607
14644	076126	020427	000003	7\$:		CMP R4,#3	: VALUE,*			7585
14645	076132	001014				BNE 9\$				
14646	076134	104455				TRAP 55				7612
14647	076136	012267				.WORD 12267				
14648	076140	011070				.WORD MSG1				
14649	076142	000000				.WORD 0				
14650	076144	112763	000006	034446		MOVB #6,WHY.DROPT(R3)	: *,*(LUN)			7613
14651	076152	010300				MOV R3,RO	: LUN,*			7614
14652	076154	104451				TRAP 51				
14653	076156	062706	000014	8\$:		ADD #14,SP				7615
14654	076162	000535				BR 14\$				
14655	076164	004767	147776	9\$:		JSR PC,CHOOSE				7619
14656	076170	010066	000036			MOV RO,#36(SP)	: *,COMMAND			
14657	076174	005005				CLR R5				7621
14658	076176	020027	045612			CMP RO,#READ	: COMMAND,*			
14659	076202	001014				BNE 10\$				
14660	076204	005205				INC R5				
14661	076206	016766	134460	000034		MOV RPTR,34(SP)	: *,PTR			7624
14662	076214	010346				MOV R3,-(SP)	: LUN,*			7625
14663	076216	010246				MOV R2,-(SP)	: WRDCNT,*			
14664	076220	016746	134446			MOV RPTR,-(SP)				
14665	076224	010146				MOV R1,-(SP)	: SECTOR,*			
14666	076226	004767	147360			JSR PC,READ				
14667	076232	000412				BR 11\$				
14668	076234	016766	134430	000034	10\$:	MOV WPTR,34(SP)	: *,PTR			7629
14669	076242	010346				MOV R3,-(SP)	: LUN,*			7630
14670	076244	010246				MOV R2,-(SP)	: WRDCNT,*			
14671	076246	016746	134416			MOV WPTR,-(SP)				
14672	076252	010146				MOV R1,-(SP)	: SECTOR,*			
14673	076254	004767	147520			JSR PC,CHECK				
14674	076260	010004			11\$:	MOV RO,R4	: *,VALUE			
14675	076262	001076				BNE 15\$				7637
14676	076264	006005				ROR R5				7642
14677	076266	103402				BLO 13\$				
14678	076270	000167	000750	12\$:		JMP 28\$				
14679	076274	016746	134370	13\$:		MOV WPTR,-(SP)				7646
14680	076300	016746	134366			MOV RPTR,-(SP)				
14681	076304	010246				MOV R2,-(SP)	: WRDCNT,*			
14682	076306	004767	146014			JSR PC,DOUBLE.CHECK				
14683	076312	062706	000006			ADD #6,SP				
14684	076316	010066	000050			MOV RO,#50(SP)	: *,DBL.VALUE			
14685	076322	001762				BEQ 12\$				
14686	076324	010346				MOV R3,-(SP)	: LUN,*			7649
14687	076326	004767	137172			JSR PC,SAYWHO				
14688	076332	012716	011216			MOV #MSG5,(SP)				7650
14689	076336	012746	007072			MOV #SAY1,-(SP)				
14690	076342	012746	000002			MOV #2,-(SP)				
14691	076346	010600				MOV SP,RO	: SP,*			

Address	OpCode	Operand1	Operand2	Operand3	Operand4	Instruction	Comments	Count	Address
14693						TRAP	14		27-Mar-1982 19:24:42
14694						MOV	56(SP), (SP)		27-Mar-1982 19:23:44
14695						MOV	@56(SP), -(SP)	: DBL.VALUE,*	TOPS
14696	076350	104414				MOV	#FMT12A, -(SP)	: DBL.VALUE,*	PA: <
14697	076352	016616	000056			MOV	#3, -(SP)		
14698	076356	017646	000056			MOV	SP, R0	: SP,*	
14699	076362	012746	006710			TRAP	14		
14700	076366	012746	000003			ADD	#10000, 64(SP)	: *, DBL.VALUE	7653
14701	076372	010600				MOV	64(SP), (SP)	: DBL.VALUE,*	7654
14702	076374	104414				MOV	@64(SP), -(SP)	: DBL.VALUE,*	
14703	076376	062766	010000	000064		MOV	#FMT12B, -(SP)		
14704	076404	016616	000064			MOV	#3, -(SP)		
14705	076410	017646	000064			MOV	SP, R0	: SP,*	
14706	076414	012746	006756			TRAP	14		
14707	076420	012746	000003			MOVB	#10, WHY.DROPT(R3)	: *,*(LUN)	7656
14708	076424	010600				TRAP	55	:	7657
14709	076426	104414				.WORD	12270		
14710	076430	112763	000010	034446		.WORD	MSG1		
14711	076436	104455				.WORD	0		
14712	076440	012270				MOV	R3, R0	: LUN,*	7658
14713	076442	011070				TRAP	51		
14714	076444	000000				ADD	#46, SP	:	7659
14715	076446	010300				BR	20\$		
14716	076450	104451				CMP	R4, #1	: VALUE,*	7637
14717	076452	062706	000046			BNE	16\$		
14718	076456	000506				MOV	#6, -(SP)	:	7667
14719	076460	020427	000001			MOV	50(SP), -(SP)	: COMMAND,*	
14720	076464	001031				MOV	R3, -(SP)	: LUN,*	
14721	076466	012746	000006			MOV	R2, -(SP)	: WRDCNT,*	
14722	076472	016646	000050			MOV	54(SP), -(SP)	: PTR,*	
14723	076476	010346				MOV	R1, -(SP)	: SECTOR,*	
14724	076500	010246				JSR	PC, RETRY		
14725	076502	016646	000054			ADD	#14, SP		
14726	076506	010146				TST	R0		
14727	076510	004767	147510			BEQ	12\$		
14728	076514	062706	000014			MOVB	#4, WHY.DROPT(R3)	: *,*(LUN)	7670
14729	076520	005700				TRAP	55	:	7671
14730	076522	001662				.WORD	12271		
14731	076524	112763	000004	034446		.WORD	MSG1		
14732	076532	104455				.WORD	0		
14733	076534	012271				MOV	R3, R0	: LUN,*	7672
14734	076536	011070				TRAP	51		
14735	076540	000000				BR	19\$:	7673
14736	076542	010300				CMP	R4, #2	: VALUE,*	7637
14737	076544	104451				BNE	17\$		
14738	076546	000450				MOVB	#5, WHY.DROPT(R3)	: *,*(LUN)	7680
14739	076550	020427	000002			TRAP	55	:	7681
14740	076554	001012				.WORD	12272		
14741	076556	112763	000005	034446		.WORD	MSG1		
14742	076564	104455				.WORD	0		
14743	076566	012272				MOV	R3, R0	: LUN,*	7682
14744	076570	011070				TRAP	51		
14745	076572	000000							
14746	076574	010300							
14747	076576	104451							

				TESTING RANDOM SECTORS, DATA, WORD COUNTS		27-Mar-1982 19:24:42	TOPS
						27-Mar-1982 19:23:44	PA:<
14749				:MLX4			
14750				:			
14751							
14752	076600	000433		BR	19\$		7683
14753	076602	020427	000003	CMP	R4,#3	: VALUE,*	7637
14754	076606	001012		BNE	18\$		
14755	076610	104455		TRAP	55	:	7688
14756	076612	012273		.WORD	12273		
14757	076614	011070		.WORD	MSG1		
14758	076616	000000		.WORD	0		
14759	076620	112763	000006	MOVB	#6,WHY.DROPT(R3)	: *,*(LUN)	7689
14760	076626	010300		MOV	R3,R0	: LUN,*	7690
14761	076630	104451		TRAP	51		
14762	076632	000416		BR	19\$:	7691
14763	076634	020427	000004	CMP	R4,#4	: VALUE,*	7637
14764	076640	001017		BNE	21\$		
14765	076642	004767	140464	JSR	PC,ISOLATE	:	7696
14766	076646	104455		TRAP	55	:	7697
14767	076650	012274		.WORD	12274		
14768	076652	011120		.WORD	MSG2		
14769	076654	000000		.WORD	0		
14770	076656	112763	000007	MOVB	#7,WHY.DROPT(R3)	: *,*(LUN)	7698
14771	076664	010300		MOV	R3,R0	: LUN,*	7699
14772	076666	104451		TRAP	51		
14773	076670	062706	000024	ADD	#24,SP	:	7700
14774	076674	000167	001044	JMP	42\$		
14775	076700	020427	000005	CMP	R4,#5	: VALUE,*	7637
14776	076704	001157		BNE	28\$		
14777	076706	004767	140420	JSR	PC,ISOLATE	:	7705
14778	076712	032767	000001	BIT	#1,ERRRUT	:	7707
14779	076720	001423		BEQ	22\$		
14780	076722	017705	135462	MOV	@ML.REG+42,R5		
14781	076726	006205		ASR	R5		
14782	076730	006205		ASR	R5		
14783	076732	006205		ASR	R5		
14784	076734	006205		ASR	R5		
14785	076736	006205		ASR	R5		
14786	076740	006205		ASR	R5		
14787	076742	042705	177700	BIC	#177700,R5		
14788	076746	010546		MOV	R5,-(SP)		
14789	076750	012746	006640	MOV	#FMT10B,-(SP)		
14790	076754	012746	000002	MOV	#2,-(SP)		
14791	076760	010600		MOV	SP,R0	: SP,*	
14792	076762	104414		TRAP	14		
14793	076764	062706	000006	ADD	#6,SP		
14794	076770	017766	135416	MOV	@ML.REG+44,52(SP)	: *,OLDSEC	7710
14795	076776	017705	135406	MOV	@ML.REG+42,R5	:	7711
14796	077002	006205		ASR	R5		
14797	077004	006205		ASR	R5		
14798	077006	006205		ASR	R5		
14799	077010	006205		ASR	R5		
14800	077012	006205		ASR	R5		
14801	077014	006205		ASR	R5		
14802	077016	042705	177700	BIC	#177700,R5		
14803	077022	010566	000042	MOV	R5,42(SP)	: *,OLDCHN	

14805									27-Mar-1982 19:24:42	TOPS
14806									27-Mar-1982 19:23:44	PA:<
14807										7713
14808	077026	012746	000001							
14809	077032	012746	046000							
14810	077036	010346								
14811	077040	010246								
14812	077042	016746	133622							
14813	077046	010146								
14814	077050	004767	147150							
14815	077054	062706	000014							
14816	077060	020027	000005							
14817	077064	001051								
14818	077066	027766	135320	000052						7716
14819	077074	001034								
14820	077076	016600	000042							
14821	077102	017705	135302							
14822	077106	006205								
14823	077110	006205								
14824	077112	006205								
14825	077114	006205								
14826	077116	006205								
14827	077120	006205								
14828	077122	042705	177700							
14829	077126	020500								
14830	077130	001016								
14831	077132	032767	000001	103120						7720
14832	077140	001404								
14833	077142	104456								
14834	077144	012275								
14835	077146	011202								
14836	077150	000000								
14837	077152	010346		23\$:						7723
14838	077154	016746	135162							
14839	077160	004767	140236							
14840	077164	000426								7716
14841	077166	032767	000001	103064	24\$:					7728
14842	077174	001415								
14843	077176	104457								
14844	077200	012276								
14845	077202	011166								
14846	077204	000000								
14847	077206	000410								7731
14848	077210	032767	000001	103042	25\$:					7737
14849	077216	001404								
14850	077220	104457								
14851	077222	012277								
14852	077224	011166								
14853	077226	000000								
14854	077230	010346		26\$:						7740
14855	077232	016746	135104							
14856	077236	004767	140642							
14857	077242	022626		27\$:						7704
14858	077244	062767	000002	133416	28\$:					7746
-359	077252	062706	000020							7572

Address	OpCode	Operand1	Operand2	Operand3	Label	Instruction	Comments	Time	Page
14861					:MLX4			27-Mar-1982 19:24:42	TOPS
14862					:		TESTING RANDOM SECTORS, DATA, WORD COUNTS	27-Mar-1982 19:23:44	PA:<
14863									7571
14864	077256	005266	000012		29\$:	INC 12(SP)	: KOUNT		
14865	077262	026666	000012	000020		CMP 12(SP),20(SP)	: KOUNT,TIMES		
14866	077270	003002				BGT 30\$			
14867	077272	000167	176376			JMP 4\$			
14868	077275	022626			30\$:	CMP (SP)+,(SP)+	:		7565
14869	077300	032767	000001	102754	31\$:	BIT #1,EFNS21	:		7757
14870	077306	001002				BNE 32\$			
14871	077310	000167	000430			JMP 42\$			
14872	077314	010300			32\$:	MOV R3,R0	: LUN,*		7764
14873	077316	006300				ASL R0			
14874	077320	016066	034500	000010		MOV TOP.SECT(R0),10(SP)			
14875	077326	016005	034460			MOV LOW.SECT(R0),R5	: *,SECTOR		
14876	077332	000577				BR 41\$			
14877	077334	010346			33\$:	MOV R3,-(SP)	: LUN,*		7766
14878	077336	012746	000400			MOV #400,-(SP)			
14879	077342	012746	022670			MOV #RBUF,-(SP)			
14880	077346	010546				MOV R5,-(SP)	: SECTOR,*		
14881	077350	004767	146236			JSR PC,READ			
14882	077354	062706	000010			ADD #10,SP			
14883	077360	020027	000005			CMP R0,#5			
14884	077364	001161				BNE 40\$			
14885	077366	004767	137740			JSR PC,ISOLATE	:		7769
14886	077372	032767	000001	102660		BIT #1,ERROUT	:		7771
14887	077400	001423				BEQ 34\$			
14888	077402	017700	135002			MOV @ML.REG+42,R0			
14889	077406	006200				ASR R0			
14890	077410	006200				ASR R0			
14891	077412	006200				ASR R0			
14892	077414	006200				ASR R0			
14893	077416	006200				ASR R0			
14894	077420	006200				ASR R0			
14895	077422	042700	177700			BIC #177700,R0			
14896	077426	010046				MOV R0,-(SP)			
14897	077430	012746	006640			MOV #FMT10B,-(SP)			
14898	077434	012746	000002			MOV #2,-(SP)			
14899	077440	010600				MOV SP,R0	: SP,*		
14900	077442	104414				TRAP 14			
14901	077444	062706	000006			ADD #6,SP			
14902	077450	017766	134736	000026	34\$:	MOV @ML.REG+44,26(SP)	: *,OLDSEC		7780
14903	077456	017700	134726			MOV @ML.REG+42,R0	:		7781
14904	077462	006200				ASR R0			
14905	077464	006200				ASR R0			
14906	077466	006200				ASR R0			
14907	077470	006200				ASR R0			
14908	077472	006200				ASR R0			
14909	077474	006200				ASR R0			
14910	077476	042700	177700			BIC #177700,R0			
14911	077502	010066	000016			MOV R0,16(SP)	: *,OLDCHN		
14912	077506	012746	000001			MOV #1,-(SP)	:		7789
14913	077512	016646	000024			MOV 24(SP),-(SP)	: COMMAND,*		
14914	077516	010346				MOV R3,-(SP)	: LUN,*		
14915	077520	012746	000400			MOV #400,-(SP)			

Address	Instruction	Operand 1	Operand 2	Operand 3	Operand 4	Comments	Time Stamp	Page
14917						:MLX4	27-Mar-1982 19:24:42	TOPS
14918						:	27-Mar-1982 19:23:44	PA:<
14919								
14920	MOV	077524	016646	000030		30(SP),-(SP) : PTR,*		
14921	MOV	077530	016646	000040		40(SP),-(SP) : OLDSEC,*		
14922	JSR	077534	04767	146464		PC,RETRY		
14923	ADD	077540	J62706	000014		#14,SP		
14924	CMP	077544	020027	000005		RO,#5		
14925	BNE	077550	001051			37\$		
14926	CMP	077552	027766	134634	000026	2ML.REG+44,26(SP) : *,OLDSEC		7798
14927	BNE	077560	001034			36\$		
14928	MOV	077562	016646	000016		16(SP),-(SP) : OLDCHN,*		
14929	MOV	077566	017700	134616		2ML.REG+42,RO		
14930	ASR	077572	006200			RO		
14931	ASR	077574	006200			RO		
14932	ASR	077576	006200			RO		
14933	ASR	077600	006200			RO		
14934	ASR	077602	006200			RO		
14935	ASR	077604	006200			RO		
14936	BIC	077606	042700	177700		#177700,RO		
14937	CMP	077612	020026			RO,(SP)+		
14938	BNE	077614	001016			36\$		
14939	BIT	077616	032767	000001	102434	#1,ERROUT :		7802
14940	BEQ	077624	001404			35\$		
14941	TRAP	077626	104456			56 :		7807
14942	.WORD	077630	012300			12300		
14943	.WORD	077632	011202			MSG4		
14944	.WORD	077634	000000			0		
14945	MOV	077636	010346			R3, -(SP) : LUN,*		7810
14946	MOV	077640	016746	134476		BOARD, -(SP)		
14947	JSR	077644	004767	137552		PC,UP.HARD.COUNT		
14948	BR	077650	000426			39\$:		7798
14949	BIT	077652	032767	000001	102400	#1,ERROUT :		7815
14950	BEQ	077660	001415			38\$:		
14951	TRAP	077662	104457			57 :		7820
14952	.WORD	077664	012301			12301		
14953	.WORD	077666	011166			MSG3		
14954	.WORD	077670	000000			0		
14955	BR	077672	000410			38\$:		7823
14956	BIT	077674	032767	000001	102356	#1,ERROUT :		7829
14957	BEQ	077702	001404			38\$:		
14958	TRAP	077704	104457			57 :		7834
14959	.WORD	077706	012302			12302		
14960	.WORD	077710	011166			MSG3		
14961	.WORD	077712	000000			0		
14962	MOV	077714	010346			R3, -(SP) : LUN,*		7837
14963	MOV	077716	016746	134420		BOARD, -(SP)		
14964	JSR	077722	004767	140156		PC,UP.SOFT.COUNT		
14965	CMP	077726	022626			(SP)+,(SP)+ :		7768
14966	INC	077730	005205			R5 : SECTOR		7764
14967	CMP	077732	020566	000010		R5,10(SP) : SECTOR,*		
14968	BHI	077736	101002			42\$		
14969	JMP	077740	000167	177370		33\$		
14970	INC	077744	005203			R3 : LUN		7557
14971	CMP	077746	020366	000012		R3,12(SP) : LUN,*		

```

14973                                     ;MLX4
14974                                     ;
14975                                     ;
14976 077752 002002                                     BGE 44$
14977 077754 000167 175540                             JMP 2$
14978 077760 005266 000030 44$: INC 30(SP)           : COUNT
14979 077764 026666 000030 000050 CMP 30(SP),50(SP) : COUNT,REPEAT
14980 077772 003002                                     BGT 45$
14981 077774 000167 175504                             JMP 1$
14982 100000 062706 000032 45$: ADD #32,SP           :
14983 100004 000207                                     RTS PC
14984
14985                                     ; Routine Size: 625 words
14986                                     ; Maximum stack depth per invocation: 38 words
14991
14992
  
```

27-Mar-1982 19:24:42 TOPS
 27-Mar-1982 19:23:44 PA:<

7554

7487

14994 :MLX4
14995 :
14996 :
14997 :
14998 :
14999 :
15000 :
15001 :
15002 :
15003 :
15004 :
15005 :
15006 :
15007 :
15008 :
15009 :
15010 :
15011 :
15012 :
15013 :
15014 :
15015 :
15016 :
15017 :
15018 :
15019 :
15020 :
15021 :
15022 :
15023 :
15024 :
15025 :
15026 :
15027 :
15028 :
15029 :
15030 :
15031 :
15032 :
15033 :
15034 :
15035 :
15036 :
15037 :
15038 :
15039 :
15040 :
15041 :
15042 :
15043 :
15044 :
15045 :
15046 :
15047 :
15048 :

```
TESTING RANDOM UNITS, SECTORS, DATA, WORD COUNT
%sbttl 'TESTING RANDOM UNITS, SECTORS, DATA, WORD COUNTS'
routine RAND4 (REPFAT) : novalue =
  begin
    !* 1 * START OF ROUTINE
  ++
  ROUTINE:      RAND4(REPEAT)
  PURPOSE:     TO CHECK RANDOM UNITS
  ARGUMENT:    REPEAT = NUMBER OF TIMES TO EXECUTE ENTIRE ROUTINE
  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)
  --
  local
    LUN,
    WRDCNT,
    SECTOR,
```

27-Mar-1982 19:24:42
27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
PA:<NEALE>MLX4.BLI.5 (47)

```

15050 :MLX4
15051 : TESTING RANDOM UNITS, SECTORS, DATA, WORD COUNT
15052 :
15053 : 7903 VALUE,
15054 : 7904 OLDSEC,
15055 : 7905 OLDCHN,
15056 : 7906 PTR,
15057 : 7907 COMMAND,
15058 : 7908 DBL_VALUE;
15059 : 7909
15060 : 7910 Label
15061 : 7911 LOOP;
15062 : 7912
15063 : 7913 PR:NTB (SAY1, RTN5D);
15064 : 7914 !'RAND4'
15065 : 7915
15066 : 7916 incr COUNT from 1 to .REPEAT do
15067 : 7917 begin !* 2 * START OF REPEAT LOOP FOR THIS ROUTINE
15068 : 7918 GEN5 ();
15069 : 7919
15070 : 7920 incr KOUNT from 1 to (.NUM_DRIVES*4) do
15071 : 7921 begin !* 3 * START OF COUNTING LOOP FOR UNIT SELECTION
15072 : 7922 LUN = RNDU ();
15073 : 7923 L$LUN = .LUN;
15074 : 7924 WPTR = WBUFF;
15075 : 7925 RPTR = RBUFF;
15076 : 7926 LOOP :
15077 : 7927 begin !* 4 * START OF LOOP THAT COMPLETELY TESTS 1 UNIT
15078 : 7928
15079 : 7929 incr KOUNT2 from 1 to 10 do
15080 : 7930 begin !* 5 * START OF COUNTING LOOP FOR SECTOR SELECTION
15081 : 7931 WRDCNT = RNDWC (); !EXPECT VALUE BETWEEN 1 AND BUFSIZ
15082 : 7932 SET PTRS (.WRDCNT);
15083 : 7933 SECTOR = RNDSEC (.LUN); !EXPECT VALUE BETWEEN LOWEST AND HIGHEST
15084 : 7934
15085 : 7935 if (.SECTOR + .WRDCNT/256) gtra HIGHEST then WRDCNT = 256*(HIGHEST - .SECTOR + 1);
15086 : 7936
15087 : 7937 VALUE = write (.LUN, .WRDCNT, .WPTR, .SECTOR);
15088 : 7938
15089 : 7939 !+
15090 : 7940 ! SEE HOW SUCCESSFUL THE WRITE WAS:
15091 : 7941 !-
15092 : 7942
15093 : 7943 selectone .VALUE of !SEE 'SYSERR' FOR DEFINITION
15094 : 7944 set !OF ERROR # CONTAINED IN 'VALUE'
15095 : 7945
15096 : 7946 [1] :
15097 : 7947 begin !* 5A * RETRY ALLOWED
15098 : 7948
15099 : 7949 if RETRY (SIX, write, .LUN, .WRDCNT, .WPTR, .SECTOR) neq 0
15100 : 7950 then !THE RETRY FAILED -- SYSTEM FATAL ERROR
15101 : 7951 begin
15102 : 7952 WHY DROPT [.LUN] = CODE_4;
15103 : 7953 ERRDF (5401, MSG1, 0); !**** OPTION 5, RAND4 ERROR 01 ****
15104 : 7954 DODU (.LUN);

```

```

15106 :MLX4
15107 :
15108 :
15109 :          7955          Leave LOOP;          !JUMP JUST BEYOND END OF BLOCK * 4 *
15110 :          7956          end;
15111 :          7957
15112 :          7958          end;          !* 5A *
15113 :          7959
15114 :          7960          [2] :
15115 :          7961          begin          !* 5B * FATAL CONTROLLER ERROR -- NO RETRY ALLOWED
15116 :          7962          WHY DROPT [.LUN] = CODE_5;
15117 :          7963          ERRDF (5402, MSG1, 0); !**** OPTION 5, RAND4 ERROR 02 ****
15118 :          7964          DODU (.LUN);
15119 :          7965          leave LOOP;          !JUMP JUST BEYOND END OF BLOCK * 4 *
15120 :          7966          end;          !* 5B *
15121 :          7967
15122 :          7968          [3] :
15123 :          7969          begin          !* 5C * FATAL DRIVE ERROR -- NO RETRY ALLOWED
15124 :          7970          ERRDF (5403, MSG1, 0); !**** OPTION 5, RAND4 ERROR 03 ****
15125 :          7971          WHY DROPT [.LUN] = CODE_6;
15126 :          7972          DODU (.LUN);
15127 :          7973          leave LOOP;          !JUMP JUST BEYOND END OF BLOCK * 4 *
15128 :          7974          end;          !* 5C *
15129 :          7975          tes;
15130 :          7976
15131 :          7977          COMMAND = CHOOSE ();
15132 :          7978
15133 :          7979          if .COMMAND eql read
15134 :          7980          then
15135 :          7981          begin
15136 :          7982          PTR = .RPTR;
15137 :          7983          VALUE = read (.LUN, .WRDCNT, .RPTR, .SECTOR);
15138 :          7984          end
15139 :          7985          else
15140 :          7986          begin
15141 :          7987          PTR = .WPTR;
15142 :          7988          VALUE = CHECK (.LUN, .WRDCNT, .WPTR, .SECTOR);
15143 :          7989          end;
15144 :          7990
15145 :          7991          !+
15146 :          7992          ! SEE HOW SUCCESSFUL THE OPERATION WAS:
15147 :          7993          !-
15148 :          7994
15149 :          7995          selectone .VALUE of          !SEE 'SYSERR' FOR DEFINITION
15150 :          7996          set          !OF ERROR # CONTAINED IN 'VALUE'
15151 :          7997
15152 :          7998          [0] :
15153 :          7999
15154 :          8000          if .COMMAND eql read
15155 :          8001          then
15156 :          8002          begin
15157 :          8003
15158 :          8004          if (DBL_VALUE = DOUBLE_CHECK (.WPTR, .RPTR, .WRDCNT)) neq 0
15159 :          8005          then
15160 :          8006          begin
    
```

```

15162 :MLX4
15163 :
15164 :
15165 :      8007
15166 :      8008
15167 :      8009
15168 :      8010
15169 :      8011
15170 :      8012
15171 :      8013
15172 :      8014
15173 :      8015
15174 :      8016
15175 :      8017
15176 :      8018
15177 :      8019
15178 :      8020
15179 :      8021
15180 :      8022
15181 :      8023
15182 :      8024
15183 :      8025
15184 :      8026
15185 :      8027
15186 :      8028
15187 :      8029
15188 :      8030
15189 :      8031
15190 :      8032
15191 :      8033
15192 :      8034
15193 :      8035
15194 :      8036
15195 :      8037
15196 :      8038
15197 :      8039
15198 :      8040
15199 :      8041
15200 :      8042
15201 :      8043
15202 :      8044
15203 :      8045
15204 :      8046
15205 :      8047
15206 :      8048
15207 :      8049
15208 :      8050
15209 :      8051
15210 :      8052
15211 :      8053
15212 :      8054
15213 :      8055
15214 :      8056
15215 :      8057
15216 :      8058

                27-Mar-1982 19:24:42      TOPS-20 Bliss-16 V2(212)
                27-Mar-1982 19:23:44      PA:<NEALE>MLX4.BLI.5 (47)

                TESTING RANDOM UNITS, SECTORS, DATA, WORD COUNT

                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 (5404, MSG1, 0); !**** OPTION 5, RAND4 ERROR 04 ****
                DODU (.LUN);
                leave LOOP;      !JUMP JUST BEYOND END OF BLOCK * 4 *
                end;

                end;

[1] :
    begin
        !* 5D * 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 (5405, MSG1, 0); !**** OPTION 5, RAND4 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 (5406, MSG1, 0); !**** OPTION 5, RAND4 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 (5407, MSG1, 0); !**** OPTION 5, RAND4 ERRGR 07 ****
        WHY DROPT [.LUN] = CODE_6;
        DODD (.LUN);
        leave LOOP;      !JUMP JUST BEYOND END OF BLOCK * 4 *
    end;
    !* 5F *

[4] :
    begin
        !* 5G * UNRECOVERABLE DATA ERROR
        ISOLATE ();
        ERRDF (5408, MSG2, 0); !**** OPTION 5, RAND4 ERROR 08 ****
        WHY DROPT [.LUN] = CODE_7;
        DODD (.LUN);
    end;
    
```



```

15218 :MLX4
15219 :
15220 :
15221 :      8059
15222 :      8060
15223 :      8061
15224 :      8062
15225 :      8063
15226 :      8064
15227 :      8065
15228 :      8066
15229 :      8067
15230 :      8068
15231 :      8069
15232 :      8070
15233 :      8071
15234 :      8072
15235 :      8073
15236 :      8074
15237 :      8075
15238 :      8076
15239 :      8077
15240 :      8078
15241 :      8079
15242 :      8080
15243 :      8081
15244 :      8082
15245 :      8083
15246 :      8084
15247 :      8085
15248 :      8086
15249 :      8087
15250 :      8088
15251 :      8089
15252 :      8090
15253 :      8091
15254 :      8092
15255 :      8093
15256 :      8094
15257 :      8095
15258 :      8096
15259 :      8097
15260 :      8098
15261 :      8099
15262 :      8100
15263 :      8101
15264 :      8102
15265 :      8103
15266 :      8104
15267 :      8105
15268 :      8106
15269 :      8107
15270 :      8108
15271 :      8109
15272 :      8110

TESTING RANDOM UNITS, SECTORS, DATA, WORD COUNT

                Leave LOOP;
                end;
                [* 5G *

[5] :
begin
ISOLATE ();
                [* 5H * RECOVERABLE DATA ERROR

if .ERROUT then PRINTB (FMT10B, .CHAN);

!' BIT 00'
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 (5409, MSG4, 0); !**** OPTION 5, RAND4 ERROR 09 ****
            UP_HARD_COUNT (.LUN, .BOARD);
        end
    else
        begin
            if .ERROUT then ERRSOFT (5410, MSG3, 0);
            UP_SOFT_COUNT (.LUN, .BOARD);
        end
    else
        begin
            if .ERROUT then ERRSOFT (5411, MSG3, 0); !**** OPTION 5, RAND4 ERROR 11 ****
            UP_SOFT_COUNT (.LUN, .BOARD);
        end
    end;
end;
                [* 5H *

tes;
WPTR = .WPTR + 2;
end;
                [* 5 * END OF COUNTING LOOP FOR SECTOR SELECTION

!+
! Test to see if this uut's address space is
! to be read for soft errors. This test is
! is intended for DMT purposes.
!-

```

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (47)

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (47)

```

15274 :MLX4
15275 :
15276 :
15277 :      8111
15278 :      8112          if .EFNS$21          !Is the background pattern to be read
15279 :      8113          then
15280 :      8114          begin
15281 :      8115
15282 :      8116
15283 :      8117 : version czmlbb changed incr to incru
15284 :      8118
15285 :      8119          incru SECTOR from LOWEST to HIGHEST do
15286 :      8120
15287 :      8121          if read (.LUN, 256, RBUFF, .SECTOR) eql 5
15288 :      8122          then
15289 :      8123          begin
15290 :      8124          ISOLATE ();          !Find the failing bank and board no.
15291 :      8125
15292 :      8126          if .ERROUT then PRINTB (FMT10B, .CHAN);
15293 :      8127
15294 :      8128          : Print where the error is
15295 :      8129
15296 :      8130          : Save the contents of the ML error location
15297 :      8131          : register so we can compare it to the new
15298 :      8132          : contents of this register after the retry.
15299 :      8133          : This is done to classify the error.
15300 :      8134
15301 :      8135          OLDSEC = .MLEL;
15302 :      8136          OLDCHN = .CHAN;
15303 :      8137
15304 :      8138          : Do a classify retry call. If the same error
15305 :      8139          : occurs then classify it as a hard error. If
15306 :      8140          : a different error occurred or the error went away
15307 :      8141          : then classify it as a soft error.
15308 :      8142
15309 :      8143
15310 :      8144          if RETRY (ONE, .COMMAND, .LUN, 256, .PTR, .OLDSEC) eql 5
15311 :      8145          then
15312 :      8146
15313 :      8147          : The same error occurred so see if it is at the same
15314 :      8148          : sector and channel number, if so then classify
15315 :      8149          : it as a hard error else classify it as a soft
15316 :      8150          : error.
15317 :      8151
15318 :      8152
15319 :      8153          if ((.MLEL eql .OLDSEC) and (.CHAN eql .OLDCHN))
15320 :      8154          then
15321 :      8155          begin          !Same error occurred 'hard'
15322 :      8156
15323 :      8157          if .ERROUT          !Print error if enabled
15324 :      8158          then
15325 :      8159          begin
15326 :      8160          ERRHRD (5412,          !Error number
15327 :      8161          MSG4,          !Error message
15328 :      8162          0);          !Additional message routine
  
```

27-Mar-1982 19:24:42
27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
PA:<NEALE>MLX4.BLI.5 (47)

```
15330 :MLX4
15331 : TESTING RANDOM UNITS, SECTORS, DATA, WORD COUNT
15332 :
15333 :         8163
15334 :         8164
15335 :         8165
15336 :         8166
15337 :         8167
15338 :         8168
15339 :         8169
15340 :         8170
15341 :         8171
15342 :         8172
15343 : P 8173
15344 : P 8174
15345 :         8175
15346 :         8176
15347 :         8177
15348 :         8178
15349 :         8179
15350 :         8180
15351 :         8181
15352 :         8182
15353 :         8183
15354 :         8184
15355 :         8185
15356 : P 8186
15357 : P 8187
15358 : P 8188
15359 :         8189
15360 :         8190
15361 :         8191
15362 :         8192
15363 :         8193
15364 :         8194
15365 :         8195
15366 :         8196
15367 :         8197
15368 :         8198
15369 :         8199
15370 :         8200
15371 :         8201
15372 :         8202
15373 :         8203
15374 :         8204
15375 :         8205
15379 :
15380 :
15384 : 100006 004167 105322

end;
UP_HARD_COUNT (.LUN, .BOARD);
end;
else
begin
!Not the same error 'soft'
if .ERROUT
!Print error if enabled
then
begin
ERRSOFT (5413, !Error number
MSG3, !Error message
0); !Additional message routine
end;
UP_SOFT_COUNT (.LUN, .BOARD);
end;
else
begin
!Not the same error 'soft'
if .ERROUT
!Print error if enabled
then
begin
ERRSOFT (5414, !Error number
MSG3, !Error message
0); !Additional message routine
end;
UP_SOFT_COUNT (.LUN, .BOARD);
end;
end;
end;
end;
!* 4 * END OF LOOP THAT COMPLETELY TESTS 1 UNIT
!* 3 * END OF COUNTING LOOP FOR UNIT SELECTION
end;
!* 2 * END OF REPEAT LOOP FOR THIS ROUTINE
return;
end;
!* 1 * END OF ROUTINE

.SBTTL RAND4 TESTING RANDOM UNITS, SECTORS, DATA, WORD COUNT
RAND4: JSR R1,$SAVE5 ;
```

Address	Offset	Value	Label	Instruction	Comment	Count
15386			:MLX4			
15387			:		TESTING RANDOM UNITS, SECTORS, DATA, WORD COUNT	
15388						
15389	100012	162706		SUB	#26,SP	
15390	100016	012746		MOV	#RTN5D,-(SP)	7913
15391	100022	012746		MOV	#SAY1,-(SP)	
15392	100026	012746		MOV	#2,-(SP)	
15393	100032	010600		MOV	SP,R0	: SP,*
15394	100034	104414		TRAP	14	
15395	100036	005066		CLR	32(SP)	: COUNT
15396	100042	000167		JMP	41\$	
15397	100046	004767	141546	JSR	PC,GEN5	7918
15398	100052	016766	134400	MOV	NUM.DRIVES,16(SP)	7920
15399	100060	006366	000016	ASL	16(SP)	
15400	100064	006366	000016	ASL	16(SP)	
15401	100070	005066	000014	CLR	14(SP)	: KOUNT
15402	100074	000563		BR	8\$	
15403	100076	004767	170234	JSR	PC,RNDU	7922
15404	100102	010002		MOV	R0,R2	: *,LUN
15405	100104	010267	101764	MOV	R2,L\$LUN	: LUN,*
15406	100110	012767	012670	MOV	#WBUF,WPTR	7924
15407	100116	012767	022670	MOV	#RBUF,RPTR	7925
15408	100124	010266	000012	MOV	R2,12(SP)	: LUN,*
15409	100130	006366	000012	ASL	12(SP)	
15410	100134	012766	034500	MOV	#TOP.SECT,10(SP)	
15411	100142	066666	000012	ADD	12(SP),10(SP)	
15412	100150	012766	000001	MOV	#1,6(SP)	: *,KOUNT2
15413	100156	004767	170022	JSR	PC,RNDWC	7931
15414	100162	010003		MOV	R0,R3	: *,WRDCNT
15415	100164	010346		MOV	R3,-(SP)	: WRDCNT,*
15416	100166	004767	146740	JSR	PC,SET.PTRS	
15417	100172	010216		MOV	R2,(SP)	: LUN,*
15418	100174	004767	170040	JSR	PC,RNDSEC	
15419	100200	010001		MOV	R0,R1	: *,SECTOR
15420	100202	010346		MOV	R3,-(SP)	: WRDCNT,*
15421	100204	012746	000400	MOV	#400,-(SP)	
15422	100210	004767	104754	JSR	PC,BL\$DIV	
15423	100214	022626		CMP	(SP)+,(SP)+	
15424	100216	060100		ADD	R1,R0	: SECTOR,*
15425	100220	020076	000012	CMP	R0,@12(SP)	
15426	100224	101410		BLOS	4\$	
15427	100226	017600	0C0012	MOV	@12(SP),R0	: SECTOR,*
15428	100232	160100		SUB	R1,R0	
15429	100234	000300		SWAB	R0	
15430	100236	105000		CLRB	R0	
15431	100240	010003		MOV	R0,R3	: *,WRDCNT
15432	100242	062703	000400	ADD	#400,R3	: *,WRDCNT
15433	100246	010216		MOV	R2,(SP)	: LUN,*
15434	100250	010346		MOV	R3,-(SP)	: WRDCNT,*
15435	100252	016746	132412	MOV	WPTR,-(SP)	
15436	100256	010146		MOV	R1,-(SP)	: SECTOR,*
15437	100260	004767	145140	JSR	PC,WRITE	
15438	100264	010004		MOV	R0,R4	: *,VALUE
15439	100266	020427	000001	CMP	R4,#1	: VALUE,*
15440	100272	001031		BNE	5\$	7943

Address	Offset	Value	Label	Instruction	Comments	Time	Page
15442						27-Mar-1982 19:24:42	TOPS
15443						27-Mar-1982 19:23:44	PA:<
15444							7949
15445	100274	012746	000006	MOV	#6,-(SP)		
15446	100300	012746	045424	MOV	#WRITE,-(SP)		
15447	100304	010246		MOV	R2,-(SP)	: LUN,*	
15448	100306	010346		MOV	R3,-(SP)	: WRDCNT,*	
15449	100310	016746	132354	MOV	WPTR,-(SP)		
15450	100314	010146		MOV	R1,-(SP)	: SECTOR,*	
15451	100316	004767	145702	JSR	PC,RETRY		
15452	100322	062706	000014	ADD	#14,SP		
15453	100326	005700		TST	R0		
15454	100330	001446		BEQ	9\$		
15455	100332	112762	000004 034446	MOVB	#4,WHY.DROPT(R2)	: *,*(LUN)	7952
15456	100340	104455		TRAP	55	:	7953
15457	100342	012431		.WORD	12431		
15458	100344	011070		.WORD	MSG1		
15459	100346	000000		.WORD	0		
15460	100350	010200		MOV	R2,R0	: LUN,*	7954
15461	100352	104451		TRAP	51		
15462	100354	000431		BR	7\$:	7955
15463	100356	020427	000002 5\$:	CMP	R4,#2	: VALUE,*	7943
15464	100362	001012		BNE	6\$		
15465	100364	112762	000005 034446	MOVB	#5,WHY.DROPT(R2)	: *,*(LUN)	7962
15466	100372	104455		TRAP	55	:	7963
15467	100374	012432		.WORD	12432		
15468	100376	011070		.WORD	MSG1		
15469	100400	000000		.WORD	0		
15470	100402	010200		MOV	R2,R0	: LUN,*	7964
15471	100404	104451		TRAP	51		
15472	100406	000414		BR	7\$:	7965
15473	100410	020427	000003 6\$:	CMP	R4,#3	: VALUE,*	7943
15474	100414	001014		BNE	9\$		
15475	100416	104455		TRAP	55	:	7970
15476	100420	012433		.WORD	12433		
15477	100422	011070		.WORD	MSG1		
15478	100424	000000		.WORD	0		
15479	100426	112762	000006 034446	MOVB	#6,WHY.DROPT(R2)	: *,*(LUN)	7971
15480	100434	010200		MOV	R2,R0	: LUN,*	7972
15481	100436	104451		TRAP	51		
15482	100440	062706	000010 7\$:	ADD	#10,SP	:	7973
15483	100444	000535		BR	14\$		
15484	100446	004767	145514 8\$:	JSR	PC,CHOOSE	:	7977
15485	100452	010066	000040 9\$:	MOV	R0,40(SP)	: *,COMMAND	
15486	100456	005005		CLR	R5	:	7979
15487	100460	020027	045612	CMP	R0,#READ	: COMMAND,*	
15488	100464	001014		BNE	10\$		
15489	100466	005205		INC	R5		
15490	100470	016766	132176 000036	MOV	RPTR,36(SP)	: *,PTR	7982
15491	100476	010246		MOV	R2,-(SP)	: LUN,*	7983
15492	100500	010346		MOV	R3,-(SP)	: WRDCNT,*	
15493	100502	016746	132164	MOV	RPTR,-(SP)		
15494	100506	010146		MOV	R1,-(SP)	: SECTOR,*	
15495	100510	004767	145076	JSR	PC,READ		
15496	100514	000412		BR	11\$		

					27-Mar-1982 19:24:42	TOPS
					27-Mar-1982 19:23:44	PA:<
15498				:MLX4		
15499				:	TESTING RANDOM UNITS, SECTORS, DATA, WORD COUNT	
15500				:		
15501	100516	016766	132146	000036 10\$:	MOV WPTR,36(SP)	: *,PTR 7987
15502	100524	010246			MOV R2,-(SP)	: LUN,* 7988
15503	100526	010346			MOV R3,-(SP)	: WRDCNT,*
15504	100530	016746	132134		MOV WPTR,-(SP)	
15505	100534	010176			MOV R1,-(SP)	: SECTOR,*
15506	100536	004767	145236		JSR PC,CHECK	
15507	100542	010004		11\$:	MOV R0,R4	: *,VALUE
15508	100544	001076			BNE 15\$: 7995
15509	100546	006005			ROR R5	: 8000
15510	100550	103402			BLO 13\$	
15511	100552	000167	000750	12\$:	JMP 28\$	
15512	100556	016746	132106	13\$:	MOV WPTR,-(SP)	: 8004
15513	100562	016746	132104		MOV RPTR,-(SP)	
15514	100566	010346			MOV R3,-(SP)	: WRDCNT,*
15515	100570	004767	143532		JSR PC,DOUBLE.CHECK	
15516	100574	062706	000006		ADD #6,SP	
15517	100600	010066	000042		MOV R0,42(SP)	: *,DBL.VALUE
15518	100604	001762			BEQ 12\$	
15519	100606	010246			MOV R2,-(SP)	: LUN,* 8007
15520	100610	004767	134710		JSR PC,SAYWHO	
15521	100614	012716	011216		MOV #MSG5,(SP)	: 8008
15522	100620	012746	007072		MOV #SAY1,-(SP)	
15523	100624	012746	000002		MOV #2,-(SP)	
15524	100630	010600			MOV SP,R0	: SP,*
15525	100632	104414			TRAP 14	
15526	100634	016616	000050		MOV 50(SP),(SP)	: DBL.VALUE,* 8010
15527	100640	017646	000050		MOV @50(SP),-(SP)	: DBL.VALUE,*
15528	100644	012746	006710		MOV #FMT12A,-(SP)	
15529	100650	012746	000003		MOV #3,-(SP)	
15530	100654	010600			MOV SP,R0	: SP,*
15531	100656	104414			TRAP 14	
15532	100660	062766	010000	000056	ADD #10000,56(SP)	: *,DBL.VALUE 8012
15533	100666	016616	000056		MOV 56(SP),(SP)	: DBL.VALUE,* 8013
15534	100672	017646	000056		MOV @56(SP),-(SP)	: DBL.VALUE,*
15535	100676	012746	006756		MOV #FMT12B,-(SP)	
15536	100702	012746	000003		MOV #3,-(SP)	
15537	100706	010000			MOV SP,R0	: SP,*
15538	100710	104414			TRAP 14	
15539	100712	112762	000010	034446	MOVB #10,WHY.DROPT(R2)	: *,*(LUN) 8015
15540	100720	104455			TRAP 55	: 8016
15541	100722	012434			.WORD 12434	
15542	100724	011070			.WORD MSG1	
15543	100726	000000			.WORD 0	
15544	100730	010200			MOV R2,R0	: LUN,* 8017
15545	100732	104451			TRAP 51	
15546	100734	062706	000042		ADD #42,SP	: 8018
15547	100740	000506		14\$:	BR 20\$	
15548	100742	020427	000001	15\$:	CMP R4,#1	: VALUE,* 7995
15549	100746	001031			BNE 16\$	
15550	100750	012746	000006		MOV #6,-(SP)	: 8026
15551	100754	016646	003052		MOV 52(SP),-(SP)	: COMMAND,*
15552	100760	010246			MOV R2,-(SP)	: LUN,*

Address	Offset	Count	Label	Instruction	Comment	Date/Time	Page
15554			:MLX4			27-Mar-1982 19:24:42	TOPS
15555			:	TESTING RANDOM UNITS, SECTORS, DATA, WORD COUNT		27-Mar-1982 19:23:44	PA:<
15556							
15557	100762	010346		MOV R3,-(SP)	: WRDCNT,*		
15558	100764	016646	000056	MOV 56(SP),-(SP)	: PTR,*		
15559	100770	010146		MOV R1,-(SP)	: SECTOR,*		
15560	100772	004767	145226	JSR PC,RETRY			
15561	100776	062706	000014	ADD #14,SP			
15562	101002	005700		TST R0			
15563	101004	001662		BEQ 12\$			
15564	101006	112762	000004	MOVB #4,WHY.DROPT(R2)	: *,*(LUN)		8029
15565	101014	104455		TRAP 55	:		8030
15566	101016	012435		.WORD 12435			
15567	101020	011070		.WORD MSG1			
15568	101022	000000		.WORD 0			
15569	101024	010200		MOV R2,R0	: LUN,*		8031
15570	101026	104451		TRAP 51			
15571	101030	000450		BR 19\$:		8032
15572	101032	020427	000002	16\$: CMP R4,#2	: VALUE,*		7995
15573	101036	001012		BNE 17\$			
15574	101040	112762	000005	MOVB #5,WHY.DROPT(R2)	: *,*(LUN)		8039
15575	101046	104455		TRAP 55	:		8040
15576	101050	012436		.WORD 12436			
15577	101052	011070		.WORD MSG1			
15578	101054	000000		.WORD 0			
15579	101056	010200		MOV R2,R0	: LUN,*		8041
15580	101060	104451		TRAP 51			
15581	101062	000433		BR 19\$:		8042
15582	101064	020427	000003	17\$: CMP R4,#3	: VALUE,*		7995
15583	101070	001012		BNE 18\$			
15584	101072	104455		TRAP 55	:		8047
15585	101074	012437		.WORD 12437			
15586	101076	011070		.WORD MSG1			
15587	101100	000000		.WORD 0			
15588	101102	112762	000006	MOVB #6,WHY.DROPT(R2)	: *,*(LUN)		8048
15589	101110	010200		MOV R2,R0	: LUN,*		8049
15590	101112	104451		TRAP 51			
15591	101114	000416		BR 19\$:		8050
15592	101116	020427	000004	18\$: CMP R4,#4	: VALUE,*		7995
15593	101122	001017		BNE 21\$			
15594	101124	004767	136202	JSR PC,ISOLATE	:		8055
15595	101130	104455		TRAP 55	:		8056
15596	101132	012440		.WORD 12440			
15597	101134	011120		.WORD MSG2			
15598	101136	000000		.WORD 0			
15599	101140	112762	000007	MOVB #7,WHY.DROPT(R2)	: *,*(LUN)		8057
15600	101146	010200		MOV R2,R0	: LUN,*		8058
15601	101150	104451		TRAP 51			
15602	101152	062706	000020	19\$: ADD #20,SP	:		8059
15603	101156	000167	001042	20\$: JMP 40\$			
15604	101162	020427	000005	21\$: CMP R4,#5	: VALUE,*		7995
15605	101166	001157		BNE 28\$			
15606	101170	004767	136136	JSR PC,ISOLATE	:		8064
15607	101174	032767	000001	BIT #1,ERRROUT	:		8066
15608	101202	001423		BEQ 22\$:		

Address	Offset	Count	Label	Instruction	Comment	Time	Page
15666						27-Mar-1982 19:24:42	TOPS
15667						27-Mar-1982 19:23:44	PA:<
15668							
15669	101432	000000		.WORD	0		
15670	101434	010246	23\$:	MOV	R2,-(SP)		8081
15671	101436	016746		MOV	BOARD,-(SP)		
15672	101442	004767	132700	JSR	PC,UP.HARD.COUNT		
15673	101446	000426		BR	27\$		8075
15674	101450	032767	000001 100602	24\$:	BIT	#1,ERROUT	8086
15675	101456	001415		BEQ	26\$		
15676	101460	104457		TRAP	57		
15677	101462	012442		.WORD	12442		
15678	101464	011166		.WORD	MSG3		
15679	101466	030000		.WORD	0		
15680	101470	000410		BR	26\$		8089
15681	101472	032767	000001 100560	25\$:	BIT	#1,ERROUT	8095
15682	101500	001404		BEQ	26\$		
15683	101502	104457		TRAP	57		
15684	101504	012443		.WORD	12443		
15685	101506	011166		.WORD	MSG3		
15686	101510	000000		.WORD	0		
15687	101512	010246		26\$:	MOV	R2,-(SP)	8097
15688	101514	016746	132622	MOV	BOARD,-(SP)		
15689	101520	004767	136360	JSR	PC,UP.SOFT.COUNT		
15690	101524	022626		27\$:	CMP	(SP)+,(SP)+	8063
15691	101526	062767	000002 131134	28\$:	ADD	#2,WPTR	8103
15692	101534	062706	000020	ADD	#20,SP		7930
15693	101540	005266	000006	INC	6(SP)		7929
15694	101544	026627	000006 000012	CMP	6(SP),#12		
15695	101552	003002		BGT	29\$		
15696	101554	000167	176376	JMP	3\$		
15697	101560	032767	000001 100474	29\$:	BIT	#1,EFN\$21	8112
15698	101566	001002		BNE	30\$		
15699	101570	000167	000430	JMP	40\$		
15700	101574	017666	000010 000010	30\$:	MOV	@10(SP),10(SP)	8119
15701	101602	016600	000012	MOV	12(SP),R0		
15702	101606	016005	034460	MOV	LOW.SECT(R0),R5		
15703	101612	000577		BR	39\$		
15704	101614	010246		31\$:	MOV	R2,-(SP)	8121
15705	101616	012746	000400	MOV	#400,-(SP)		
15706	101622	012746	022670	MOV	#RBUF,-(SP)		
15707	101626	010546		MOV	R5,-(SP)		
15708	101630	004767	143756	JSR	PC,READ		
15709	101634	062706	000010	ADD	#10,SP		
15710	101640	020027	000005	CMP	R0,#5		
15711	101644	001161		BNE	38\$		
15712	101646	004767	135460	JSR	PC,ISOLATE		8124
15713	101652	032767	000001 100400	BIT	#1,ERROUT		8126
15714	101660	001423		BEQ	32\$		
15715	101662	017700	132522	MOV	@ML.REG+42,R0		
15716	101666	006200		ASR	R0		
15717	101670	006200		ASR	R0		
15718	101672	006200		ASR	R0		
15719	101674	006200		ASR	R0		
15720	101676	006200		ASR	R0		

Address	Offset	Count	Label	Instruction	Comment	Time	Page
15722			:MLX4			27-Mar-1982 19:24:42	TOPS
15723			:		TESTING RANDOM UNITS, SECTORS, DATA, WORD COUNT	27-Mar-1982 19:23:44	PA:<
15724							
15725	101700	006200		ASR	R0		
15726	101702	042700	177700	BIC	#177700,R0		
15727	101706	010046		MOV	R0,-(SP)		
15728	101710	012746	006640	MOV	#FMT10B,-(SP)		
15729	101714	012746	000002	MOV	#2,-(SP)		
15730	101720	010600		MOV	SP,R0	: SP,*	
15731	101722	1044:		TRAP	14		
15732	101724	062706	000006	ADD	46,SP		
15733	101730	017766	132456	MOV	@ML.REG+44,24(SP)	: *,OLDSEC	8135
15734	101736	01770C	132446	MOV	@ML.REG+42,R0	:	8136
15735	101742	006200		ASR	R0		
15736	101744	006200		ASR	R0		
15737	101746	006200		ASR	R0		
15738	101750	006200		ASR	R0		
15739	101752	006200		ASR	R0		
15740	101754	006200		ASR	R0		
15741	101756	042700	177700	BIC	#177700,R0		
15742	101762	010066	000020	MOV	R0,20:SP	: *,OLDCHN	
15743	101766	012746	000001	MOV	#1,-(SP)	:	8144
15744	101772	016646	000032	MOV	32(SP),-(SP)	: COMMAND,*	
15745	101776	010246		MOV	R2,-(SP)	: LUN,*	
15746	102000	012746	000400	MOV	#400,-(SP)		
15747	102004	016646	000036	MOV	36(SP),-(SP)	: PTR,*	
15748	102010	016646	000036	MOV	36(SP),-(SP)	: OLDSEC,*	
15749	102014	004767	144204	JSR	PC,RETRY		
15750	102020	062706	000014	ADD	#14,SP		
15751	102024	020027	000905	CMP	R0,#5		
15752	102030	001051		BNE	35\$		
15753	102032	027766	132354	CMP	@ML.REG+44,24(SP)	: *,OLDSEC	8153
15754	102040	001034		BNE	34\$		
15755	102042	016646	000020	MOV	20(SP),-(SP)	: OLDCHN,*	
15756	102046	017700	132336	MOV	@ML.REG+42,R0		
15757	102052	006200		ASR	R0		
15758	102054	006200		ASR	R0		
15759	102056	006200		ASR	R0		
15760	102060	006200		ASR	R0		
15761	102062	006200		ASR	R0		
15762	102064	006200		ASR	R0		
15763	102066	042700	177700	BIC	#177700,R0		
15764	102072	020026		CMP	R0,(SP)+		
15765	102074	001016		BNE	34\$		
15766	102076	032767	000001	BIT	#1,ERROUT	:	8157
15767	102104	001404		BEQ	33\$:	8162
15768	102106	104456		TRAP	56		
15769	102110	012444		.WORD	12444		
15770	102112	011202		.WORD	MSG4		
15771	102114	000000		.WORD	0		
15772	102116	010246		MOV	R2,-(SP)	: LUN,*	8165
15773	102120	016746	132216	MOV	BOARD,-(SP)		
15774	102124	004767	135272	JSR	PC,UP.HARD.COUNT		
15775	102130	000426		BR	37\$:	8153
15776	102132	032767	000001	BIT	#1,ERROUT	:	8170

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (48)

```

15821 :MLX4
15822 :
15823 :
15824 :      8206 %sbttl 'DEFINITION OF OPTION 5'
15825 :      8207 routine OPT5 : novalue =
15826 :      8208         begin
15827 :      8209
15828 :      8210
15829 :      8211 :++ ROUTINE:      OPT5
15830 :      8212
15831 :      8213 : PURPOSE:      TO EXERCISE THE ML11 SYSTEMS UNDER TEST IN A RANDOM
15832 :      8214 :                MANNER, SO AS TO SIMULATE THE FLEXIBILITY OF TESTING
15833 :      8215 :                THAT WOULD BE DONE BY AN OPERATING SYSTEM.
15834 :      8216
15835 :      8217 :                THERE ARE 4 RANDOM TESTS WHICH ARE CALLED BY 'OPT5'
15836 :      8218 :                TO ACCOMPLISH ALL TESTING. IT IS THE RESPONSIBILITY
15837 :      8219 :                OF THIS ROUTINE TO DECIDE HOW MANY TIMES THOSE 4
15838 :      8220 :                RANDOM TESTS WILL BE EXECUTED. REFER TO 'RAND1' TO
15839 :      8221 :                'RAND4' ABOVE FOR MORE INFORMATION.
15840 :      8222 :--
15841 :      8223
15842 :      8224 local
15843 :      8225 REPEAT;
15844 :      8226
15845 :      8227 PRINTB (SAY2, WRD34, RTN5);
15846 :      8228 !'RUNNING OPT5'
15847 :      8229 REPEAT = 9 - .NUM_DRIVES;
15848 :      8230 !FEWER TIMES EACH RANDOM TEST WILL BE RUN.
15849 :      8231
15850 :      8232 incr LUN from 0 to (.LSUNIT - 1) do
15851 :      8233         begin
15852 :      8234
15853 :      8235         if .DRIVE_STATUS [.LUN] eql ACTIVE
15854 :      8236         then
15855 :      8237         begin
15856 :      8238         UNIT = .DRIVE;
15857 :      8239
15858 :      8240         if .MLDT
15859 :      8241         then
15860 :      8242         begin
15861 :      8243         REPEAT = 1;
15862 :      8244         !LOWER THE REPEAT TIME FOR ALL TESTABLE UNITS.
15863 :      8245         exitloop;
15864 :      8246         end;
15865 :      8247
15866 :      8248         end;
15867 :      8249
15868 :      8250         end;
15869 :      8251
15870 :      8252 :+
15871 :      8253 : AT THIS POINT, 'REPEAT' IS BASED ON THE SYSTEM CONFIGURATION.
15872 :      8254 : NOW WEIGHT THE LOOP COUNTS OF THE 4 RANDOM TESTS SO THAT RAND1
15873 :      8255 : AND RAND2 ARE MUCH QUICKER THAN RAND3 AND RAND4. RAND3 IS LONGEST.
15874 :      8256 :--
15875 :      8257 RAND1 (.REPEAT);
  
```

!* 1 *

!THE MORE UNITS THERE ARE ON THE SYSTEM, THE

!SEE IF THERE ARE ANY UNITS WHICH HAVE 64K CHIPS (ML11-B)

!* 3 *

!* 4 *
 !THERE IS AT LEAST 1 UNIT WHICH IS AN ML11-B, SO

!* 4 *

!* 3 *

!* 2 *

!TEST USING RANDOM DATA

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (48)
 !TEST USING RANDOM DATA, WORD COUNTS
 !TEST USING RANDOM SECTORS, DATA, WORD COUNTS
 !TEST USING RANDOM UNITS, SECTORS, DATA, WORD COUNTS

!* 1 *

15877 :MLX4
 15878 :
 15879 :
 15880 : 8258 RAND2 (.REPEAT);
 15881 : 8259 RAND3 (.REPEAT);
 15882 : 8260 RAND4 ((.REPEAT*16));
 15883 : 8261 return;
 15884 : 8262 end;

DEFINITION OF OPTION 5

15889				.SBTTL	OPT5 DEFINITION OF OPTION 5		
15893	102272	004167	103016	OPT5:	JSR R1,\$SAVE4	:	8207
15894	102276	012746	007656		MOV #RTN5,-(SP)	:	8227
15895	102302	012746	007344		MOV #WRD34,-(SP)		
15896	102306	012746	007100		MOV #SAY2,-(SP)		
15897	102312	012746	000003		MOV #3,-(SP)		
15898	102316	010600			MOV SP,R0	: SP,*	
15899	102320	104414			TRAP 14		
15900	102322	012703	000011		MOV #11,R3	: *,REPEAT	8229
15901	102326	166703	132124		SUB NUM.DRIVES,R3	: *,REPEAT	
15902	102332	016704	077454		MOV LSUNIT,R4	:	8232
15903	102336	005001			CLR R1	: LUN	
15904	102340	000450			BR 3\$		
15905	102342	010102		1\$:	MOV R1,R2	: LUN,*	8235
15906	102344	006202			ASR R2		
15907	102346	006202			ASR R2		
15908	102350	006202			ASR R2		
15909	102352	062702	034442		ADD #DRIVE.STATUS,R2		
15910	102356	010246			MOV R2,-(SP)		
15911	102360	010146			MOV R1,-(SP)	: LUN,*	
15912	102362	042716	177770		BIC #177770,(SP)		
15913	102366	012746	000001		MOV #1,-(SP)		
15914	102372	005046			CLR -(SP)		
15915	102374	004767	101756		JSR PC,BLSGT2		
15916	102400	062706	000010		ADD #10,SP		
15917	102404	005300			DEC R0		
15918	102406	001074			BNE 2\$		
15919	102410	010102			MOV R1,R2	: LUN,*	8238
15920	102412	006302			ASL R2		
15921	102414	016202	034422		MOV PTABLE.ADDR(R2),R2		
15922	102420	016200	000006		MOV 6(R2),R0		
15923	102424	042700	177770		BIC #177770,R0		
15924	102430	142777	000007	131720	BICB #7,2ML.REG+10		
15925	102436	150077	131714		BISB R0,2ML.REG+10		
15926	102442	032777	000001	131724	BIT #1,2ML.REG+26	:	8240
15927	102450	001403			BEQ 2\$		
15928	102452	012703	000001		MOV #1,R3	: *,REPEAT	8243
15929	102456	000403			BR 4\$:	8245
15930	102460	005201		2\$:	INC R1	: LUN	8232
15931	102462	020104		3\$:	CMP R1,R4	: LUN,*	

```
15933 ;MLX4
15934 ;
15935
15936 102464 002726
15937 102466 010316
15938 102470 004767 166044
15939 102474 010316
15940 102476 004767 170422
15941 102502 010316
15942 102504 004767 172734
15943 102510 010300
15944 102512 006300
15945 102514 006300
15946 102516 006300
15947 102520 006300
15948 102522 010016
15949 102524 004767 175256
15950 102530 062706 000010
15951 102534 000207
15952
15953
15954
15959
15960
```

DEFINITION OF OPTION 5

```
4$: BLT 1$
MOV R3,(SP) ; REPEAT,* 8257
JSR PC,RAND1
MOV R3,(SP) ; REPEAT,* 8258
JSR PC,RAND2
MOV R3,(SP) ; REPEAT,* 8259
JSR PC,RAND3
MOV R3,R0 ; REPEAT,* 8260
ASL R0
ASL R0
ASL R0
MOV R0,(SP)
JSR PC,RAND4
ADD #10,SP ; 8207
RTS PC
```

: Routine Size: 82 words
: Maximum stack depth per invocation: 13 words

27-Mar-1982 19:24:42
27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
PA:<NEALE>MLX4.BLI.5 (49)

```
15962 :MLX4
15963 :
15964 :
15965 :      8263 %sbttl 'THE OPTION SCHEDULER'
15966 :      8264 BGNST;
15967 :      8265
15968 :      8266 !++
15969 :      8267 ROUTINE:      T1
15970 :      8268
15971 :      8269 PURPOSE:      THIS IS THE ONE MAIN TEST OF THE EXERCISER. IT LOOKS
15972 :      8270 AT THE AVAILABILITY OF TEST OPTIONS AND MAKES CALLS TO
15973 :      8271 THE OPTIONS WHEN APPROPRIATE.
15974 :      8272
15975 :      8273 THE QUICK VERIFY PASS INCLUDES THE FOLLOWING ROUTINES:
15976 :      8274
15977 :      8275 (1) INTEGRITY
15978 :      8276 (2) OPT1
15979 :      8277 (3) OPT2
15980 :      8278 (4) OPT3
15981 :      8279
15982 :      8280 SUBSEQUENT PASSES INCLUDE:
15983 :      8281
15984 :      8282 (1) OPT1
15985 :      8283 (2) OPT2
15986 :      8284 (3) OPT3
15987 :      8285 (4) OPT4
15988 :      8286 (5) OPT5
15989 :      8287 !--
15990 :      8288
15991 :      8289 if .QUICK neq 0
15992 :      8290 then !THIS IS THE QUICK VERIFY PASS
15993 :      8291 begin
15994 :      8292 PRINTB (CRLF);
15995 :      8293 PRINTB (SAY3, WRD2, PHR2, WRD4);
15996 :      8294 !'BEGIN QUICK VERIFY PASS'
15997 :      8295 INTEGRITY ();
15998 :      8296 end;
15999 :      8297
16000 :      8298 if .DROP1 eql 0
16001 :      8299 then !OPTION 1 IS AVAILABLE
16002 :      8300 OPT1 ();
16003 :      8301
16004 :      8302 if .DROP2 eql 0
16005 :      8303 then !OPTION 2 IS AVAILABLE
16006 :      8304 OPT2 ();
16007 :      8305
16008 :      8306 if .DROP3 eql 0
16009 :      8307 then !OPTION 3 IS AVAILABLE
16010 :      8308 OPT3 ();
16011 :      8309
16012 :      8310 if .QUICK eql 0
16013 :      8311 then
16014 :      8312 begin
16015 :      8313
16016 :      8314 if .DROP4 eql 0
```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (49)

```

16018 ;MLX4
16019 :
16020 :
16021 :      8315      then
16022 :      8316      OPT4 ();
16023 :      8317
16024 :      8318      if .DROP5 eql 0
16025 :      8319      then
16026 :      8320      OPT5 ();
16027 :      8321
16028 :      8322      end;
16029 :      8323
16030 :      8324      ENDTST;
    
```

!OPTION 4 IS AVAILABLE

!OPTION 5 IS AVAILABLE

Address	Hex	Hex	Hex	Label	Instruction	Comment	Address
16034					.SBTTL	\$T1 THE OPTION SCHEDULER	
16035				\$T1:	TST	QUICK	8289
16039	102536	005767	130132		BEQ	1\$	
16040	102542	001426			MOV	#CRLF,-(SP)	8292
16041	102544	012746	007066		MOV	#1,-(SP)	
16042	102550	012746	000001		MOV	SP,RO	: SP,*
16043	102554	010600			TRAP	14	
16044	102556	104414			MOV	#WRD4,(SP)	8293
16045	102560	012716	007212		MOV	#PHR2,-(SP)	
16046	102564	012746	007730		MOV	#WRD2,-(SP)	
16047	102570	012746	007200		MOV	#SAY3,-(SP)	
16048	102574	012746	007112		MOV	#4,-(SP)	
16049	102600	012746	000004		MOV	SP,RO	: SP,*
16050	102604	010600			TRAP	14	
16051	102606	104414			JSR	PC,INTEGRITY	8295
16052	102610	004767	144416		ADD	#14,SP	8291
16053	102614	062706	000014		TST	DROP1	8298
16054	102620	005767	077412	1\$:	BNE	2\$	
16055	102624	001002			JSR	PC,OPT1	8300
16056	102626	004767	146626		TST	DROP2	8302
16057	102632	005767	077402	2\$:	BNE	3\$	
16058	102636	001002			JSR	PC,OPT2	8304
16059	102640	004767	151210		TST	DROP3	8306
16060	102644	005767	077372	3\$:	BNE	4\$	
16061	102650	001002			JSR	PC,OPT3	8308
16062	102652	004767	155174		TST	QUICK	8310
16063	102656	005767	130012	4\$:	BNE	6\$	
16064	102662	001012			TST	DROP4	8314
16065	102664	005767	077354		BNE	5\$	
16066	102670	001002			JSR	PC,OPT4	8316
16067	102672	004767	157546		TST	DROP5	8318
16068	102676	005767	077346	5\$:	BNE	6\$	
16069	102702	001002			JSR	PC,OPT5	8320
16070	102704	004767	177362		RTS	PC	8262
16071	102710	000207		6\$:			
16072							

27-Mar-1982 19:24:42 TOPS
27-Mar-1982 19:23:44 PA:<

```
16074 :MLX4
16075 ;
16076 THE OPTION SCHEDULER
16077 ; Routine Size: 54 words
16078 ; Maximum stack depth per invocation: 6 words
16083
16084
16088
16089 .SBTTL T1 THE OPTION SCHEDULER
16093 T1::
16094 1$: JSR PC,$T1 ;
16095 102712 004767 177620 TRAP 66
16096 102716 104466 ROR R0
16097 102720 006000 BLO 1$
16098 102722 103773 RTS PC
16099
16100 ; Routine Size: 6 words
16101 ; Maximum stack depth per invocation: 0 words
16106
16107
```

8322

27-Mar-1982 19:24:42
27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
PA:<NEALE>MLX4.BLI.5 (50)

16109 :MLX4
16110 :
16111 :
16112 :
16113 :
16114 :
16115 :
16116 :
16117 :
16118 :
16119 :
16120 :
16121 :
16122 :
16123 :
16124 :
16125 :
16126 :
16127 :
16128 :
16129 :
16130 :
16131 :
16132 :
16133 :
16134 :
16135 :
16136 :
16137 :
16138 :
16139 :
16140 :
16141 :
16142 :
16143 :
16144 :
16145 :
16146 :
16147 :
16148 :
16149 :
16150 :
16151 :
16152 :
16153 :
16154 :
16155 :
16156 :
16157 :
16158 :
16159 :
16160 :
16161 :
16162 :
16163 :

```
END OF PASS SUMMARY
8325 %sbttl 'END OF PASS SUMMARY'
8326 routine EOP : novalue =
8327 begin
8328
8329
8330 ++ ROUTINE: EOP
8331
8332 PURPOSE: TO PRINT A STATUS REPORT FOR EACH DRIVE. IF AN
8333 ARRAY HAS ANY HARD OR SOFT ERRORS, THEN ITS ERROR
8334 COUNT WILL APPEAR, AND IF THE COUNT IS TOO HIGH,
8335 THEN A DIAGNOSIS TO RUN THE ML11 PROM MAINTENANCE
8336 PROGRAM WILL ALSO APPEAR.
8337
8338 THE FOLLOWING IS A SAMPLE REPORT FOR 2 DRIVES:
8339
8340 PERFORMANCE SUMMARY
8341
8342 NUMBER OF MBYTES TRANSFERRED:
8343 1028 MBYTES WRITTEN
8344 250 MBYTES READ
8345 1145 MBYTES WRITE CHECKED
8346
8347 LOGICAL UNIT: 0 DRIVE: 1 SERIAL #: 1234
8348 SOFT ERROR COUNT: 9
8349 ARRAY 3: 9
8350 HARD ERROR COUNT: 11
8351 ARRAY 0: 6
8352 ARRAY 10: 2
8353 ARRAY 15: 3
8354 TRANSFER RETRIES: 0
8355
8356 LOGICAL UNIT: 1 DRIVE: 1 SERIAL #: 9876
8357 DRIVE DROPPED (CONTROLLER FATAL ERROR)
8358 SOFT ERROR COUNT: 100
8359 ARRAY 1: 9
8360 ARRAY 13: 10 --> RUN ML11 PROM MAINTENANCE PROGRAM
8361 ARRAY 14: 1
8362 ARRAY 15: 2 --> RUN ML11 PROM MAINTENANCE PROGRAM
8363 HARD ERROR COUNT: 2
8364 ARRAY 14: 1
8365 ARRAY 15: 1
8366 TRANSFER RETRIES: 0
8367
8368
8369
8370 Local
8371 SFT_TOT.
8372 HRD_TOT.
8373 TRY_TOT.
8374
8375 if .EOPSUM
8376 then
```

!THE OPERATOR HAS ALLOWED THE SUMMARY TO PRINT

27-Mar-1982 19:24:42
27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(21?)
PA:<NEALE>MLX4.BLI.5 (50)

```
16165 :MLX4  
16166 :  
16167 :  
16168 : 8377 begin  
16169 : 8378 PRINTB (CRLF);  
16170 : 8379 PRINTB (SAY1, PHR17);  
16171 : 8380 !'PERFORMANCE SUMMARY'  
16172 : 8381 PRINTB (CRLF);  
16173 : 8382 PRINTB (SAY1, PHR12);  
16174 : 8383 !NUMBER OF MBYTES TRANSFERED:  
16175 : 8384 PRINTB (FMT14, .WR MILLIONS, PHR19);  
16176 : 8385 !'XXXXX MBYTES WRITTEN'  
16177 : 8386 PRINTB (FMT14, .LW MILLIONS, PHR20);  
16178 : 8387 !'XXXXX MBYTES READ'  
16179 : 8388 PRINTB (FMT14, .WC MILLIONS, PHR13);  
16180 : 8389 !'XXXXX MBYTES WRITE CHECKED'  
16181 : 8390  
16182 : 8391 incr LUN from 0 to (.LSUNIT - 1) do  
16183 : 8392 begin  
16184 : 8393 UNIT = .DRIVE;  
16185 : 8394  
16186 : 8395 if .DPR eq 0  
16187 : 8396 then  
16188 : 8397 PRINTB (FMT4A, PHR7, .LUN, WRD11, .DRIVE)  
16189 : 8398 !'LOGICAL UNIT: X DRIVE: Y'  
16190 : 8399 else  
16191 : 8400 SAYWHO (.LUN);  
16192 : 8401  
16193 : 8402 !'LOGICAL UNIT: X DRIVE: Y SERIAL #: ZZZZ'  
16194 : 8403  
16195 : 8404 if .DROPT_DRIVES [.LUN] eq 1 ACTIVE  
16196 : 8405 then  
16197 : 8406 begin  
16198 : 8407 PRINTB (SAY2, WRD11, WRD21);  
16199 : 8408 !'DRIVE DROPPED'  
16200 : 8409  
16201 : 8410 selectone .WHY_DROPT [.LUN] of  
16202 : 8411 set  
16203 : 8412  
16204 : 8413 [CODE 1] :  
16205 : 8414 PRINTB (FMT2, CAUSE1);  
16206 : 8415 !' (NOT POWERED UP)'  
16207 : 8416  
16208 : 8417 [CODE 2] :  
16209 : 8418 PRINTB (FMT2, CAUSE2);  
16210 : 8419 !' (NOT AN ML11 UNIT)'  
16211 : 8420  
16212 : 8421 [CODE 3] :  
16213 : 8422 PRINTB (FMT2, CAUSE3);  
16214 : 8423 !' (OPERATOR SELECTED TEST LIMITS INCORRECTLY)'  
16215 : 8424  
16216 : 8425 [CODE 4] :  
16217 : 8426 PRINTB (FMT2, CAUSE4);  
16218 : 8427 !' (ALL RETRIES FAILED FOR A NON-FATAL ERROR)'  
16219 : 8428
```

!* 2 *

!* 3 *

!* 4 *

```

16221 :MLX4
16222 :
16223 :
16224 : 8429
16225 : 8430
16226 : 8431
16227 : 8432
16228 : 8433
16229 : 8434
16230 : 8435
16231 : 8436
16232 : 8437
16233 : 8438
16234 : 8439
16235 : 8440
16236 : 8441
16237 : 8442
16238 : 8443
16239 : 8444
16240 : 8445
16241 : 8446
16242 : 8447
16243 : 8448
16244 : 8449
16245 : 8450
16246 : 8451
16247 : 8452
16248 : 8453
16249 : 8454
16250 : 8455
16251 : 8456
16252 : 8457
16253 : 8458
16254 : 8459
16255 : 8460
16256 : 8461
16257 : 8462
16258 : 8463
16259 : 8464
16260 : 8465
16261 : 8466
16262 : 8467
16263 : 8468
16264 : 8469
16265 : 8470
16266 : 8471
16267 : 8472
16268 : 8473
16269 : 8474
16270 : 8475
16271 : 8476
16272 : 8477
16273 : 8478
16274 : 8479
16275 : 8480

END OF PASS SUMMARY

[CODE 5] :
PRINTB (FMT2, CAUSE5);
!' (CONTROLLER FATAL ERROR)'

[CODE 6] :
PRINTB (FMT2, CAUSE6);
!' (DRIVE FATAL ERROR)'

[CODE 7] :
PRINTB (FMT2, CAUSE7);
!' (ECC HARD ERROR)'

[CODE 8] :
PRINTB (FMT2, CAUSE8);
!' (ECC LOGIC FAILURE)'
tes;

end;          !* 4 *

SFT_TOT = 0;
HRD_TOT = 0;
TRY_TOT = 0;

incr ARRAY from 0 to 15 do
begin
SFT_TOT = .SFT_TOT + .SOFTS [.LUN, .ARRAY, 0, 16, 0];
HRD_TOT = .HRD_TOT + .HARDS [.LUN, .ARRAY, 0, 16, 0];
TRY_TOT = .TRY_TOT + .TRIES [.LUN, .ARRAY, 0, 16, 0];
end;

PRINTB (FMT7, PHR5, .SFT_TOT);
!' SOFT ERROR COUNT: DDDDD'

if .SFT_TOT neq 0
then

incr ARRAY from 0 to 15 do
if .SOFTS [.LUN, .ARRAY, 0, 16, 0] neq 0
then
begin
PRINTB (FMT9, .ARRAY, .SOFTS [.LUN, .ARRAY, 0, 16, 0]);
!' ARRAY XX: YYYYY'

: VER CZMLBB THE FOLLOWING CODE HAS BEEN COMMENTED OUT BECAUSE SOFT ERRORS
: SHOULD NOT BE COUNTED TOWARD THE CALLING OUT OF THE PROM MAINENANCE PROGRAM
:
: +
: NOW SEE IF THRESHOLDS FOR SOFT ERRORS
: HAVE BEEN REACHED FOR THIS ARRAY BOARD:
:
if ((.ARR_TYP eql 0) and (.SOFTS [.LUN, .ARRAY, 0, 16, 0] geq S16K_LIMIT))

```

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (50)

```

16277 :MLX4
16278 :
16279 :
16280 : 8481 :
16281 : 8482 :
16282 : 8483 :
16283 : 8484 :
16284 : 8485 :
16285 : 8486 :
16286 : 8487 :
16287 : 8488 :
16288 : 8489 :
16289 : 8490 :
16290 : 8491 :
16291 : 8492 :
16292 : 8493 :
16293 : 8494 :
16294 : 8495 :
16295 : 8496 :
16296 : 8497 :
16297 : 8498 :
16298 : 8499 :
16299 : 8500 :
16300 : 8501 :
16301 : 8502 :
16302 : 8503 :
16303 : 8504 :
16304 : 8505 :
16305 : 8506 :
16306 : 8507 :
16307 : 8508 :
16308 : 8509 :
16309 : 8510 :
16310 : 8511 :
16311 : 8512 :
16312 : 8513 :
16313 : 8514 :
16314 : 8515 :
16315 : 8516 :
16316 : 8517 :
16317 : 8518 :
16318 : 8519 :
16319 : 8520 :
16320 : 8521 :
16321 : 8522 :
16322 : 8523 :
16323 : 8524 :
16324 : 8525 :
16325 : 8526 :
16326 : 8527 :
16327 : 8528 :
16328 : 8529 :
16329 : 8530 :
16330 : 8531 :
16331 : 8532 :

END OF PASS SUMMARY

then
  PRINTB (FMT2, MSG2);
!' --> RUN ML11 PRGM MAINTENANCE PROGRAM'
if ((.ARR_TYP eq 1) and (.SOFTS [.LUN, .ARRAY, 0, 16, 0] geq S64K_LIMIT))
then
  PRINTB (FMT2, MSG2);
!' --> RUN ML11 PROM MAINTENANCE PROGRAM'
end;

PRINTB (FMT7, PHR18, .HRD_TOT);
!' HARD ERROR COUNT: DDDDD'
if .HRD_TOT neq 0
then
  incr ARRAY from 0 to 15 do
    if .HARDS [.LUN, .ARRAY, 0, 16, 0] neq 0
    then
      begin
        PRINTB (FMT9, .ARRAY, .HARDS [.LUN, .ARRAY, 0, 16, 0]);
        !' ARRAY XX: YYYYY'
        !+
        ! NOW SEE IF THRESHOLDS FOR HARD ERRORS
        ! HAVE BEEN REACHED FOR THIS ARRAY BOARD:
        !-
      end
    end
  end

VER CZMLBB CHANGED TESTING HARDS [TABLE] TO TESTING PM_SBE_CNT [TABLE]

if ((.ARR_TYP eq 0) and (.PM_SBE_CNT [.LUN, .ARRAY, PM_SBE$_SUM] geq H16K_LIMIT))
then
  PRINTB (FMT2, MSG2);
!' --> RUN ML11 PROM MAINTENANCE PROGRAM'

VER CZMLBB CHANGED TESTING HARDS [TABLE] TO TESTING PM_SBE_CNT [TABLE]

if ((.ARR_TYP eq 1) and (.PM_SBE_CNT [.LUN, .ARRAY, PM_SBE$_SUM] geq H64K_LIMIT))
then
  PRINTB (FMT2, MSG2);
!' --> RUN ML11 PROM MAINTENANCE PROGRAM'
end;

PRINTB (FMT7, PHR6, .TRY_TOT);

```

27-Mar-1982 19:24:42
27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
PA:<NEALE>MLX4.BLI.5 (50)

27-Mar-1982 19:24:42
27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
PA:<NEALE>MLX4.BLI.5 (50)

16333 :MLX4
16334 :
16335 :
16336 : 8533
16337 : 8534
16338 : 8535
16339 : 8536
16340 : 8537
16341 : 8538
16342 : 8539
16343 : 8540
16344 : 8541
16345 : P 8542
16346 : P 8543
16347 : 8544
16348 : 8545
16349 : 8546
16350 : 8547
16351 : 8548
16352 : 8549
16353 : 8550
16354 : 8551
16355 : 8552
16356 : 8553
16357 : 8554
16358 : 8555
16359 : 8556
16360 : 8557
16361 : 8558
16362 : 8559
16363 : 8560
16364 : 8561
16365 : 8562
16366 : 8563
16367 : 8564
16368 : 8565
16369 : 8566
16370 : 8567
16371 : 8568
16372 : 8569
16373 : 8570
16374 : 8571
16375 : 8572
16376 : 8573
16377 : 8574
16378 : P 8575
16379 : P 8576
16380 : P 8577
16381 : P 8578
16382 : P 8579
16383 : 8580
16384 : 8581
16385 : 8582
16386 : 8583
16387 : 8584

END OF PASS SUMMARY

```

!' TRANSFER RETRIES: DDDDD'
if .TRY_TOT neq 0
then
    incr ARRAY from 0 to 15 do
        if .TRIES [.LUN, .ARRAY, 0, 16, 0] neq 0
        then
            PRINTB (FMT9, .ARRAY,
                .TRIES [.LUN,
                    .ARRAY, 0, 16, 0]);
!' ARRAY XX: YYYYY'
end;
!* 3 *

+
Examine the number of single
bit errors detected during
execution of the exerciser
thus far. If any were detected
then print them out to the
operator.

if .SBES_COUNT gtr ZERO
then
begin
PRINTB (CRLF);
PRINTB (SAY1, PHR21);          !Print single bit log summary report
PRINTB (CRLF);
PRINTB (SAY1, SBES$HEADER);   !Print the column headings

+
Print to the console terminal for the
operator review all the detected single
bit errors during the execution of this
exerciser thus far.

incr index from 0 to .SBES_COUNT - 1 do    !Print all errors logged in this table
begin
PRINTB (FMT16,                !This is the printing format
    .SBE_LOG [.index, UNIT$ SBE], !Print the failing unit of this sbe
    .SBE_LOG [.index, BRD$ SBE], !Print the failing board number
    .SBE_LOG [.index, BNK$ SBE], !Print the failing bank number
    .SBE_LOG [.index, BIT$ SBE], !Print the failing bit number
    .SBE_LOG [.index, SUM$ SBE]); !Print the total sbe's found
end;
end;

```

16389 :MLX4
 16390 :
 16391 :
 16392 : 8585
 16393 : 8586
 16394 : 8587
 16395 : 8588
 16396 : 8589

END OF PASS SUMMARY

PRINTB (CRLF);
 end;
 return;
 end;

27-Mar-1982 19:24:42
 27-Mar-1982 19:23:44

TOPS-20 Bliss-16 V2(212)
 PA:<NEALE>MLX4.BLI.5 (50)

!* 2 *

!* 1 *

Address	Offset	Hex	Hex	Label	Instruction	Comment	Address
16400					.SBTTL	EOP END OF PASS SUMMARY	
16401				EOP:	JSR	P1,\$SAVE5	8326
16405	102726	004167	102402		SUB	#10,SP	
16406	102732	162706	000010		BIT	#1,EOPSUM	8375
16407	102736	032767	000001	077312	BNE	1\$	
16408	102744	001002			JMP	32\$	
16409	102746	000167	001744		MOV	#CRLF,-(SP)	8378
16410	102752	012746	007066	1\$:	MOV	#1,-(SP)	
16411	102756	012746	000001		MOV	SP,R0	: SP,*
16412	102762	010600			TRAP	14	
16413	102764	104414			MOV	#PHR17,(SP)	8379
16414	102766	012716	010342		MOV	#SAY1,-(SP)	
16415	102772	012746	007072		MOV	#2,-(SP)	
16416	102776	012746	000002		MOV	SP,R0	: SP,*
16417	103002	010600			TRAP	14	
16418	103004	104414			MOV	#CRLF,(SP)	8381
16419	103006	012716	007066		MOV	#1,-(SP)	
16420	103012	012746	000001		MOV	SP,R0	: SP,*
16421	103016	010600			TRAP	14	
16422	103020	104414			MOV	#PHR12,(SP)	8382
16423	103022	012716	010202		MOV	#SAY1,-(SP)	
16424	103026	012746	007072		MOV	#2,-(SP)	
16425	103032	012746	000002		MOV	SP,R0	: SP,*
16426	103036	010600			TRAP	14	
16427	103040	104414			MOV	#PHR19,(SP)	8384
16428	103042	012716	010410		MOV	WR.MILLIONS,-(SP)	
16429	103046	016746	131246		MOV	#FMT14,-(SP)	
16430	103052	012746	007046		MOV	#3,-(SP)	
16431	103056	012746	000003		MOV	SP,R0	: SP,*
16432	103062	010600			TRAP	14	
16433	103064	104414			MOV	#PHR20,(SP)	8386
16434	103066	012716	010430		MOV	RD.MILLIONS,-(SP)	
16435	103072	016746	131230		MOV	#FMT14,-(SP)	
16436	103076	012746	007046		MOV	#3,-(SP)	
16437	103102	012746	000003		MOV	SP,R0	: SP,*
16438	103106	010600			TRAP	14	
16439	103110	104414			MOV	#PHR13,(SP)	8388
16440	103112	012716	010240		MOV	WC.MILLIONS,-(SP)	
16441	103116	016746	131212		MOV	#FMT14,-(SP)	
16442	103122	012746	007046		MOV	#3,-(SP)	
16443	103126	012746	000003		MOV		

Address	OpCode	Operand1	Operand2	Operand3	Operand4	Instruction	Comments	Address	Time	Page
16445										
16446									27-Mar-1982 19:24:42	TOPS
16447									27-Mar-1982 19:23:44	PA:<
16448	103132	010600				MOV SP,R0	: SP,*			
16449	103134	104414				TRAP 14				
16450	103136	016766	076650	000046		MOV LSUNIT,46(SP)				8391
16451	103144	005005				CLR R5	: LUN			
16452	103146	000167	001246			JMP 27\$				
16453	103152	010500			2\$:	MOV R5,R0	: LUN,*			8393
16454	103154	006300				ASL R0				
16455	103156	016003	034422			MOV PTABLE.ADDR(R0),R3				
16456	103162	016304	000006			MOV 6(R3),R4				
16457	103166	042704	177770			BIC #177770,R4				
16458	103172	142777	000007	131156		BICB #7,@ML.REG+10				
16459	103200	150477	131152			BISB R4,@ML.REG+10				
16460	103204	032777	000400	131146		BIT #400,@ML.REG+12	:			8395
16461	103212	001022				BNE 3\$				
16462	103214	016003	034422			MOV PTABLE.ADDR(R0),R3	:			8397
16463	103220	016346	000006			MOV 6(R3),-(SP)				
16464	103224	012746	007240			MOV #WRD11,-(SP)				
16465	103230	010546				MOV R5,-(SP)	: LUN,*			
16466	103232	012746	010050			MOV #PHR7,-(SP)				
16467	103236	012746	006216			MOV #FMT4A,-(SP)				
16468	103242	012746	000005			MOV #5,-(SP)				
16469	103246	010600				MOV SP,R0	: SP,*			
16470	103250	104414				TRAP 14				
16471	103252	062706	000012			ADD #12,SP				
16472	103256	000403				BR 4\$:			8395
16473	103260	010546			3\$:	MOV R5,-(SP)	: LUN,*			8400
16474	103262	004767	132236			JSR PC,SAYWHO				
16475	103266	010500			4\$:	MOV R5,R0	: LUN,*			8404
16476	103270	006200				ASR R0				
16477	103272	006200				ASR R0				
16478	103274	006200				ASR R0				
16479	103276	062700	034444			ADD #DROPT.DRIVES,R0				
16480	103302	010046				MOV R0,-(SP)				
16481	103304	010546				MOV R5,-(SP)	: LUN,*			
16482	103306	042716	177770			BIC #177770,(SP)				
16483	103312	012746	000001			MOV #1,-(SP)				
16484	103316	005046				CLR -(SP)				
16485	103320	004767	101032			JSR PC,BLSGT2				
16486	103324	062706	000010			ADD #10,SP				
16487	103330	005300				DEC R0				
16488	103332	001160				BNE 14\$				
16489	103334	012746	007322			MOV #WRD21,-(SP)	:			8407
16490	103340	012746	007240			MOV #WRD11,-(SP)				
16491	103344	012746	007100			MOV #SAY2,-(SP)				
16492	103350	012746	000003			MOV #3,-(SP)				
16493	103354	010600				MOV SP,R0	: SP,*			
16494	103356	104414				TRAP 14				
16495	103360	005003				CLR R3	:			8410
16496	103362	156503	034446			BISB WHY.DROPT(R5),R3	: *(LUN),*			
16497	103366	020327	000001			CMP R3,#1				
16498	103372	001011				BNE 5\$				
16499	103374	012746	010460			MOV #CAUSE1,-(SP)	:			8414

Address	OpCode	Operand1	Operand2	Operand3	Instruction	Comments	Address	Time	Page
16501								27-Mar-1982 19:24:42	TOPS
16502								27-Mar-1982 19:23:44	PA:<
16503									
16504	103400	012746	006160		MOV #FMT2,-(SP)				
16505	103404	012746	000002		MOV #2,-(SP)				
16506	103410	010600			MOV SP,R0	: SP,*			
16507	103412	104414			TRAP 14				
16508	103414	000523			BR 12\$				
16509	103416	020327	000002	5\$:	CMP R3,#2	:			8410
16510	103422	001011			BNE 6\$				
16511	103424	012746	010502		MOV #CAUSE2,-(SP)	:			8418
16512	103430	012746	006160		MOV #FMT2,-(SP)				
16513	103434	012746	000002		MOV #2,-(SP)				
16514	103440	010600			MOV SP,R0	: SP,*			
16515	103442	104414			TRAP 14				
16516	103444	000507			BR 12\$				
16517	103446	020327	000003	6\$:	CMP R3,#3	:			8410
16518	103452	001011			BNE 7\$				
16519	103454	012746	010526		MOV #CAUSE3,-(SP)	:			8422
16520	103460	012746	006160		MOV #FMT2,-(SP)				
16521	103464	012746	000002		MOV #2,-(SP)				
16522	103470	010600			MOV SP,R0	: SP,*			
16523	103472	104414			TRAP 14				
16524	103474	000473			BR 12\$				
16525	103476	020327	000004	7\$:	CMP R3,#4	:			8410
16526	103478	001011			BNE 8\$				
16527	103484	012746	010602		MOV #CAUSE4,-(SP)	:			8426
16528	103490	012746	006160		MOV #FMT2,-(SP)				
16529	103514	012746	000002		MOV #2,-(SP)				
16530	103520	010600			MOV SP,R0	: SP,*			
16531	103522	104414			TRAP 14				
16532	103524	000457			BR 12\$				
16533	103526	020327	000005	8\$:	CMP R3,#5	:			8410
16534	103532	001011			BNE 9\$				
16535	103534	012746	010656		MOV #CAUSE5,-(SP)	:			8430
16536	103540	012746	006160		MOV #FMT2,-(SP)				
16537	103544	012746	000002		MOV #2,-(SP)				
16538	103550	010600			MOV SP,R0	: SP,*			
16539	103552	104414			TRAP 14				
16540	103554	000443			BR 12\$				
16541	103556	020327	000006	9\$:	CMP R3,#6	:			8410
16542	103562	001011			BNE 10\$				
16543	103564	012746	010710		MOV #CAUSE6,-(SP)	:			8434
16544	103570	012746	006160		MOV #FMT2,-(SP)				
16545	103574	012746	000002		MOV #2,-(SP)				
16546	103600	010600			MOV SP,R0	: SP,*			
16547	103602	104414			TRAP 14				
16548	103604	000427			BR 12\$				
16549	103606	020327	000007	10\$:	CMP R3,#7	:			8410
16550	103612	001011			BNE 11\$				
16551	103614	012746	010734		MOV #CAUSE7,-(SP)	:			8438
16552	103620	012746	006160		MOV #FMT2,-(SP)				
16553	103624	012746	000002		MOV #2,-(SP)				
16554	103630	010600			MOV SP,R0	: SP,*			
16555	103632	104414			TRAP 14				

Address	Instruction	Comment	Symbol	Label	Register/Value	Symbol	Address	Time	Symbol
16557				:MLX4			27-Mar-1982 19:24:42	TOPS	
16558				:	END OF PASS SUMMARY		27-Mar-1982 19:23:44	PA:<	
16559									
16560	103634	000413			BR 12\$				
16561	103636	020327	000010	11\$:	CMP R3,#10	:			8410
16562	103642	00012			BNE 13\$:			
16563	103644	012746	010756		MOV #CAUSE8,-(SP)	:			8442
16564	103650	012746	006160		MOV #FMT2,-(SP)	:			
16565	103654	012746	000002		MOV #2,-(SP)	:			
16566	103660	010600			MOV SP,R0	: SP,*			
16567	103662	104414			TRAP 14				
16568	103664	062706	000006	12\$:	ADD #6,SP				8406
16569	103670	062706	000010	13\$:	ADD #10,SP	:			8448
16570	103674	005066	000046	14\$:	CLR 46(SP)	: SFT.TOT			8449
16571	103700	005066	000044		CLR 44(SP)	: HRD.TOT			8450
16572	103704	005066	000042		CLR 42(SP)	: TRY.TOT			8454
16573	103710	010500			MOV R5,R0	: LUN,*			
16574	103712	006300			ASL R0				
16575	103714	006300			ASL R0				
16576	103716	006300			ASL R0				
16577	103720	006300			ASL R0				
16578	103722	010004			MOV R0,R4				
16579	103724	005002			CLR R2	: ARRAY			8452
16580	103726	010403		15\$:	MOV R4,R3	: :			8454
16581	103730	060203			ADD R2,R3	: ARRAY,*			
16582	103732	006303			ASL R3				
16583	103734	066366	032714 000046		ADD SOFTS(R3),46(SP)	: *,SFT.TOT			
16584	103742	066366	033314 000044		ADD HARDS(R3),44(SP)	: *,HRD.TOT			8455
16585	103750	066366	033714 000042		ADD TRIES(R3),42(SP)	: *,TRY.TOT			8452
16586	103756	005202			INC R2	: ARRAY			8452
16587	103760	020227	000017		CMP R2,#17	: ARRAY,*			
16588	103764	003760			BLE 15\$				
16589	103766	016616	000046		MOV 46(SP),(SP)	: SFT.TOT,*			8459
16590	103772	012746	010004		MOV #PHR5,-(SP)				
16591	103776	012746	006446		MOV #FMT7,-(SP)				
16592	104002	012746	000003		MOV #3,-(SP)				
16593	104006	010600			MOV SP,R0	: SP,*			
16594	104010	104414			TRAP 14				
16595	104012	005760	000054		TST 54(SP)	: SFT.TOT			8462
16596	104016	001425			BEQ 18\$				
16597	104020	005002			CLR R2	: ARRAY			8465
16598	104022	010403		16\$:	MOV R4,R3	:			8467
16599	104024	060203			ADD R2,R3	: ARRAY,*			
16600	104026	006303			ASL R3				
16601	104030	016303	032714		MOV SOFTS(R3),R3				
16602	104034	001412			BEQ 17\$				
16603	104036	010346			MOV R3,-(SP)	:			8470
16604	104040	010246			MOV R2,-(SP)	: ARRAY,*			
16605	104042	012746	006534		MOV #FMT9,-(SP)				
16606	104046	012746	000003		MOV #3,-(SP)				
16607	104052	010600			MOV SP,R0	: SP,*			
16608	104054	104414			TRAP 14				
16609	104056	062706	000010		ADD #10,SP	:			8469
16610	104062	005202		17\$:	INC R2	: ARRAY			8465
16611	104064	020227	000017		CMP R2,#17	: ARRAY,*			

Address	OpCode	Operand1	Operand2	Operand3	Operand4	Instruction	Comments	Seq
16613						:MLX4		
16614						:	END OF PASS SUMMARY	
16615								
16616	104070	003754				BLE 16\$		
16617	104072	016616	000052			MOV 52(SP), (SP)	: HRD.TOT,*	8493
16618	104076	012746	010366			MOV #PHR18, -(SP)		
16619	104102	012746	006446			MOV #FMT7, -(SP)		
16620	104106	012746	000003			MOV #3, -(SP)		
16621	104112	010600				MOV SP, R0	: SP,*	
16622	104114	104414				TRAP 14		
16623	104116	005766	000060			TST 60(SP)	: HRD.TOT	8496
16624	104122	001471				BEQ 23\$		
16625	104124	005001				CLR R1	: ARRAY	8499
16626	104126	010403				MOV R4, R3	: ARRAY,*	8501
16627	104130	060103				ADD R1, R3		
16628	104132	006303				ASL R3		
16629	104134	016302	033314			MOV HARDS(R3), R2		
16630	104140	001456				BEQ 22\$		
16631	104142	010246				MOV R2, -(SP)	: ARRAY,*	8504
16632	104144	010146				MOV R1, -(SP)		
16633	104146	012746	006534			MOV #FMT9, -(SP)		
16634	104152	012746	000003			MOV #3, -(SP)		
16635	104156	010600				MOV SP, R0	: SP,*	
16636	104160	104414				TRAP 14		
16637	104162	032777	002000	130202		BIT #2000, @ML.REG+24		8516
16638	104170	001016				BNE 20\$		
16639	104172	026327	012264	000012		CMP PM.SBE.CNT(R3), #12		
16640	104200	002412				BLT 20\$		
16641	104202	012746	011120			MOV #MSG2, -(SP)	: ARRAY,*	8518
16642	104206	012746	006160			MOV #FMT2, -(SP)		
16643	104212	012746	000002			MOV #2, -(SP)		
16644	104216	010600				MOV SP, R0	: SP,*	
16645	104220	104414				TRAP 14		
16646	104222	062706	000006			ADD #6, SP		
16647	104226	032777	002000	130136	20\$:	BIT #2000, @ML.REG+24		8525
16648	104234	001416				BEQ 21\$		
16649	104236	026327	012264	000012		CMP PM.SBE.CNT(R3), #12		
16650	104244	002412				BLT 21\$		
16651	104246	012746	011120			MOV #MSG2, -(SP)	: ARRAY,*	8527
16652	104252	012746	006160			MOV #FMT2, -(SP)		
16653	104256	012746	000002			MOV #2, -(SP)		
16654	104262	010600				MOV SP, R0	: SP,*	
16655	104264	104414				TRAP 14		
16656	104266	062706	000006			ADD #6, SP		
16657	104272	062706	000010		21\$:	ADD #10, SP	: ARRAY,*	8503
16658	104276	005201			22\$:	INC R1	: ARRAY,*	8499
16659	104300	020127	000017			CMP R1, #17		
16660	104304	003710				BLE 19\$		
16661	104306	016616	000056		23\$:	MOV 56(SP), (SP)	: TRY.TOT,*	8532
16662	104312	012746	010026			MOV #PHR6, -(SP)		
16663	104316	012746	006446			MOV #FMT7, -(SP)		
16664	104322	012746	000003			MOV #3, -(SP)		
16665	104326	010600				MOV SP, R0	: SP,*	
16666	104330	104414				TRAP 14		
16667	104332	005766	000064			TST 64(SP)	: TRY.TOT	8535


```

16725
16726
16727
16728 104562 006204
16729 104564 006204
16730 104566 006204
16731 104570 006204
16732 104572 006204
16733 104574 042704 177600
16734 104600 010446
16735 104602 111346
16736 104604 042716 177774
16737 104610 111304
16738 104612 006204
16739 104614 006204
16740 104616 042704 177760
16741 104622 010446
16742 104624 011304
16743 104626 006104
16744 104630 006104
16745 104632 006104
16746 104634 006104
16747 104636 042704 177770
16748 104642 010446
16749 104644 012746 005736
16750 104650 012746 000006
16751 104654 010600
16752 104656 104414
16753 104660 062706 000016
16754 104664 005202
16755 104666 020205
16756 104670 002723
16757 104672 062706 000016
16758 104676 012716 007066
16759 104702 012746 000001
16760 104706 010600
16761 104710 104414
16762 104712 062706 000042
16763 104716 062706 000010
16764 104722 000207
16765
16766
16767
16772
16773

```

```

;MLX4
;
END OF PASS SUMMARY

```

```

ASR R4
ASR R4
ASR R4
ASR R4
ASR R4
BIC #177600,R4
MOV R4,-(SP)
MOVB (R3),-(SP)
BIC #177774,(SP)
MOVB (R3),R4
ASR R4
ASR R4
BIC #177760,R4
MOV R4,-(SP)
MOV (R3),R4
ROL R4
ROL R4
ROL R4
ROL R4
BIC #177770,R4
MOV R4,-(SP)
MOV #FMT16,-(SP)
MOV #6,-(SP)
MOV SP,R0
TRAP 14
ADD #16,SP
INC R2
30$: CMP R2,R5
BLT 29$
ADD #16,SP
31$: MOV #CRLF,(SP)
MOV #1,-(SP)
MOV SP,R0
TRAP 14
32$: ADD #42,SP
ADD #10,SP
RTS PC

```

```

; SP,*
;
; INDEX
; INDEX,*
;
;
; SP,*
;
;
;

```

```

8574
8573
8560
8585
8377
8326

```

```

; Routine Size: 511 words
; Maximum stack depth per invocation: 40 words

```

27-Mar-1982 19:24:42 TOPS-20 Bliss-16 V2(212)
 27-Mar-1982 19:23:44 PA:<NEALE>MLX4.BLI.5 (51)

```

16775 :MLX4
16776 :
16777 :
16778 :      8590 %sbttl 'CLEANUP CODING SECTION'
16779 :      8591 BGNCLN;
16780 :      8592 !+
16781 :      8593 ! THE CLEANUP CODING SECTION IS EXECUTED AFTER
16782 :      8594 ! EACH PASS THROUGH THE EXERCISER.
16783 :      8595 !-
16784 :      8596 EOP_COUNT = .EOP_COUNT + 1;
16785 :      8597
16786 :      8598 if .EOP_COUNT eql 1
16787 :      8599 then
16788 :      8600     PRINTB (SAY3, WRD3, PHR2, WRD4)
16789 :      8601     !'END QUICK VERIFY PASS'
16790 :      8602 else
16791 :      8603     PRINTB (FMT3, WRD3, WRD4, .EOP_COUNT);
16792 :      8604
16793 :      8605 !'** END PASS XX **'
16794 :      8606 EOP ();
16795 :      8607 return;
16796 :      8608 ENDCLN;
16800
16801
  
```

```

16801          .SBTTL      LCLEAN CLEANUP CODING SECTION
16805 104724 005267 125754          LCLEAN: INC      EOP_COUNT          :      8596
16806 104730 026727 125750 000001  CMP      EOP_COUNT,#1      :      8598
16807 104736 001015          BNE      1$                :
16808 104740 012746 007212          MOV      #WRD4,-(SP)        :      8600
16809 104744 012746 007730          MOV      #PHR2,-(SP)        :
16810 104750 012746 007206          MOV      #WRD3,-(SP)        :
16811 104754 012746 007112          MOV      #SAY3,-(SP)       :
16812 104760 012746 000004          MOV      #4,-(SP)          :
16813 104764 010600          MOV      SP,R0              : SP,*
16814 104766 104414          TRAP    14                  :
16815 104770 000414          BR      2$                  :      8598
16816 104772 016746 125706          1$:  MOV      EOP_COUNT,-(SP) :      8603
16817 104776 012746 007212          MOV      #WRD4,-(SP)        :
16818 105002 012746 007206          MOV      #WRD3,-(SP)        :
16819 105006 012746 006166          MOV      #FMT3,-(SP)       :
16820 105012 012746 000004          MOV      #4,-(SP)          :
16821 105016 010600          MOV      SP,R0              : SP,*
16822 105020 104414          TRAP    14                  :
16823 105022 004767 175700          2$:  JSR      PC,EOP         :      8606
16824 105026 062706 000012          ADD      #12,SP             :      8589
16825 105032 000207          RTS      PC
  
```

: Routine Size: 36 words
 : Maximum stack depth per invocation: 5 words

16837
16838
16842
16843
16847
16848
16849
16850
16851
16852
16853
16858
16859
16860
16861
16862
16866
16867
16868
16869
16870
16871
16872
16873
16874
16875
16876
16877
16881
16882
16883
16884

105034
105034 004767 177664
105040 104412
105042 000207

```
.SBTTL L$CLEAN CLEANUP CODING SECTION
L$CLEAN::
    JSR    PC,L$CLEAN
    TRAP  12
    RTS    PC
```

8607

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

8609 LASTAD;
8610 BGNSETUP (0);
8611 ENDSETUP;

```
BL$LAS::.WORD T$FREE
          .WORD <<T$FREE-<BL$LAS+4>>/2>
T$FREE::.WORD 0
```

```
L$LAST==          BL$LAS+4
T$PTHV==          0
```

```
.SBTTL SEND.LINK CLEANUP CODING SECTION
SEND.LINK::
    RTS    PC
```

8608

```
; Routine Size: 1 word
```

27-Mar-1982 19:24:42 TOPS
27-Mar-1982 19:23:44 PA:<

```

16886      ;MLX4
16887      ;
16888      ; CLEANUP CODING SECTION
16889      ; Maximum stack depth per invocation: 0 words
16894
16895
16896      :      8612 end
16897      :      8613
16898      :      8614 eludom
16902
16903      ; OTS external references
16904      .GLOBL BL$GT2, $$SAVE5, $$SAVE4, $$SAVE3
16905      .GLOBL $$SAVE2, BL$PU2, BL$$SHF, BL$DIV
16906      .GLOBL BL$MOD, BL$MUL
16907
16908
16909
16910
16911
16912
16913      : Size:          10337 code + 5852 data words
16914      : Run Time:       01:43.8
16915      : Elapsed Time:  02:08.4
16916      : Memory Used:   161 pages
16917      : Compilation Complete
16918
16919      000001          .END

```


SYMBOL TABLE

ADF = 000020 G	C\$AU = 000052	DIVMOD 005006	F\$END = 000041	I\$HRD = 000041
ASSEMB = 000010	C\$AUTO = 000061	DOUBLE 044326	F\$HARD = 000004	I\$INIT = 000041
BANK 034344	C\$BRK = 000022	DRIVE 034442 G	F\$HW = 000013	I\$MOD = 000041
BASE.A 032706	C\$BSEG = 000004	DROPT 002234 G	F\$INIT = 000006	I\$MSG = 000041
BIT.NU 012666	C\$BSUB = 000002	DROPT. 034444 G	F\$JMP = 000050	I\$PROT = 000040
BIT0 = 000001 G	C\$CEFG = 000045	DROP1 002236 G	F\$MOD = 000000	I\$PTAB = 000041
BIT00 = 000001 G	C\$CLCK = 000062	DROP2 002240 G	F\$MSG = 000011	I\$PWR = 000041
BIT01 = 000002 G	C\$CLEA = 000012	DROP3 002242 G	F\$PROT = 000021	I\$RPT = 000041
BIT02 = 000004 G	C\$CLOS = 000035	DROP4 002244 G	F\$PWR = 000017	I\$SEEG = 000041
BIT03 = 000010 G	C\$CLP1 = 000006	DROP5 002250 G	F\$RPT = 000012	I\$SETU = 000041
BIT04 = 000020 G	C\$CVEC = 000036	ECCDIS 002254 G	F\$SEG = 000003	I\$SFT = 000041
BIT05 = 000040 G	C\$DCLN = 000044	EFNS21 002262 G	F\$SOFT = 000005	I\$SRV = 000041
BIT06 = 000100 G	C\$DODU = 000051	EF.CON = 000036 G	F\$SRV = 000010	I\$SUB = 000041
BIT07 = 000200 G	C\$DRPT = 000024	EF.NEW = 000035 G	F\$SUB = 000002	I\$TST = 000041
BIT08 = 000400 G	C\$DU = 000053	EF.PWR = 000034 G	F\$SW = 000014	I.A.M.D 034336
BIT09 = 001000 G	C\$EDIT = 000003	EF.RES = 000037 G	F\$TEST = 000001	J\$JMP = 000167
BIT1 = 000002 G	C\$ERDF = 000055	EF.STA = 000040 G	GEN1 040732	LAU 005650
BIT10 = 002000 G	C\$ERHR = 000056	END.RB = 032670	GEN2 041432	LAUTO 005504
BIT11 = 004000 G	C\$ERRO = 000060	END.WB = 022670	GEN3 041466	L\$CLEAN 104724
BIT12 = 010000 G	C\$ERSF = 000054	EOP 102726 G	GEN4 041564	LDU 005516
BIT13 = 020000 G	C\$ERSO = 000057	EOPSUM 002256 G	GEN5 041620	LIMIT 002222 G
BIT14 = 040000 G	C\$ESCA = 000010	EOP.CO 032704	GETCNT 041106	LINIT 040474
BIT15 = 100000 G	C\$ESEG = 000005	ERRBLK 002200 G	GET.WR 047176	LOE = 040000 G
BIT2 = 000004 G	C\$ESUB = 000003	ERRMSG 002176 G	G\$CNT0 = 000200	LOT = 000010 G
BIT3 = 000010 G	C\$ETST = 000001	ERRNBR 002174 G	G\$DELM = 000372	LOW.SE 034460 G
BIT4 = 000020 G	C\$EXIT = 000032	ERROUT 002260 G	G\$DISP = 000003	LRPT 005466
BIT5 = 000040 G	C\$GETB = 000026	ERRTYP 002172 G	G\$EXCP = 000400	L\$SECT 002226 G
BIT6 = 000100 G	C\$GETW = 000027	EVL = 000004 G	G\$HILI = 000002	L\$ACP 002110 G
BIT7 = 000200 G	C\$GMAN = 000043	E\$END = 002100	G\$LOLI = 000001	L\$APT 002036 G
BIT8 = 000400 G	C\$GPHR = 000042	E\$LOAD = 000035	G\$NO = 000000	L\$AU 005652 G
BIT9 = 001000 G	C\$GPLO = 000030	FILLER 041266	G\$OFFS = 000400	L\$AUT 002070 G
BL\$DIV 005170 G	C\$GPRI = 000040	FMT1A = 006126	G\$OFFSI = 000376	L\$AUTO 005506 G
BL\$GT1 004234 G	C\$INIT = 000011	FMT1B = 006142	G\$PRMA = 000001	L\$CCP 002106 G
BL\$GT2 004356 G	C\$INLP = 000020	FMT10A = 006564	G\$PRMD = 000002	L\$CLEA 105034 G
3L\$LAS 105044 G	C\$MANI = 000050	FMT10B = 006640	G\$PRML = 000000	L\$CO 002032 G
BL\$MOD 005202 G	C\$MEM = 000031	FMT11 = 006654	G\$RADA = 000140	L\$DEPO 002011 G
BL\$MUL 004744 G	C\$MSG = 000023	FMT12A = 006710	G\$RADB = 000100	L\$DESC 002130 G
BL\$PU1 004520 G	C\$OPEN = 000034	FMT12B = 006756	G\$RADL = 000040	L\$DESP 002076 G
BL\$PU2 004614 G	C\$PNTB = 000014	FMT13 = 007024	G\$RADL = 000120	L\$DEVP 002060 G
BL\$SHF 005214 G	C\$PNTF = 000017	FMT14 = 007046	G\$RADO = 000020	L\$DISP 002204 G
BOARD 034342	C\$PNTS = 000016	FMT15 = 007060	G\$XFER = 000004	L\$DLY 002116 G
BOE = 000400 G	C\$PNTX = 000015	FMT16 = 005736	G\$YES = 000010	L\$DTP 002040 G
BR.LEV 032712	C\$QIO = 000377	FMT2 = 006160	HARDS 033314	L\$DTYP 002034 G
CAUSE1 = 010460	C\$RDBU = 000007	FMT3 = 006166	HELP = 000000	L\$DU 005640 G
CAUSE2 = 010502	C\$REFG = 000047	FMT4 = 006216	HOE = 100000 G	L\$DUT 002072 G
CAUSE3 = 010526	C\$RESE = 000033	FMT4A = 006252	IBE = 010000 G	L\$DVTY 002122 G
CAUSE4 = 010602	C\$REVI = 000003	FMT4C = 006270	IDU = 000040 G	L\$EF 002052 G
CAUSE5 = 010656	C\$RFLA = 000021	FMT5 = 006320	IER = 020000 G	L\$ENVI 002044 G
CAUSE6 = 010710	C\$RPT = 000025	FMT6 = 006376	INIT.A 035206	L\$ERRT 002172 G
CAUSE7 = 010734	C\$SEFG = 000046	FMT7 = 006446	INTEGR 047232	L\$ETP 002102 G
CAUSE8 = 010756	C\$SPRI = 000041	FMT8 = 006466	ISOLAT 037332	L\$EXP1 002046 G
CHECK 046000	C\$SVEC = 000037	FMT9 = 006534	ISR = 000100 G	L\$EXP4 002064 G
CHOOSE 046166	C\$TPRI = 000013	F\$AU = 000015	IXE = 004000 G	L\$EXP5 002066 G
CLRTBL 034554	DATA.C 032700	F\$AUTO = 000020	ISAU = 000041	L\$HARD 002266 G
COMP.C 032702	DECODE 037176	F\$BGN = 000040	ISAUTO = 000041	L\$HIME 002120 G
CONFIG 036002	DFPTBL 002210 G	F\$CLEA = 000007	ISCLN = 000041	L\$HPCP 002016 G
CRLF = 007066	DIAGMC = 000000	F\$DU = 000016	ISDU = 000041	L\$HPTP 002022 G

SYMBOL TABLE

L\$HW	002210	G	MSG4	=	011202	P.AAD	006126	P.ACI	007500	P.AEN	011070		
L\$ICP	002104	G	MSG5	=	011216	P.AAE	006142	P.ACJ	007504	P.AEO	011120		
L\$INIT	040722	G	NUM.DR	034456	G	P.AAF	006160	P.ACK	007510	P.AEP	011166		
L\$LADP	002026	G	ONEFIL	=	000001	P.AAG	006166	P.ACL	007514	P.AEQ	011202		
L\$LAST	=	105050	ONLY	=	002232	G	P.AAH	006216	P.ACM	007520	P.AER	011216	
L\$LOAD	002100	G	OPT1	051460		P.AAI	006252	P.ACN	007524	QH1	002332		
L\$LUN	002074	G	OPT2	054054		P.AAJ	006270	P.ACO	007530	QH2	002351		
L\$MREV	002050	G	OPT3	060052		P.AAK	006320	P.ACP	007534	QH3	002374		
L\$NAME	002000	G	OPT4	062444		P.AAL	006376	P.ACQ	007542	QH4	002425		
L\$PRIO	002042	G	OPT5	102272		P.AAM	006446	P.ACR	007550	QS1	002664		
L\$PROT	004126	G	OSAPTS	=	000001	P.AAN	006466	P.ACS	007554	QS10	003534		
L\$PRT	002112	G	OSAU	=	000001	P.AAO	006534	P.ACT	007560	QS11	003554		
L\$REPP	002062	G	OSBGNR	=	000001	P.AAP	006564	P.ACU	007564	QS12	003637		
L\$REV	002010	G	OSBGNS	=	000001	P.AAQ	006640	P.ACV	007570	QS13	003657		
L\$RPT	005474	G	OSDU	=	000001	P.AAR	006654	P.ACW	007574	QS14	003724		
L\$SOFT	002450	G	OSERRT	=	000001	P.AAS	006710	P.ACX	007626	QS15	003763		
L\$SPC	002056	G	OSGNSW	=	000001	P.AAT	006756	P.ACY	007634	QS16	004030		
L\$SPCP	002020	G	OSPOIN	=	000001	P.AAU	007024	P.ACZ	007642	QS17	004066		
L\$SPTP	002024	G	OSSETU	=	000001	P.AAV	007046	P.ADA	007650	QS2	002732		
L\$STA	002030	G	PATTBL	034520	G	P.AAW	007060	P.ADB	007656	QS3	003007		
L\$SW	002222	G	PATTER	032676		P.AAX	007066	P.ADC	007664	QS4	003047		
L\$TEST	002114	G	PHR1	=	007714	P.AAY	007072	P.ADD	007672	QS5	003107		
L\$TIML	002014	G	PHR10	=	010120	P.AAZ	007100	P.ADE	007700	QS6	003147		
L\$UNIT	002012	G	PHR11	=	010140	P.ABA	007112	P.ADF	007706	QS7	003454		
L10000	002220		PHR12	=	010202	P.ABB	007130	P.ADG	007714	QS8	003474		
L10001	002264		PHR13	=	010240	P.ABC	007152	P.ADH	007730	QS9	003514		
L10002	002332		PHR14	=	010266	P.ABD	007200	P.ADI	007746	QUICK	032674		
L10003	002664		PHR15	=	010304	P.ABE	007206	P.ADJ	007770	RANDOM	005464	G	
MARPAT	002246	G	PHR16	=	010322	P.ABF	007212	P.ADK	010004	RAND1	070540		
MLB10	=	007472	PHR17	=	010342	P.ABG	007220	P.ADL	010026	RAND2	073124		
MLB11	=	007500	PHR18	=	010366	P.ABH	007230	P.ADM	010050	RAND3	075444		
MLB12	=	007504	PHR19	=	010410	P.ABI	007240	P.ADN	010066	RAND4	100006		
MLB13	=	007510	PHR2	=	007730	P.ABJ	007246	P.ADO	010100	RANGE	002224	G	
MLB14	=	007514	PHR20	=	010450	P.ABK	007256	P.ADP	010120	RBUFF	022670		
MLB15	=	007520	PHR21	=	005662	P.ABL	007264	P.ADQ	010140	RCBUFF	=	023670	
MLB16	=	007524	PHR3	=	007746	P.ABM	007272	P.ADR	010202	RDBUFF	=	022670	
MLB17	=	007530	PHR4	=	007770	P.ABN	007300	P.ADS	010240	RD.COU	034322		
MLB18	=	007534	PHR5	=	010004	P.ABO	007312	P.ADT	010266	RD.MIL	034326		
MLB19	=	007542	PHR6	=	010026	P.ABP	007322	P.ADU	010304	RD.THO	034324		
MLB2	=	007430	PHR7	=	010050	P.ABQ	007332	P.ADV	010322	READ	045612		
MLB20	=	007550	PHR8	=	010066	P.ABR	007336	P.ADW	010342	REFRES	002252	G	
MLB21	=	007554	PHR9	=	010100	P.ABS	007344	P.ADX	010366	RETRY	046224		
MLB22	=	007560	PM.SBE	012264		P.ABT	007354	P.ADY	010410	RETRYI	034340		
MLB23	=	007564	PNT	=	001000	G	P.ABU	007370	P.ADZ	010430	RE2	005362	
MLB24	=	007570	PRI	=	002000	G	P.ABV	007400	P.AEA	010444	RE3	005360	
MLB3	=	007434	PRI00	=	000000	G	P.ABW	007412	P.AEB	010446	RE4	005356	
MLB4	=	007440	PRI01	=	000040	G	P.ABX	007416	P.AEC	010450	RN	005370	G
MLB5	=	007444	PRI02	=	000100	G	P.ABY	007424	P.AED	010454	RNDSEC	070240	
MLB6	=	007450	PRI03	=	000140	G	P.ABZ	007430	P.AEE	010460	RNDU	070336	
MLB7	=	007454	PRI04	=	000200	G	P.ACA	007434	P.AEF	010502	RNDWC	070204	
MLB8	=	007460	PRI05	=	000240	G	P.ACB	007440	P.AEG	010526	RPTR	032672	
MLB9	=	007464	PRI06	=	000300	G	P.ACC	007444	P.AEH	010602	RTN0	=	007574
ML.REG	034346	G	PRI07	=	000340	G	P.ACD	007450	P.AEI	010656	RTN1	=	007626
MSG0	=	011002	PTABLE	034422	G	P.ACE	007454	P.AEJ	010710	RTN2	=	007634	
MSG1	=	011070	P.AAA	005662		P.ACF	007460	P.AEK	010734	RTN3	=	007642	
MSG2	=	011120	P.AAB	005736		P.ACG	007464	P.AEL	010756	RTN4	=	007650	
MSG3	=	011166	P.AAC	006042		P.ACH	007472	P.AEM	011002	RTN5	=	007656	

RTNSA = 007664	SVCSUB= 000001	T\$NS1 = 000021	VEC = 032710	WRD36 = 007370
RTNSB = 007672	SVCTAG= 000001	T\$PTHV= 000000 G	WAITER 044420	WRD37 = 007400
RTNSC = 007700	SVCTST= 000001	T\$PTNU= 000000	W\$BUFF 012670	WRD38 = 007412
RTNSD = 007706	SYSERR 041710	T\$SAVL= 177777	WC\$BUFF= 013670	WRD4 = 007212
SAYWHO 035524	\$SLSYM= 010000	T\$SEGL= 177777	WC.COU 034330	WRD40 = 007416
SAY1 = 007072	TOP.SE 034500 G	T\$SUBN= 000000	WC.MIL 034334	WRD41 = 007424
SAY2 = 007100	TRIES 033714	T\$TAGL= 177777	WC.THO 034332	WRD6 = 007220
SAY3 = 007112	TRT00 = 010444	T\$TAGN= 010005	WDBUFF= 012670	WRD7 = 007230
SAY4 = 007130	TRT01 = 010446	T\$TEMP= 000000	WHY.DR 034446 G	WRITE 045424
SAY5 = 007152	TRT10 = 010450	T\$TEST= 000000	WPTR 032670	WR.COU 034314
SBESHE= 006042	TRT11 = 010454	T\$TSTM= 177777	WRD11 = 007240	WR.MIL 034320
SBES.C 012664	TSECT 002230 G	T\$TSTS= 000000	WRD15 = 007246	WR.THO 034316
SBE.LO 011264	T\$ARGC= 000002	T\$SHAR= 010002	WRD16 = 007256	X\$ALWA= 000000
SEED1 005456 G	T\$CODE= 020130	T\$SHW = 010000	WRD17 = 007264	X\$FALS= 000040
SEED2 005460 G	T\$ERRN= 000000	T\$SPRO= 010004	WRD18 = 007272	X\$OFFS= 000400
SEED3 005462 G	T\$EXCP= 000000	T\$SOF = 010003	WRD19 = 007300	X\$TRUE = 000020
SELPAT 041066	T\$FREE 105050 G	T\$SSW = 010001	WRD2 = 007200	\$END.L 105052 G
SERVIC 034544 G	T\$GMAN= 000000	T1 102712 G	WRD20 = 007312	\$PATCH 004134 G
SET.PT 047132	T\$HILI= 000012	UAM = 000200 G	WRD21 = 007322	\$SAVE2 005262 G
SFPTBL 002222 G	T\$LAST= 000000	UP.HAR 037422	WRD24 = 007332	\$SAVE3 005276 G
SOFTS 032714	T\$LOLI= 000001	UP.RD. 045240	WRD25 = 007336	\$SAVE4 005314 G
START. 044424	T\$LSYM= 010000	UP.SOF 040104	WRD3 = 007206	\$SAVE5 005334 G
SVCGBL= 000001	T\$NEST= 177777	UP.WC. 045332	WRD34 = 007344	\$T1 102536
SVCINS= 000001	T\$NS0 = 000000	UP.WR. 045146	WRD35 = 007354	

. ABS. 105054 000
000000 001

ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 31919 WORDS (125 PAGES)
 DYNAMIC MEMORY: 21558 WORDS (82 PAGES)
 ELAPSED TIME: 00:09:52
 CZMLBB.BIN,CZMLBB/CR/-SP=SVC/ML,CZMLBB.DGC,MLX2,OTS,RANDOM,MLX3,MLX4

SYMBOL	VALUE	CROSS REFERENCE	REFERENCES
ADR	= 000020	G	#83-1773
ASSEMB	= 000010		6-13
BANK	034344		#82-1685 *98-2978 *99-2991 100-3052 103-3237 105-3302 108-3485 109-3530 155-6371
BASE.A	032706		*155-6383
BIT.NU	012666		#81-1650 *88-2289 89-2338 97-2909
BIT0	= 000001	G	#81-1637 *99-3005 103-3226 104-3290 108-3474 109-3522
BIT00	= 000001	G	#83-1754
BIT01	= 000002	G	#83-1744
BIT02	= 000004	G	#83-1743
BIT03	= 000010	G	#83-1742
BIT04	= 000020	G	#83-1741
BIT05	= 000040	G	#83-1740
BIT06	= 000100	G	#83-1739
BIT07	= 000200	G	#83-1738
BIT08	= 000400	G	#83-1737
BIT09	= 001000	G	#83-1736
BIT1	= 000002	G	#83-1735
BIT10	= 002000	G	#83-1753
BIT11	= 004000	G	#83-1734
BIT12	= 010000	G	#83-1733
BIT13	= 020000	G	#83-1732
BIT14	= 040000	G	#83-1731
BIT15	= 100000	G	#83-1730
BIT2	= 000004	G	#83-1729
BIT3	= 000010	G	#83-1752
BIT4	= 000020	G	#83-1751
BIT5	= 000040	G	#83-1750
BIT6	= 000100	G	#83-1749
BIT7	= 000200	G	#83-1748
BIT8	= 000400	G	#83-1747
BIT9	= 001000	G	#83-1746
BL\$DIV	005170	G	#83-1745
BL\$GT1	004234	G	#37-1170 185-8022 205-9118 209-9381 225-10222 273-13132 282-13730 297-14575 298-14593
BL\$GT2	004356	G	313-15422 341-16905
BL\$LAS	105044	G	#13-169
BL\$MOD	005202	G	#18-384 164-6887 180-7728 200-8836 205-9143 220-9931 242-11178 265-12395 273-12846
BL\$MUL	004744	G	281-13705 297-14560 322-15915 333-16485 341-16904
BL\$PU1	004520	G	#340-16868 340-16869 340-16873
BL\$PU2	004614	G	#38-1201 118-4116 261-12206 263-12297 265-12380 265-12404 341-16906
BL\$SHF	005214	G	#34-1022 95-2787 341-16906
BOARD	034342		#23-592
			#29-829 47-220 47-231 50-339 85-2133 85-2140 111-3672 112-3685 118-4112
			341-16905
			13-181 13-184 13-188 13-192 14-217 19-406 19-409 19-413 19-417
			19-438 23-595 23-599 23-605 24-632 29-838 29-842 29-848 30-880
			#39-1236 88-2294 341-16905
			#82-1684 *99-2996 100-3053 104-3246 104-3271 105-3306 108-3490 109-3534 155-6370
			155-6379 *155-6382 168-7119 169-7140 171-7267 171-7284 185-7996 185-8013 187-8108
			187-8125 204-9088 205-9109 209-9350 209-9367 211-9467 211-9484 225-10196 225-10213
			227-10308 227-10325 247-11460 248-11481 252-11739 253-11760 257-12018 258-12039 259-12125
			259-12142 273-13106 273-13123 273-13216 273-13233 286-13980 287-14001 288-14085 289-14106
			302-14838 302-14855 304-14946 304-14963 318-15671 318-15688 319-15773 320-15794

SYMBOL	CROSS REFERENCE	VALUE	REFERENCES
BOE	=	000400 G	#83-1777
BR.LEV	=	032712	#81-1653 *88-2291 88-2292
CAUSE1	=	010460	#83-1905 94-2733 333-16499
CAUSE2	=	010502	#83-1906 94-2756 334-16511
CAUSE3	=	010526	#83-1907 96-2831 96-2858 334-16519
CAUSE4	=	010602	#83-1908 334-16527
CAUSE5	=	010656	#83-1909 334-16535
CAUSE6	=	010710	#83-1910 334-16543
CAUSE7	=	010734	#83-1911 334-16551
CAUSE8	=	010756	#83-1912 335-16563
CHECK	=	046000	#147-5961 149-6059 153-6265 154-6301 154-6312 169-7147 169-7152 171-7238 182-7826 201-8923 206-9185 222-10026 244-11290 249-11569 254-11848 273-12939 283-13815 299-14673 302-14809 315-15506
CHOOSE	=	046166	#149-6052 181-7804 201-8905 206-9166 221-10004 244-11269 249-11548 254-11827 273-12918 283-13797 299-14655 314-15484
CLRTBL	=	034554	#84-2078 111-3644
COMP.C	=	032702	#81-1646 *116-3994 118-4102
CONFIG	=	036002	#93-2703 112-3678
CRLF	=	007066	#83-1814 88-2301 89-2330 154-6342 325-16041 332-16410 332-16419 337-16696 337-16705 338-16758
C\$AU	=	000052	#6-13
C\$AUTO	=	000061	#6-13
C\$BRK	=	000022	#6-13
C\$BSEG	=	000004	#6-13
C\$BSUB	=	000002	#6-13
C\$CEFG	=	000045	#6-13
C\$CLCK	=	000062	#6-13
C\$CLEA	=	000012	#6-13
C\$CLOS	=	000035	#6-13
C\$CLP1	=	000006	#6-13
C\$CVEC	=	000036	#6-13
C\$DCLN	=	000044	#6-13
C\$DODU	=	000051	#6-13
C\$DRPT	=	000024	#6-13
C\$DU	=	000053	#6-13
C\$EDIT	=	000003	#6-13
C\$ERDF	=	000055	#6-13
C\$ERHR	=	000056	#6-13
C\$ERRO	=	000060	#6-13
C\$ERSF	=	000054	#6-13
C\$ERSO	=	000057	#6-13
C\$ESCA	=	000010	#6-13
C\$ESEG	=	000005	#6-13
C\$ESUB	=	000003	#6-13
C\$ETST	=	000001	#6-13
C\$EXIT	=	000032	#6-13
C\$GETB	=	000026	#6-13
C\$GETW	=	000027	#6-13
C\$GMAN	=	000043	#6-13
C\$GPHR	=	000042	#6-13
C\$GPLO	=	000030	#6-13
C\$GPRI	=	000040	#6-13

6-65

SYMBOL	CROSS REFERENCE	VALUE	REFERENCES
CSINIT	=	000011	#6-13
CSINLP	=	000020	#6-13
CSMANI	=	000050	#6-13
CSMEM	=	000031	#6-13
CSMSG	=	000023	#6-13
CSOPEN	=	000034	#6-13
CSPNTB	=	000014	#6-13
CSPNTF	=	000017	#6-13
CSPNTS	=	000016	#6-13
CSPNTX	=	000015	#6-13
CSQIO	=	000377	#6-13
CSRDBU	=	000007	#6-13
CSREFG	=	000047	#6-13
CSRESE	=	000033	#6-13 #6-13
CSREVI	=	000003	#6-13 6-65
CSRFLA	=	000021	#6-13
CSRPT	=	000025	#6-13
CSSFFG	=	000046	#6-13
CSSPRI	=	000041	#6-13
CSSVEC	=	000037	#6-13
CSTPRI	=	000013	#6-13
DATA.C		032700	#81-1644 *116-3993 118-4104
DECODE		037176	#98-2972 100-3049 155-6373
DGPTBL		002210	G #7-117
DIAGMC	=	000000	G 6-13 6-13
DIVMOD		005006	#35-1091 37-1171 38-1202
DOUBLE		044326	#133-5198 166-6966 182-7835 201-8931 206-9193 222-10035 244-11299 249-11578 254-11857
DRIVE.		034442	G 273-12948 284-13828 299-14682 315-15515
DROPNE		002234	G #8-160 83-1723 85-2146
DROPT.		034444	G 46-201 47-225 #82-1695 85-2135 333-16479
DROP1		002236	G #8-161 83-1723 *85-2148 325-16054
DROP2		002240	G #8-162 83-1724 *85-2149 325-16057
DROP3		002242	G #8-163 83-1724 *85-2150 325-16060
DROP4		002244	G #8-164 83-1724 *85-2151 325-16065
DROP5		002250	G #8-166 83-1724 *85-2152 325-16068
ECCDIS		002254	G #8-169 83-1725 88-2296 131-4772 131-4807 131-4858 131-4956 138-5458
EFNS21		002262	G #8-192 83-1722 185-8027 210-9390 225-10227 258-12048 273-13137 287-14008 303-14869
EF.CON	=	000036	G 318-15697 #83-1757
EF.NEW	=	000035	G #83-1758
EF.PWR	=	000034	G #83-1759
EF.RES	=	000037	G #83-1756
EF.STA	=	000040	G #83-1755
END.RB	=	032670	#83-1790
END.WB	=	022670	#83-1789 156-6439
EOP		102726	G 44-31 44-39 #332-16405 339-16823
EOPSUM		002256	G #8-188 83-1725 332-16407
EOP.CO		032704	#81-1648 *84-2101 *339-16805 339-16806 339-16816
ERRBLK		002200	G #7-94
ERRMSG		002176	G #7-94

SYMBOL	CROSS REFERENCE VALUE	REFERENCES	6-13	6-13	6-13	6-13	6-13	6-13	6-13	6-39
FSHARD	= 000004	#6-13	6-13	6-13	6-13	6-13	6-13	6-13	6-13	6-39
FSHW	= 000013	8-277	8-339	9-411	8-314	8-316	8-319	8-322	8-324	8-329
FSINIT	= 000006	#6-13	7-117	7-134						
FSJMP	= 000050	#6-13								
FSMOD	= 000000	#6-13	6-39	9-411						
FSMSG	= 000011	#6-13								
FSPROT	= 000021	#6-13	9-379	9-385						
FSPWR	= 000017	#6-13								
FSRPT	= 000012	#6-13								
FSSEG	= 000003	#6-13								
FSSOFT	= 000005	#6-13	8-302	8-314	8-316	8-319	8-322	8-324	8-329	8-339
FSSRV	= 000010	#6-13								
FSSUB	= 000002	#6-13								
FSSW	= 000014	#6-13	8-143	8-196						
FSTEST	= 000001	#6-13								
GEN1	040732	#114-3793	180-7743							
GEN2	041432	#120-4173	199-8818							
GEN3	041466	#121-4241	164-6873	220-9917						
GEN4	041564	#123-4310	242-11156	242-11160						
GEN5	041620	#125-4371	273-12832	281-13694	297-14549	313-15397				
GETCNT	041106	#116-3959	120-4174	123-4311						
GET.WR	047176	#157-6486	165-6903	180-7746	200-8850	205-9158	220-9946	273-12861		
GSCNTO	= 000200	#6-13								
GSDLM	= 000372	#6-13								
GSDISP	= 000003	#6-13								
GSEXCP	= 000400	#6-13								
GSHIL!	= 000002	#6-13								
GSLOLI	= 000001	#6-13								
GSNO	= 000000	#6-13								
GSOFFS	= 000400	#6-13	8-272	8-273	8-274	8-275	8-313	8-315	8-317	8-318
		8-320	8-321	8-323	8-325	8-326	8-327	8-328	8-330	8-331
		8-332	8-333	8-334	8-335	8-336				
GSOFSI	= 000376	#6-13	8-272	8-273	8-274	8-275	8-313	8-315	8-317	8-318
		8-320	8-321	8-323	8-325	8-326	8-327	8-328	8-330	8-331
		8-332	8-333	8-334	8-335	8-336				
GSPRMA	= 000001	#6-13	8-272	8-273						
GSPRMD	= 000002	#6-13	8-274	8-275	8-317	8-318	8-320	8-323	8-330	
GSPRML	= 000000	#6-13	8-313	8-315	8-321	8-325	8-326	8-327	8-328	8-331
		8-332	8-333	8-334	8-335	8-336				
GSRADA	= 000140	#6-13								
GSRADB	= 000000	#6-13								
GSRADD	= 000040	#6-13	8-320	8-323	8-330					
GSRADL	= 000120	#6-13	8-313	8-315	8-321	8-325	8-326	8-327	8-328	8-331
		8-332	8-333	8-334	8-335	8-336				
GSRADO	= 000020	#6-13	8-272	8-273	8-274	8-275	8-317	8-318		
G\$XFER	= 000004	#6-13	8-314	8-316	8-319	8-322	8-324	8-329		
G\$YES	= 000010	#6-13	8-272	8-273	8-274	8-275	8-313	8-315	8-317	8-318
		8-320	8-321	8-323	8-325	8-326	8-327	8-328	8-330	8-331
		8-332	8-333	8-334	8-335	8-336				
HARDS	033314	#82-1660	*84-2092	*103-3214	335-16584	336-16629				

CZMLBB
SYMBOL CROSS REFERENCE
SYMBOL VALUE
HELP = 000000

CREATED BY MACRO ON 29-MAR-82 AT 13:44

PAGE 6
REF

SEQ 0369

SYMBOL	VALUE	REFERENCE	6-8	6-30	6-48	6-67	7-103	7-119	8-145	#8-200
HOE	= 100000	G #83-1784								
IBE	= 010000	G #83-1781								
IDU	= 000040	G #83-1774								
IER	= 020000	G #83-1782								
INIT.A	035206	#88-2286		111-3664						
INTEGR	047232	#164-6862		325-16052						
ISOLAT	037332	#100-3048		167-7046	167-7058	170-7195	170-7206	183-7918	184-7935	186-8047
		203-9027		208-9277	208-9289	210-9406	223-10118	224-10135	226-10247	246-11386
		251-11665		251-11678	256-11944	256-11957	258-12064	273-13030	273-13046	273-13156
		285-13919		287-14024	301-14765	301-14777	303-14885	316-15594	316-15606	318-15712
ISR	= 000100	G #83-1775								
IXE	= 004000	G #83-1780								
ISAU	= 000041	#6-13								
ISAUTO	= 000041	#6-13								
ISCLN	= 000041	#6-13								
ISDU	= 000041	#6-13								
ISHRD	= 000041	#8-259	#8-277							
ISINIT	= 000041	#6-13								
ISMOD	= 000041	#6-13	6-39	#6-39	9-411	#9-411				
ISMSG	= 000041	#6-13								
ISPROT	= 000040	#6-13	#9-379							
ISPTAB	= 000041	#6-13								
ISPR	= 000041	#6-13								
ISRPT	= 000041	#6-13								
ISSEG	= 000041	#6-13								
ISSETU	= 000041	#6-13								
ISSFT	= 000041	#8-302	#8-339							
ISSRV	= 000041	#6-13								
ISSUB	= 000041	#6-13								
ISTST	= 000041	#6-13								
I.A.M.D	034336	#82-1680	*83-1927	*138-5465	138-5471	138-5477				
JSJMP	= 000167	#6-13								
LAU	005650	#48-296	49-315							
LAUTO	005504	#45-101	45-119							
LCLEAN	104724	#339-16805	340-16848							
LDU	005516	#47-210	47-257							
LIMIT	002222	G #8-153	83-1722	95-2794						
LINIT	040474	#111-3636	112-3710							
LOE	= 040000	G #83-1783								
LOT	= 000010	G #83-1772								
LOW.SE	034460	G #82-1702	*85-2122	*96-2850	97-2881	164-6894	180-7738	185-8033	200-8845	205-9153
		210-9396	220-9941	225-10233	243-11200	258-12054	263-12285	273-12856	273-13143	282-13717
		287-14014	297-14573	303-14875	318-15702					
		#44-39	44-58							
LQPT	005466	#8-157	83-1723	*95-2806	*95-2814	96-2848	96-2850	96-2865		
LSECT	002226	G #6-65								
LSACP	002110	G #6-65								
LSAPT	002036	G #6-65								
LSAU	005652	G #6-65	#49-315							
LSAUT	002070	G #6-65								
LSAUTO	005506	G #6-65	#45-119							

SYMBOL	VALUE		REFERENCES															
L\$CCP	002106	G	#6-65															
L\$CLEA	105034	G	6-65	#340-16847														
L\$CO	002032	G	#6-65															
L\$DEPO	002011	G	#6-65															
L\$DESC	002130	G	6-65	#7-85														
L\$DESP	002076	G	#6-65															
L\$DEVP	002060	G	#6-65															
L\$DISP	002204	G	6-65	#7-101														
L\$DLY	002116	G	#6-65	137-5400	137-5428	138-5442												
L\$DTP	002040	G	#6-65															
L\$DTYP	002034	G	#6-65															
L\$DU	005640	G	6-65	#47-257														
L\$DUT	002072	G	#6-65															
L\$DVTY	002122	G	6-65	#6-80														
L\$EF	002052	G	#6-65															
L\$ENVI	002044	G	#6-65															
L\$ERRT	002172	G	6-65	#7-94														
L\$ETP	002102	G	#6-65															
L\$EXP1	002046	G	#6-65															
L\$EXP4	002064	G	#6-65															
L\$EXP5	002066	G	#6-65															
L\$HARD	002266	G	6-65	8-259	#8-259													
L\$HIME	002120	G	#6-65															
L\$HPCP	002016	G	#6-65															
L\$HPTP	002022	G	#6-65															
L\$HW	002210	G	6-65	7-117	#7-117													
L\$ICP	002104	G	#6-65															
L\$INIT	040722	G	6-65	#112-3710														
L\$LADP	002026	G	#6-65															
L\$LAST	= 105050	G	6-65	#340-16873														
L\$LOAD	002100	G	#6-65															
L\$LUN	002074	G	#6-65	46-201	47-210	47-216	47-221	47-227	83-1722	*85-2119	*111-3648							
			*164-6891	*180-7733	*200-8840	*205-9148	*220-9936	*242-11183	*273-12851	*282-13714	*297-14565							
			*313-15405															
L\$MREV	002050	G	#6-65															
L\$NAME	002000	G	#6-65															
L\$P7IO	002042	G	#6-65															
L\$PROT	004126	G	6-65	#9-379														
L\$PRT	002112	G	#6-65															
L\$REPP	002062	G	#6-65															
L\$REV	002010	G	#6-65															
L\$RPT	005474	G	6-65	#44-58														
L\$SOFT	002450	G	6-65	8-302	#8-302													
L\$SPC	002056	G	#6-65															
L\$SPCP	002020	G	#6-65															
L\$SPTP	002024	G	#6-65															
L\$STA	002030	G	#6-65															
L\$SW	002222	G	6-65	8-143	#8-143													
L\$TEST	002114	G	#6-65															
L\$TIML	002014	G	#6-65															
L\$UNIT	002012	G	#6-65	83-1722	84-2079	111-3645	164-6874	180-7715	199-8819	205-9130	220-9918							
			242-11165	265-12372	265-12379	265-12382	265-12403	273-12833	281-13691	297-14546	322-15902							

SYMBOL	CROSS REFERENCE	VALUE	REFERENCES
L10000		002220	333-16450
L10001		002264	7-117 #7-134
L10002		002332	8-143 #8-196
L10003		002664	8-259 #8-277
MARPAT	G	002246	8-302 #8-339
MLB10	=	007472	#8-165 83-1724 242-11154 242-11157
MLB11	=	007500	#83-1853 131-4992
MLB12	=	007504	#83-1854 131-5002
MLB13	=	007510	#83-1855 131-5013
MLB14	=	007514	#83-1856 131-5021
MLB15	=	007520	#83-1857 131-5029
MLB16	=	007524	#83-1858 131-5037
MLB17	=	007530	#83-1859 131-5045
MLB18	=	007534	#83-1860 131-5055
MLB19	=	007542	#83-1861 131-5066
MLB2	=	007430	#83-1862 131-5074
MLB20	=	007550	#83-1845 131-4919
MLB21	=	007554	#83-1863 131-4831
MLB22	=	007560	#83-1864 131-4870
MLB23	=	007564	#83-1865 131-4839
MLB24	=	007570	#83-1866 131-4850
MLB3	=	007434	#83-1867 131-4878
MLB4	=	007440	#83-1846 131-4927
MLB5	=	007444	#83-1847 131-4935
MLB6	=	007450	#83-1848 131-4945
MLB7	=	007454	#83-1849 131-4860 131-4958
MLB8	=	007460	#83-1850 131-4966
MLB9	=	007464	#83-1851 131-4974
ML.REG	G	034346	#83-1852 131-4984
			#82-1689 *89-2340 90-2411 90-2421 90-2424 90-2430 90-2451 94-2714 94-2715
			94-2716 94-2745 95-2770 95-2780 95-2798 97-2887 98-2972 99-2997 100-3048
			100-3054 131-4732 131-4738 131-4740 131-4742 131-4745 131-4747 131-4749 131-4751
			131-4753 131-4756 131-4758 131-4761 131-4763 131-4765 131-4767 131-4770 131-4775
			131-4777 131-4779 131-4781 131-4783 131-4785 131-4791 131-4793 131-4795 131-4798
			131-4800 131-4802 131-4805 131-4810 131-4812 131-4829 131-4837 131-4848 131-4856
			131-4868 131-4876 131-4917 131-4925 131-4933 131-4943 131-4951 131-4964 131-4972
			131-4980 131-4982 131-4990 131-5000 131-5011 131-5019 131-5027 131-5035 131-5043
			131-5053 131-5061 131-5072 137-5416 137-5417 137-5422 137-5425 138-5439 138-5452
			138-5453 138-5454 138-5457 138-5460 138-5461 138-5462 138-5463 138-5464 138-5469
			138-5474 167-7061 168-7079 168-7080 168-7099 168-7102 170-7209 170-7223 170-7224
			171-7247 171-7250 184-7938 184-7952 184-7953 184-7972 184-7975 186-8050 186-8064
			186-8065 186-8084 186-8087 203-9030 203-9044 203-9045 204-9068 204-9071 208-9292
			208-9306 208-9307 208-9326 209-9333 210-9409 210-9423 210-9424 211-9447 211-9450
			224-10138 224-10152 224-10153 224-10172 224-10175 226-10250 226-10264 226-10265 226-10284
			226-10287 246-11402 247-11420 247-11421 247-11440 247-11443 251-11681 251-11695 252-11700
			252-11719 252-11722 256-11960 256-11974 256-11975 257-11998 257-12001 258-12067 258-12081
			258-12082 259-12105 259-12108 273-13049 273-13063 273-13064 273-13083 273-13086 273-13159
			273-13173 273-13174 273-13193 273-13196 285-13922 286-13940 286-13941 286-13960 286-13963
			287-14027 287-14041 287-14042 288-14065 288-14068 301-14780 301-14794 301-14795 302-14818
			302-14821 303-14888 303-14902 303-14903 304-14926 304-14929 317-15013 317-15627 317-15628
			317-15647 317-15650 318-15715 319-15733 319-15734 319-15753 319-15756 322-15924 322-15925
			322-15926 333-16458 333-16459 333-16460 336-16637 336-16647

SYMBOL	CROSS REFERENCE	VALUE	REFERENCES
MSG0	=	011002	#83-1913 139-5504
MSG1	=	011070	#83-1914 165-6926 165-6936 165-6945 166-6993 167-7019 167-7029 167-7038 169-7164 169-7174 170-7187 181-7776 181-7787 181-7796 182-7863 183-7889 183-7900 183-7911 200-8875 201-8889 201-8898 202-8962 202-8984 203-8998 203-9007 207-9224 207-9246 207-9256 207-9265 221-9976 221-9987 221-9996 222-10063 223-10089 223-10100 223-10109 243-11229 243-11240 244-11253 245-11331 245-11353 246-11368 246-11377 248-11506 248-11517 248-11526 250-11610 250-11632 251-11647 251-11656 253-11785 253-11795 253-11805 255-11889 255-11911 256-11926 256-11935 273-12890 273-12901 273-12910 273-12915 273-13001 273-13012 273-13021 283-13771 283-13781 283-13790 284-13855 284-13876 285-13890 285-13899 298-14625 298-14635 299-14648 300-14713 300-14734 300-14744 301-14757 314-15458 314-15468 314-15477 315-15542 316-15567 316-15577 316-15586
MSG2	=	011120	#83-1915 104-3278 167-7049 170-7198 183-7921 203-9018 208-9280 223-10121 246-11389 251-11668 256-11947 273-13033 285-13910 301-14768 316-15597 336-16641 336-16651
MSG3	=	011166	#83-1916 169-7130 169-7137 171-7274 171-7281 185-8003 185-8010 187-8115 187-8122 204-9095 204-9102 209-9357 209-9364 211-9474 211-9481 225-10203 225-10210 227-10315 227-10322 247-11467 248-11478 252-11746 253-11757 257-12025 258-12036 259-12132 259-12139 273-13113 273-13120 273-13223 273-13230 286-13987 287-13998 288-14092 288-14099 302-14845 302-14852 304-14953 304-14960 318-15678 318-15685 320-15784 320-15791
MSG4	=	011202	#83-1917 168-7116 171-7264 185-7993 187-8105 204-9085 209-9347 211-9464 225-10193 227-10305 247-11457 252-11736 257-12015 259-12122 273-13103 273-13213 286-13977 288-14082 302-14835 304-14943 317-15664 319-15770
MSG5	=	011216	#83-1918 166-6972 182-7841 202-8941 206-9199 222-10041 245-11309 250-11588 254-11863 273-12954 284-13834 299-14688 315-15521 46-202 *47-232 #82-1700 *84-2102 111-3661 *111-3666 313-15398 322-15901
NUM. DR		034456	G 46-202 *47-232 #82-1700 *84-2102 111-3661 *111-3666 313-15398 322-15901
ONEFIL	=	000001	G #2-5 4-1665 5-1666 6-34 8-223 9-429
ONLY		002232	G #8-159 83-1723 95-2800 95-2810
OPT1		051460	#179-7702 325-16056
OPT2		054054	#199-8781 325-16059
OPT3		060052	#219-9903 325-16062
OPT4		062444	#242-11146 325-16067
OPT5		022272	#322-15893 325-16070
OSAPTS	=	000001	#6-13 #6-46 6-65
OSAU	=	000001	#6-13 #6-46 6-65
OSBGNR	=	000001	#6-13 #6-46 6-65
OSBGNS	=	000001	#6-13 #6-46 6-65
OSDU	=	000001	#6-13 #6-46 6-65
OSERRT	=	000001	#6-13 #6-46 6-65
OSGNSW	=	000001	#6-13 #6-46 6-65
OSPOIN	=	000001	#6-13 #6-46 6-46 6-65
OSSETU	=	000001	#6-13 #6-46 6-65
PATTBL		034520	G #82-1706 116-3993 116-3994
PATTER		032676	#81-1643 *115-3870 115-3871 *115-3873 120-4173 *199-8789 199-8800 199-8802 199-8809 199-8817
PHR1	=	007714	#83-1881 139-5495 148-5990 153-6268
PHR10	=	010120	#83-1890 131-4822 131-4895 131-4909
PHR11	=	010140	#83-1891 131-5104
PHR12	=	010202	#83-1892 332-16423
PHR13	=	010240	#83-1893 332-16440
PHR14	=	010266	#83-1894 96-2839 96-2864
PHR15	=	010304	#83-1895 96-2866
PHR16	=	010322	#83-1896 96-2837
PHR17	=	010342	#83-1897 332-16414

SYMBOL	CROSS REFERENCE	VALUE	REFERENCES
PHR18	=	010366	#83-1898 336-16618
PHR19	=	010410	#83-1899 332-16428
PHR2	=	007730	#83-1882 325-16046 339-16809
PHR20	=	010430	#83-1900 332-16434
PHR21	=	005662	#83-1791 337-16700
PHR3	=	007746	#83-1883 89-2324
PHR4	=	007770	#83-1884 97-2910
PHR5	=	010004	#83-1885 335-16590
PHR6	=	010026	#83-1886 336-16662
PHR7	=	010050	#83-1887 90-2438 333-16466
PHR8	=	010066	#83-1888 90-2452 90-2462
PHR9	=	010100	#83-1889 199-8803 199-8812
PM. SBE	=	012264	#81-1633 *86-2176 *104-3273 104-3274 336-16639 336-16649
PNT	=	001000	G #83-1778
PRI	=	002000	G #83-1779
PRI00	=	000000	G #83-1770
PRI01	=	000040	G #83-1769
PRI02	=	000100	G #83-1768
PRI03	=	000140	G #83-1764
PRI04	=	000200	G #83-1763
PRI05	=	000240	G #83-1762
PRI06	=	000300	G #83-1761
PRI07	=	000340	G #83-1760
PTABLE	=	034422	G #82-1690 90-2434 94-2711 94-2718 *111-3660 137-5413 138-5449 322-15921 333-16462 333-16455
P.AAA	=	005662	#70-944 83-1791
P.AAB	=	005736	#70-959 83-1792
P.AAC	=	006042	#70-982 83-1793
P.AAD	=	006126	#70-1003 83-1794
P.AAE	=	006142	#70-1007 83-1795
P.AAF	=	006160	#70-1012 83-1796
P.AAG	=	006166	#70-1014 83-1797
P.AAH	=	006216	#70-1022 83-1798
P.AAI	=	006252	#70-1032 83-1799
P.AAJ	=	006270	#70-1037 83-1800
P.AAK	=	006320	#71-1049 83-1801
P.AAL	=	006376	#71-1065 83-1802
P.AAM	=	006446	#71-1079 83-1803
P.AAN	=	006466	#71-1085 83-1804
P.AAO	=	006534	#72-1102 83-1805
P.AAP	=	006564	#72-1110 83-1806
P.AAQ	=	006640	#72-1125 83-1807
P.AAR	=	006654	#72-1129 83-1808
P.AAS	=	006710	#72-1139 83-1809
P.AAT	=	006756	#73-1156 83-1810
P.AAU	=	007024	#73-1169 83-1811
P.AAV	=	007046	#73-1175 83-1812
P.AAW	=	007060	#73-1179 83-1813
P.AAX	=	007066	#73-1181 83-1814
P.AAY	=	007072	#73-1183 83-1815
P.AAZ	=	007100	#73-1185 83-1816
P.ABA	=	007112	#73-1189 83-1817

CZMLRB
 SYMBOL CROSS REFERENCE
 SYMBOL VALUE

CREATED BY MACRO ON 29-MAR-82 AT 13:44

PAGE 11
 CREF

M 13

SEO 0374

REFERENCES

P.ABB	007130	#73-1194	83-1818
P.ABC	007152	#73-1200	83-1819
P.ABD	007200	#74-1212	83-1823
P.ABE	007206	#74-1214	83-1824
P.ABF	007212	#74-1216	83-1825
P.ABG	007220	#74-1218	83-1826
P.ABH	007230	#74-1221	83-1827
P.ABI	007240	#74-1224	83-1828
P.ABJ	007246	#74-1226	83-1829
P.ABK	007256	#74-1229	83-1830
P.ABL	007264	#74-1231	83-1831
P.ABM	007272	#74-1233	83-1832
P.ABN	007300	#74-1235	83-1833
P.ABO	007312	#74-1239	83-1834
P.ABP	007322	#74-1242	83-1835
P.ABQ	007332	#74-1245	83-1836
P.ABR	007336	#74-1247	83-1837
P.ABS	007344	#74-1249	83-1838
P.ABT	007354	#74-1252	83-1839
P.ABU	007370	#74-1256	83-1840
P.ABV	007400	#74-1259	83-1841
P.ABW	007412	#75-1267	83-1842
P.ABX	007416	#75-1269	83-1843
P.ABY	007424	#75-1271	83-1844
P.ABZ	007430	#75-1273	83-1845
P.ACA	007434	#75-1275	83-1846
P.ACB	007440	#75-1277	83-1847
P.ACC	007444	#75-1279	83-1848
P.ACD	007450	#75-1281	83-1849
P.ACE	007454	#75-1283	83-1850
P.ACF	007460	#75-1285	83-1851
P.ACG	007464	#75-1287	83-1852
P.ACH	007472	#75-1289	83-1853
P.ACI	007500	#75-1291	83-1854
P.ACJ	007504	#75-1293	83-1855
P.ACK	007510	#75-1295	83-1856
P.ACL	007514	#75-1297	83-1857
P.ACM	007520	#75-1299	83-1858
P.ACN	007524	#75-1301	83-1859
P.ACO	007530	#75-1303	83-1860
P.ACP	007534	#75-1305	83-1861
P.ACQ	007542	#75-1307	83-1862
P.ACR	007550	#75-1309	83-1863
P.ACS	007554	#75-1311	83-1864
P.ACT	007560	#75-1313	83-1865
P.ACU	007564	#75-1315	83-1866
P.ACV	007570	#75-1317	83-1867
P.ACW	007574	#76-1323	83-1868
P.ACX	007626	#76-1332	83-1869
P.ACY	007634	#76-1334	83-1870
P.ACZ	007642	#76-1336	83-1871
P.ADA	007650	#76-1338	83-1872

SYMBOL	CROSS REFERENCE	SYMBOL	VALUE	REFERENCES	CREF
P.ADB			007656	#76-1340	83-1873
P.ADC			007664	#76-1342	83-1874
P.ADD			007672	#76-1344	83-1878
P.ADE			007700	#76-1346	83-1879
P.ADF			007706	#76-1348	83-1880
P.ADG			007714	#76-1350	83-1881
P.ADH			007730	#76-1354	83-1882
P.ADI			007746	#76-1359	83-1883
P.ADJ			007770	#76-1365	83-1884
P.ADK			010004	#76-1369	83-1885
P.ADL			010026	#77-1379	83-1886
P.ADM			010050	#77-1385	83-1887
P.ADN			010066	#77-1390	83-1888
P.ADO			010100	#77-1394	83-1889
P.ADP			010120	#77-1400	83-1890
P.ADQ			010140	#77-1406	83-1891
P.ADR			010202	#77-1418	83-1892
P.ADS			010240	#77-1428	83-1893
P.ADT			010266	#78-1440	83-1894
P.ADU			010304	#78-1445	83-1895
P.ADV			010322	#78-1450	83-1896
P.ADW			010342	#78-1456	83-1897
P.ADX			010366	#78-1463	83-1898
P.ADY			010410	#78-1469	83-1899
P.ADZ			010430	#78-1475	83-1900
P.AEA			010444	#78-1479	83-1901
P.AEB			010446	#78-1480	83-1902
P.AEC			010450	#78-1481	83-1903
P.AED			010454	#78-1483	83-1904
P.AEE			010460	#78-1485	83-1905
P.AEF			010502	#79-1495	83-1906
P.AEG			010526	#79-1502	83-1907
P.AEH			010602	#79-1517	83-1908
P.AEI			010656	#79-1532	83-1909
P.AEJ			010710	#79-1541	83-1910
P.AEK			010734	#80-1552	83-1911
P.AEL			010756	#80-1558	83-1912
P.AEM			011002	#80-1565	83-1913
P.AEN			011070	#80-1583	83-1914
P.AEO			011120	#80-1591	83-1915
P.AEP			011166	#81-1608	83-1916
P.AEQ			011202	#81-1612	83-1917
P.AER			011216	#81-1616	83-1918
QH1			002332	8-272	#8-286
QH2			002351	8-273	#8-287
QH3			002374	8-274	#8-288
QH4			002425	8-275	#8-289
QS1			002664	8-313	#9-347
QS10			003534	8-328	#9-362
QS11			003554	8-323	8-330
QS12			003637	8-331	#9-364
QS13			003657	8-332	#9-365

#9-363

SYMBOL	CROSS REFERENCE	VALUE	REFERENCES
QS14		003724	8-333 #9-366
QS15		003763	8-334 #9-367
QS16		004030	8-335 #9-368
QS17		004066	8-336 #9-369
QS2		002732	8-315 #9-348
QS3		003007	8-317 #9-349
QS4		003047	8-318 #9-350
QS5		003107	8-320 #9-351
QS6		003147	8-321 #9-352
QS7		003454	8-325 #9-359
QS8		003474	8-326 #9-360
QS9		003514	8-327 #9-361
QUICK		032674	#81-1642 *111-3643 *112-3691 199-8790 205-9127 325-16039 325-16063
RANDOM	G	005464	*42-27 *42-37 83-1726 125-4379 149-6053 149-6055 *261-12203 261-12204 *263-12292
RAND1		070540	263-12295 *265-12377 265-12378 *265-12381 265-12385 265-12391 265-12399 265-12401 *265-12405
RAND2		073124	#273-12823 323-15938
RAND3		075444	#281-13682 323-15940
RAND4		100006	#297-14537 323-15942
RANGE		002224	#312-15384 323-15949
RBUFF	G	022670	#8-155 83-1722 95-2796
			#81-1639 83-1787 83-1788 83-1790 166-6958 166-6964 166-7005 168-7093 180-7735
			185-8037 200-8842 205-9150 210-9400 220-9938 225-10237 258-12058 273-12853 273-13150
			282-13719 287-14018 297-14567 303-14879 313-15407 318-15706
RCBUFF	=	023476	#83-1788 42-11164
RDBUFF	=	022670	#83-1787 242-11163
RD.CO		034322	#82-1668 *85-2111 141-5627 *141-5628 *141-5632
RD.MIL		034326	#82-1672 *85-2113 *141-5636 *141-5638 332-16435
RD.THO		034324	#82-1670 *85-2112 *141-5631 141-5633 *141-5635
READ		045612	#145-5863 149-6057 153-6275 154-6315 154-6327 166-6960 166-7002 168-7090 181-7807
			181-7815 185-8039 201-8908 201-8916 206-9170 206-9178 210-9402 221-10007 221-10015
			225-10239 244-11275 244-11283 249-11554 249-11562 254-11833 254-11841 258-12060 273-12921
			273-12932 273-13152 283-13800 283-13808 287-14020 299-14658 299-14666 303-14881 314-15487
			314-15495 318-15708
REFRES	G	002252	#8-167 83-1725 89-2318 138-5455
RETRY		046724	#153-6245 165-6919 166-7007 168-7095 169-7157 171-7243 181-7768 183-7881 184-7968
			186-8080 200-8868 202-8976 204-9064 207-9238 208-9322 211-9443 221-9968 223-10081
			224-10168 226-10280 243-11221 245-11345 247-11436 248-11498 250-11624 252-11715 253-11777
			255-11903 257-11994 259-12101 273-12882 273-12993 273-13079 273-13189 282-13760 284-13869
			286-13956 288-14061 298-14618 300-14727 302-14814 304-14922 314-15451 316-15560 317-15643
			319-15749
RETRY:		034340	#82-1682 *84-2103 131-4815 131-5102 138-5479 144-5779 144-5790 146-5877 146-5888
			148-5975 148-5986 *153-6246 *155-6385
RE2		005362	41-1349 #41-1379
RE3		005360	41-1357 #41-1378
RE4		005356	41-1366 #41-1377
RN		005370	#42-12 83-1726 125-4377 149-6052 261-12202 263-12291 265-12376
RNDSEC	G	070240	#263-12282 298-14589 313-15418
RNDU		070336	#264-12367 313-15403
RNDWC		070204	#261-12202 282-13724 298-14584 313-15413
RPTR		032672	#81-1641 *156-6444 *180-7735 181-7810 181-7813 182-7833 *200-8842 201-8911 201-8914
			201-8929 *205-9150 206-9176 206-9176 206-9191 *220-9938 221-10010 221-10013 222-10033
			*273-12853 273-12924 273-12930 273-12946 *282-13719 283-13803 283-13806 284-13826 *297-14567

SYMBOL CROSS REFERENCE
SYMBOL VALUE

REFERENCES

6-65	6-65	6-65	6-65	6-65	6-65	6-65	6-65	6-65	5-65
6-65	6-65	6-65	6-65	6-65	6-65	6-65	6-65	6-65	6-65
6-65	6-65	6-65	6-65	6-65	6-65	6-65	6-65	6-65	6-65
6-65	6-65	6-65	6-65	6-65	6-65	6-65	6-65	6-65	6-65
6-65	6-65	6-65	6-65	6-65	6-65	6-65	6-65	6-65	6-65
6-65	6-65	6-65	6-65	6-65	6-65	6-65	6-65	6-65	6-65
6-80	6-80	6-80	6-80	6-80	6-80	7-85	7-85	7-85	6-80
7-85	7-85	7-101	7-101	7-101	7-101	7-101	7-101	7-101	7-85
7-117	7-117	8-143	8-143	8-143	8-143	8-259	8-259	8-259	7-117
8-272	8-272	8-272	8-272	8-272	8-272	8-272	8-272	8-272	8-272
8-272	8-272	8-273	8-273	8-273	8-273	8-273	8-273	8-273	8-273
8-273	8-273	8-273	8-273	8-273	8-273	8-274	8-274	8-274	8-273
8-274	8-274	8-274	8-274	8-274	8-274	8-274	8-274	8-274	8-274
8-274	8-274	8-275	8-275	8-275	8-275	8-275	8-275	8-275	8-274
8-275	8-275	8-275	8-275	8-275	8-275	8-275	8-275	8-275	8-275
8-277	8-277	8-302	8-302	8-302	8-302	8-313	8-313	8-313	8-277
8-313	8-313	8-313	8-313	8-313	8-313	8-313	8-313	8-313	8-313
8-315	8-315	8-315	8-315	8-315	8-315	8-314	8-314	8-314	8-315
8-316	8-316	8-317	8-317	8-317	8-317	8-315	8-315	8-315	8-316
8-317	8-317	8-317	8-317	8-317	8-317	8-317	8-317	8-317	8-317
8-318	8-318	8-318	8-318	8-318	8-318	8-317	8-317	8-317	8-318
8-318	8-318	8-318	8-318	8-318	8-318	8-318	8-318	8-318	8-318
8-320	8-320	8-318	8-318	8-318	8-318	8-319	8-319	8-319	8-318
8-320	8-320	8-320	8-320	8-320	8-320	8-319	8-319	8-319	8-320
8-320	8-320	8-320	8-320	8-320	8-320	8-320	8-320	8-320	8-320
8-321	8-321	8-320	8-320	8-320	8-320	8-320	8-320	8-320	8-320
8-321	8-321	8-321	8-321	8-321	8-321	8-321	8-321	8-321	8-321
8-323	8-323	8-321	8-321	8-321	8-321	8-321	8-321	8-321	8-321
8-323	8-323	8-323	8-323	8-323	8-323	8-322	8-322	8-322	8-323
8-323	8-323	8-323	8-323	8-323	8-323	8-322	8-322	8-322	8-323
8-325	8-325	8-323	8-323	8-323	8-323	8-323	8-323	8-323	8-323
8-326	8-326	8-325	8-325	8-325	8-325	8-324	8-324	8-324	8-325
8-327	8-327	8-326	8-326	8-326	8-326	8-324	8-324	8-324	8-326
8-328	8-328	8-327	8-327	8-327	8-327	8-325	8-325	8-325	8-327
8-329	8-329	8-328	8-328	8-328	8-328	8-326	8-326	8-326	8-328
8-330	8-330	8-328	8-328	8-328	8-328	8-327	8-327	8-327	8-329
8-330	8-330	8-330	8-330	8-330	8-330	8-328	8-328	8-328	8-330
8-331	8-331	8-330	8-330	8-330	8-330	8-328	8-328	8-328	8-330
8-331	8-331	8-331	8-331	8-331	8-331	8-330	8-330	8-330	8-331
8-332	8-332	8-331	8-331	8-331	8-331	8-330	8-330	8-330	8-331
8-333	8-333	8-332	8-332	8-332	8-332	8-331	8-331	8-331	8-332
8-334	8-334	8-332	8-332	8-332	8-332	8-331	8-331	8-331	8-332
8-335	8-335	8-333	8-333	8-333	8-333	8-332	8-332	8-332	8-333
8-336	8-336	8-333	8-333	8-333	8-333	8-333	8-333	8-333	8-333
8-339	8-339	8-334	8-334	8-334	8-334	8-333	8-333	8-333	8-334
		8-335	8-335	8-335	8-335	8-334	8-334	8-334	8-335
		8-336	8-336	8-336	8-336	8-335	8-335	8-335	8-336
		8-336	8-336	8-336	8-336	8-336	8-336	8-336	8-336

SVC SUB = 000001
 SVCTAG = 000001
 SVCTST = 000001
 SYSERR = 041710
 S\$LSYM = 010000
 TOP.SE = 034500

G

#6-13	#6-21								
#6-13	#6-23	7-134	8-196	8-277	8-339				
#6-13	#6-20								
#131-4722	139-5511								
#6-13	#7-134	#8-196	#8-277	#8-339					
#82-1704	95-2777	165-6902	180-7739	180-7745	185-8032	200-8846	200-8849	205-9154	
205-9157	210-9395	220-9942	220-9945	225-10232	243-11202	258-12053	263-12287	263-12293	
273-12857	273-12860	273-13142	282-13720	287-14013	297-14570	303-14874	313-15410		
#82-1661	*84-2093	*155-6391	335-16585	337-16677					
#83-1901	97-2891								
#83-1902	97-2895								
#83-1903	97-2899								

CZMLBB CREATED BY MACRO ON 29-MAR-82 AT 13:44

PAGE 17
CREF

SEQ 0380

SYMBOL CROSS REFERENCE

VALUE REFERENCES

T\$TAGL = 177777
T\$TAGN = 010005
T\$TEMP = 000000

#6-13	7-117	7-117	#7-117	8-143	8-143	#8-143	8-259	8-259
#6-13	8-302	8-302	#8-302	9-379	9-379	#9-379		
#8-259	7-101	7-101	#7-101	#7-134	7-134	#8-196	8-196	#8-272
#7-101	#8-272	8-272	#8-272	8-272	#8-273	8-273	#8-273	8-273
8-272	8-273	#8-274	8-274	#8-274	8-274	#8-274	8-274	#8-275
#8-273	8-275	8-275	#8-275	8-275	#8-277	8-277	#8-313	8-313
8-275	8-313	8-313	#8-313	8-313	#8-315	8-315	#8-315	#8-315
#8-313	8-317	8-317	#8-317	8-317	#8-317	8-317	#8-318	8-318
8-315	8-318	8-318	#8-318	8-318	#8-320	8-320	#8-320	#8-320
#8-318	8-320	8-321	#8-321	8-321	#8-321	8-321	#8-321	#8-323
8-320	8-323	8-323	#8-323	8-323	#8-325	8-325	#8-325	#8-325
#8-323	8-326	8-326	#8-326	8-326	#8-326	8-326	#8-327	8-327
8-325	8-327	8-327	#8-327	8-327	#8-328	8-328	#8-328	#8-328
#8-327	8-330	8-330	#8-330	8-330	#8-330	8-330	#8-331	8-331
8-328	8-331	8-331	#8-331	8-331	#8-332	8-332	#8-332	#8-332
#8-331	8-333	8-333	#8-333	8-333	#8-333	8-333	#8-334	8-334
8-332	8-334	8-334	#8-334	8-334	#8-335	8-335	#8-335	#8-335
#8-334	8-336	8-336	#8-336	8-336	#8-336	8-336	#8-339	8-339
8-335	9-385	9-385	#9-411	9-411				
#9-385								

T\$TEST = 000000
T\$TSTM = 177777
T\$TSTS = 000000
T\$\$SHAR = 010002
T\$\$HW = 010000
T\$\$PRO = 010004
T\$\$SOF = 010003
T\$\$SW = 010001
T1 = 102712
UAM = 000200
UP.HAR = 037422

G
G

#6-13									
#6-13									
#6-13									
#8-259	8-259	8-277							
#7-117	7-117	7-134							
#9-379									
#8-302	8-302	8-339							
#8-143	8-143	8-196							
7-101	#326-16093								
#83-1776									
#103-3203	168-7120	171-7268	185-7997	187-8109	204-9089	209-9351	211-9468	225-10197	
227-10309	247-11461	252-11740	257-12019	259-12126	273-13107	273-13217	286-13981	288-14086	
302-14839	304-14947	318-15672	319-15774						
#141-5624	146-5880								
#108-3452	169-7141	171-7285	185-8014	187-8126	205-9110	209-9368	211-9485	225-10214	
227-10326	248-11482	253-11761	258-12040	259-12143	273-13124	273-13234	287-14002	289-14107	
302-14856	304-14964	318-15689	320-15795						
#142-5688	148-5978								
#140-5560	144-5782								
#81-1652	*88-2290	89-2349							
#134-5254	137-5421	138-5473							
#81-1638	83-1785	83-1786							
*122-4258	*125-4372	*125-4373	83-1789	*114-3806	*114-3817	120-4175	*122-4250	*122-4251	
169-7145	*125-4374	*125-4375	*125-4374	*125-4379	156-6441	165-6907	165-6917	166-6963	
297-14566	169-7155	171-7241	180-7734	200-8841	205-9149	220-9937	273-12852	282-13718	
#83-1786	313-15406								
#82-1674	242-11159	242-11162							
#82-1678	*85-2114	142-5691	*142-5692	*142-5696					
#82-1676	*85-2116	*142-5700	*142-5702	332-16441					
#83-1785	*85-2115	*142-5695	142-5697	*142-5699					
#82-1698	242-11155	242-11161							
*166-6990	*85-2141	*94-2738	*95-2765	*96-2844	*96-2871	*165-6923	*165-6933	*165-6947	
	*167-7016	*167-7026	*167-7040	*167-7051	*169-7161	*169-7171	*170-7189	*170-7200	

G

WCBUFF = 013670
WC.CO = 034330
WC.MIL = 034334
WC.THO = 034332
WDBUFF = 012670
WHY.DR = 034446

SYMBOL	CROSS REFERENCE VALUE	REFERENCES
		*181-7773 *181-7784 *181-7799 *182-7860 *183-7886 *183-7897 *183-7908 *183-7924 *200-8872
		*201-8886 *201-8900 *202-8959 *202-8981 *203-8995 *203-9009 *203-9020 *207-9221 *207-9243
		*207-9253 *207-9267 *208-9282 *221-9973 *221-9984 *221-9999 *222-10060 *223-10086 *223-10097
		*223-10112 *223-10124 *243-11226 *243-11237 *244-11256 *245-11328 *245-11350 *246-11365 *246-11380
		*246-11392 *248-11503 *248-11514 *249-11533 *250-11607 *250-11629 *251-11644 *251-11659 *251-11671
		*253-11782 *253-11793 *254-11812 *255-11886 *255-11908 *255-11919 *256-11938 *256-11950 *273-12887
		*273-12898 *273-12913 *273-12973 *273-12998 *273-13009 *273-13024 *273-13039 *282-13764 *283-13778
		*283-13792 *284-13852 *284-13873 *285-13887 *285-13901 *285-13912 *298-14622 *298-14632 *299-14650
		*300-14710 *300-14731 *300-14741 *301-14759 *301-14770 *314-15455 *314-15465 *314-15479 *315-15539
		*316-15564 *316-15574 *316-15588 *316-15599 333-16496
WPTR	032670	#81-1640 156-6438 *156-6441 156-6442 *180-7734 180-7752 181-7766 182-7821 182-7824
		182-7832 185-8018 *185-8019 *200-8841 200-8856 200-8866 201-8918 201-8921 201-8928
		205-9114 *205-9115 *205-9149 206-9180 206-9183 206-9190 209-9377 *209-9378 *220-9937
		220-9952 220-9962 221-10017 222-10024 222-10032 225-10218 *225-10219 *273-12852 273-12867
		273-12880 273-12934 273-12937 273-12945 273-13128 *273-13129 *282-13718 282-13748 282-13758
		283-13810 283-13813 283-13821 *287-14005 *297-14566 298-14606 298-14616 299-14668 299-14671
		299-14679 302-14812 *302-14858 *313-15406 313-15435 314-15449 315-15501 315-15504 315-15512
		*318-15691
WRD11	= 007240	#83-1828 90-2436 94-2720 94-2728 94-2751 96-2826 96-2853 333-16464 333-16490
WRD15	= 007246	#83-1829 100-3055 139-5497 144-5793 146-5891 148-5989
WRD16	= 007256	#83-1830 138-5485 144-5794 153-6258
WRD17	= 007264	#83-1831 138-5488 146-5892 153-6278
WRD18	= 007272	#83-1832 153-6257 153-6267 153-6277 154-6337 155-6361
WRD19	= 007300	#83-1833 154-6336
WRD2	= 007200	#83-1823 153-6259 153-6269 153-6279 325-16047
WRD20	= 007312	#83-1834 100-3056 155-6360
WRD21	= 007322	#83-1835 94-2727 94-2750 96-2825 96-2852 333-16489
WRD24	= 007332	#83-1836 242-11189 244-11264 254-11822
WRD25	= 007336	#83-1837 249-11543
WRD3	= 007206	#83-1824 339-16810 339-16816
WRD34	= 007344	#83-1838 89-2313 164-6866 180-7709 199-8784 219-9906 242-11149 322-15895
WRD35	= 007354	#83-1839 242-11184
WRD36	= 007370	#83-1840 89-2320
WRD37	= 007400	#83-1841 88-2298 89-2327
WRD38	= 007412	#83-1842 89-2311
WRD4	= 007212	#83-1825 325-16045 339-16808 339-16817
WRD40	= 007416	#83-1843 89-2312 89-2325
WRD41	= 007424	#83-1844 89-2309
WRD6	= 007220	#83-1826 95-2772
WRD7	= 007230	#83-1827 95-2775
WRITE	045424	#143-5765 153-6255 153-6288 153-6294 154-6307 154-6322 165-6909 165-6914 180-7754
		180-7759 200-8858 200-8863 220-9954 220-9959 243-11211 243-11216 248-11488 248-11493
		253-11767 253-11772 273-12869 273-12877 282-13750 282-13755 298-14608 298-14613 313-15437
		314-15446
WR.COU	034314	#82-1662 *84-2104 140-5563 *140-5564 *140-5568
WR.MIL	034320	#82-1666 *84-2106 *140-5572 *140-5574 332-16429
WR.THO	034316	#82-1664 *84-2105 *140-5567 140-5569 *140-5571
X\$ALWA	= 000000	#6-13 8-319 8-324
X\$FALS	= 000040	#6-13 8-314 8-316
X\$OFFS	= 000400	#6-13 8-314 8-316 8-319 8-322 8-324 8-329
X\$TRUE	= 000020	#6-13 8-322 8-329
\$END.L	105052 G	#340-16881

MACRO CROSS REFERENCE	MACRO NAME	REFERENCES								
BGNHRD		8-259								
BGNHW		#7-117								
BGNMOD		6-39								
BGNPRO		9-379								
BGNSFT		#8-302								
BGNSW		#8-143								
DESCRI		#7-85								
DEV TYP		#6-80								
DISPAT		7-101								
ENDHRD		#8-277								
ENDHW		7-134								
ENDMOD		9-411								
ENDPRO		#9-385								
ENDSFT		8-339								
ENDSW		8-196								
ERRTBL		7-94								
GPRMA		#8-272	#8-273							
GPRMO		8-274	8-275	8-317	8-318	8-320	8-323	8-330		
GPRML		#8-313	#8-315	#8-321	#8-325	#8-326	#8-327	#8-328	#8-331	#8-332
		#8-334	#8-335	#8-336						#8-333
HEADER		6-65								
MSBYTE		#6-65	#6-65	#6-65	#6-65					
MSCNTO		#8-272	8-272	#8-273	8-273	#8-274	8-274	#8-275	8-275	#8-313
		#8-315	8-315	#8-317	8-317	#8-318	8-318	#8-320	8-320	#8-321
		#8-323	8-323	#8-325	8-325	#8-326	8-326	#8-327	8-327	#8-328
		#8-330	8-330	#8-331	8-331	#8-332	8-332	#8-333	8-333	#8-334
		#8-335	8-335	#8-336	8-336					
MSDATA		#6-65	6-65	6-65	6-65	6-65	6-65	6-65	6-65	6-65
		6-65	6-65	6-65	6-65	6-65	6-65	6-65	6-65	6-65
		6-65	6-65	6-65	6-65	6-65	6-65	6-65	6-65	6-65
		6-65	6-65	6-65	6-65	6-65	6-65	6-65	6-65	6-65
		6-80	#7-85	7-85						#6-80
MSDECR		#7-134	7-134	#8-196	8-196	#8-277	8-277	#8-339	8-339	#9-385
		#9-411	9-411							9-385
MSDEFA		#8-272	#8-272	#8-273	#8-273	#8-274	#8-274	#8-275	#8-275	#8-313
		#8-315	#8-315	#8-317	#8-317	#8-318	#8-318	#8-320	#8-320	#8-321
		#8-323	#8-323	#8-325	#8-325	#8-326	#8-326	#8-327	#8-327	#8-328
		#8-330	#8-330	#8-331	#8-331	#8-332	#8-332	#8-333	#8-333	#8-334
		#8-335	#8-335	#8-336	#8-336					
MSENDE		#7-134	#8-196	#8-277	#8-339	#9-411				
MSEXCP		#8-272	8-272	8-272	#8-339	8-273	8-273	#8-274	8-274	#8-275
		8-275	8-275	#8-317	8-317	8-317	#8-318	8-318	#8-320	8-320
		8-320	#8-323	8-323	8-323	#8-330	8-330	8-330		
MSGEN		#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65
		#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65
		#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65
		#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65
		#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65
		#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65
		#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65
		#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65
		#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65
		#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65
		#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65
		#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65
		#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65
		#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65
		#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65
		#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65
		#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65	#6-65
		#7-94	#7-94	#7-101	#7-101	#7-117	#7-117	#7-117	#7-117	#7-134

