CR80 AMOS TAPE CTRL. TEST

DOCUMENT NO.: CSS/0719/USM/0049

ISSUE: 3.0

DATE: 870407

PREPARED BY: Peter Clark P.J. Clark.

AUTHORIZED BY: Poul Høgh Andersen

APPROVED BY:

AUTHORITY	DATE	SIGNATURE
Claus Gry	870408	Claus Gry
Poul Høgh Andersen	870408	PHA

REVISION RECORD

CHANGE	DATE	AUTHORIZATION OF CHANGE	LAST PAGE	BRIEF DESCRIPTION OF CHANGE
Issue 1.0	810616	JHO	58	Initial Issue
Issue 2.0	840102	NMJ	58	
Issue 3.0	870407	PHA	74	The Edit tests have been removed.

REVISION RECORD

LIST OF UPDATES	ISSUE	COMMENT	PAGE
3			1
2			1
2			3
2			4
3			6
2			6
2			8
2			13
2			14
2			63

TABLE OF CONTENTS PAGE

1 Scope

3> This document describes the use of version 3 of the
"TEST-MAGTAPE" program, performing test of the CR80 Mag.
3> Tape Controller & RAM. The program is available as a
3> bootstrap module (CSS/719) for MXAMOS installation and as a
2> utility program used in connection with the terminal
2> operating system.

1.1 Introduction

The purpose of this document is to provide:

- a general description of the test philosophy used in the test.
- a description of the commands to the program.
- a list of the possible error conditions detected in the Mag.-Tape Controller.
- a list of the error messages output from the test program.
- a description of the program invocation, including the various files used by the program.
- examples of test sequences using the program.

1.2 Abbreviations

Throughout this document the following abbreviations are used:

	MTC	Mag. Tape Controller
	BOT	Beginning of Tape
	EOT	End of Tape
	BMO	Byte Mode (see 4.29)
	BAD	Byte Address (see 4.30)
	PE	Phase Encoded
	NRZI	Non Return to Zero Inverted
2>	AMOS	Advanced Multiprocessor, Operating System
2>	MX-AMOS	AMOS for CR80MX

2 Applicable Documents

2> 1. CR80 Minicomputer Handbook
2> 82/83

2. CR80 TAPE CTRL, Dual Bus
CR8045D
CSD/005/PSP/0043

3. PERTEC FR9000 Engineering Spec.
PERTEC Equipment 1977

3 Overview

This section contains:

- Software and hardware requirements to facilitate the test.

- A general description of the test philosophy used during the test.

- Description of the functions performed by the test program, describing both standard tests and user defined tests.

- Syntactical rules for input commands.

3.1 Environment

3> Required hardware configuration for the Tape Controller
3> Test is a MX Type 1 or a MX Type 2 PU and CU:
3>
3> <u>MX Type 1 Hardware Configuration:</u>
3>

3>	1 PU CRATE	1 CU CRATE
3>	1 PSU	2 PSU
3>	1 MAP	2 CIA
3>	1 CPU CACHE	1 FD Controller
3>	1 RAM	1 SFA
3>	1 MIA	1 Tape Controller
3>		1 Tape Controller adapter

3> MX Type 2 Hardware Configuration:

3>

3>	1 PU CRATE	1 CU CRATE
3>	1 PSU	2 PSU
3>	1 CMR	2 CMI
3>	1 DIA	2 MIC
3>	1 MIC	1 FD Controller
3>	1 DCI	1 SFA
3>	1 Internal Data	1 Tape Controller
3>	channel	1 Tape controller adapter

3>

3> Other required Hardware

3>

3> 1 O.C
3> 1 V24 Cable
3> 2 Data Channel Cables
3> 1 Data Channel Terminator
3> 1 Tape station

SW: The bootstrap version is self supporting. The utility version requires a standard CR80 software development system, including a Pascal Runtime system

3.2 Functions performed

The overall module test is divided into 2 groups.

The first group consists of predefined test sequences performing a check that the module can execute all commands in a correct way. Furthermore it will be attempted to execute the instructions so that all error conditions are tested, e.g. writing on a writeprotected tape. The read and write test of this group involves writing and reading of some well suited test-patterns.

The second group is user-defined tests, where it is possible to submit either a command-buffer to be executed by the MTC, or mnemonics associated to the separate MTC commands. In this group it is possible to specify a set of commands, which may be repeated.

Furthermore commands are available to utilize

- bus switch in dual-bus systems. (Module initialization)

- printout of status registers and data buffers

- generation of a test tape containing data used in the predefined tests.

- definition of testpatterns fitting special purposes

3.3 Syntactical Rules

In this paragraph the syntactical rules for input are
2> given. The rules are denoted in an extended Backus-
2> Naur-Form (EBNF) using the metasymboles and to enclose
non-terminal symbols, and to enclose structures which may
be repeated.

```
<commandfile>         ::= <device> <unit> <commands> END

<device>              ::= DEVICE:<const>   MEMORY:<const>

<unit>                ::= UNIT:<const>

<commands>            ::= <commands> <command>
                      ::= <command>

<command>             ::= <function>             <cr>
                      ::= <controller-command>   <cr>
                      ::= <normaltest>           <cr>
                      ::= <specialtest>          <cr>
                      ::= <device>               <cr>
                      ::= <unit>                 <cr>

<function>            ::= GENERATETAPE
                      ::= USERSECTION
                      ::= SYSSECTION
                                    /TRANSPORT\
                      ::= DISPLAY   |TRANSFER |
                                    |REWIND   |
                                    \ALL      /
```

```
                    ::= PRINT    B: <const>
                    ::= BLOCK: <const>
                    ::= READY
                                     1
                    ::= INIT: {0 | 1}
                                     1
                                                    10
                    ::= DEFINE TP <const>: {<const>}
                                                     1
                    ::= BYTEMODE: {0 | 1}
                    ::= BYTEADDRESS: {0 | 1}
                    ::= REPORT: <level>
                    ::= LIST

<level>             ::= {0 | 1 | 2 |}

                                           1
<controller-command>::= REWIND {<unitspec>}
                                           0

                                           1
                    ::= UNLOAD {<unitspec>}
                                           0

                                           1
                    ::= STATUS {<unitspec>}
                                           0

                             /                 \1 /        \1
                             |DATA:<const>     |  |        |
                    ::= SKIP |FMS:<const>      |  |FORWARD |
                             |DATAORFMS:<const>|  |REVERSE |
                             |EOT              |  \        /0
                             \                 /1
                    ::= WRITE B:<const> <output>
                                                   1
                    ::= READ B:<const>{OFFSET:<const>}
                                                   0

                                            1            1
                                {<treshold>}  {<exp.data>}
                                            0            0

                    ::= READREVERSE B:<const>

                                                 1
                                      {<exp.data>}
                                                 0

                    ::= WRITERM
                                            1
                    ::= ERASE {FIXED|<const>}
                                            1

<unitspec>          ::= <const>

                        /          \1
<output>            ::= |TP <const>|
                        |/<string>/|
                        \          /1

                             /      \1
                             |LOW   |
<treshold>          ::= THR: |NORMAL|
                             |HIGH  |
                             \      /1
```

```
<exp.data>          ::= TP <const>
                                      00
<string>            ::= {<characters>}
                                      1

<normaltest>        ::= TEST <const>

<specialtest>       ::= REPEAT <times> <subtest>
                    ::= LOAD <commandbuffer>
                    ::= OPCODE <const>

                         /                      \1
                         |<command>              |
<subtest>           ::=  |BEGIN <commands> END   |
                         \                      /1

<times>             ::=  <const>

                                  7
<commandbuffer>     ::=  {<const>}
                                  7

<const>                  /<decimal-constant>     \
                         \<hexadecimal-constant>/

<character>           ::= <any ascii character>
```

4 Function Description

In this chapter all of the program functions are described in detail.

This includes a description of the operations performed by the predefined test sequences.

The following abbreviations are used:

I	for input syntax
O	for output message
E	for possible error messages
FCT	for function performed

NOTE:
In general, correct execution of a command will not imply any output message.

4.1 Define Device Number

FCT: Define haedware device number of the module to be tested. The device number may be changed during the the test, to facilitate a check that the module
2> responds to different device numbers.
2> When running MX-AMOS the crate number must be in
2> bit 12-15 of the device number.

I: DEVICE: <const>

E: S2, S3, S5
DEVICE CAN NOT BE RESERVED

4.2 Define Device Memory

2> FCT: Define device memory. This parameter may be changed
2> together with the change of device number (see 4.1).
2> The start address of the device memory counted in
2> offsets of 1K must be given. When running MX-AMOS
2> the offset is from start of the crate else from
2> start of page 0.

I: MEMORY: <const>

E: S2, S3, S5

4.3 Define Device Priority

FCT: Define hardware priority. This priority may be changed together with the change of device number (see 4.1).

I: PRIORITY: <const>

E: S2, S3, S5

4.4 Define Test Unit

FCT: Define the physical unit number of the tape transport to be used in the test.

This number can be changed during the test, to select a new transport on which the test may run. Used in connection with the REWIND or UNLOAD-commands it is possible to rewind or unload another tape than the one currently used during the test. The unit number must be in the range 0...7.

I: UNIT: <const>

E: S2, S3, S5

4.5 Disable Tape Controller

FCT: Disable the Tape Controller from both main-busses and reset to a well defined state.

I: INIT: 0

4.6 Enable Tape Controller

FCT: Disable the Tape Controller from the remote main bus and enable it onto the local main bus.

I: INIT: 1

4.7 REWIND

FCT: Rewind a specified unit.
There are two cases:

- The rewind concerns current test unit. The program waits until the operation has completed, and then reports back the time used to rewind.

- The rewind concerns another unit. In this case the program initiates the operation, and then returns to user control, hereby facilitating rewind of several units concurrently, or rewind of one or more units, and for instance reading from current test unit.

If the unitspecification parameter is not present the rewind concerns current test unit.

```
                        1
I:      REWIND {UNIT: <NO>}
                        0
```

```
                                                   1
O:      REWIND UNIT <no> COMPLETE {TIME USED: <no> SEC}
                                                   0
```

E: S1, S2, S3, S5
M1

4.8 UNLOAD

FCT: Rewind and unload a specified unit.
Except for the unload sequence the remarks in 4.6 are valid. The unload sequence performs a slow rewind from the load point, removing all tape from the take-up reel.

```
                         1
I:     UNLOAD {UNIT: <no>}
                         0

O:     UNLOAD UNIT <no> COMPLETE
                           1
       {TIME USED: <no> SEC}
                           0
       (see 4.14)
```

E: S1, S2, S3, S3, S5, M1

4.9 STATUS

FCT: Read and display in binary the transport status word of a specified unit.

Transport Status

Data Bit	STATUS	SYMBOL
15, 13, 12	NOT USED (="0")	-
14	TAPE CONTROLLER READY	TCR
11	ON LINE	ONL
10	READY	RDY
9	REWINDING	RDW
8	WRITE PROTECTED	WPT
7	BEGINNING OF TAPE	BOT
6	END OF TAPE	EOT
5	NRZI	NRZ
4	SPEED	SPD
3	SINGLE HEAD	SGL
2-0	TRANSPORT NUMBER	TN 2-0

```
                        1
I:  STATUS {UNIT: <NO>}
                        0
```

O: status word in binary

E: M1

4.10 SKIP

FCT: This command is used to:

- Skip data records, forward or reverse. (DATA)
- Skip filemarks, forward or reverse. (FMS)
- Skip data records or filemarks, forward or reverse. (DATAORFMS)
- Skip to logical end-of-tape, forward (EOT)

```
          /DATA     : <const>\1
I:   SKIP |FMS      : <const>|  /FORWARD\1
          |DATAORFMS: <const>|  \REVERSE/0
          \EOT              /1
```

If the FOR/REV parameter is not used, forward is assumed.

E: Data records:
S1, S2, S3
M1, M6, M12, M13, M18, M19, M20

Filemarks:
S1, S2, S3
M1, M6, M12, M13, M18

Data or filemarks:
S1, S2, S3
M1, M6, M12, M13, M18

Logical end-of tape:

S1

M1, M6, M12, M13

4.11 WRITE

FCT: Write a record of <const> bytes. The data to be written may be either a testpattern generated by the program or a string of characters submitted with the command.

If the string is shorter than <const> indicates, the string will be repeated to occupy <const> bytes.

For a specification of the testpatterns please consult appendix B.

```
I:     WRITE B: <const> / TP <n>      \1
                        \ /<string>/ /1
```

Where B: <no> is number of bytes and TP <no> is testpattern with number <no>.

E: S1, S2, S3, S4, S5
M1..M7, M12..M16

NOTE:
If the output is a string, only the 20 first characters are used, the rest are ingnored.

4.12 ERASE

FCT: Erase either fixed or variable length of tape.
Erase fixed length erases 3.7 inch of tape.
Erase variable length erases a length of tape according to the below written formula:

Coding	length erased	maximum
PE	$\left(\frac{\langle no \rangle}{1600} + 0.6 \right)$ inch	41.6 inch
NRZI	$\left(\frac{\langle no \rangle}{800} + 0.6 \right)$ inch	82.6 inch

```
I:     ERASE / FIXED   \1
             \ <const> /1
```

E: S1
M1, M6, M13, M14, M15, M16

4.13 READ

```
                                  1
I:   READ B: <n> {OFFSET: <p>}
                                  0

                           1              1
               {<threshold>}  {<exp.data>}
                           0              0
```

<exp.data> is expected data.

FCT: Read a data record of <n> bytes. If the offset parameter is used <p> bytes of the record are skipped.

The threshold parameter decides the threshold to be used. If omitted NORMAL is assumed. Low threshold is used to read data with a very low amplitude. High threshold may be used to perform a read-after-write check to ensure that data is properly written.

If the expected data parameter is used, the program compares the read data to the specified testpattern and reports back the result of the comparison. When using the expected data parameter, the offset parameter should be used with care.

The print command may be used to have the read data printed for a visual check.

4.14 READREVERSE

I: READREVERSE B: <const> <exp.data>$\}_{0}^{1}$

exp.data is expected data.

FCT: Read a data record of <const> bytes reverse.
Use of the expected data parameter causes a comparison between the read data and the specified testpattern.
The print command may be used to have the read data printed for a visual check.

O: If the expected data parameter is used: result of the comparison.

E: S1, S2, S3, S5
M1..M5, M12..M16, M18.
D2

O: If the expected data parameter is used: result of the comparison

E: S1, S2, S3, S5
M1, M6, M12..M17, M19, M22
D1

4.15 Write Filemarks

FCT: Write a filemark.
A filemark consists of a 3.5 inch filemark-gap followed by an actual mark.

I: WRITEFM

O: If read-after-write heads are used, the error message M19 (filemark sensed) is a correct message.

E: M1, M6, M15

4.16 Load Command Buffer

FCT: This command enables the set up of an instruction buffer to be executed by the MTC. The instruction buffer must be specified as 7 number. The memory address (location 3) is, however, still decided by the program, and may just as well be a zero. A detailed instruction buffer layout is available in appendix C.

I:

```
LOAD <word1> <word2> <word3>
     <word4> <word5> <word6>
     <word7>
```

where word1...word7 are integers.

E: S3

M: all

4.17 Test Illegal Operation Codes

FCT: This command is used to test proper reaction by illegal operation codes submitted to the MTC. The code must be in range #14..#1D (20..29).

I: OPCODE <no>

O: Errormessage M11 (invalid operation)

E: S3, S5

4.18 Normal Tests

This paragraph contains a description of the predefined test sequences. The introduction supplies common information and s short function overview. For detailed information, consult the sub-paragraphs. A tape containing data generated by the GENERATETAPE command must be used as all commands start by reading and checking the identification block. Some of the tests require operator actions, e.g. pressing buttons on the front plate of the tape transport.

The functions are subdivided into a group of "general" tests and a group of "special" tests. General tests verify proper operation in normal use situations. Special tests verify proper operation in cases that should imply error messages from the module. During the special tests, correct module error messages are masked off, so that only the ones indication improper operation are displayed.

Function overview:
General tests:

TEST 1: read forward
TEST 2: read reverse
TEST 3: write
TEST 5: skip
TEST 6: erase
TEST 7: read status
TEST 8: perform 1..6 in sequence

Special tests:

TEST 11:	read forward
TEST 12:	read reverse
TEST 13:	write
TEST 15:	skip
TEST 16:	erase

Output messages:

In case of proper operation the program outputs:

```
TEST <n>. <p> OK
```

Where <n> is the test number, and <p> is a unique identification of a test, defined in the sub-paragraphs.

In case of improper operationf the program outputs:

```
TEST <n>. <p> ERROR
```

Where <n> and <p> are as described above, and TP <no> is the eventual testpattern where the error occured. This error situation message is followed by a printout of the error codes.

4.18.1 TEST 1

FCT: Perform a general read forward test.

Operation sequence:

1: Read and check testpatterns 0..7.
2: Read and check data records with different block sizes.
3: Read and check testpatterns using offset.
4: Read and check testpatterns using low threshold.
5: Read and check testpatterns using high threshold.

I: TEST 1

E: M4...M7, M17, M19, M20...M22
W1
D1

4.18.2 TEST 2

FCT: Perform a general read reverse test.

Operation sequence:

1: Read testpatterns reverse and check.

I: TEST 2

E: M4..M7, M17, M18, M21, M22
D2

4.18.3 TEST 3

FCT: Perform a general write record test.

Each write sequence is followed by a read and verify sequence.

Operation sequence:

1: Write testpatterns 0..7.
2: Write data with varying block lengths
3: Write data from different memory addresses.

I: TEST 3

E: M2..M7, M15, M16
W1
D3

4.18.5 TEST 5

FCT: Perform a general skip test.

Operation sequence:

1: Skip data records forward
2: Skip data records reverse
3: Skip filemarks forward
4: Skip filemarks reverse
5: Skip data records or filemarks forward
6: Skip data records or filemarks reverse
7: Skip to logical end-of-tape

The skip operations are verified by checking the test patterns written on the test tape.

I: TEST 5

E: M6, M18, (M19), M20
W1

4.18.6 TEST 6

FCT: Perform a general erase test.

Operation sequence:

1: Erase fixed length

The erasing is verified by trying to read the data fields that were present before the erase. These fields are rewritten prior to the termination of the command, to facilitate repeating.

I: TEST 6

E: M6, M15
W1

4.18.7 TEST 7

FCT: Perform a general read status test.

This test should verify that the transport status word corresponds to the pushbutton settings on the front plate of the tape transport, the write protection and the tape position.

The execution of the test requires some operator actions for depressing buttons and mounting of tape.

The command flow for each status check is like this

A. Tell the operator to perform an operation
B. The operator does it (hopefully)
C. The operator presses "return" on the console
D. The appropriate bit of the status word is checked and possible an error message is printed.

Status check sequence:

1. Write protection
2. Transport ready
3. Rewind
4. Beginning of tape
5. End of tape
6. (NRZ1 coding)

O: STATUS BIT OK
STATUS BIT NOT OK

4.18.8 TEST 8

FCT: Perform TEST 1... TEST 6 in sequence.

The result of this command should give a quite good hint to whether or not the MTC is functioning properly.

It should be the first command executed at an installation test.

4.17.11 TEST 11

FCT: Perform special read forward test.

Operation sequence:

1. Read too short data record
2. Read too long data record
3. Read max. length record (64K bytes)
4. Read less than 3 bytes
5. Read filemark
6. Read with all record as offset
7. Read with more than all record as offset
8. Read with tape pos. at logical EOT

I: TEST 11

E: M: all
D1

4.18.12 TEST 12

FCT: Perform a special read reverse test.

Operation sequence:

1. Read too short data record
2. Read too long data record
3. Read max. length record (64K bytes)
4. Read less than 3 bytes
5. Read filemarks
6. Read with tape pos. at beginning of tape

I: TEST 12

E: M: all
D2

4.18.13 TEST 13

FCT: Perform a special write record test.

Operation sequence:

1. Write on write protected tape
2. Write record with less than 3 bytes

I: TEST 13

E: M: all
D3

4.18.15 TEST 15

FCT: Perform a special skip test.

Operations sequence:

1. Skip reverse past BOT
2. Skip forward to logical EOT.
3. Skip whole tape reverse

I: TEST 15

E: M: all

4.18.16 TEST 16

FCT: Perform a special erase test.

Operation sequence:

1. Erase write protected tape
2. Erase maximum (64K)
3. Erase less than 3 bytes

I: TEST 16

E: M: all

4.19 REPEAT

FCT: Specify a command or a set of commands to be executed several times.
A repeat statement can not contain another REPEAT statement, it may contain any other command.
The commands to be executed should be well considered before submitting a REPEAT.

I: REPEAT <times><subtest>

```
             /                        \1
             | <command>              |
<subtest>::= |                        |
             |                  00    |
             | BEGIN {<command>}      |
             \                  0     /1
```

O: Output from the single commands
(see REPORT, 4.32).

E: Union of the messages from the single commands.

4.19 GENERATE TEST TAPE

FCT: Generate a test tape to be used during the predefined tests, and possibly during the user defined tests. The tape will be generated according to the transport currently used concerning coding (PE or NRZI). The density will be as selected by the pushbutton on the front plate of the tape transport. It is important specific test configuration are generated on a similar configuration, i.e. coding (PE/NRZI) and density on a test tape must correspond to tape transport specifications. Furthermore it is important when using this command, that the MTC used is functioning properly.

After this command is executed, the tape contains:

- an identification block
- data records written with different block lengths
- data records containing various test patterns
- a logical end-of-tape mark

The above named tape layout occupies approx. 1% of a 2400 feet tape. The rest is intended to be user defined, where it is possible to perform further test.

In the following the data generated by the GENERATETAPE command is called "system-section" and the rest is called "user-section".

A detailed tape layout of the system section can be found in appendix A.

```
I:     GENERATETAPE
       xxx TEST TAPE GENERATED xxx
                CODING: {PE|NRZI}

E:     W1
```

Note:

After the tape generation the tape is rewound, and the identification block is read and checked. At last the tape is rewound.

4.20 Move Tape to Usersection

FCT: Position tape at beginning of user-section. In practice this command corresponds to the command sequence:

```
REWIND
SKIP DATAORFMS: 40
SKIP EOT
SKIP FMS: 1
```

I: USERSECTION

4.21 Move Tape to System-section

FCT: Position tape at beginning of system-action.
The command causes the following to be executed:
REWIND
Reading and checking of the identification block.
REWIND

I: SYSSECTION

E: W1

4.22 Move Tape to a Specified System Block

FCT: Position tape before block <no> of system-section. This command will typically be usde when wanting to read some of the testpatterns included in the system-section. For a detailed tape layout please consult appendix A.

I: BLOCK: <no>

E: S2, S3, S5

4.23 Define Test Pattern

FCT: With this command it is possible to define test-patterns, made to fit a special purpose. These self-made patterns may be used both as output to a write-command and as expected data for read and read reverse commands.

The patterns must consist of 1...10 words. When used in a write-command, the pattern is repeated to fill the block.

The numbers of the patterns must be in the range 11...15, and they may be redefined. An overview of the patterns is obtained by submitting the LIST command.

```
                          10
I:    DEFINE TP <no>: {<no>}
                           1
```

O:

E: S1, S2, S3, S5

4.24 List Testpattern Overview

FCT Print an overview of the testpatterns available

I: LIST

O: Testpattern overview

E: None

4.25 Display Status

FCT: Display a status word (transport, transfer or rewind) or the 8 word command buffer of the MTC.

```
I:     DISPLAY / TRANSPORT \1
               | TRANSFER  |
               | REWIND    |
               \ ALL       /1
```

O: Status word or command buffer in binary

E: S1

4.26 Print out Contents of On-board RAM

FCT: Output contents of the MTC on-board RAM

I: PRINT B: <number-of-bytes>

O: RAM contents as hexadecimal words and ASC11 characters.

E: S2, S3, S5

4.27 Show Controller READY Status

FCT: Inspect the "tape controller ready"-bit and report back. Can be used to check whether the controller is ready during a rewind operation.

I: READY

O: READY OR BUSY

E: None

4.28 Change Byte Mode

FCT: Set/change the byte mode.

Byte mode = 0:
Lower byte (right byte) is used before upper byte (left byte)

Byte mode = 1:
Upper byte (left byte) is used before lower byte (right byte)

The default mode is 0.

```
                    1
I:     BYTEMODE: {0|1}
                    1
```

E: S2, S3, S5

4.29 Change Byte Address

FCT: Set/change the byte address.

Byte address = 0
First byte used is lower byte

Byte address = 1
First byte used is upper byte

The byte address concerns the first byte in a transfer to or from RAM.

```
                          1
I:     BYTEADDRESS: {0|1}
                          1
```

E: S2, S3, S5

4.30 Change Report Level

FCT: Set/change the amount of messages to be output from the program. This command is especiallly useful when using the REPEAT command.

Reporting levels:

O: No messages are output during the repeating, but a summary of detectd errors is output when the repeating stops.

1: Only data comparison errors are output during the repeating. A summmary of errors is output at the end of the repeating.

2: All error messages are output when they occur.

Note:
Syntax- and semantic errors are not suppressed, but are always output when they occur.

Default level is 2.

I: REPORT: <level>

E: S2, S3, S5

5 Activation

Utility program:

Submit

```
                         1              1
TEST MAGTAPE { I: <file> }  { O: <file> }   <cr>
                         0              0
```

to the command interpreter

I: file is a file containing commands to the program. If omitted, current input is used.

O: file is a file intended for output messages and error messages. If omitted, current output is used.

Bootstrap module:

Submit

```
                1
Boot FD { 0 | 1 }
                1
```

to the AMU program.

The program will then be bootstrap loaded and start execution.

6 Error Codes and Messages

The error codes and messages are split into 4 groups:

- Syntax and semantic errors of input commands
- Warnings
- Module error reports fetched from the MTC statuswords
- Data error messages resulting from comparison between read data and expected data.

6.1 Syntax and Semantic Errors

This group consists of an 'S' followed by an error number.

Number	Meaning
S 1	illegal command
S 2	':' expected
S 3	number expected
S 4	'/' expected
S 5	illegal number

In case of a syntax- or semantic error, the rest of the input line is skipped.

6.2 Warnings

This group consists of a 'W' followed by a number.

Number	Meaning
W 1	Unable to read identification block of test tape.

6.3 Module Error Messages

2> This group consists of an 'M' followed by a number and an
2> abbreviation of the meaning. For an explanation you may consult ref. 2.

Number	Meaning	Abbreviation
M 1	unit not available	UNA
M 2	tape hard error	THE
M 3	tape conrrectable error	TCE
M 4	cable parity error	CPE
M 5	memory parity error	MPE
M 6	end-of-tape sensed	ETS
M 7	data timing error	DTE
M 8	\	
M 9	| not used	
M 10	/	
M 11	invalid operation	IVO
M 12	tape transport is off-line	TOL
M 13	tape transport not ready	TRN
M 14	tape transport rewinding	TRW
M 15	tape is write protected	TWP
M 16	invalid record length	IRL
M 17	missing identifier at PE	MID
M 18	beginning of tape sensed	BTS
M 19	file mark sensed	FMS
M 20	logical end-of-tape sensed	LET
M 21	short data record	SDR
M 22	long data record	LDR
M 23	command buffer parity error	CPE
M 24	invalid read off-set	IOR

6.4 Data Errors

Error messages of this kind are output when the comparison between data actually read and expected data gives a conflict.

The format of the message is like this:

```
D <error> TP <no> W <no>
```

D	is an identification of this kind of errors
<error>	is the number of the error (see table below)
TP <no>	is the testpattern number where the error occured
W <no>	is the word number relative to buffer start

The message is only output for the first comparison error.

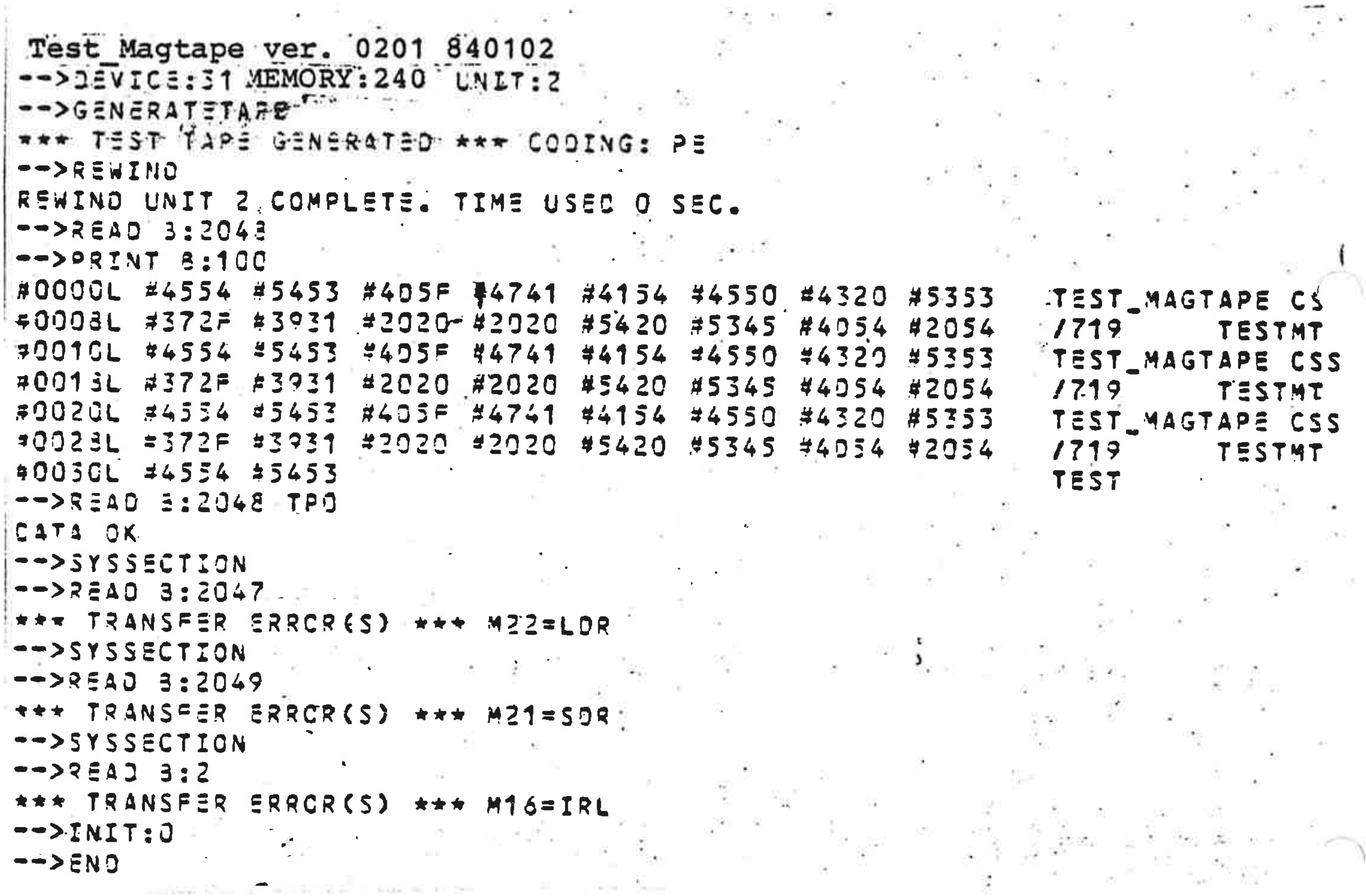
Number	meaning
D 1	Error occurred during READ
D 2	Error occurred during READREVERSE
D 3	Error occurred during read after write check

When the comparison has completed, the message:

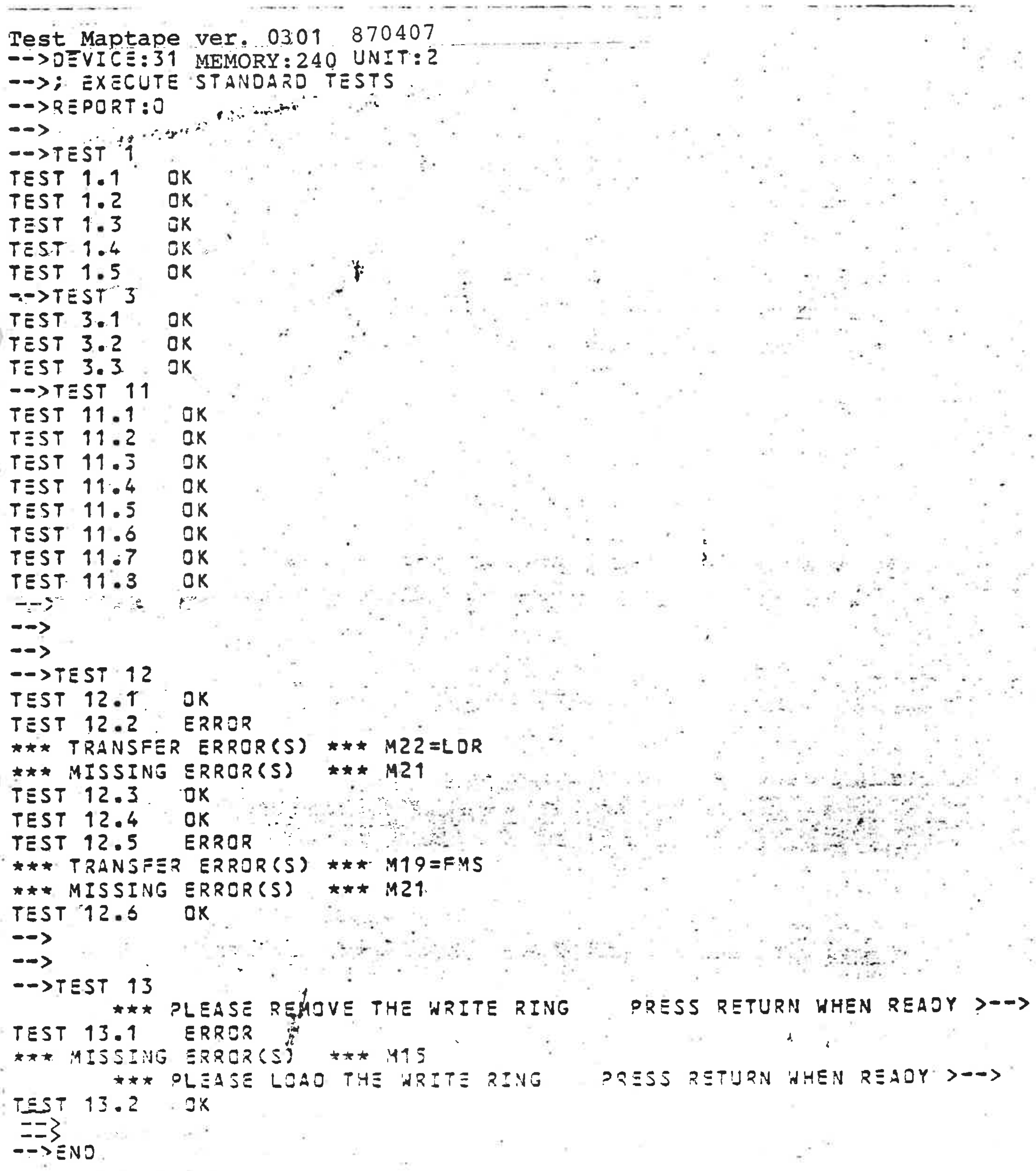
```
D-ERRORS: <errorcount>
```

Which gives the number of comparison errors, detected is output.

7 Examples



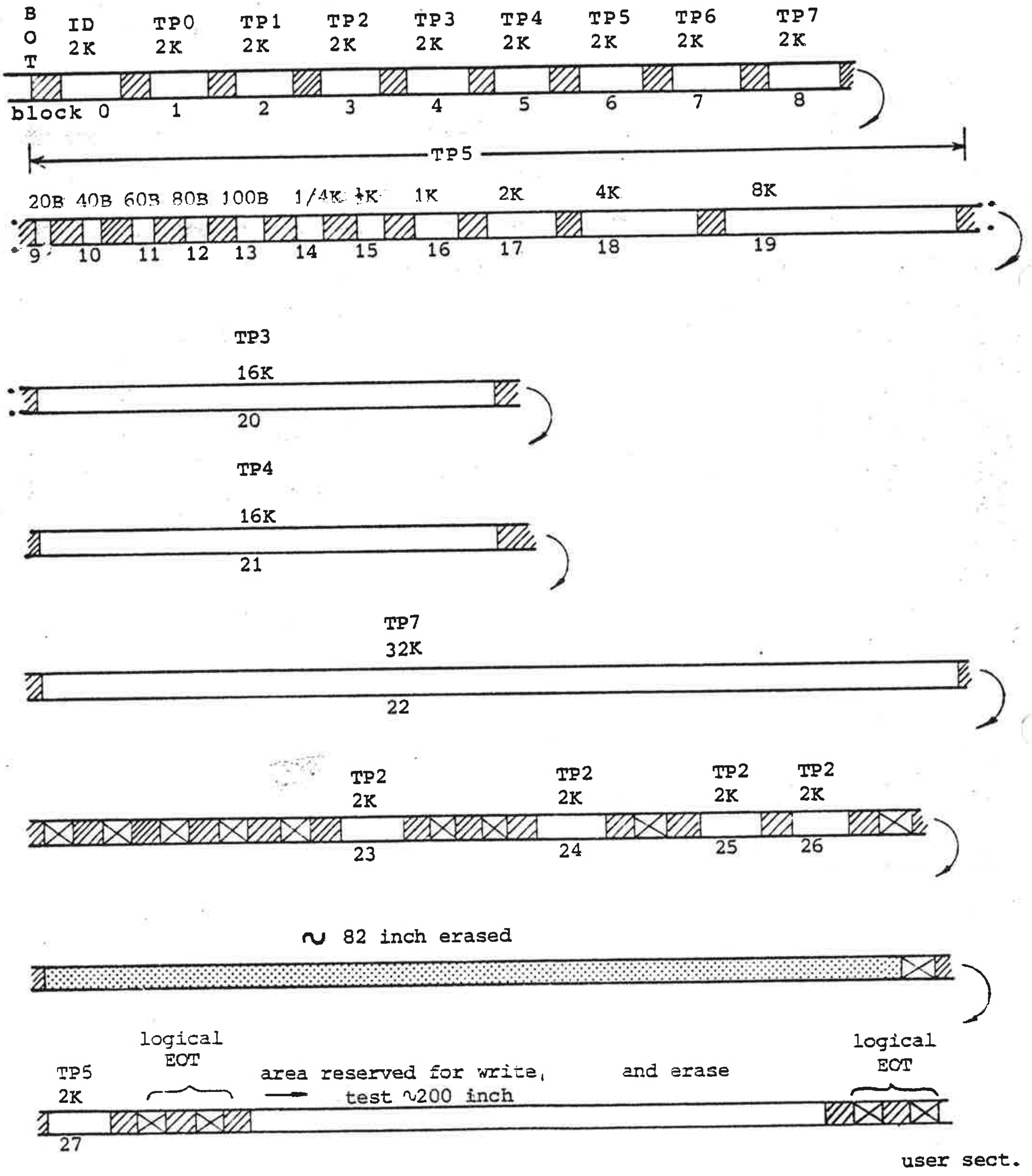
```
Test_Magtape ver. 0201 840102
-->DEVICE:31 MEMORY:240  UNIT:2
-->GENERATETAPE
*** TEST TAPE GENERATED *** CODING: PE
-->REWIND
REWIND UNIT 2 COMPLETE. TIME USED 0 SEC.
-->READ 3:2048
-->PRINT 8:100
#0000L #4554 #5453 #405F #4741 #4154 #4550 #4320 #5353   TEST_MAGTAPE CS
#0008L #372F #3931 #2020 #2020 #5420 #5345 #4054 #2054   /719      TESTMT
#0010L #4554 #5453 #405F #4741 #4154 #4550 #4320 #5353   TEST_MAGTAPE CSS
#0018L #372F #3931 #2020 #2020 #5420 #5345 #4054 #2054   /719      TESTMT
#0020L #4554 #5453 #405F #4741 #4154 #4550 #4320 #5353   TEST_MAGTAPE CSS
#0028L #372F #3931 #2020 #2020 #5420 #5345 #4054 #2054   /719      TESTMT
#0030L #4554 #5453                                       TEST
-->READ 3:2048 TPO
DATA OK
-->SYSSECTION
-->READ 3:2047
*** TRANSFER ERRCR(S) *** M22=LDR
-->SYSSECTION
-->READ 3:2049
*** TRANSFER ERRCR(S) *** M21=SDR
-->SYSSECTION
-->READ 3:2
*** TRANSFER ERRCR(S) *** M16=IRL
-->INIT:0
-->END
```



```
Test Maptape ver. 0301  870407
-->DEVICE:31 MEMORY:240 UNIT:2
-->; EXECUTE STANDARD TESTS
-->REPORT:0
-->
-->TEST 1
TEST 1.1    OK
TEST 1.2    OK
TEST 1.3    OK
TEST 1.4    OK
TEST 1.5    OK
-->TEST 3
TEST 3.1    OK
TEST 3.2    OK
TEST 3.3    OK
-->TEST 11
TEST 11.1    OK
TEST 11.2    OK
TEST 11.3    OK
TEST 11.4    OK
TEST 11.5    OK
TEST 11.6    OK
TEST 11.7    OK
TEST 11.8    OK
-->
-->
-->
-->TEST 12
TEST 12.1    OK
TEST 12.2    ERROR
*** TRANSFER ERROR(S) *** M22=LOR
*** MISSING ERROR(S)  *** M21
TEST 12.3    OK
TEST 12.4    OK
TEST 12.5    ERROR
*** TRANSFER ERROR(S) *** M19=FMS
*** MISSING ERROR(S)  *** M21
TEST 12.6    OK
-->
-->
-->TEST 13
       *** PLEASE REMOVE THE WRITE RING    PRESS RETURN WHEN READY >-->
TEST 13.1    ERROR
*** MISSING ERROR(S)  *** M15
       *** PLEASE LOAD THE WRITE RING    PRESS RETURN WHEN READY >-->
TEST 13.2    OK
-->
-->END
```

APPENDIX A

Test tape layout



☒ = file mark

▨ = gap

NOTE →

Note: Each block in the system section consists of the testpattern, with length in bytes as specified.

APPENDIX B

Test pattern description

Description of system defined test patterns.

Overview:

TP0:	Walking zero
TP1:	Walking one
TP2:	Checkerboard
TP3:	Count up, words
TP4:	Count down, words
TP5:	Count up, bytes
TP6:	Data = inverted address, except 1 bit
TP7:	Random numbers (reproducible)

TP0: Walking zero pattern

Word	contents	Hex
0	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0	FFFE
1	1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1	FFFD
2	1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1	FFFB
3	1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1	FFF7
.	.	
.	.	
.	.	
14	1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1	BFFF
15	0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	7FFF
16	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0	FFFE
.	.	
.	.	
.	.	

TP1: Walking on pattern

Word	contents	Hex
0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1	0 0 0 1
1	0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0	0 0 0 2
2	0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0	0 0 0 4
3	0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0	0 0 0 8
.	.	
.	.	
.	.	
14	0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0	4 0 0 0
15	1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	8 0 0 0
16	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1	0 0 0 1
.	.	
.	.	
.	.	

TP2: Checkerboard

Word	contents (hex)
0	5 5 5 5
1	A A A A
2	5 5 5 5
3	A A A A
4	5 5 5 5
	.
	.
	.

TP3: Count up, words

Word	contents (hex)
0	0 0 0 0
1	0 0 0 1
2	0 0 0 2
3	0 0 0 3
.	
.	
.	
15	0 0 0 F
16	0 0 1 0
.	.
.	.
.	.

TP4: Count down, words

Word	contents (hex)
0	F F F F
1	F F F E
2	F F F D
3	F F F C
4	F F F B
5	F F F A
6	F F F 9
.	.
.	.
.	.

TP5: Count up, bytes

Word	contents
0	0100
1	0302
2	0504
3	0706
.	.
.	.
.	.
127	FFFE
128	0100
129	0302
.	.
.	.
.	.
.	.
255	FFFE
.	.
.	.
.	.

TP6: Data = inverted address, except 1 bit

Word	contents	Hex
0	0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	7 F F F
1	1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 0	B F F E
2	1 1 0 1 1 1 1 1 1 1 1 1 1 1 0 1	D F F D
3	1 1 1 0 1 1 1 1 1 1 1 1 1 1 0 0	E F F C
4	1 1 1 1 0 1 1 1 1 1 1 1 1 0 1 1	F 7 F B
.	.	
.	.	
.	.	

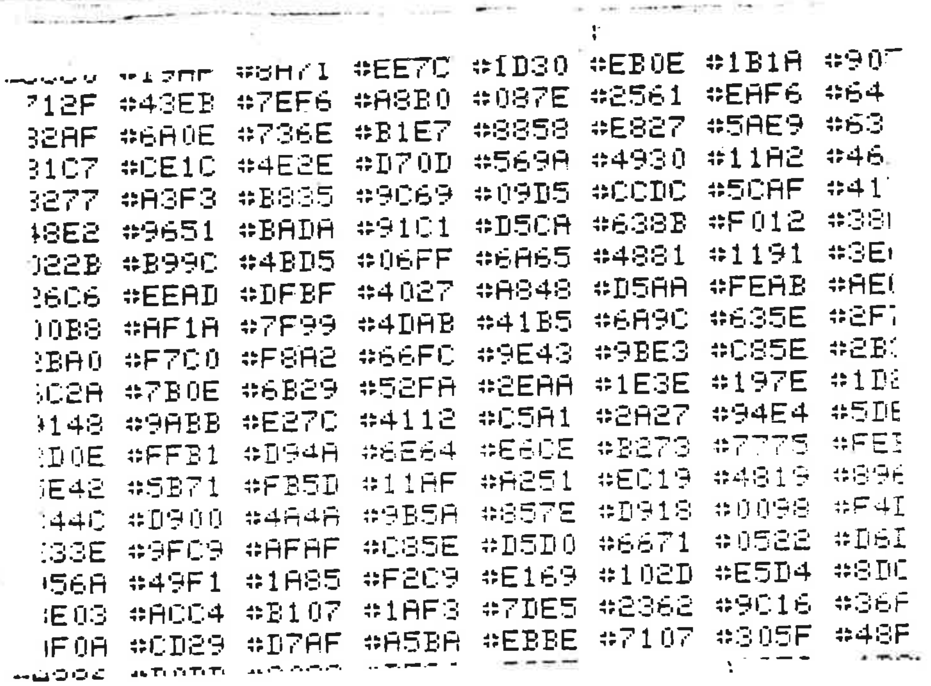
Formula for calculating:

$$\text{contents} = \text{INV (address) XOR } 2^{15-(\text{address MOD } 16)}$$

Where address = word number relative to buffer start.

TP7: Reproducible random numbers

< 2



```
                   #EE7C #1D30 #EB0E #1B1A #90
 712F #43EB #7EF6 #A8B0 #087E #2561 #EAF6 #64
 32AF #6A0E #736E #B1E7 #8858 #E827 #5AE9 #63
 31C7 #CE1C #4E2E #D70D #569A #4930 #11A2 #46
 3277 #A3F3 #B835 #9C69 #09D5 #CCDC #5CAF #41
 48E2 #9651 #BADA #91C1 #D5CA #638B #F012 #38
 022B #B99C #4BD5 #06FF #6A65 #4881 #1191 #3E
 26C6 #EEAD #DFBF #4027 #A848 #D5AA #FEAB #AE
 00B8 #AF1A #7F99 #4DAB #41B5 #6A9C #635E #2F
 2BA0 #F7C0 #F8A2 #66FC #9E43 #9BE3 #C85E #2B
 5C2A #7B0E #6B29 #52FA #2EAA #1E3E #197E #1D
 9148 #9ABB #E27C #4112 #C5A1 #2A27 #94E4 #5DE
 2D0E #FFB1 #D94A #6E64 #E6CE #B273 #7775 #FE
 5E42 #5B71 #FB5D #11AF #A251 #EC19 #4819 #89
 144C #D900 #4A4A #9B5A #857E #D918 #0098 #F4
 133E #9FC9 #AFAF #C85E #D5D0 #6671 #0522 #D6
 156A #49F1 #1A85 #F2C9 #E169 #102D #E5D4 #8D
 1E03 #ACC4 #B107 #1AF3 #7DE5 #2362 #9C16 #36F
 1F0A #CD29 #D7AF #A5BA #EBBE #7107 #305F #48F
```

< 2

APPENDIX C

MTC Command buffer layout

FUNCTION	LOCATION (b2b1b0)	CPU Read/Write	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Used by Controller	0																	
TRANSFER CONTROL	1	(R)/W	Not Used				BMO	BAD	OFS	DIN	OP 4-0					TN 2-0		
TRANSFER COUNT	2	R/W	CT 15-0															
MEMORY ADDRESS	3	(R)/W	AD 15-0															
READ OVERFLOW/ READ OFFSET	4	R/W	OV 15-0/ OF 15-0															
READ OVERFLOW/OFFSET, REWIND MASK	5	R/W	RM 7-0								OV 23-16/ OF 23-16							
TRANSPORT STATUS	6	R/W	O	TCR	CO		ONL	RDY	RWD	WPT	BOT	EOT	NRZ	SPD	SGL	TN 2-0		
TRANSFER STATUS			PFL	TCR	ERP	SC 3-0				THE	TCE	CPE	MPE	ETS	DTE	TN2-0		
REWIND, CONTROLLER STATUS	7	R	O	MER	ARE	CRE	ODP	HTS			RW 7-0							

DISPLAY ALL:

#0300
0
0
0
0
0
#4000
#0C00

STATUS:

#4C80 (ON LINE)   #4000 (OFF LINE)

DISPLAY ALL

0300
0016
0
0
0
0
4C80
0C00

Test opstilling :

P3 - 8000 (224)

Tape Ctrl.: # 1F MEM : P2 - C000 (176)

1 2 3 4 5 6 7 8
|- OFF -|- ON -|
1 1 1 1 1 0 0 0

1 2 3 4 5 6
OFF | ON | OFF
1 1 0 1 1 1

RAM: P0-0000 => P3-FFFF

Load RT test:

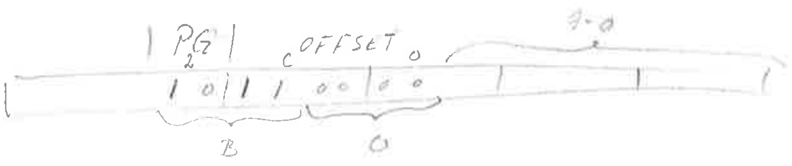
```
@ TEST_MAGTAPE

--> DEVICE: 31   MEMORY: 176   UNIT: 0

--> GENERATETAPE
--> REPORT: 1
--> REPEAT 3   TEST 8
```

```
Test 1.1 OK
   :
TEST 4.1 ERROR   M19 "FMS"
   :
Test 6.2 OK
Test 1.1 OK
   :
Test 4.1 ERROR   M2 "THS"  M31 "SOR"
   :
Test 6.2 OK
Test 1.1 OK
   :
Test 4.1 ERROR   M2        M21
   :
Test 6.2 OK
```

MEMORY:



PG
2
C
OFFSET
0
9-0
1 0 1 1
00 00
B
C