128 KByte
memoryexpansion
with
centronics
interface

Users manual for the 99/4 A peripheral

MECHATRONIC

TABLE OF CONTENTS

INTRODUCTION

The MECHATRONIC 128 kByte MEMORYEXPANSION is a system which provides two devices in one.
First it increases the memory of your TI 99/4A Home Computer.
It adds 128 kBytes of Random Access Memory (RAM) to the 16 kBytes of RAM available with the computer.
These 128 kBytes are organized in four blocks of 32 kBytes.
The first block directly increases the computer RAM.
The three other blocks are organized as a so called RAMDISK. The RAMDISK stores informations like a disk drive but not on diskettes but in built in semiconductor circuits.

Second it supports you with a Centronics parallel interface which enables your system to transfer your data to a printer.

This manual shows you how to connect and test this peripheral, describes the MEMORYEXPANSION and presents the BASIC instructions that are used to operate it.
The second section gives you informations over the interface.
As the last part the appendices provide special informations.

SET-UP INSTRUCTIONS

Connecting the 128 kByte MEMORYEXPANSION

Setting up the 128 kByte MEMORYEXPANSION is a simple process. First the peripheral is attached at the right side of the TI 99/4A console. To do so, the TI 99/4A and all attached devices must be switched off.

CAUTION

Never plug in the 128 kByte MEMORYEXPANSION if it or any other peripheral is turned on. Otherwise this peripheral or others can be damaged.

Open the slot at the right side of the console and push in gently the peripheral with its plug. Other peripherals can now be attached at the right side of the 128 kByte MEMORYEXPANSION. Of course it is possible to connect both, a Peripheral Expansion Box or stand-alone devices. However, since the first 32 kByte RAM block in the 128 kByte MEMORYEXPANSION must use the same adresses as the 32 kByte Memory Expansion Card does, it is necessary to remove one of them, either the card or the 128 kByte peripheral. Keep this fact in mind since otherwise confusion in the system can result.

CAUTION

The electronic components of the 128 kByte MEMORYEXPANSION can be damaged by discharges of static electricity. To avoid damage, do not touch the connector contacts or expose them to static electricity.

Powering-up the System

To turn on the complete system the following sequence has to be obeyed.
First switch on all other peripherals. Then power up the 128 kByte MERORYEXPANSION. At least switch on the monitor and the TI 99/4A console.

The 128 kByte MEMORYEXPANSION has no ON/OFF switch. Therefore you need only attach the power cord to the small jack on the back of the 128 kByte MEMORYEXPANSION and plug the AC adapter that comes with it in a standard wall outlet.
TO NOT CONNECT ANY OTHER ADAPTER TO THIS PERIPHERAL.

Testing the Peripheral

Be sure that all devices are turned on in the proper sequence.

Attention: The following program performs a DELETE "RAMFORM" command. Consequently all data stored in the RAMDISK will be erased. If you intend the present contents for further use, save them on a diskette.

Type in the following program to test, if the 128 kByte MEMORY-EXPANSION is working correctly.

```
100 DELETE "RAMFORM"
110 OPEN #1:"DSKR.TEST"
120 INPUT "SOMETHING":S$
130 PRINT #1:S$
140 CLOSE #1
150 DELETE "CATALOG"
160 OPEN #1:"DSKR.TEST"
170 INPUT #1:S$
180 PRINT S$
190 CLOSE #1
200 END
```

Be cautious to input the previous lines exactly as they are since each period, colon, quotes etc. must match the listed program. Otherwise difficulties can occur, that are not related to the RAMDISK.

After the program is correctly typed in, enter RUN to execute the test.
First the dialog "SOMETHING" will be displayed. Enter any line of text, e.g. "THIS IS A TEST"
Now a catalog of the RAMDISK will be displayed, showing that a file named "TEST" exists in the RAMDISK.
Pressing any key will now bring back on the screen the line you entered just before.
The message ** DONE ** indicates that the program was terminated and the RAMDISK is functioning.

THE 128 kBYTE MEMORYEXPANSION

Memory map

1. 32 kByte direct RAM Memoryexpansion

The total storage capacity of this device amounts 128 kByte RAM. This memory is available for programs and all write- or read processes.
32 kByte of this memory increase directly the RAM of the TI 99/4A. The Extended Basic Modul, Editor-Assembler, TI-Writer, TI Logo, Multiplan and other command-moduls can access this memory. So, with this memory expansion in Extended Basic 13928 Bytes are available for string-variables and variable-lists. 24488 Bytes are available for programs and numeric variables and 8192 Bytes for machineprograms. To check the available memory for programs use the SIZE command (Displays memory except space reserved for machineprograms).

2. 96 kByte Memory-Blocks

This RAM expansion contains another 96 kByte RAM which can be accessed by ROM-based software or from normal programs and modules.
Also this part of memory can be accessed for machineprograms. The total memory consisting of four 32 kByte blocks will be reached by so called BANK-SWITCHING.
The switching is performed by output of CRU-bits:
CRU-address >14C0 >14C2

0	0	32 kByte RAM block 0
1	0	32 kByte RAM block 1
0	1	32 kByte RAM block 2
1	1	32 kByte RAM block 3

The routines for the BANK-SWITCHING should be located outside of all RAM supported by the computer and the 128 kByte MEMORY-EXPANSION, for example in the Mini-Memory Modul.

The RAMDISK Principle

The memory in the 96 kByte block is organized as a so called RAMDISK, this means that the application to this RAM is performed as to the disk-system. The advance of this RAMDISK is that it is about five times faster than a disk-drive. Time consuming file accesses for example for sorting will so be done with much higher speed. The neccessary files have to be copied for these actions in the RAMDISK. Now they can be processed as usual.
For the file processing the options in the open statement have to be changed to the new file-names.
Attention: Please note that data will be stored in the RAMDISK only while the device is powered on. Also if the power supply of the RAMDISK is maintained switching the console or other peripherals on and off can cause a loss of data.

RAMDISK commands

1. Initializing

The command DELETE "RAMFORM" initializes the RAMDISK. You will find that in 99.9% of all cases the RAMDISK will also show no malfuncion without initializing. Due to undefined cases it can happen that the RAMDISK does not proper work without initializing. Therefore after turning on the command DELETE "RAMFORM" should be performed.
ATTENTION: After the command DELETE "RAMFORM" all files that ar stored in the RAMDISK are erased. The command DELETE "RAMFORM" can be used at any time, also in BASIC-programs.

2. Cataloging

By the use of the command DELETE "CATALOG" the catalog of the RAMDISK appears on the screen:

```
         RAMDISK CATALOG

    NAME     BLK BYTE   TYP
    Name    Block Bytes  Filetyp

    PRESS 1 FOR COPY FILES
    ANY OTHER KEY TO RETURN
```

Name - At this position the names of files which are stored jus now in the RAMDISK are listed in this column. Maximum display a three files since each file or programm occupies one block.

Block - At this position the number of blocks is listed which a occupied by the file.

Bytes - Lists the number of Bytes taken from the file.

Filetyp - Specifies program or data file (see file processing).

By pressing any key the RAMDISK-software returns to the calling program. By the special use of the DELETE command for catalog display it is possible to use this command also outside of a BASIC-program.
Avoid calling the catalog from programs which have switched the screen to 40 characters/line since in this case reading the screen will be difficult (Example editor of TI-Writer).

a) COPY FILES

By pressing the 1 key the optional routine COPY FILES is chosen. With this routine a file can be copied from the RAMDISK to a diskette or vice versa.
First specify the device-name and file-name of the file that is to be copied.
Second specify device-name and file-name where to copy.

Example

```
        MASTERFILE:

        DSKR.RAMFILE

        COPYFILE:

        DSK1.DSKFILE
```

In the case that COPY FILES was chosen by error the BACK key (FCTN 9) turns back to the catalog. If an error will occur while copying, only the second digit of the error-code is displayed (See appendix A - error codes in BASIC).

Attention: Only data files but no program files can be copied. This restriction was neccessary to be sure that this function works with very much programs.
Furthermore the COPY FILES function occupies some memory which can also be used by machineprograms or Extended Basic programs. The memory section concerned is >A000 to >A2D0.
For Extended Basic programs you should test with the SIZE statement if at least 720 Bytes of program area are free.

The catalog itself can be chosen without doubts, also it is possible to return with the BACK key, without that this memory area is occupied or changed. While copying files, often strange characters are displayed on the screen. This is explained since the screen is used as a temporary buffer for the copied data.

Note: Although the RAMDISK provides about as much memory space as a single sided, single density diskette, due to different memory usage it can occur that a file stored in the RAMDISK can not be copied on a diskette. For example a file that is stored in FIXED 129 format and occupies all RAMDISK MEMORY cannot be copied on a diskette.

b) Copying Programs to RAMDISK

To store a program in the RAMDISK do the following:
First load the program as usual e.g. with

```
OLD "CS1"
```

or with

```
OLD "DSK1.TEST"
```

Then save the program in the RAMDISK with

```
SAVE "DSKR.TEST"
```

The programs saved by this may be reloaded by

```
OLD "DSKR.TEST" or
RUN "DSKR.TEST"
```

The last command is only possible with Extended Basic.

C) Files from the Tape Recorder

Also files stored on a tape recorder can be loaded easily in the RAMDISK by means of a short BASIC program:

```
100 REM Open with the particular file attributes
110 OPEN #1:"CS1",INPUT,INTERNAL,FIXED 192
120 OPEN #2:"DSKR.TEST1",OUTPUT,INTERNAL,FIXED 192
130 REM Write number of records in the first record.
140 INPUT #1:A
150 FOR X=1 TO A
160 REM Match variable-list with the particular file
170 INPUT #1:C,D,F$
180 PRINT #2:C,D,F$
190 NEXT X
200 CLOSE #1
210 CLOSE #2
220 END
```

RAMDISK Compatibility with Modules

The RAMDISK also supports modules like TI-WRITER or Editor-Assembler. With these it can be used instead of a disk drive to store data.
Especially with the Editor-Assembler results an important increase of speed for the assembling-process.

Generally the RAMDISK can be operated with all modules which can store on diskettes. Often results a distinct decrease of processing time.

Operating the RAMDISK with BASIC

Without difficulties the RAMDISK can be operated with TI BASIC and TI Extended BASIC.
As already shown in the recent examples each file is specified by a name which may have a length of up to 10 charcters. Avoid periods and spaces in the name. Also if the RAMDISK does not response with an error this can cause malfunction with other devices.
To apply to the RAMDISK its device-name "DSKR." is to specify before the file name.

Maximum 3 files or programs can be saved at a time in the RAMDISK. Always each file occupies a 32 kByte block of memory. If a file becomes longer than 32 kByte it occupies another memory block if available. In this case only two files can be stored. Of course a file can occupy all three blocks. From the 98304 Bytes of memory the RAMDISK operating system occupies only 96 Bytes. So 98208 Bytes are available for storage of data.

Loading and Saving Programs

To load or save programs the commands known from the tape recorder or the disk-system are available.

The use of this commands with the RAMDISK is totally equal. You have only to change the file-names to "DSKR.NAME".

<u>OLD</u>

Format:

```
        OLD "DSKR.file-name"
```

Example of OLD command:

OLD "DSKR.DEMO"	The command loads the program DEMO from the RAMDISK in the RAM of the console.

<u>SAVE</u>

Format:

```
        SAVE "DSKR.file-name"
```

Example of SAVE command:

SAVE "DSKR.DEMO"	The command saves the program with the name DEMO in the RAMDISK.

File Processing

There are seven main TI BASIC statements that are used to acces files in the RAMDISK:
OPEN, CLOSE, INPUT, PRINT, EOF, RESTORE, and DELETE.
The following discussion of each of these statements relates to their use with the 128 kByte MEMORYEXPANSION.

OPEN

The OPEN statement prepares a BASIC program to use data files stored in the RAMDISK. It provides a link between a file-number used in a program and the file in the RAMDISK, and it describes file's characteristic so that the program can proccess or creat the file. If the file already exsists, the description that is given in the program must match the actual characteristics of t file.

Format:

```
        OPEN #file-number:"device.file-name" [,file-organizatio
                                             [,file-type]
                                             [,open-mode]
                                             [,record-type]
```

The file-number and device.file-name must be included in the OF statement. The other information may be in any order or may be omitted. If an item is omitted, the computer assumes certain defaults, which are described below.

- file-number - The file-number (1 through 255 or an expression) is assigned to a particular file by the OPE statement. (File number 0 is the keyboard and screen of the computer. It cannot be used for other files and is always open.) You may assign file numbers as you wish, with each file having a different number.

  The file-number is entered as a number sign (#) followe by a number or a numeric expression that, when rounded the nearest integer, is a number from 1 to 255 and is n the number of a file that is already open.

- device.file-name - The device is here the RAMDISK. Therefore the device name has to be specified with "DSKR." .
  The file-name is the name under which the file is store To reopen the file exact the same file-name has to be used.

- file-organization - The records in a file can be access either sequentially or randomly. Records accessed seque tially are read or written one after the other. Records accessed randomly can be read or written in any order, including one after the other.

To indicate which access method you wish to use, enter either SEQUENTIAL for sequential accessing or RELATIVE for random accessing. If you are creating a file, you may optionally specify the number of records on a file by following the word SEQUENTIAL or RELATIVE with a number or a numeric expression. If you do not specify the file-organization, the default is SEQUENTIAL. Normally processing time for SEQUENTIAL files is less than for RELATIVE ones, but random access is often more effective.

- file-type - Files can be stored in RAMDISK either as easily readable ASCII characters or in machine-readable binary form. If the information is going to be printed or displayed for people to use, ASCII format is usually a better choice. In most cases, binary records are preferred because they take up less space and are processed faster by the computer.

  To specify that you wish the file to be in ASCII format, enter DISPLAY. (The length of a DISPLAY-type record is limited to approximately 150 bytes.) To specify binary format, enter INTERNAL. If you do not specify a file-type, the default is DISPLAY.

- open-mode - This entry instructs the computer that the file may be both read and written upon (UPDATE), may only be read (INPUT), may only to be written to (OUTPUT), or may only be added to (APPEND).

  APPEND mode can only be specified for VARIABLE length records. If you do not specify an open-mode, the computer assumes the default UPDATE.

  Note: If a file already exists on the RAMDISK, specifying an open mode of OUTPUT to the same file-name writes over the existing file with the new file. You can prevent this by opening in the update mode and reading all the existing records so that you move to the end of the file or by using the RESTORE statement with the proper record number.

- record-type - File records may be all the same length (FIXED) or may vary in length (VARIABLE). If they are all FIXED, shorter records are padded to make up the difference. Any that are longer may be truncated to the proper length. Files that have FIXED-length records are processed faster than files with VARIABLE-length records but usually take up more space in the RAMDISK.

  If you like, you may specify a maximum length of a record by following VARIABLE or FIXED with a numeric expression. The maximum length for a VARIABLE file is 254 bytes, and the maximum for a FIXED file is 255 bytes. If you do not specify a record length, the default is 80.

RELATIVE files must have fixed-length records. If you d
not specify a record-type for a RELATIVE file, the
default is FIXED 80 .
SEQUENTIAL files have either FIXED or VARIABLE-length
records. If you do not specify a record-type for a
SEQUENTIAL file, the default is VARIABLE. A file with
FIXED-length records may be reopened for either RELATIV
or SEQUENTIAL access.

Examples of OPEN statements:

OPEN #1:"DSKR.TESTFILE"	Creates or reopens a file in the RAMDI with the name "TESTFILE". This file is SEQUENTIAL file in the UPDATE mode wit DISPLAY format and VARIABLE length records having a maximum length of 80 bytes. (These are the default attribut assigned by the computer.)
OPEN #230:O$,INTERNAL	Creates or reopens a file in the RAMDI with a name TEST if O$ equals DSKR.TES All assigned attributes are default.

CLOSE

The CLOSE statement discontinues the association between a file and a program. After the CLOSE statement is performed, the file is not available unless it is opened again with an OPEN statement. Files may optionally be deleted by adding :DELETE to the end of the CLOSE statement.

Format:

```
CLOSE #file-number [:DELETE]
```

The file-number has to be the same number which was used in the OPEN statement to open the file.

Examples of CLOSE statements:

CLOSE #1	Closes the file numbered #1.
CLOSE #3:DELETE	Closes the file #3 and deletes it after having it closed.

If you do not close a file, data on it may be lost. If a program ends due to a BREAK statement, by pressing CLEAR, or because of an error, files may not be closed even if you have a CLOSE statement later in the program.
NEW, or BYE if you wish to leave BASIC, or Editing the program also automatically closes any open files.
Note: If you leave TI BASIC by pressing QUIT, data may be lost. Leave TI BASIC only by entering BYE when you are processing files.

DELETE

The DELETE statement is used to delete files.

Format:

```
DELETE "DSKR.file-name"
```

A file-number would cause the error "* INCORRECT STATEMENT". The DELETE statement can also be used as a command.
Special use of the DELETE statement is made by RAMDISK-software without affecting this normal function.

Example of DELETE statement:

DELETE "DSKR.DEMO1"	The file named DEMO1 stored in the RAMDISK will be erased.

INPUT

The INPUT statement allows you to read data from files.
It only can be used with files opened in INPUT or UPDATE mode.

Format:

```
        INPUT #file-number [,REC record-number]:variable-list
```

The file-number and variable-list must be included in the INPUT statement. The record-number may optionally be included when reading random-access files.

- o file-number - The file-number is the number assigned to a particular file by the OPEN statement. The file-number is entered as a number sign (#) followed by a number or a numeric expression that, when rounded to the nearest integer, is a number from 1 to 255 and is the number of an open file.
- o record-number - A record-number refers to the record on th file which you want to read. The record-number can only be specified for RELATIVE files. (SEQUENTIAL files are read i sequential order.)
- o variable-list - The variable-list is the list of variables into which you want the data from the file to read. It consists of string or numeric variables separated by comma

Examples of INPUT statements:

INPUT #1:X$	Puts into X$ the next value available the file that was opened as #1.
INPUT #20:A,B,O$	Puts into A, B, and O$ the next three values from the file that was opened a #20.
INPUT #19,REC 13:OL1	Puts into OL1 the first value of recor number 13 of the file that was opened #19.
INPUT #6:O,P,R,	Puts into O, P, and R the next three values from the file that was opened a #6. The comma after R creates a pendin input condition. When the next INPUT statement using this file is performed one of the following actions occurs. If the next INPUT statement has no REC clause, the computer uses the data beginning where the previous INPUT statement stopped. If the next INPUT statement includes a REC clause, the computer terminates th pending input condition and reads the specified record.

PRINT

The PRINT statement allows you to write data onto files.
It only can be used with files opened in OUTPUT, UPDATE, or APPEND mode.

Format:

```
PRINT #file-number [,REC record-number][:print-list]
```

The file-number must be included in the PRINT statement. The record-number may optionally be included when writing to random-access (RELATIVE) files. The print-list is also optional.

- file-number - The file-number is the number assigned to a particular file by the OPEN statement. The file-number is entered as a number sign (#) followed by a number that, when rounded to the nearest integer, is a number from 1 to 255 and is the number of an open file.

- record-number - A record-number refers to the record on the file which you want to write. The record-number can only be specified for RELATIVE files.

- print-list - The print-list is the list of values that you want to put on the file. It consists of string or numeric variables separated by commas, colons or semicolons.

Examples of PRINT statements:

PRINT #2:X$	Puts the value of X$ into the next position of the file that was opened as #2.
PRINT #20:O,P;"2TIMES"	Puts the value of O,P and the string "2TIMES" into the next record of the file that was opened as #20.
PRINT #6:O,P,R,	Puts the values of O, P, and R into the next three positions in the file that was opened as #6. The comma after R creates a pending print condition. When the next PRINT statement is performed, one of the following actions occurs. If the next PRINT statement has no REC clause, the computer places the data immediately following the previous data. If the next PRINT statement has a REC clause, the computer writes the pending record onto the file at the position indicated by the internal counter and performs the new PRINT statement as usual.

EOF

The EOF (end-of-file) function indicates whether there is anothe
record to be read from a file.

Format:

```
          EOF(file-number)
```

The value of the file-number must correspond to the number of an open file.
The EOF function always assumes that the next record is going to be read sequentially, even if you are using a RELATIVE file.
The value of the EOF function depends on where you are in the file. If you are not at the end of the file, the value returned is zero (0). If you are at the end of the file, the function returnes a value of one (1). If the RAMDISK is full and you are at the end of the file, the function returns a negative value of one (-1).

Examples of the EOF function:

PRINT EOF(6)	Prints a value of 0, 1, or -1, accordin to whether you are at the end of the file that was opened as #6.
IF EOF(1)<>0 THEN 20	If you are at the end of the file that was opened as #1, control is transferre to line 20.

The usual way to keep track of the last record in RELATIVE files is to maintain a dummy record as the first record in the file. This record contains the number of records in the file. Each tim
you change the length of the file, you must update this record.

RESTORE

The RESTORE statement is used to get positioned at a specified record on a file.

Format:

```
        RESTORE #file-number [,REC record-number]
```

The file-number must be included in the RESTORE statement when using it with the RAMDISK. The record-number may optionally be included.

- file-number - The file-number is the number assigned to a particular file by the OPEN statement. The file-number is entered as a number sign (#) followed by a number or a numeric expression that, when rounded to the nearest integer, is a number from 1 to 255 and is the number of an open file.

- record-number - The record-number indicates which record on the file you want to access. Omitting the record-number (as for SEQUENTIAL files) sets the internal pointer to the first record.

Examples of RESTORE statements:

RESTORE #13	Resets the file pointer to the first record on the file that was opened as #13 with the consequence that the next record to be processed by PRINT or INPUT is the first in the file.
RESTORE #6,REC 11	Resets the file pointer to the twelfth record on the file that was opened as #6. (Remember that counting of records begins with number zero!)

With RELATIVE files, RESTORE is generally used only to prepare them for the use of the EOF function since the record to be accessed can also be specified in the PRINT or INPUT statements.

THE CENTRONICS INTERFACE

Connecting Devices to the Interface

After the 128 kByte MEMORYEXPANSION is connected and tested, a printer or other device can be attached to this peripheral.
The interface connector is located at the back of the housing next to the power jack.
To be sure that a given device is compatible, check its users manual for the cabling of its port.

Operating the Interface

The file-name for the CENTRONICS interface is "PLOT".
It is allowed to be operated with the following statements: OPEN, PRINT, and CLOSE.
As for file-processing refer to the explanations in the previous chapter.
The file-type for the CENTRONICS interface has to be specified only with DISPLAY allowing only OUTPUT mode.

An easy example how to tranfer data with TI BASIC to a printer shows the following program:

```
100 OPEN #1:"PLOT"
110 INPUT A$
120 PRINT #1:A$
130 GOTO 110
140 STOP
```

When performing an OPEN statement the interface sends out a 0-character, which has to be echoed by the printer. Without response an error-message will be displayed after a time.
The output of characters to the printer can be cancelled at any time by pressing the CLEAR key (FCTN 4). This will also cause an error.

To list a program, enter:

```
LIST "PLOT"
```

Linesize of Printouts

The default of the record-length if not specified in the OPEN statement, for the transfer to the printer is 80 characters.
By use of the option VARIABLE XXX in the OPEN statement (not allowed with LIST) the linesize can be changed for printers that can print more than 80 characters per line.
To output lines with 132 characters per line specify in the OPEN statement:

```
OPEN #1:"PLOT",DISPLAY,VARIABLE 132
```

Linefeed and Carriage Return Options at the end of line

Each record (line) is terminated from the interface by a character for carriage return (CR) coded as CHR$(13) and a linefeed coded as CHR$(10). For some printers these signals have to be supressed. This can be done with the options "PLOT.LF" (linefeed), "PLOT.CR" (carriage return) or both "PLOT.LF.CR" .
These options must be specified in the OPEN statement and are also allowed with LIST.
Attention: Normally for graphic-printouts both LF and CR have to be supressed to get correct ones.

Parallel Port Cabling

The pins are connected as listed below:

Pin 1: Strobe
Pin 2 through 9: Data; 2 = LSB 9 = MSB.
Pin 10: Busy
Pin 11 and pin 16: signal ground.
Pin 12: +5 V (max. 50 mA)

If looking at the rear panel on the connector Pin 1 is upside right, pin 2 is located under pin 1, and pin 3 is left beside pin 1 and so on.

APPENDIX A

Error Codes in BASIC

The following error codes relate to the RAMDISK when operated in BASIC including those for I/O errors.
The normal error codes for the computer are given in the User's Reference Guide.

Error codes are two digit numbers. The first digit indicates the command or statement involved in the error, and the second digit tells you the type of error.
In the catalog only the second digit is displayed.

FIRST DIGIT	COMMAND OR STATEMENT
0	OPEN
1	CLOSE
2	INPUT
3	PRINT
4	RESTORE
5	OLD
6	SAVE
7	DELETE
9	EOF

SECOND DIGIT	TYPE OF ERROR
0	The device specified could not be found.
1	DEVICE or FILE WRITE PROTECTED Usually this error does not occur with the RAMDISK
2	BAD OPEN ATTRIBUTE (Invalid option in the OPEN statement) One or more options were illegal or didn't match the file's actual characteristics. Sometimes if trying to open a file when the RAMDIS was not initialized.
3	ILLEGAL OPERATION In the most cases this error is caused by attempting to write to files opened with INPUT or attempting to read from files opened with OUTPUT.
4	OUT OF SPACE All three blocks of memory in the RAMDISK are full or attempt to open more than three files in the RAMDISK.
5	ATTEMPT TO READ PAST END OF FILE
6	DEVICE ERROR May occur if RAMDISK and computer were disconnecte
7	FILE ERROR The indicated file doesn't exist or you are trying to read a PROGRAM-type file as if it were data.

APPENDIX B

In Case of Difficulties

If the 128 kByte MEMORYEXPANSION does not appear to be working properly, check the following:

1. Power - Be sure all devices are plugged in. Then turn on the power to the units in the proper sequence: printer(s) disk drives and Peripheral Expansion System first, followed by the 128 kByte MEMORYEXPANSION the monitor and the console.

2. Cable - Check to be sure that the proper cable for the power supply is used. Check the cable for loose or broken leads. Check to see that the cable ist properly connected.

3. Software - Be sure all commands and statements are used as described in this manual. If the 128 k MEMORYEXPANSION works properly with modules but not with a program, the problem is probably within the program. Especially check the use of OPEN, INPUT, and PRINT.

4. If the above actions do not restore the correct function, disconnect all peripherals from the console and check the console. For the case of proper function of the console, connect each single peripheral to the console and check for correct function to find the defect peripheral.

5. 32 kByte Memory Expansion Card - Since the whole system should only be operated with the card or the 128 kByte MEMORYEXPANSION, the use of both can cause confusion in the system. Therefore be sure to have one of them removed. As for removing the card from the Peripheral Expansion Box look up the appropriate manual.
A damage of the devices however, cannot be caused by using both.

In case that the 128 k MEMORYEXPANSION is defect, our Service-Facility will be at your disposal.

MECHATRONIC 128 kByte RAM EXPANSION with CENTRONICS INTERFACE

THREE-MONTH LIMITED WARRANTY

MECHATRONIC EXTENDS THIS WARRANTY TO THE ORIGINAL CONSUMER PURCHASER OF THE 128 kByte RAM EXPANSION with CENTRONICS INTERFACE

WARRANTY DURATION

This MEMORYEXPANSION is warranted for a period of three (3)months from the date of original purchase by the consumer.

WARRANTY COVERAGE

This MEMORYEXPANSION is warranted against defective materials or workmanship. THIS WARRANTY IS VOID IF THE ACESSORY HAS BEEN DAMAGED BY ACCIDENT, UNREASONABLE USE, NEGLECT, IMPROPER SERVICE OR OTHER CAUSES NOT ARISING OF DEFECTS IN MATERIALS OR WORKMANSHIP.

WARRANTY DISCLAIMERS

ANY IMPLIED WARRANTIES ARISING OUT OF THIS SALE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, ARE LIMITED IN DURATION TO THE ABOVE THREE-MONTH PERIOD. MECHATRONIC SHALL NOT BE LIABLE FOR LOSS USE OF THE SYSTEM OR OTHER INCIDENTAL OR CONSEQUENTIAL COSTS EXPENSES, OR DAMAGES INCURRED BY THE CONSUMER OR ANY OTHER USER.

Some states do not allow the exclusion or limitation of implied warranties or consequential damages, so the above limitations or exclusions may not apply to you.

LEGAL REMEDIES

This warranty gives you specific legal rights, and you may also have other rights that vary from state to state.

WARRANTY PERFORMANCE

During the above three-month warranty period, your 128 kByte MEMORYEXPANSION will be repaired or replaced with a new or reconditioned unit of the same or equivalent model (at option of MECHATRONIC) when the unit is returned by prepaid shipment to MECHATRONIC. The repaired or replacement unit will be warranted for three months from date of repair or replacement. Other than the postage requirement, no charge will be made for the repair or replacement of in-warranty units.

MECHATRONIC Strongly recommends that you insure the unit for value, prior to shipment.

This manual was developed and translated by:
Peter Richter

With contributions and support by:
MECHATRONIC GmbH

Layout and typewriting were coordinated and executed by

GRS-Labors
Burggrafenstraße 1
D-8000 München 80
Western Germany

Published by:

MECHATRONIC GmbH
Dresdener Straße 21
D-7032 Sindelfingen