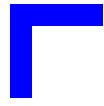


Modular Computers®



Bootstrap Loader

A software package for
on-site programming of FLASH devices

Manual Order Nr. 16878

User's Manual

Index 0110



Modular Computers®





Revision History

Index	Comment	Date	Initiator
0100	First Issue	September,1997	acb
0110	Load/remove of OS-9 modules	Jan. 1998	mf

This document contains proprietary information of PEP Modular Computers. It may not be copied or transmitted by any means, passed to others, or stored in any retrieval system or media, without the prior consent of PEP Modular Computers or its authorized agents.

The information in this document is, to the best of our knowledge, entirely correct. However, PEP Modular Computers cannot accept liability for any inaccuracies, or the consequences thereof, nor for any liability arising from the use or application of any circuit, product, or example shown in this document.

PEP Modular Computers reserve the right to change, modify, or improve this document or the product described herein, as seen fit by PEP Modular Computers without further notice.

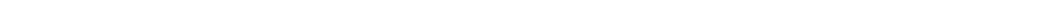




Table of Contents

Chapter 1

Introduction 1

Chapter 2

System Operation 2

2.1 *Startup* 2

2.2 *Entering the Command Mode* 2

2.3 *Preparing the Image* 3

2.4 *Programming with Motorola S-Records* 3

2.5 *Example using OS-9 development system* 3

2.6 *Example using DOS* 4

2.7 *Monitoring programming progress* 4

2.8 *Programming from an Absolute Address* 5

2.9 *BootWaitTime* 5

2.10 *ISaGRAF Start Flag* 6

2.11 *Example using Hyperterminal from Windows 95[®]
or Windows NT[®]* 6

Chapter 3

OS-9 Module Support 9

3.1 *General* 9

3.2 *Display OS-9 Modules* 9

3.3 *Removing OS-9 Modules* 9



3.4 *Loading and Updating OS-9 Modules* 9
 3.5 *Example using OS-9 development system*..... 10
 3.6 *Example using Hyperterminal from Windows 95[®]
 or Windows NT[®]* 10

Chapter

4

***Command Reference* 11**

4.1 *Boot Wait* 11
 4.2 *Load Flash* 11
 4.3 *Memory Display* 11
 4.4 *Port Format* 12
 4.5 *ISaGRAF Start Flag* 12
 4.6 *Reset Systems* 13
 4.7 *Help* 13
 4.8 *Load and Update Modules* 13
 4.9 *List Modules* 13
 4.10 *Remove Modules* 14

Appendices

Appendix A ISaGRAF applications on CM-302 15
 A.1 *List Modules* 15
 A.2 *Remove Modules* 15
 A.3 *Load and Update Modules* 15

Appendix B Utility 'genflash' 16

Appendix C Bootstrap Loader Start - 1 sec BREAK Signal 16



BOOTSTRAP LOADER FOR VM 30, VM 162, VM 172, VM(6)62, VM(6)42, VSBC 32, IUC-32, (V)IUC, VSBC-4, SMART I/O AND SMART II (CM302)

1 Introduction

The Bootstrap Loader is a stand-alone software located in FLASH memory which allows the user to safely update the contents of the FLASH and delay the boot process for a specified time.

The Bootstrap Loader has the capability of programming FLASH memory from “MOTOROLA S-RECORDS” or from an absolute memory address. If the programmed image does not work, the Bootstrap Loader can be entered again. The memory contents can be examined and another programming cycle initiated.

The Bootstrap Loader is delivered already installed on PEP’s CPU boards equipped with FLASH devices.

Please read this user manual before reprogramming any FLASH memory.

WARNING !

*When programming FLASH memory, ***NEVER*** press the RESET button or switch off the main-power! This may damage the Bootstrap Loader and will consequently leave the board unusable due to corrupt FLASH contents.*

However, the ABORT button may be used to cancel a running operation safely.



2 System Operation

2.1 Startup

After system reset, the Bootstrap Loader is started. It searches the FLASH memory area for a valid start key. If this start key is found, the Bootstrap Loader checks the 'BootWaitTime' from serial EEPROM or RTC. If the entry is valid, the continuation of the boot process is delayed by this time while flashing the green front panel LED to indicate that the system is alive but waiting for continuation. If the entry is not valid, a default of 5 seconds is used. After the BootWaitTime has passed, the program in FLASH is started.

The Bootstrap Loader has two modes of operation: non-interactive start mode as described above and the interactive command mode.

For normal board operation, only the non-interactive start mode is used to start a program in FLASH. This is done automatically without any user interaction. The interactive command mode is used to re-program the FLASH memory contents or change the BootWaitTime.

The serial term port operates at 9600 Baud, 8 bits / character, 1 stop bit and no parity.

2.2 Entering the Command Mode

There are two possible cases:

If no valid start key was found, the Bootstrap Loader's command mode is entered automatically¹⁾.

If the user wants to enter the Bootstrap Loader manually (e.g. for re-programming the FLASH contents) he must use the ABORT button on the front panel.

On CPUs without an ABORT button, the Bootstrap Loader will enter the command mode on detecting a BREAK character of at least one second length at the terminal connector.

Note: The ABORT button must not be pushed until the green LED appears, because this button generates a NMI and the exception vector tables must be initialized correctly to serve this NMI. Pressing the ABORT prior to the green LED leads to HALT in most cases. In this case, press the RESET button and try again.

Note: The ABORT button must, however, be pushed before the green LED stops flashing (BootWaitTime), because system control is passed to the downloaded binary image afterwards. The LED is cycled every 0.25 sec so if 1 second is specified as BootWaitTime, the LED will only flash 2 times.

Note: CPUs without an user LED or without an ABORT button uses a BootWaitTime of at least 5 seconds.

CTRL-x deletes the complete input line while CTRL-a restores the last input line.

¹⁾ The start key is a special combination of data appended at the end of the load program or inside the load program itself.



2.3 Preparing the Image

The image must be compiled / linked to run from the FLASH base address of the CPU. The image must start with the ResetSP / ResetPC vectors as usual for ROM / FLASH images on 68000 processor boards.

A binary image must be converted to Motorola S-records or loaded to a VME memory board with battery-backup, FLASH or EPROM population.

2.4 Programming with Motorola S-Records

Programming is done with the *If* command.

The *If* command accepts S1, S2 and S3 records. Operation is terminated by the appropriate S9, S8 or S7 record. Other types of records are ignored.

The checksum of every record is checked; bad records are refused by the Bootstrap Loader. The address range of every record is also checked; records that try to overwrite the Bootstrap Loader are refused. Additionally, every record must match the programmable area exactly. To give the user an overview of the available ranges, the startup banner includes address information.

If S1 or S2 record input is preferred, please note that these records only include 16 and 24-bit wide addresses. Therefore, in order to reach the FLASH area an address offset must be specified using the '-o=...' option of the *If* command. Additionally, it must be ensured that the code is not larger than the covered address range.

Note: The If command cannot be used to program Motorola S-records to RAM areas.

For sending programming data, the lower (*term*) or upper (*serO*) RJ12 front panel connectors can be used. The *serO* port should be preferred because in this configuration it is possible to monitor the progress of the operation via the *term* port.

If not otherwise specified, sectors which are not touched by the programming operation are not erased. If you want to erase all sectors while programming, the '-c' option can be specified along with the *If* command. This is useful for software which searches memory during startup and should not find any old modules (e.g. OS-9).

Make sure that the XON/XOFF protocol is used on the host side. This is a fixed setting and cannot be changed. Additionally, make sure that your host does not stop transmission after a number of lines (e.g. OS-9: use the 'nopause' attribute).

Serial parameters can be modified with the *pf* command.

Note: On CPUs without a ser0 serial connection the Bootstrap Loader will refuse loading the data from ser0.

2.5 Example using OS-9 development system

The host is assumed to be an OS-9 development system. A serial cable is used to connect the *ser0* port of the board to program to *t0* of the development system. Additionally, we assume that we want to program a PEPbug image which is available as a file '*pbVM42*' in a binary image format. The serial connection should run at 38400 Baud. The following steps must be performed:

Bootstrap Loader



Host:

```
xmode /t0 baud=38400 nopause  
iniz /t0
```

Target:

```
pf ser0 38400  
lf -u
```

Host:

```
binex -s3 -a=4000000 pbVM42 >/t0
```

2.6 Example using DOS

The host is assumed to be a PC with Windows, Windows95 or WindowsNT. A serial cable is used to connect the *ser0* port of the board to program to *COM2* of the PC. Additionally, we assume that we want to program a Motorola S-record built for address 0, e.g the VxWorks file *bootrom.hex*. The serial connection should run at 19200 Baud. The following steps must be performed:

Host:

Under MSDOS, configure the *COM 2* port to the correct parameters:

```
mode com2: baud=19200 parity=n data=8 stop=1
```

Target:

```
pf ser0 19200 lf -o=4000000
```

Host:

```
type bootrom.hex >com2:
```

2.7 Monitoring programming progress

In both examples, the programming can be monitored over the *term* port. The characters displayed have the following meaning:

- r Read S-record; valid and in range
- t Protected sector touched
- e Erase sector
- c Copy to buffer, program later
- p Program record

None of the above characters indicate an error. The first sector (which includes Reset SP / PC) and the last sector (which includes the Bootstrap Loader itself) are protected. These sectors are not immediately programmed like the other sectors. The contents of these protected sectors are buffered in RAM and programmed at the end of the operation. This is done to limit the time the Bootstrap Loader itself is not in FLASH or not startable, because if the Bootstrap Loader crashes during this critical period of time, it will not start again afterwards.





WARNING !

*When programming FLASH memory, *NEVER* press the RESET button or cycle power! This may damage the Bootstrap Loader and will consequently leave the board unusable due to damaged FLASH contents. The ABORT button may be used to cancel a running operation.*

'-q' suppresses all messages and warnings except error messages.

Programming over the *term* port is also supported, but in this case the loader programs in the background by default and the propagation of the process cannot be monitored.

It is recommended that by default the programming over the *ser0* port should be used.

If the process must be aborted, press the ABORT button and try again.

On CPUs without an ABORT button the process cannot be aborted.

2.8 Programming from an Absolute Address

The second possibility to program FLASH memory is to program it from an absolute address. The image to program must be located in a visible address range, for example on the VMEbus. A memory card with battery-backup, FLASH or EPROM can be used to hold the image to program. If we assume that the image is located at 0x87000000 and is 0x123456 bytes large we must type the following at the command prompt of the Bootstrap Loader:

```
If -m=87000000 -l=123456
```

The characters which are displayed now have the same meaning as if we are programming from S-records, but the time needed for each step to complete may be longer because the loader tries to program with the largest possible block size that it can manage.

Again, '-c' can be used to clear untouched sectors.

Background operation is not supported and it is also not possible to specify an offset.

The programming cannot be aborted with ABORT.

Note: On CPUs without an ABORT button the process cannot be aborted.

2.9 BootWaitTime

The command *bw* can be used to display / change the current BootWaitTime. Available delays are 1-2-5-10-20-50 seconds. CPUs without an user LED or without an ABORT button have a BootWaitTime of at least 5 seconds. The Bootstrap Loader refuses setting a BootWaitTime of 1-2 seconds.

Note: The BootWaitTime is stored in the boot section of the serial EEPROM or RTC. This section is validated with a CRC code to avoid the setting of random parameters. If the CRC of the Boot section is not valid, changing the BootWaitTime has no effect because the bw command does not validate an invalid CRC to avoid undesired side effects. In this case, the default of 5 seconds is always used. On CPU equipped with a RTC which holds the BootWaitTime there is no CRC value included.

To validate an invalid CRC, the appropriate utility from an operating system must be used (e.g. *ee_config* from OS-9).

Bootstrap Loader



2.10 ISaGRAF Start Flag

The command *si* can be used to display / change the setting of the ISaGRAF Start Flag. Valid values are 0 and 1.

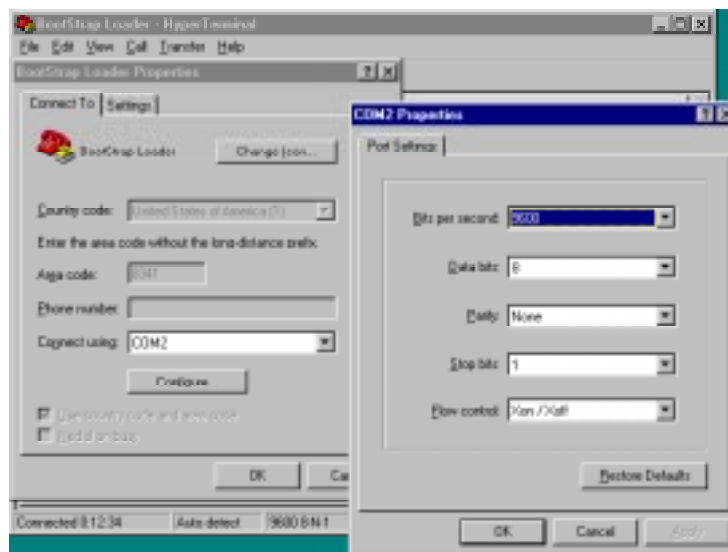
See pep-vdme.wri on ISaGRAF Workbench for more information about the usage of this flag.

Note: The ISaGRAF Start Flag is stored in the boot section of the serial EEPROM or RTC. This section is validated with a CRC code to avoid the setting of random parameters. If the CRC of the Boot section is not valid, changing the ISaGRAF Start Flag has no effect because the *bw* command does not validate an invalid CRC to avoid undesired side effects. In this case, the default of 5 seconds is always used. On CPU equipped with a RTC which holds the ISaGRAF Start Flag there is no CRC value included.

Note: The setting of this flag has no influence on the behaviour of the Bootstrap Loader.

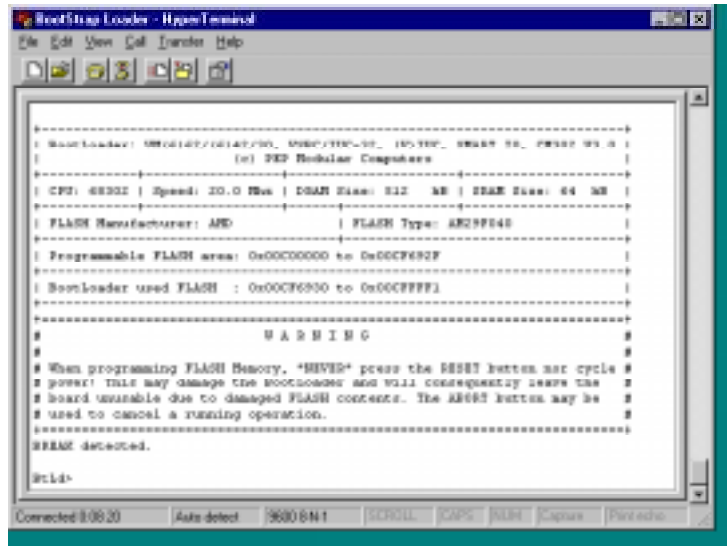
2.11 Example using Hyperterminal from Windows 95® or Windows NT®

The host is assumed to be a PC with Windows 95 or Windows NT. A serial cable is used to connect the term port of the board to program to COM2 of the PC. Additionally, we assume that we want to program a MOTOROLA S-record built for the FLASH address of the board. The serial connection runs at the default configuration of 9600 baud with no parity and one stopbit. The properties should look like the following:

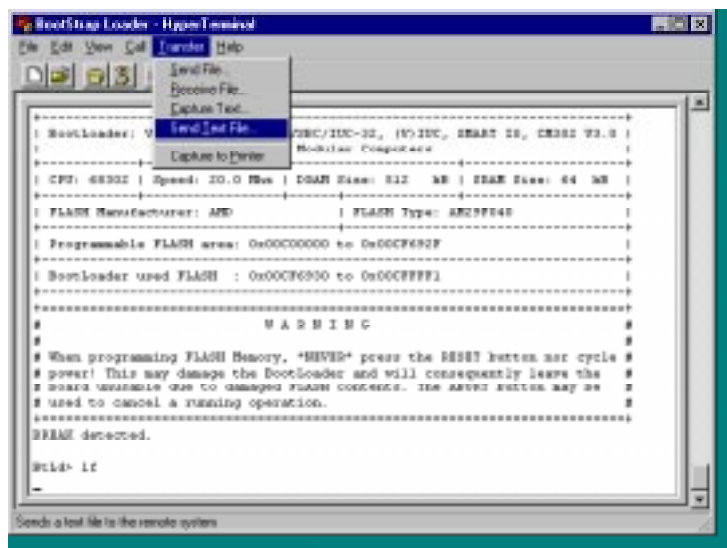




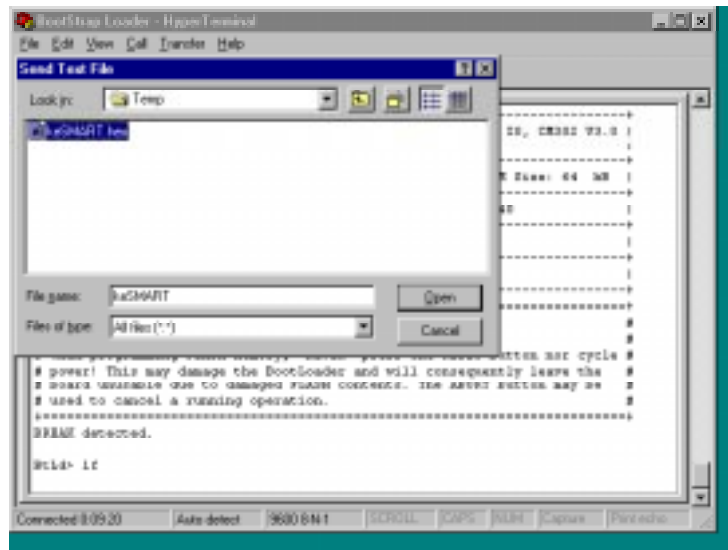
After entering the Command Mode the following screen should appear:



Enter 'lf' on the target. The target now waits until the MOTOROLA S-records have been transferred over the term. Select **Transfer/Send Text file** dialogue box and select the file containing the MOTOROLA S-records.



Bootstrap Loader



After selecting **Open** the dialogue box disappears and the MOTOROLA S-records are transferred. As soon as the complete file is sent, the prompt of the Bootstrap Loader appears again.

Note: No status information is displayed.





3 OS-9 Module Support

3.1 General

The commands *lsmod*, *rmmod* and *ldmod* can be used to display, delete, load and update OS-9 modules on base of a single module.

The progress of searching, loading, removing and programming modules can be monitored over the *term* port. None of the displayed characters indicate an error. The characters displayed have the following meaning:

- s Valid OS-9 module found
- r Read S-record; valid and in range
- e Rease sector
- p Programm record

Note: Every OS-9 module contains a header with a parity value and a CRC value for the whole module. Both values are checked and only modules having a valid header parity and a valid CRC value are accepted as OS-9 modules.

3.2 Display OS-9 Modules

The command *lsmod* displays a list of all valid OS-9 modules in the programmable FLASH area. It is always an extended list displayed which is similar to the output of the OS-9 utility 'mdir -e'.

If there is any gap between two modules the command produces a separate line, giving information about the position and the size of the gap. This line is indicated by '*** gap ***'.

After the last valid module is displayed the command produces a special line giving information about the position and the size of the free FLASH area. This line is indicated by '*** gap ***'.

3.3 Removing OS-9 Modules

Every OS-9 module displayed with *lsmod* can be removed with *rmmod*. Removing a module will unrecoverable remove the module from the FLASH. The resulting gap in the FLASH is filled by shifting all modules behind the removed module.

Note: At least one module must stay in the FLASH. This module cannot be removed.

Note: Removed modules cannot be recovered.

3.4 Loading and Updating OS-9 Modules

With the command *ldmod* existing OS-9 modules can be updated and new modules can be loaded over the serial connections *term* or *ser0* using S-records.

Loading a module which is already in the FLASH will remove the existing module from the FLASH. Removing a module will unrecoverable remove the module from the FLASH. Loading an existing module will always update the existing module in the FLASH. There is no check of the module edition number. The new module is always programmed behind the last module in the FLASH.

Bootstrap Loader



Note: The maximum size of the binary module which can be loaded or updated is 256 Kbytes. If a module greater than 256 kBytes is sent to the Bootstrap Loader each S-record which exceeds this size will be refused. As a result the loaded module is declined due to an invalid CRC value.

Note: If only one module is in the FLASH, this module cannot be updated.

Note: Loading or updating modules from an absolute memory address is not supported.

3.5 Example using OS-9 development system

The host is assumed to be an OS-9 development system. A serial cable is used to connect the *ser0* port of the board to program to *t0* of the development system. The serial connection should run at 38400 baud. The following steps must be performed to update the 'sysgo' module:

Host:

```
xmode /t0 baud=38400 nopause  
iniz /t0
```

Target:

```
pf ser0 38400  
ldmod -u
```

Host:

```
binex -s=3 sysgo >/t0
```

3.6 Example using Hyperterminal from Windows 95® or Windows NT®

See 'Example using Hyperterminal from Windows 95® or Windows NT®' on page 6 point 2.11.



4 Command Reference

4.1 Boot Wait

Syntax

`bw [<time>]`

Description

Without parameters, `bw` displays the current setting.

For `<time>` 1, 2, 5, 10, 20 and 50 may be specified as time in seconds. Other values are not supported.

Note: CPUs without an user LED or without an ABORT button have a `BootWaitTime` of at least 5 seconds. The Bootstrap Loader refuses setting a `BootWaitTime` of 1-2 seconds.

4.2 Load Flash

Syntax

`If [-o[=]<offset>] [-u] [-q] [-c] [-m[=]<adr> -l[=]<len>]`

Description

Without parameters, the FLASH is loaded using S-records over the `term` port.

'`offset`' is a signed 32 bit offset which is added to every record and can be used to move the S-records to the FLASH position.

Note: This option must be used if S1 or S2 records are used.

'`u`' must be used to download over `ser0`.

'`q`' suppress all messages except error messages.

'`c`' clears all untouched sectors and leaves no old code fragments.

For a Load Flash from an absolute address, the `-m` / `-l` options must be used.

Note: On CPUs without a `ser0` serial connection the Bootstrap Loader will refuse a command including the '`u`' option.

4.3 Memory Display

Syntax

`md [<adr>]`

Description

Without parameters specified, the FLASH contents starting at the beginning of the programmable FLASH area are displayed. This function is not limited to FLASH and other address ranges can be specified.

Note: The `ResetPC` in FLASH is not identical to the `ResetPC` from the programming source (S-records memory block).



4.4 Port Format

Syntax

```
pf [<port> [<baud>][/ [<bitschar>][/ [<parity>][/ <stops>]]]]
```

Description

Without parameters specified, the current serial port settings are displayed.

<port> specifies the serial port. Valid values are *term* or *ser0*.

<baud> specifies the baud rate. The values 50, 75, 110, 134.5, 150, 300, 600, 1200, 1800, 2000, 2400, 3600, 4800, 7200, 9600, 19200 and 38400 Baud can be specified.

<bitschar> specifies the bits / character. Valid values are 7 or 8.

<parity> specifies if parity should be checked / generated. The value n specifies none, o for odd and e for even parity.

<stops> specifies the stopbits which will be generated. Valid values are 1 or 2.

Note: No spaces are allowed between the options. Options must be separated with the '/'. Not all options must be specified, but the '/' characters must be present to distinguish the different options from each other. The sequence can be aborted after every option.

Note: On CPUs without a ser0 serial connection the Bootstrap Loader will refuse setting ser0 port parameters and will not display ser0 port parameters.

Examples

Setting *term* to 300 Baud, 7 Bits/char, odd parity and 2 stopbits:

```
pf term 300/7/o/n
```

Set the bits / character field to 7 for *ser0* only:

```
pf ser0 /7
```

Set the stopbits field to 2 for *ser0*:

```
pf ser0 ///2
```

4.5 ISaGRAF Start Flag

Syntax

```
si [<flag>]
```

Description:

Without parameters specified, the current setting of the ISaGRAF start flag is displayed. <flag> specifies the new setting. Valid values are 0 and 1, where 1 means start of ISaGRAF.

Note: The value of this flag has no influence on the behavior of the Bootstrap Loader.





4.6 Reset System

Syntax

rs

Description

This command exits the Bootstrap Loader and resets the system. It terminates the Bootstrap Loader command mode and resets the complete system, generating a system reset with the on-board watchdog.

4.7 Help

Syntax

? or *help*

Description

This command prints the online help page. The display of the help text varies between the different CPUs reflecting the difference of the CPUs.

4.8 Load and Update Modules

Syntax

lmod [-qu]

Description

Without any parameters, the module is loaded or updated using S-records over the *term* port.

'-u' must be used to download over *ser0*.

'-q' suppress all messages except error messages.

Note: On CPUs without a *ser0* serial connection the Bootstrap Loader will refuse a command including the '-u' option.

4.9 List Modules

Syntax

lsmod

Description

List all OS-9 modules in the programmable FLASH area. The display output is similar to the output of the OS-9 utility 'mdir -e'.



4.10 Remove Modules

Syntax

`rmmod [-q] <name>`

Description

Remove module with the <name> from the FLASH.
[-q] suppress all messages except error messages.



Appendix A *ISaGRAF applications on CM-302*

On the CM-302 80 Kbytes of the FLASH are reserved for ISaGRAF applications. The commands `ldmod`, `lsmod` and `rmmod` have a different behavior on the CM-302 than on the other CPUs concerning the handling of this special FLASH area for ISaGRAF applications.

Each module with a name according to 'isa?1' is treated as ISaGRAF application on the CM-302.

Note: Do not include any ISaGRAF applications in the image programmed via `lf`. Removing or updating these modules with `rmmod` or `ldmod` is not possible.

A.1 List Modules

Due to the fixed location of the ISaGRAF applications in the FLASH, there is a gap between the last OS-9 module and the first application. The display shows two lines of free FLASH, indicated with '*** free ***'. The first shows the free FLASH for 'normal' OS-9 modules and the second for ISaGRAF applications.

A.2 Remove Modules

If the name of the module is an ISaGRAF application, the module is only searched in the special area of the FLASH.

A.3 Load and Update Modules

Loading a new module or updating an existing ISaGRAF application will always go into the special FLASH area for ISaGRAF applications.



Appendix B *Utility 'genflash'*

This utility is included in the 'PEP Utilities'.

'genflash' takes the image of the Bootstrap Loader '-l' and an user-images '-i' and generates a FLASH image '-o' for a specific board '-b'. The output-file has the exact size '-s' and contents of the installed FLASHs. This image can directly programmed into the FLASHs.

Syntax

```
genflash -b=<board> -i=<file> -o=<file> -s=<size> [-v]
```

Options:

-b=<board>	board type: VM30, VM42, VIUC, IUC, SMART, VSBC4, VSBC32PB, CM302
-i=<file>	filename of user-image
-l=<file>	bootloader filename
-o=<file>	output filename
-s=<size>	size of FLASH in Kbytes: 256, 512, 1024, 2048, 4096
-v	verbose flag
-?	help message

Note: To get images for VM62/VM642/VM662 use board type VM42.

Note: The possible size of FLASH depends on the chosen board type.





Appendix C *Bootstrap Loader Start - 1 sec BREAK Signal*

On CPUs without an ABORT button, the Bootstrap Loader will enter the command mode on detecting a BREAK character of at least one second length at the terminal connector.

Since not all terminals are capable of generating a BREAK signal lasting for one second, the modification described below helps achieve it.

One possibility to generate the BREAK character is to use a special adapter. This adapter contains a switch which can disconnect the sending line from the host to the target and connect the disconnected line from the target to the DTR line of the host (CTS on target). The DTR line of the host has the level of a BREAK character.

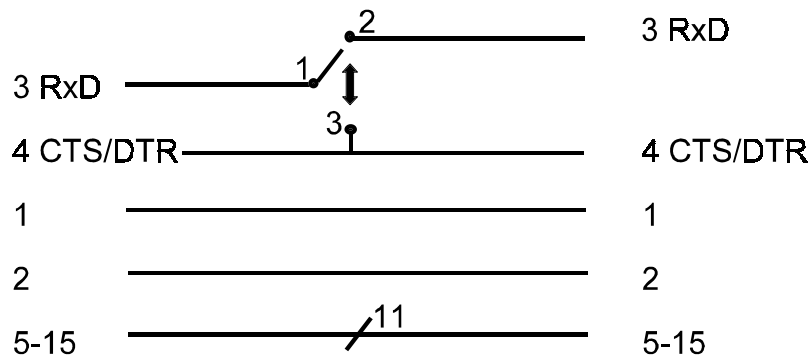
This switch may be integrated at any position in the connection line, as long as the DTR/CTS line is connected to the host.

After setting the target, the adapter should be configured so that 1-3 is shorted. On CPUs containing a user LED, the LED indicates that the Bootstrap Loader is looking for the BREAK. After two flashes the adapter can be configured to 1-2 and immediately the startmenu should appear. If the LED still flashes, reconfigure to 1-3 and wait for another two flashes. On CPUs without an user LED wait for approximately 2 seconds and than configure to 1-2. If the startmenu does not appears immediately, reconfigure to 1-3, wait for approximately another 2 seconds.

A) Adapter at 15 pin connector on OS-9 side

15-pin male

15-pin female



Normal Operation: 1-2

Sending BREAK: 1-3

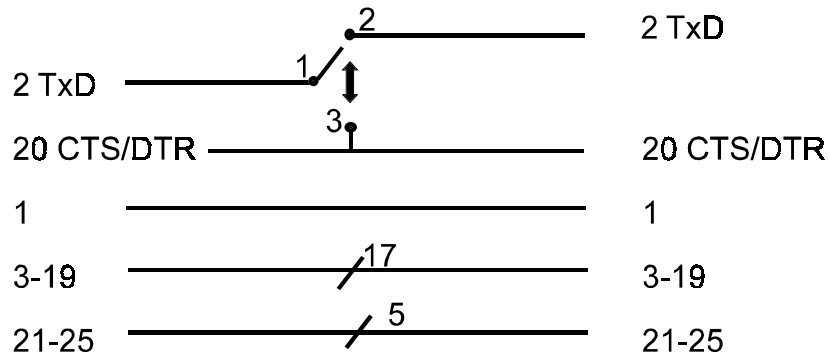
Bootstrap Loader



B) Adapter at 25 pin connector on PC side

25-pin male

25-pin female



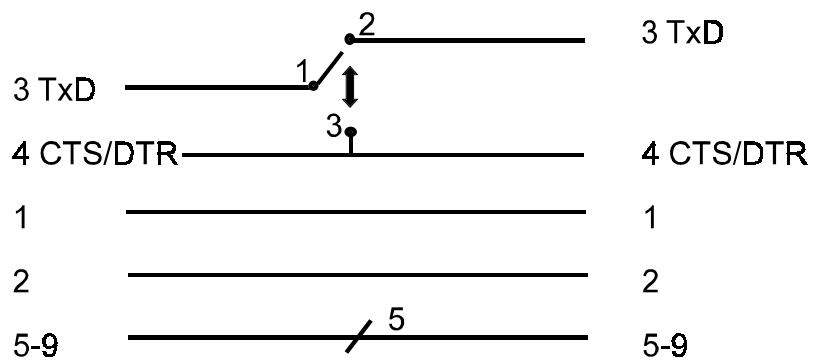
Normal Operation: 1-2

Sending BREAK: 1-3

C) Adapter at 9 pin connector on PC side

9-pin male

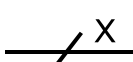
9-pin female



Normal Operation: 1-2

Sending BREAK: 1-3

Note: Always use a serial cable for CTS/DTR handshake from the adapter to the host.

 **Note:** This description stands for x separate lines.