A SOLECTRON COMPANY

PowerBoot

Rel 2.4

for PPMC-280

Instruction Set

P/N 221378 Revision AF
April 2004

Copyright

A SOLECTRON COMPANY

World Wide Web: www.forcecomputers.com
24-hour access to on-line manuals, driver updates, and application notes is provided via SMART, our SolutionsPLUS customer support program that provides current technical and services information.

Headquarters

The Americas	Europe	Asia
Corporate Headquarters/CA	**Force Computers GmbH**	**Force Computers Japan KK**
Force Computers	Lilienthalstr. 15	Shibadaimon MF Building 4F
4211 Starboard Drive	D-85579 Neubiberg/München	2-1-16 Shiba Daimon
Fremont, CA 94538	Germany	Minato-ku, Tokyo 105-0012 Japan
Tel.: +1 (510) 624-8274	Tel.: +49 (89) 608 14-0	Tel.: +81 (03) 3437 6221
Fax: +1 (510) 445-6007	Fax: +49 (89) 609 77 93	Fax: +81 (03) 3437 6223
Email: support@fci.com	Email: support-de@fci.com	Email: support-de@fci.com

Contents

Tables

Using This Guide

This guide explains how to use the PowerBoot command-line interface (CLI) on PPMC-280 modules. The guide also details how you can boot Linux on PPMC-280.

The guide details instructions for PowerBoot Release Version 2.4 for PPMC-280.

This guide provides the following information:

Chapter Number	Name of Chapter	Description
1	"PowerBoot Instruction Set" chapter	Details generic PowerBoot commands.
2	"PowerBoot Differences for PPMC-280" chapter	Details PowerBoot PPMC-280 board specific commands.
3	"Getting Started with PowerBoot" chapter	Details how to set up your system for PowerBoot, features of PowerBoot, how to enter and exit from PowerBoot.

This guide is for designers, developers, and system integrators who are designing and building PPMC-280 modules into application systems. Manufacturing and field technicians and support specialists can also use the information in this manual to help configure systems and diagnose problems.

The guide assumes you have experience with system design and the module's bus design and specifications.

Conventions

Notation	Description
	All numbers are decimal numbers except when used with the following notations:
0000.0000_{16}	Typical notation for hexadecimal numbers (digits are 0 through F), e.g. used for addresses and offsets. Note the dot marking the 4th (to its right) and 5th (to its left) digit.
0000_2	Same for binary numbers (digits are 0 and 1)
Bold	Character format used to emphasize a word

Notation	Description
Courier	Character format used for on-screen output
Courier+Bold	Character format used to characterize user input and to separate it from system output
Italics	Character format for references and for table and figure descriptions.
<text>	Typical notation used for variables and keys.
[text]	Typical notation for buttons.
Note:	No danger encountered. Pay attention to important information marked using this layout.
Caution ⚠	Possibly dangerous situation; slight injuries to people or damage to objects possible.
Danger 🛑	Dangerous situation; injuries to people or severe damage to objects possible.

Abbreviations

ANSI	American National Standards Institute
ARP	Address Resolution Protocol
CLI	Command Line Interface
CPU	Central processing unit
CRC	Cyclic Redundancy Checking
DRAM	Dynamic random access memory
I/O	Input/output
JFFS2	Journalling Flash File System Version 2
L2	Level 2 (cache)
LED	Light emitting diode
MAC	Media Access Control
MMU	Memory Management Unit

NFS	Network File System
NVRAM	Nonvolatile random access memory
PCI	Peripheral Component Interconnect (bus)
PMC	PCI mezzanine card
POT	Power-on test
PPC	PowerPC
RAM	Random access memory
RARP	Reverse Address Resolution Protocol
SDRAM	Synchronous dynamic random access memory
TFTP	Trivial File Transfer Protocol
UART	Universal asynchronous receiver-transmitter
VPD	Vital Product Data

Revision History

Order Number	Revision	Date	Description
221378 410 000	AA	July 2003	Initial release Release A1.0
221378 410 000	AB	July 2003	Release 1.1 Editorial Changes
221378 410 000	AC	October 2003	Release 2.0
221378 410 000	AD	January 2004	Release 2.2
221378 410 000	AE	March 2004	Release 2.3
221378 410 000	AF	April 2004	Release 2.4

Other Sources of Information

For further information refer to the following documents:

Company	Web Address	Document
Force Computers	www.forcecomputers.com	The following documents are available via http://splus.forcecomputers.com /cgi-bin/user/account/services
VITA Standards Organization	www.vita.com	Processor PMC Standard for Processor PCI Mezannine Cards, VITA32

1

PowerBoot Instruction Set

PowerBoot Commands

PowerBoot is firmware that provides basic test and debug commands. It is stored in the on-board boot ROM or Flash.

PowerBoot provides the following commands:

- command to fill a memory area with values or with an ASCII string:

  "BF – Filling Blocks" on page 1-6
- command to copy the contents of a source memory area to a destination memory area:

  "BM – Copying Blocks" on page 1-7
- command to search a memory area for values or for an ASCII string:

  "BS – Searching Blocks" on page 1-7
- command to test the memory:

  "BT – Testing the Memory" on page 1-8
- command to compare two memory areas:

  "BV – Verifying Blocks" on page 1-10
- commands to set and get the MAC address for the two on-board Ethernet ports

  "SETMAC-Set the MAC Address" on page 1-26

  "GETMAC-Get the MAC Address" on page 1-17
- commands to erase, to verify or program flash memory devices:

  "FERASE – Erasing Flash Memory" on page 1-11,

  "FPROG – Programming Flash Memory" on page 1-13,

  "FREAD-Reading Flash Memory" on page 1-14

  "FVERIFY- Verifying Flash Memory" on page 1-15
- commands to read and write to EEPROM devices

  "FRCEEPROMWRITE-Writing to EEPROM Device" on page 1-16

  "FRCEEPROMREAD-Reading the Contents of EEPROM Devices" on page 1-17
- command to start a binary image located in the DRAM memory:

"GO – Starting Binary Images without Cyclic Redundancy Check Header" on page 1-18

"GOLINUX– Loading Linux using RAMDISK" on page 1-18

- command to display a help message:

  "HELP – Displaying Help Message" on page 1-19

- commands to modify or to display the memory contents:

  "M – Modifying the Memory" on page 1-20,

  "MD – Displaying Memory Contents" on page 1-23

- commands to download and execute binary images from a remote system via a network interface or to save a memory area via network into a file:

  "NETLOAD – Loading a Binary Image Through Ethernet" on page 1-23

  "Initial Boot Screen" on page 1-26 - Resets the board

- stores/executes a set of upto 10 commands in boot flash.

  "BATCH - Stores/Executes a set of commands" on page 1-29

- changes the serial port baudrate for the session

  "BAUD - Change Baudrate" on page 1-29

- changes the netload/bootp boot file name. This command also updates the boot parameters stored in Boot-flash.

  "CHANGE_BOOT - changes Bootimage filename" on page 1-30

- prints or sets command line to be passed to Linux kernel, when linux is booted using GOLINUX.

  "LINUXCMDLINE - edits Linux command line" on page 1-30

Specifying Addresses

Hexadecimal addresses, e.g. $0000.4C00_{16}$, can be entered in their complete form in the command line, i.e. 00004C00, or the preceding zeros can be left out, i.e. it is also possible to enter only 4C00.

Long-aligned addresses are addresses which are divisible by 4 without a remainder.

Command Description Structure

The command description is structured as follows:

1. Short description of the command.

2. Syntax description consisting of:

   command parameter1 parameter2 [parameter3]

   The parameter given in brackets is optional. The following sections explain respective parameters.

3. Detailed description of the command completing the short command description given at the beginning of the respective section.

4. Example.

BF – Filling Blocks

BF fills a specified RAM memory area with a byte, a word, or a long constant or with a user defined ASCII string.

SYNTAX	bf begin end value
begin	is the first byte of the memory area used for filling.
end	is the first byte of the memory area not used for filling.
value	specifies the constant used for filling. It can be one of the following constants:

- byte value = a byte constant followed by the character B (e.g. A5 B)
- word value = a word constant followed by the character W (e.g. A5B6 W)
- long value = a long constant followed by the character L (e.g. A5B6C7D8 L)
- string value = an ASCII string followed by the character P, the ASCII string has to be set in inverted commas (e.g. "Hello" P)

EXAMPLE

In the following example a 4-KByte memory beginning at 0000.2D00$_{16}$ and ending at OOOO.3D00$_{16}$ is filled with the word constant A5A5$_{16}$. Thereby the 4-KByte memory is filled:

```
PowerBoot>bf 2D00 3D00 A5A5 W
PowerBoot>
```

In the second example an 8-KByte memory beginning at 0000.4C00$_{16}$ and ending at 0000.6C00$_{16}$ is filled with the ASCII string "Hello":

```
PowerBoot>bf 4C00 6C00 “Hello” P
PowerBoot>
```

BM – Copying Blocks

BM copies the contents of a specified source memory area to a memory area starting at a specified destination address. However, if the source and the destination areas overlap, only the destination area will be complete.

SYNTAX bm begin end destination

begin specifies the starting address of the source memory area, i.e. the first byte of the source memory area used for copying.

end specifies the end address of the source memory area, i.e. the first byte of the source memory area not used for copying.

destination

specifies the starting address of the destination memory area where the first byte of the source memory area is copied to.

EXAMPLE In the first example a 4-KByte memory is copied from address $0000.2D00_{16}$ to address $0000.5E00_{16}$:

```
PowerBoot>bm 2D00 3D00 5E00
PowerBoot>
```

In the second example an 8-KByte memory is copied from address $0000.4C00_{16}$ to address $0000.5C00_{16}$. In this case the areas overlap each other:

```
PowerBoot>bm 4C00 6C00 5C00
PowerBoot>
```

BS – Searching Blocks

BS searches a memory area for a byte, a word, or a long constant or for a user defined ASCII string. There are two tasks which can be done by means of BS:

- In order to display the locations where the searched byte, word, long constant, or ASCII string is found, select syntax 1.
- In order to display the locations where the searched byte, word, long constant, or ASCII string is not found, select syntax 2.

SYNTAX bs begin end value

begin specifies the starting address of the memory area which is searched, i.e. the first byte of the memory area which is read.

end specifies the end address of the memory area which is searched, i.e. the first byte of the memory area which is not read.

value specifies the constant to search for. It can be one of the following constants:

– *byte value* = a byte constant followed by the character *B* (e.g. *3F* B)
– *word value* = a word constant followed by the character W (e.g. *3F3F* W)
– *long value* = a long constant followed by the character L (e.g. *EFEFEFEF* L)
– *string value* = an ASCII string followed by the character P, the ASCII string has to be set in inverted commas (e.g. "*FORCE*" P)

EXAMPLE In this example a 4-KByte memory from $0000.2D00_{16}$… $0000.3D00_{16}$ is searched for the byte constant $3F_{16}$. The byte constant is found at addresses $0000.2E01_{16}$ and $0000.2E05_{16}$.

```
PowerBoot>bs 2D00 3D00 3F B
Search:  00002E01 = 3F
Search:  00002E05 = 3F
PowerBoot>
```

BT – Testing the Memory

BT executes a set of basic binary tests to locate memory errors.

SYNTAX bt begin end [e]

begin specifies the first byte of the memory area to be tested.

end specifies the first byte of the memory area not to be tested.

e achieves that the test runs endlessly.

DESCRIPTION bt executes the following test types:

Test name	**Access size**		
	8-bit	**16-bit**	**32-bit**
Checkerboard test (reverse)	x	x	x
Walking ones test (reverse)	x	x	x
Walking zeros test (reverse)	x	x	x
Increment test	x	x	x
Prime test	–	–	x
March-b test	–	–	x

All tests, except the prime test, fill the memory area specified by *begin* and *end*.. During this action fill appears on the screen. After the memory area has been filled, the testing starts indicated by:

- test forward
- or test backward

test forward means that the memory is tested from *begin* to *end*, whereas test backward indicates that the memory is tested from *end* to *begin*. When the test has been finished, done appears on the screen.

If an error occurs during the test, the following error message will appear:

```
error at location 0x12345678 value found:0xAB should be:0xBA
```

EXAMPLE

In the following example the memory area 0001.0000$_{16}$ … 0002.0000$_{16}$ is tested:

```
PowerBoot> _
PowerBoot> bt 10000 20000
Blocktest at single mode, testing 65536 bytes from 0x10000..0x1FFFF

Checkerboard test 8bit......done
Checkerboard test reverse...done

Walking ones test 8bit......done
Walking ones test reverse...done

Walking zeros test 8bit.....done
Walking zeros test reverse..done

Increment test 8bit.........done

Checkerboard test 16bit.....done
Checkerboard test reverse...done
```

```
Walking ones test 16bit.....done
Walking ones test reverse...done

Walking zeros test 16bit....done
Walking zeros test reverse..done

Increment test 16bit........done

Checkerboard test 32bit.....done
Checkerboard test reverse...done

Walking ones test 32bit.....done
Walking ones test reverse...done

Walking zeros test 32bit....done
Walking zeros test reverse..done

Increment test 32bit........done

Prime test 32bit............done
Prime test 32bit R/T........done

March-b test ...............done

--- Blocktest runs 1 times - 00000000 errors ---

PowerBoot>_
```

BV – Verifying Blocks

BV compares two memory areas on a byte-organized level.

SYNTAX bv begin end destination

begin specifies the starting address of the source memory area, i.e. the first byte of the source memory area which is read.

end specifies the end address of the source memory area, i.e. the first byte of the source memory area which is not read.

destination

specifies the starting address of the destination memory area, i.e. the first byte of the destination memory area which is compared with the source memory.

DESCRIPTION If the compared memory areas are not equal, the different values and the memory location will be displayed. There is no overlap of memory areas allowed.

EXAMPLE In the first example a 4-KByte memory from $0000.2D00_{16}$… $0000.3D00_{16}$ is compared with a destination memory starting at address $0000.5E00_{16}$:

```
PowerBoot>bv 2D00 3D00 5E00
PowerBoot>
```

In the second example an 8-KByte memory from $0000.4C00_{16}$… $0000.6C00_{16}$ is compared with a destination memory starting at address $0000.7C00_{16}$. The two compared memory areas are not identical at the locations $0000.4C10_{16}$ and $0000.7C10_{16}$:

```
PowerBoot>bv 4C00 6C00 7C00
Verify:  00004C10 = EE  00007C10    = 3F
PowerBoot>
```

FERASE – Erasing Flash Memory

FERASE erases a whole flash memory device or a specified area of the flash memory device. The specified area must exactly match the page boundaries of the flash memory device. If a user flash consists, for example, of a 28F008 device (1 Mbit * 8 bit), the minimum size of the erasable area is 64 KByte, i.e. the size of one sector.

There are two tasks, which can be done by FERASE:

- In order to erase a whole flash memory device, select syntax 1.
- In order to erase a specified area of a flash memory device, select syntax 2.

SYNTAX 1 ferase flashDevice

SYNTAX 2 ferase flashDevice flashOffset length

flashDevice

specifies the name of the flash memory device to be erased. If only *flashDevice* is defined, the whole flash memory device will be erased. The following names are currently supported:

BOOT_FLASH	boot flash

USER_FLASH1	first user flash
USER_FLASH2	second user flash

flashOffset

specifies the destination offset within the flash memory device area to be erased. The offset must be sector aligned. Specify the offset at proper boundaries of 256 Kbytes.

length specifies the number of bytes within the flash memory device area to be erased. The length must be sector aligned. The flash device can be erased only in valid block size of 256 Kbytes.

Caution

Ensure that you do not erase Flash memory and re-program it, unless you are certain. Faulty re-programming of Boot Flash will result in a damaged and in-operational board.

EXAMPLE Boot Flash is write-protected through switch setting. When FERASE checks for flash write protection, the following messages appear:

```
PowerBoot> ferase user_flash 0 40000

Are you sure you wish to erase the Flash:[yes = 1,no = 0] : 1
Erasing flash memory ...   done.

/* Common Errors */
PowerBoot> ferase user_flash 0 7000

Are you sure you wish to erase the Flash:[yes = 1,no = 0] : 1
Offset and size parameters are not sector aligned!

PowerBoot> ferase user_flash1 0 3000000
Flash Size is too large

PowerBoot>

PowerBoot> fverify boot_flash 1000000 0 22AC0
Cannot verify for BOOT_FLASH.
```

User flash 1 is erased only after you enter 1 to confirm the action.

FPROG – Programming Flash Memory

FPROG programs flash memory devices. There are three tasks which can be done by means of FPROG:

- In order to program a whole flash memory device, select syntax 1.
- In order to program the space between a specified offset and the end of the flash memory device, select syntax 2.
- In order to program a specified number of bytes starting at a specified destination offset, select syntax 3.

SYNTAX 1 fprog flashDevice source

SYNTAX 2 fprog flashDevice source flashOffset

SYNTAX 3 fprog flashDevice source flashOffset length

flashDevice

specifies the name of the flash memory device to be programmed. The following names are currently supported:

BOOT_FLASH	boot flash
USER_FLASH1	first user flash
USER_FLASH2	second user flash

source specifies the source address containing the data with which the flash memory device is programmed.

flashOffset

specifies the destination offset within the flash memory device.

length specifies the number of bytes to be programmed.

Caution

Ensure that you do not erase Flash memory and re-program it, unless you are certain. Faulty re-programming of Boot Flash will result in a damaged and in-operational board.

EXAMPLE In this example the user wants to reprogram the write-protected boot flash 1 with the data stored at source address 0010.0000_{16}. Since FPROG checks whether the flash is write-protected or not, the following message appears:

```
PowerBoot> fprog user_flash 1000000 0 40000
Programming flash memory

  0 |##################################################| 100%

Done.
```

Reprogramming occurs only upon entering 1 to confirm.

FREAD-Reading Flash Memory

FREAD reads the contents of Flash Memory into RAM.

SYNTAX fread flashdevice destination flashoffset length

flashDevice

specifies the name of the Flash memory device to be verified. The following names are currently supported:
*USER_FLASH1*first user flash
USER_FLASH2 second user flash

Destination

specifies the memory address where the read bytes from flash are to be kept.

flashOffset

specifies the destination offset within the flash memory device.

length specifies the number of byte to be verified.

EXAMPLE In this example the user wants to read the contents of the User Flash 1 memory into RAM

```
PowerBoot> fread user_flash 1000000 0 200
```

Here, fread reads contents of User Flash1
where,
user_flash1: User Flash bank 1
1000000: Memory address where the read bytes of flash are to be kept
0: destination offset within the flash memory device

300000: Length of image in User Flash to be read

FVERIFY- Verifying Flash Memory

FVERIFY compares bytewise the contents of a specified flash bank with the contents of a specified memory location.

SYNTAX 1 fverify flashDevice source flashOffset length

flashDevice

specifies the name of the flash memory device to be verified. The following names are currently supported:

USER_FLASH1	first user flash
USER_FLASH2	second user flash

source specifies the source address containing the data with which the flash memory device is verified.

flashOffset

specifies the destination offset within the flash memory device.

length specifies the number of bytes to be verified.

EXAMPLE In this example the user wants to verify the User Flash 1 with the data stored at source address 0010.0000

```
PowerBoot> fverify user_flash1 1000000 0 1D3A9C
Verifying flash contents

  0 |##################################################| 100%

Done.
```

```
PowerBoot>
```

Here, FVERIFY reads contents of User Flash1, offset 0 and compares it with data at 0x1000000, upto length 0x1D3A9C.
where,

USER_FLASH1	User Flash bank 1
1000000	Start address in memory of image
0	Offset into User Flash
1D3A9C	Length of image in User Flash to be verified

FRCEEPROMWRITE-Writing to EEPROM Device

This command writes to the EEPROM Device.

SYNTAX FRCEEPROMWRITE Device Address Offset Number of Bytes RAM Address

Device Address	specifies the address of EEPROM to be written.
Offset	specifies the offset within the EEPROM in hexadecimal
Number of Bytes	number of bytes to be written
RAM Address	specifies the source address containing data with which the EEPROM device is to be programmed.

EXAMPLE PowerBoot>**FRCEEPROMWRITE a6 20 100 2000000**
where
a6: the device address of EEPROM to be written
20: offset within the EERPOM in hexadecimal
100: Number of bytes to be written in hexadecimal
2000000:The source address containing data that is to be programmed into the EEPROM device.

Note: In the device a8, the memory area is protected from 1ff0-1fff containing the MAC address.

FRCEEPROMREAD-Reading the Contents of EEPROM Devices

Reads the contents of EEPROM devices.

SYNTAX FRCEEPROMREAD Device Addressoffset Number of Bytes RAM Address

Device Address	specifies the address of EEPROM to be read
offset	specifies the offset within the EEPROM device (hexadecimal)
Number of Bytes	Number of bytes to be read (hexadecimal)
RAM Address	Location in memory where the contents are to be placed.

EXAMPLE PowerBoot>**FRCEEPROMREAD a6 10 10 1000000**
where
a6 is the device address of EEPROM to be read
10: Offset within EEPROM device
10: Number of bytes to be read
1000000: Location in RAM memory where the read contents are to be placed

GETMAC-Get the MAC Address

getmac gets the MAC address for the two on-board Ethernet ports.

SYNTAX GETMAC <Eth Interface>

EXAMPLE

```
PowerBoot> getmac 0

Current MAC for Eth0 : 10:20:0C:B9:1C:30

PowerBoot> getmac 1

Current MAC for Eth1 : 40:50:60:70:73:93
```

In this example, the getmac command gets the MAC address for the two on-board Ethernet ports where the addresses are : 00:33:65:67:98:BA and 00:33:65:67:98:BB.

GO – Starting Binary Images without Cyclic Redundancy Check Header

GO executes a binary image located in the DRAM memory. After entering GO you exit from PowerBoot.

SYNTAX go address

address specifies the starting address of the binary image.

EXAMPLE The following example starts a binary image at address 0001.0000_{16}:

```
PowerBoot> go 10000
```

GOLINUX– Loading Linux using RAMDISK

GOLINUX starts the binary image from the two memory locations, namely the Kernel Image, and the Ramdisk Image. After entering GOLINUX, you exit from PowerBoot.

SYNTAX golinux kernel_start_address ramdisk_start_address ramdisk_end_address

kernel_start_address

specifies the starting address of the downloaded kernel image in memory

ramdisk_start_address

specifies the starting address of the ramdisk image in memory

ramdisk_end_address

specifies the end address of the ramdisk image in memory

EXAMPLE The following example starts a kernel binary image at the address 0100.0000_{16} .The RAMDISK start address is 0200.0000_{16} and the RAMDISK end address is 0333.F394.

```
PowerBoot> golinux 1000000 2000000 333F394
```

HELP – Displaying Help Message

HELP displays all commands provided by PowerBoot.

SYNTAX 1 HELP
command used to display the help messages for the entire set of commands.

EXAMPLE

```
PowerBoot> help

 --- Command words ---

       BATCH - Stores/Executes a set of commands
        BAUD - Changes Console Baudrate
          BF - Fills Block of Memory Region
          BM - Copy Block of Memory Region
          BS - Searches a Pattern in a Block of Memory
          BT - Tests the Specified Memory Region
     BUSSHOW - Displays PCI Devices (PCI Scan)
          BV - Verify Block of Memory Region
 CHANGE_BOOT - Changes Bootimage Filename in NVRAM
   CONFIG_RD - Performs a PCI Bus Configuration Read Cycle
   CONFIG_WR - Performs a PCI Bus Configuration Write Cycle
FACTORY_DEFAULTS - Sets Setboot parameters to factory defaults

Page 1 : Enter CR for more Commands


      FERASE - Erases Boot/User Flash Memory
       FPROG - Programs Boot/User Flash Memory
FRCEEPROMREAD - Reads data from EEPROM Device
FRCEEPROMWRITE - Writes data to EEPROM Device
       FREAD - Reads User Flash Memory
     FVERIFY - Verify Boot/User Flash Memory
      GETMAC - Get the MAC Address
          GO - Executes a Binary Images located in the Memory
     GOLINUX - Loading Linux using RAMDISK
        HELP - Displays Help Messages
LINUXCMDLINE - Edits Linux command line
           M - Modifies the Memory Contents

Page 2 : Enter CR for more Commands


          MD - Displays Memory Contents
      MEMMAP - Displays Memory Map of the System
     NETLOAD - Loads a Binary Image into memory through Network
     NETSAVE - Saving Data through Network into File
    PROBEPCI - Displays PCI Devices
       RESET - Does a warm reboot
     SETBOOT - Edits Auto Boot Parameters
```

```
      SETMAC - Set the MAC Address
     VERSION - Displays version Information


  "Help <command>": Displays Syntax and Usage of the individual
commands


PowerBoot>
```

SYNTAX 2 HELP <command-name>
command can be used to display the help for a single command.

EXAMPLE PowerBoot> help bf
Command : BF
Usage : BF - Filling Blocks

BF fills a specified RAM memory area with a byte, a word, or a long constant or with a user defined ASCII string.

SYNTAX: BF <begin> <end> <value> [,B|W|L|P]

begin	is the first byte of the memory area used for filling.
end	is the first byte of the memory area not used for filling.
value	specifies the constant used for filling.

M – Modifying the Memory

M displays and modifies the memory contents of an address.
There are three tasks, which can be done by M:

- In order to display and to modify the memory contents of an address, select syntax 1. Syntax 1 includes the size option L (default).
- In order to specify the memory access type and to display and modify the memory contents of an address, select syntax 3.

SYNTAX 1 M address

SYNTAX 2 M address type

address	is the first byte which is read for displaying and modifying the memory contents.
size	The *type* options O and E override the options B, W, and L. With the exception of access type O, the other access options B, W, L, N, and E check whether the write access has been successful by performing a read access after the write ac-

cess. If the written and the read data do not match, the command is terminated and an error message displayed.

size can be one of the following:

- B = the memory access is byte-sized (8 bit)
- W = the memory access is word-sized (16 bit)
- L = the memory access is long-word-sized (32 bit)

type specifies the memory access type. It can be one of the following:

- N = the memory access is limited to writing, the current contents are not displayed
- E = the memory access is byte-sized and refers only to even addresses
- O = the memory access is byte-sized and refers only to odd addresses

DESCRIPTION M supports a number of commands which can be entered instead of a new memory address. These commands do not change the access option in the command line. The following commands are supported:

Command	New memory address =
cr (key)	current address + 1
=	same as current address
-	current address – 1 x size
–*count*	current address – *count* x size
+	current address + 1 x size
+*count*	current address + *count* x size
#*address*	*address*
.	exit to the PowerBoot command line

EXAMPLE In the following example the memory contents of address 0000.1000_{16} are modified:

```
PowerBoot> m 1000 b
00001000 3f : 4f
00001001 3f : =
00001001 3f : =
00001001 3f : -
```

```
000010003f : #2000
00002000 4f : .
PowerBoot>
```

MD – Displaying Memory Contents

MD displays the memory contents byte-wise and in ASCII code representation.

SYNTAX md address [size]

address is the first byte which is read in order to display the memory contents. The access to the address space is byte-sized.

size specifies the memory access size.

DESCRIPTION The data is displayed in hexadecimal notation and ASCII code. If the data cannot be displayed in ASCII code, a full stop is displayed instead.

MD displays 16 lines, each representing 16 byte. Then the user has the possibility to continue the display or to return to the command line by using the full stop . character.

EXAMPLE In the following example the memory contents starting at FF70.0400$_{16}$ are displayed:

```
PowerBoot> md ff700400
FF700400:  46 FC 27 00 41 FB 81 70  00 01 2F AE 4E 7B 88 01 F.'.A..p../.N...
FF700410:  70 07 4E 7B 00 00 4E 7B  00 01 0E B9 00 00 00 03 p.N...N.........
FF700420:  FF 00 2E 3B 81 70 00 01  D7 82 0E B9 78 00 00 03 ...;.p......x...
FF700430:  FF 00 02 87 FF FF E0 00  28 47 DE B9 FF 71 C8 6C ........(G...q.l
FF700440:  2E 47 26 7B 81 70 00 01  C4 2C 22 2C 10 40 14 2C .G&..p...,",.@.,
FF700450:  10 09 02 02 00 F4 67 12  2E 3B 81 70 00 01 D7 70 ......g..;.p...p
FF700460:  02 87 00 1F FB FF 29 47  10 40 30 6C 10 16 39 7B ......)G.@0l..9.
FF700470:  81 70 00 01 D7 58 10 16  22 6C 10 00 29 7B 81 70 .p...X.."l..)..p
FF700480:  00 01 D7 30 10 00 24 2C  10 50 26 2C 10 54 29 7B ...0..$,.P&,.T).
FF700490:  81 70 00 01 D7 22 10 50  29 7B 81 70 00 01 D7 1C .p...".P)..p....
FF7004A0:  10 54 28 2C 10 60 2A 2C  10 64 2C 2C 10 70 2E 2C .T(,.`*,.d,,.p.,
FF7004B0:  10 74 29 7B 81 70 00 01  D7 0A 10 74 29 7B 81 70 .t)..p.....t)..p
FF7004C0:  00 01 D6 FC 10 70 B7 FC  FF FF FF FF 67 46 48 D3 .....p......gFH.
FF7004D0:  00 FF D7 FC 00 00 00 20  26 EC 10 80 26 EC 10 84 ........&...&...
FF7004E0:  26 EC 10 90 26 EC 10 94  26 EC 10 A0 26 EC 10 A4 &...&...&...&...
FF7004F0:  26 EC 10 B0 26 EC 10 B4  26 EC 10 C0 26 EC 10 C4 &...&...&...&...
More (cr) ? .
PowerBoot>
```

NETLOAD – Loading a Binary Image Through Ethernet

NETLOAD loads a binary image through TFTP (Trivial File Transfer Protocol) from a host acting as server for the specified Ethernet address. For creating a connection to the server, the following two procedures are supported:

- RARP (Reverse Address Resolution Protocol)
- ARP (Address Resolution Protocol).

For these protocols the CPU board is referred to as target.

Note: This release supports only Ethernet port 0.

SYNTAX 1 NETLOAD [-c port] <filename> <address> [targetIP#] [serverIP#]

Port option is used to specify the on-board ehernet port to be used for netload/netsave. The valid values are 0 & 1.

filename	is the absolute file name (including the path name) to access the file to be loaded. The length of the file name is limited to 128 characters. The file has to be a binary image in big endian notation.
address	is the first byte of the memory used for the binary image. The size of the loaded file is the only parameter which determines the number of bytes written to the memory. The binary image must be adapted to the address, because after the loading no relocation will be done. *address* must be long-aligned.
targetIP#	defines the Internet IP address of the CPU board (target) written in the following manner gg.hh.ii.kk.
serverIP#	defines the Internet IP address of the server accessed for downloading the TFTP file.

DESCRIPTION The memory used for the binary image will be made available to the processor if necessary. To receive the image from the server, NETLOAD broadcasts a RARP or an ARP request via Ethernet. If RARP is supported, NETLOAD waits for the Ethernet frame of the server responding at first. If ARP is supported, the specified server will be connected. Then the image is downloaded octet-wise via TFTP.

EXAMPLE In the following example the specified file is downloaded to 0001.0000_{16} on the CPU board with the Ethernet address 00:80:42:0E:88:88 by using the ARP.

NETSAVE- Saving Data Through Network into File

NETSAVE saves a specified memory area via TFTP into a file located in a host system. The file cannot be created, i.e. it must already exist on the host with correct write permissions. NETSAVE overrides the content of the file. For creating a connection to the server, the following 2 procedures are supported:

RARP (Reverse Address Resolution Protocol)
and ARP (Address Resolution Protocol).
For these protocols the CPU board is referred to as target.

SYNTAX 1 NETSAVE [-c port] <filename> <startAddr> <endAddr> [targetIP#] [serverIP#]

Port option is used to specify the on-board ehernet port to be used for netload/net-save. The valid values are 0 & 1.

filename	specifies the name of the file in the host system.
Start address	specifies the starting address of the memory area to be saved into the file on the host.
End address	specifies the end address of the memory area to be saved into the file on the host.
targetIP#	defines the Internet IP address of the CPU board (target) written in the following manner gg:hh:ii:kk.
serverIP#	defines the Internet IP address of the server accessed for uploading the TFTP file.

EXAMPLE In the following example the memory area 0010.010016…0010.011F16 is saved into the file test:

```
PowerBoot> netsave vmlinux 1000000 119ce38 192.168.1.228
192.168.1.18
        Eth0 Link is Up
        GMII/MII : Port works in full-duplex mode
        Port works at 100 Mbps

 Transmitting ARP Request ... Reception of ARP Reply
Filename : vmlinux
Load Address : 0x01000000

 Transmitting TFTP Request to Server 192.168.1.18
Connected to TFTP Server
 - Saved From 0x01000000 to 0x0119CE38 (1691192 bytes)


/* Common Errors */
PowerBoot> netsave
Parameter mismatch
```

```
NETSAVE    [] [Trgt-IP# Server-IP#]
```

SETMAC-Set the MAC Address

setmac sets the MAC address for the two on-board Ethernet ports.

SYNTAX SETMAC <Eth Interface>

EXAMPLE

```
PowerBoot> setmac 1

Current MAC for Eth1 : 04:05:60:70:73:93
New MAC for Eth1     : 40:50:60:70:73:93
MAC Address Updated Sucessfully


/* Errors*/
PowerBoot> getmac 2
ETH2 is not supported
PowerBoot> setmac
SETMAC

PowerBoot> setmac 1

Current MAC for Eth1 : 40:50:60:70:73:93
New MAC for Eth1     :
Retaining previous MAC Address
```

In this example the command sets the MAC addresses for the two on-board Ethernet ports where the MAC addresses are 00:33:65:67:98:BA and 00:33:65:67:98:BB.

Initial Boot Screen

```
Init serial MPSC0 port done

Force PowerPMC-280
Copyright Force Computers, Ltd.,  2003 - 2004

Total Memory detected : 0x40000000
Error Checking and Correction is enabled on onboard SDRAM
Memory detection and configuration complete
Persistent Memory is Disabled
```

```
Executing in Monarch mode
Init DTLB/ITLB for block translation, enable MMU
Instruction Cache is enabled
Init exception vectors starting at address: 0x00000100
Init Shared Memory Serial interface at 0x783000

Found CPU MPC7447 at 1000MHz PVR=80020101.

Onboard Memory DRAM: 1024MB  0x00000000 - 0x3FFFFFFF
GT Register space mapped at 0xF1000000 - 0xF100FFFF

Initializing on-board ethernet
Initializing Eth0 ... done
Eth0 HWaddr 00:C0:8B:06:F8:66
Initializing Eth1 ... done
Eth1 HWaddr 00:C0:8B:06:F9:66

Verifying Boot Parameters.......... done
Testing RAM ....................... done


IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
II     IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII   IIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
II IIII IIIIIIIIIIIIIIIIIIIIIIIIIIIIIII II IIIIIIIIIIIIIIIII IIIIIIIIIIIIIIII
II IIII IIIIIIIIIIIIIIIIII   IIIIIIIIII III IIIIIIIIIIIIII     IIIIIIIIIIII
II     III   IIII III III III II I  III    IIII   IIII   III IIIIIIIIIIII
II IIIIII III II IIIII II    III   IIII III II III II III II IIIIIIIIII
II IIIIII III II II II II IIIIII IIIIII III II III II III II III IIII
II IIIIII    IIII II IIIII   III IIIIII    III    III    III    III
IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
                Copyright Force Computers, Ltd.,  2003 - 2004

                PowerBoot Version:   2.4
                PPMC280   Version:   D/B.0
                Build Date       :   Apr 13 2004
                Build Time       :   10:50:20

                TLA Number       :   120049

Checking Boot Options ...  Key Pressed, Aborting

Shared Memory interface enabled

PowerBoot>
PowerBoot>
```

Version

This command displays the version number and build details of PowerBoot. The screen capture is provided in the following section.

```
PowerBoot> version

IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
II     IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII   IIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
II IIII IIIIIIIIIIIIIIIIIIIIIIIIIIIIIII II IIIIIIIIIIIIIIIII IIIIIIIIIIIIIIII
II IIII IIIIIIIIIIIIIIIIII   IIIIIIIIII III IIIIIIIIIIIIII     IIIIIIIIIIII
II     III   IIII III III III II I  III    IIII   IIII   III IIIIIIIIIIII
II IIIIII III II IIIII II    III   IIII III II III II III II IIIIIIIIII
II IIIIII III II II II II IIIIII IIIIII III II III II III II III IIII
II IIIIII    IIII II IIIII   III IIIIII    III    III    III    III
IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
                Copyright Force Computers, Ltd.,  2003 - 2004

                PowerBoot Version:   2.4
                PPMC280   Version:   D/B.0
                Build Date      :   Apr 13 2004
                Build Time      :   10:50:20

                TLA Number      :   120049


PowerBoot>
```

memmap

This command displays the memory map details. The screen capture is provided below.

```
PowerBoot> memmap
System/PowerBoot Area:          0x00000000 - 0x007FFFFF
PowerBoot Code Area:            0x00200000 - 0x0022841C
Stack and Device Buffers:       0x0022841C - 0x007FFFFF

SDRAM Mapping:
Bank0 : Base 0x00000000 Size 512 MB

GT Register space mapped at    0xF1000000 - 0xF100FFFF
Internal SRAM is mapped  at    0xF2000000 - 0xF203FFFF
User Flash1 mapped at          0xA0000000 - 0xA1FFFFFF
User Flash2 mapped at          0xA2000000 - 0xA3FFFFFF

PCI Memory mapped at           0x80000000 - 0x807FFFFF
PCI I/O mapped at              0x88000000 - 0x8801FFFF

PowerBoot>
```

BATCH - Stores/Executes a set of commands

Stores/Executes a set of upto 10 commands in boot flash.

SYNTAX: - BATCH [-c[reate]|-d[isplay]]

```
To create a set of commands

PowerBoot> BATCH -c
netload vmlinux.bin 1000000 10.208.1.10 10.208.1.1
netload ramdisk.gz 1000000 10.208.1.10 10.208.1.1
golinux 1000000 2000000 335a368
END
```

Note:

- **END denotes finish of command input.**
- **Blank lines are also counted as an command.**

```
Following  executes commands stored in Flash:
PowerBoot> BATCH


Following displays the commands stored in Flash:
PowerBoot> BATCH -d
netload vmlinux.bin 1000000 10.208.1.10 10.208.1.1
netload ramdisk.gz 1000000 10.208.1.10 10.208.1.1
golinux 1000000 2000000 335a368
```

BAUD - Change Baudrate

Changes the serial port baudrate for the session

SYNTAX: `BAUD <baud-rate>`

```
Supported Baudrates are:
600|1200|2400|4800|9600|19200|38400|57600|115200
```

Note: The baud is updated only for the current session and is not updated in Boot parameters. To specify a new baudrate for next boot, use SETBOOT command.

CHANGE_BOOT - changes Bootimage filename

Changes the netload/bootp boot file name. This command also updates the boot parameters stored in Boot-flash.

SYNTAX:

```
CHANGE_BOOT <Boot File name>
```

EXAMPLE:

```
PowerBoot> change_boot
Syntax : CHANGE_BOOT <Boot File Name>

PowerBoot> change_boot PowerBoot.bin
Updating Boot Filename
```

LINUXCMDLINE - edits Linux command line

Prints or sets command line to be passed to Linux kernel, when linux is booted using GOLINUX.

SYNTAX:

```
LINUXCMDLINE
LINUXCMDLINE -s "command-line"
```

EXAMPLE:

```
PowerBoot> linuxcmdline -s "console=ttyS0,115200 root=/dev/ram
mtdparts=0:3072k(kernel),22528k(Ramdisk),-(JFFS2)"

PowerBoot> LINUXCMDLINE
console=ttyS0,115200 root=/dev/ram
mtdparts=0:3072k(kernel),22528k(Ramdisk),-(JF
FS2)
```

Factory Defaults - restores default values

This command is used to restore setboot parameters in NVRAM to factory default values.

SYNTAX

```
FACTORY_DEFAULTS
```

EXAMPLE

```
Powerboot> Factory_defaults
Powerboot>
```

2

PowerBoot Differences for PPMC-280

PowerBoot Differences

The PPMC-280 implementation of the PowerBoot Command-line Interface (CLI) supports the generic set of commands described in the “PowerBoot Instruction Set” chapter, plus an additional set of board-specific commands. The PPMC-280 implementation of board-specific PowerBoot commands is detailed in this chapter.

RESET – Restarting the Module

Restarts the PPMC-280 module.

Note: RESET initiates only a jump to the /HRESET exception vector.

SYNTAX

```
reset
```

DESCRIPTION

RESET restarts the PPMC-280 module. This command forces a hard reset of the processor. The initialization of all the components on the board have to be done again as the execution of this command is similar to a manual power OFF and power ON.

EXAMPLE

```
PowerBoot> reset
Init serial MPSC0 port done

Force PowerPMC-280
Copyright Force Computers, Ltd.,  2003 - 2004

Total Memory detected : 0x40000000
Error Checking and Correction is enabled on onboard SDRAM
Memory detection and configuration complete
Persistent Memory is Disabled

Executing in Monarch mode
Init DTLB/ITLB for block translation, enable MMU
Instruction Cache is enabled
Init exception vectors starting at address: 0x00000100
Init Shared Memory Serial interface at 0x783000

Found CPU MPC7447 at 1000MHz PVR=80020101.

Onboard Memory DRAM: 1024MB  0x00000000 - 0x3FFFFFFF
```

```
GT Register space mapped at 0xF1000000 - 0xF100FFFF

Initializing on-board ethernet
Initializing Eth0 ... done
Eth0 HWaddr 00:C0:8B:06:F8:66
Initializing Eth1 ... done
Eth1 HWaddr 00:C0:8B:06:F9:66

Verifying Boot Parameters.......... done
Testing RAM ....................... done


IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
II     IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII   IIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
II IIII IIIIIIIIIIIIIIIIIIIIIIIIIIIIIII II IIIIIIIIIIIIIIIII IIIIIIIIIIIIIIII
II IIII IIIIIIIIIIIIIIIIII   IIIIIIIIII III IIIIIIIIIIIIII     IIIIIIIIIIII
II     III   IIII III III III II I  III    IIII   IIII   III IIIIIIIIIIII
II IIIIII III II IIIII II    III   IIII III II III II III II IIIIIIIIII
II IIIIII III II II II II IIIIII IIIIII III II III II III II III IIII
II IIIIII    IIII II IIIII   III IIIIII    III    III    III    III
IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
                Copyright Force Computers, Ltd.,  2003 - 2004

                PowerBoot Version:   2.4
                PPMC280   Version:   D/B.0
                Build Date       :   Apr 13 2004
                Build Time       :   10:50:20

                TLA Number       :   120049

Checking Boot Options ...  Key Pressed, Aborting

Shared Memory interface enabled

PowerBoot>
PowerBoot>
```

SETBOOT – Editing Auto Boot Parameters

Prompts the user to enter parameter values for automatic booting. The defined parameters become valid after the next power-on or when RESET is entered. The parameters are stored near the top of boot flash memory. SETBOOT calculates a checksum to protect the boot flash contents containing the edited auto boot parameters.

SYNTAX setboot

DESCRIPTION

SETBOOT prompts the user to assign a value to each auto boot parameter. Where appropriate, the prompt provides the possible values of each parameter and its current setting.

The parameters provide the following booting information:

- Define the location of the automatically loaded or pre-programmed binary image (Boot select).
- Determine the type of booting and the location at which the binary image is to start (Auto boot, Boot address, Load address).
- Select a delay for auto booting after power-on: (Auto boot delay).
- Define if Cyclical Redundancy Checking Version is to be enabled or disabled. (Enable CRC check).
- Select the type of file system to be loaded into Flash (Load filesystem)
- Define the Linux kernel start address in Flash (start address of LINUX KERNEL in Flash).
- Define the RAMDISK start address in Flash (start address of RAMDISK in Flash).
- Define the name and the path of the file loaded during auto boot (TFTP Boot file name).
- Select the Internet protocol for connecting a server to the board (RARP or ARP protocol).
- Select the protocol numbers for selecting the Trivial File Transfer Protocol (TFTP) file server and for identifying the board (Server-IP#, Target-IP#).
- Define whether or not the power on test (POT) should be executed after the next power-on or reset (Power On Test POT).

PROMPTS

Boot select [0=GO, 1=Net, 2=BOOTP, 3=FLASH, 4=BATCH] (1) :

Defines the location of the automatically-loaded or pre-programmed binary image:

- 0 = Autoload a binary image to the Load address via Ethernet (TFTP and front-panel connector). You must link the download application image to the Load address. The Boot address parameter is used when starting the downloaded image.
- 1 = Boots the device using RARP protocol

- 2 = Boots the device using BOOTP to load the OS image
- 3 = Copy the user flash memory contents to the Load address and start executing at the Boot address. The complete flash bank containing the image is always copied regardless of the type and size of the devices present and the amount of already programmed or used user flash storage capacity.
- 4 = Executes the commands stored using BATCH command when autoboot is enabled.

Auto boot [0=disable, 1=enable], (0):

Enables or disables auto booting:

- 0 = Disable auto booting. All auto boot parameters are ignored. The PowerBoot debugger is invoked.
- 1 = Enable auto booting. Auto boot will take place after the next power-on or RESET.

Enable CRC check [Disable=0 | Enable=1], (0) :

Enables selecting a CRC version enabled, or a Non-CRC version enabled

- 0= Enables a Non- CRC version
- 1= Enables a CRC version

Load filesystem = [NFS=0 | RAMDISK=1], :

Allows selecting the type of file system, to be loaded into the Flash.

- 0= Loads NFS into Flash
- 1= Loads RAMDISK into Flash

Enter start address of LINUX KERNEL in Flash (A0000000):

Enter the start address of the Linux Kernel in Flash.

Enter start address of RAMDISK in Flash (A3000000):

Enter the start address of the RAMDISK image in Flash.

Auto boot delay [0..99s], (55):

Specifies to delay the auto booting for a preset period of time, ranging from 0 to 99 seconds. During the auto boot delay, you can stop the auto booting by pressing any user key on the serial console line.

Load address (00000000):

Specifies the location in the CPU addressing space at which the opcode is downloaded. This parameter does not depend on other auto boot parameters. The binary image is always downloaded to the Load address if you specify Boot Select parameters 0 or 1.

Boot address (00010000):

Specifies the location in the CPU addressing space at which the opcode is started. This parameter does not depend on other auto boot parameters. The binary image always starts at Boot address, regardless of whether it is downloaded during power-on to Load address, stored in the user-programmable region of boot flash, or stored in the user flash.

On Board Ethernet Port [0/1] (00000000):

Specify the on-board Ethernet port.(Port 0, Port 1 or Port 2).

- 0= Specify the on-board Ethernet port as Port 0
- 1= Specify the on-board Ethernet port as Port 1

RARP [1] or ARP [2] protocol, (1):

Selects the Internet protocol used for connecting a server to the board.

- 1 = RARP (Reverse Address Resolution Protocol). If you select RARP, the parameters Server-IP# and Target-IP# are ignored.
- 2 = ARP (Address Resolution Protocol). If you select ARP, the values entered for Server-IP# and Target-IP# are used.

If you enter a value other than 1 or 2, the default setting, 1, is assigned to the parameter.

Server-IP# [aaa.bbb.ccc.ddd]:

Specifies an Internet protocol number that selects a TFTP file server, for example, *123.3.255.255*. If the Internet protocol RARP is selected, this parameter is ignored. The value is stored as a string.

Target-IP# [aaa.bbb.ccc.ddd]:

Specifies an Internet protocol number that identifies the board to the Internet layer, for example, *3.255.37.67*. If the Internet protocol RARP is selected, this parameter is ignored. The value is stored as a string.

TFTP Boot file name:

Defines the name and path of a boot file to be loaded during auto booting (if Boot select = 0). The file name and path must not exceed 128 characters. You must fully specify the name and path and must observe correct upper and lower casing.

If you use TFTP for booting (Boot select = 0), the host must be set up as the TFTP server. The host must provide the desired file via Ethernet (TFTP and front panel connector).

Boot Image Type [Linux = 1 | Others = 0]: (0) :

This option is used along with BOOTP to identify the image type. This enables BOOTP to pass Linux command line parameters if image type is Linux.
Image type is set to 1 if the loaded image is Linux else, the image type is set to 0.

Select Serial Baud (600|1200|2400|4800|9600|19200|38400|57600|115200)[115200]:

Specifies the baudrate to be used on next boot. Only one of the above values can be specified.

Current Linux CMDLINE ("console=ttyS0,115200 root=/dev/ram mtdparts=0:3072k(kernel),22528k(Ramdisk),-(JFFS2)") New Cmd-line :

Specifies the commandline to be passed to linux after next reboot of PowerBoot.

Ramdisk file name : ramdisk.gz:

Specifies a ramdisk file to be loaded when BOOT NET option is selected. If this file is specified, the BOOT file is booted using GOLINUX cmd during autoboot and this ramdisk is passed to the command.

If the filename is empty, the beahaviour of BOOT NET remains same and it uses GO command.

Ramdisk Load address (02000000) :

Specifies the location where ramdisk image has to be downloaded.

Power On Test POT [1=disable, 0=enable], (0):

Disables or enables the Power On Test (POT) that is executed at boot time:

- 1 = Disable the Power On Test.
- 0 = Enable the Power On Test (default).

EXAMPLE

For a RAMDISK file system

```
PowerBoot> setboot
This will modify Boot Flash Contents.Do you want to
continue(Y/[N])? y

 -General Boot Parameters-

Boot select [0=GO, 1=Net, 2=BOOTP, 3=FLASH, 4=BATCH] (1) :
Select Serial Baud
(600|1200|2400|4800|9600|19200|38400|57600|115200)[115200]:
Auto boot   [0=disable, 1=enable], (1)   :
Auto boot delay [0..99s], (5) :
Load address (01000000) :
Boot address (01000000) :

 -TFTP Ethernet/Harddisk boot file parameters-

RARP [1] or ARP [2] protocol : (2) :
Server-IP# [aaa.bbb.ccc.ddd] : 192.168.1.20 :
Target-IP# [aaa.bbb.ccc.ddd] : 192.168.2.171 :
TFTP/Disk Boot file name :
Boot Image Type [Linux = 1 | Others = 0]: (0) :
vmlinux-ramdisk.bin :

TFTP Ramdisk file name :
ramdisk2.gz :
Ramdisk Load address (02000000) :
Current Linux CMDLINE ()
 New Cmd-line :
Shared Memory Interface [0=disable, 1=enable], (0)   :

 -Power On Test (POT) parameters-

Power ON Test POT [1=disable, 0=enable], (0)   :
CSUM : 0x11DE
PowerBoot>
```

Note:

- **All other parameters are for internal use only. Do not change their default settings.**
- **Shared memory interface is disabled by default; it may be enabled using `setboot` command.**

Persistent Memory

Provides a mechanism to retain the contents of memory over a specified range, even after reset, using either software or hardware.

Enabling Persistent memory feature

MAGIC VALUE	Magic value is used enable this feature. Magic value used is 0x26A1BD0F. M command is used to set this value at location 0x1F0 in main memory.
START ADDRESS	Here 8MB aligned start address should be specified at location 0x1E0 in main memory using M command.
END ADDRESS	Here 8MB aligned end address should be specified at location 0x1E8 in main memory using M command.

EXAMPLE:

```
PowerBoot>
PowerBoot> m 1f0
000001F0  00000000 : 26a1bd0f
000001F4  00000000 : .
PowerBoot> m 1e0
000001E0  00000000 : 1000000
000001E4  00000000 :
000001E8  00000000 : 2000000
000001EC  00000000 : .
PowerBoot>
```

After Hard Reset

```
Init serial MPSC0 port done

Force PowerPMC-280
Copyright Force Computers, Ltd.,  2003 - 2004

Total Memory detected : 0x40000000
Error Checking and Correction is enabled on onboard SDRAM
```

```
Memory detection and configuration complete
Persistent Memory is Enabled [0x01000000 - 0x02000000]

Executing in Monarch mode
Init DTLB/ITLB for block translation, enable MMU
Instruction Cache is enabled
Init exception vectors starting at address: 0x00000100
Init Shared Memory Serial interface at 0x783000

Found CPU MPC7447 at 1000MHz PVR=80020101.

Onboard Memory DRAM: 1024MB  0x00000000 - 0x3FFFFFFF
GT Register space mapped at 0xF1000000 - 0xF100FFFF

Initializing on-board ethernet
Initializing Eth0 ... done
Eth0 HWaddr 00:C0:8B:06:F8:66
Initializing Eth1 ... done
Eth1 HWaddr 00:C0:8B:06:F9:66

Verifying Boot Parameters.......... done
Testing RAM ....................... done

IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
II     IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII   IIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
II IIII IIIIIIIIIIIIIIIIIIIIIIIIIIIIIII II IIIIIIIIIIIIIIIII IIIIIIIIIIIIIIII
II IIII IIIIIIIIIIIIIIIIII   IIIIIIIIII III IIIIIIIIIIIIII     IIIIIIIIIIII
II     III   IIII III III III II I  III    IIII   IIII   III IIIIIIIIIIII
II IIIIII III II IIIII II    III   IIII III II III II III II IIIIIIIIII
II IIIIII III II II II II IIIIII IIIIII III II III II III II III IIII
II IIIIII    IIII II IIIII   III IIIIII    III    III    III    III
IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
           Copyright Force Computers, Ltd.,  2003 - 2004

           PowerBoot Version:   2.4
           PPMC280   Version:   D/B.0
           Build Date       :   Apr 13 2004
           Build Time       :   10:50:20

           TLA Number       :   120049

Checking Boot Options ... Autoboot disabled.

Shared Memory interface enabled

PowerBoot>
```

Command for Accessing the PCI Bus

The PPMC-280 implementation of PowerBoot provides a command for accessing the PCI bus: BUSSHOW.

Caution

This command is valid only if the PPMC-280 is running in monarch mode and the baseboard is a CompactPCI slot 1 board. If the module is running in non-monarch/PCI-agent mode, these commands may lock up the system.

BUSSHOW - Displaying PCI Devices

Scans all PCI bus segments and, for each PCI agent found, prints the vendor/device ID, bus number, device number, and the device function number.

SYNTAX

```
busshow
```

EXAMPLE

```
PowerBoot> busshow
Scaning PCI 0 interface
Found PCI device: Device/VendorID: 0x646011AB at bus 0, device 0, function 0
Found PCI device: Device/VendorID: 0x646011AB at bus 0, device 0, function 1
Found PCI device: Device/VendorID: 0x646011AB at bus 0, device 0, function 2
Found PCI device: Device/VendorID: 0x646011AB at bus 0, device 0, function 3
Found PCI device: Device/VendorID: 0x646011AB at bus 0, device 0, function 4
Found PCI device: Device/VendorID: 0x646011AB at bus 0, device 0, function 5
Found PCI device: Device/VendorID: 0x646011AB at bus 0, device 0, function 6
Found PCI device: Device/VendorID: 0x646011AB at bus 0, device 0, function 7
Found PCI device: Device/VendorID: 0xB1548086 at bus 0, device 8, function 0
Found a Bridge
PowerBoot>
```

CONFIG_RD - Performing a PCI Bus Configuration Read Cycle

Performs a PCI bus configuration read cycle on a specified address. You specify a PCI bus device, the address/register offset of the location to be read, and the size of the data (8, 16, or 32 bits).

SYNTAX

```
config_rd <bus-nr><dev-nr><fun-nr><reg-nr> [B|W|L]
```

Table 1:*Parameter Table*

Parameter	Description
bus-nr	Specifies a PCI bus segment number.
dev-nr	Specifies a PCI bus device number.
fun-nr	Specifies a PCI bus device function number.
reg-nr	Specifies the address/register offset of the location to be read.

[B/W/L]	Specifies the size of the data to be read:
	B = byte (8 bits)
	W = word (16 bits)
	L = longword (32 bits)

EXAMPLE

The following command performs a PCI bus configuration read cycle for 32 bits of data:

```
PowerBoot> config_rd 0 18 0 4 1
address: 0x8000C004
0xFFFFFFFF
PowerBoot>
```

CONFIG_WR - Performing a PCI Bus Configuration Write Cycle

Performs a PCI bus configuration write cycle to a specified address. You specify a PCI bus device, the address/register offset of the location to be written, the data to be written, and the size of the data (8, 16, or 32 bits).

SYNTAX

```
config_wr <bus-nr><dev-nr><fun-nr><reg-nr><data> [B|W|L]
```

Table 2:*Parameter Table*

Parameter	Description
bus-nr	Specifies a PCI bus segment number.
dev-nr	Specifies a PCI bus device number.
fun-nr	Specifies a PCI bus device function number.

Table 2:*Parameter Table*

Parameter	Description
reg-nr	Specifies the address/register offset of the location to be read.

[B/W/L]	Specifies the size of the data to be read:
	B = byte (8 bits)
	W = word (16 bits)
	L = longword (32 bits)

EXAMPLE

```
The following command performs a PCI bus configuration write
cycle for 8 bits of data:
PowerBoot> config_wr 0 18 0 19 1 b
address: 0x8000C019
```

Commands for Programming and Verifying Flash Memory

PowerBoot provides the FERASE, FPROG and FVERIFY commands for programming and verifying flash memory devices. These commands are described in detail in the the “PowerBoot Instruction Set” chapter. This section describes the use of these commands in connection with the PPMC-280 onboard flash memory.

Caution

Ensure that you do not erase Flash memory and re-program it, unless you are certain. Faulty re-programming will result in a damaged and in-operational board.

Reprogramming Boot Flash

The PPMC-280 Boot Flash device (**boot_flash**) provides a total of 1 MB of flash memory with PowerBoot firmware pre-installed. 1 MB of Boot Flash is mirrored eight times in the 8 MB Boot Flash region.

For example, you can use PowerBoot commands to place an operating system loader image, such as a VxWorks boot ROM image, into the user-programmable 8 MB region of boot flash.

Note: When you reprogram the PPMC-280 Boot Flash device using FPROG, an FERASE is performed automatically to erase the target region. Other platforms may require you to manually issue an FERASE before the FPROG.

Table 3:*Boot Flash Address Range*

Component	Starting Address	Ending Address	Size
BOOT_FLASH	FF80 0000	FFFF FFFF	8 MB
NVRAM AREA	FFF4 0000	FFF4 FFFF	64 KB

You can program the Boot Flash device as follows:

1. Start PowerBoot by powering on the PPMC-280 module.

```
PowerBoot> _
```

2. In order to reprogram the Boot Flash device, you need to load the new program image into DRAM memory. For example, you can use the PowerBoot NETLOAD command. (The NETLOAD command loads a binary image via TFTP from a host acting as a server; see the the “PowerBoot Instruction Set” chapter for more information.)

```
PowerBoot> netload power.bin 1000000 10.208.1.208 10.208.1.158
 PHY-Device at 100MB/s negotiated
 WANCOM MAC ADDRESS : FE:FF:FF:00:43:00
 Transmitting ARP-REQUEST... Reception of ARP-REPLY
 Transmitting TFTP-REQUEST to server 00:D0:09:F9:FF:5B, IP 10.208.1.158
 PACKET:1 PACKET:50 PACKET:100 PACKET:150 PACKET:200 PACKET:250
PACKET:300 PACKET:350 PACKET:400 PACKET:414 - loaded
$01000000..$01033A5A (211547 bytes)
PowerBoot> _
```

3. Verify the DRAM memory contents by using the PowerBoot MD command:

```
PowerBoot> md 1000000
01000000:  46 4f 52 43 45 20 46 4f  52 43 45 20 46 4f 52 43
01000010:  45 20 46 4f 52 43 45 20  46 4f 52 43 45 20 46 4f
01000020:  52 43 45 20 46 4f 52 43  45 20 46 4f 52 43 45 20
01000030:  46 4f 52 43 45 20 46 4f  52 43 45 20 46 4f 52 43
01000040:  45 20 46 4f 52 43 45 20  46 4f 52 43 45 20 46 4f
01000050:  52 43 45 20 46 4f 52 43  45 20 46 4f 52 43 45 20
01000060:  46 4f 52 43 45 20 46 4f  52 43 45 20 46 4f 52 43
01000070:  45 20 46 4f 52 43 45 20  46 4f 52 43 45 20 46 4f
01000080:  52 43 45 20 46 4f 52 43  45 20 46 4f 52 43 45 20
01000090:  46 4f 52 43 45 20 46 4f  52 43 45 20 46 4f 52 43
010000a0:  45 20 46 4f 52 43 45 20  46 4f 52 43 45 20 46 4f
010000b0:  52 43 45 20 46 4f 52 43  45 20 46 4f 52 43 45 20
010000c0:  46 4f 52 43 45 20 46 4f  52 43 45 20 46 4f 52 43
010000d0:  45 20 46 4f 52 43 45 20  46 4f 52 43 45 20 46 4f
010000e0:  52 43 45 20 46 4f 52 43  45 20 46 4f 52 43 45 20
010000f0:  46 4f 52 43 45 20 46 4f  52 43 45 20 46 4f 52 43
More (cr) ? .

PowerBoot>
```

4. Program the image residing in DRAM memory into the Boot Flash device by using the PowerBoot FPROG command. You specify the Boot Flash device, the source DRAM address, and the destination Boot Flash offset.

Note: The Boot Flash offset you specify to the FPROG command is relative to the base address of the boot flash, FF800000.

For example, the following command programs the DRAM memory contents beginning at 10.0000_{16} into the boot flash beginning at offset 0000.0000_{16}, which corresponds to address FF800000:

```
PowerBoot> fprog boot_flash 1000000 0 33a5b
Warning, error during programming could render board un-operational!
Confirm Boot Flash programming [0=no, 1=yes] (0) : 1
Programming Powerboot flash memory
Will need to power cycle after completion
PowerBoot>
```

5. To view the contents of the programmed Boot Flash device, use the PowerBoot MD command:

```
PowerBoot> md ff800000
ff800000:  46 4f 52 43 45 20 46 4f  52 43 45 20 46 4f 52 43
ff800010:  45 20 46 4f 52 43 45 20  46 4f 52 43 45 20 46 4f
ff800020:  52 43 45 20 46 4f 52 43  45 20 46 4f 52 43 45 20
ff800030:  46 4f 52 43 45 20 46 4f  52 43 45 20 46 4f 52 43
ff800040:  45 20 46 4f 52 43 45 20  46 4f 52 43 45 20 46 4f
ff800050:  52 43 45 20 46 4f 52 43  45 20 46 4f 52 43 45 20
ff800060:  46 4f 52 43 45 20 46 4f  52 43 45 20 46 4f 52 43
ff800070:  45 20 46 4f 52 43 45 20  46 4f 52 43 45 20 46 4f
ff800080:  52 43 45 20 46 4f 52 43  45 20 46 4f 52 43 45 20
ff800090:  46 4f 52 43 45 20 46 4f  52 43 45 20 46 4f 52 43
ff8000a0:  45 20 46 4f 52 43 45 20  46 4f 52 43 45 20 46 4f
ff8000b0:  52 43 45 20 46 4f 52 43  45 20 46 4f 52 43 45 20
ff8000c0:  46 4f 52 43 45 20 46 4f  52 43 45 20 46 4f 52 43
ff8000d0:  45 20 46 4f 52 43 45 20  46 4f 52 43 45 20 46 4f
ff8000e0:  52 43 45 20 46 4f 52 43  45 20 46 4f 52 43 45 20
ff8000f0:  46 4f 52 43 45 20 46 4f  52 43 45 20 46 4f 52 43
More (cr) ?

PowerBoot>
```

6. To execute the programmed image, you must reset the board.

Programming User Flash

You can program the user flash device as follows:

1. Start PowerBoot by powering on the PPMC-280 module.

```
PowerBoot> _
```

2. Erase the user flash using the FERASE command. For example, you can use the following command to erase user flash:

```
PowerBoot> ferase user_flash1

Are you sure you wish to erase the Flash:[yes = 1,no = 0] : 1
```

User Flash1 is erased only when you enter 1 to confirm.

Note:

- **The user flash device must be erased prior to reprogramming.**
- **The FERASE command shown erases the entire 32 MB or 64 MB memory of the user flash device. Erasing the entire user flash is a lengthy operation. If you only want to erase smaller parts of the user flash device memory, you need to use offset and length parameters. For a list of these parameters, see the the “Power-Boot Instruction Set” chapter.**

---

3. In order to reprogram the user flash device you have just erased, you need to load the new program image into DRAM memory. For example, you can use the PowerBoot NETLOAD command. (The NETLOAD command loads a binary image via TFTP from a host acting as a server; see the the “PowerBoot Instruction Set” chapter for more information.)

```
PowerBoot> netload linux_exp 100000 10.208.1.208 10.208.1.158
Init Ethernet Controller MII-Port and PHY-Device
PHY-Device at 10MB/s negotiated
LAN-controller at address FE870000 set to Ethernet 00:80:42:0E:00:69
Transmitting RARP-REQUEST... Reception of RARP-REPLY
Transmitting TFTP-REQUEST to server 02:80:42:0A:0D:79, IP 192.168.41.1
PACKET:307 - loaded $00100000..$001265CF (157136 bytes)

PowerBoot> _
```

If you want to load the word “FORCE” into user flash, you can load it into DRAM memory by using the PowerBoot BF command:

```
PowerBoot> bf 100000 500000 “FORCE” p
```

4. Verify the DRAM memory contents by using the PowerBoot MD command:

```
PowerBoot> md 100000
00100000:  46 4f 52 43 45 20 46 4f  52 43 45 20 46 4f 52 43
00100010:  45 20 46 4f 52 43 45 20  46 4f 52 43 45 20 46 4f
00100020:  52 43 45 20 46 4f 52 43  45 20 46 4f 52 43 45 20
00100030:  46 4f 52 43 45 20 46 4f  52 43 45 20 46 4f 52 43
00100040:  45 20 46 4f 52 43 45 20  46 4f 52 43 45 20 46 4f
00100050:  52 43 45 20 46 4f 52 43  45 20 46 4f 52 43 45 20
00100060:  46 4f 52 43 45 20 46 4f  52 43 45 20 46 4f 52 43
00100070:  45 20 46 4f 52 43 45 20  46 4f 52 43 45 20 46 4f
```

```
00100080:  52 43 45 20 46 4f 52 43  45 20 46 4f 52 43 45 20
00100090:  46 4f 52 43 45 20 46 4f  52 43 45 20 46 4f 52 43
001000a0:  45 20 46 4f 52 43 45 20  46 4f 52 43 45 20 46 4f
001000b0:  52 43 45 20 46 4f 52 43  45 20 46 4f 52 43 45 20
001000c0:  46 4f 52 43 45 20 46 4f  52 43 45 20 46 4f 52 43
001000d0:  45 20 46 4f 52 43 45 20  46 4f 52 43 45 20 46 4f
001000e0:  52 43 45 20 46 4f 52 43  45 20 46 4f 52 43 45 20
001000f0:  46 4f 52 43 45 20 46 4f  52 43 45 20 46 4f 52 43
More (cr) ? .

PowerBoot>
```

5. Program the image residing in DRAM memory into the user flash device by using the PowerBoot FPROG command. You specify the user_flash1 device, the source DRAM address, and the destination user flash offset.

Note: The user flash offset you specify to the FPROG command is relative to the base address of the user flash, A0000000 for User_Flash1 and A2000000 for User_Flash2.

For example, the following command programs the DRAM memory contents beginning at 10.0000_{16}, for a length of 32 MB or 64 MB (based on the PPMC-280 user flash size), into the user flash beginning at offset 0000.0000_{16}, which corresponds to address A0000000**:**

```
PowerBoot> fprog user_flash1 100000 0
Programming flash memory
0 ... 100%
Done.

PowerBoot>
```

6. To view the contents of the programmed user flash device, use the PowerBoot MD command:

```
PowerBoot> md A0000000
50000000:  46 4f 52 43 45 20 46 4f  52 43 45 20 46 4f 52 43
50000010:  45 20 46 4f 52 43 45 20  46 4f 52 43 45 20 46 4f
50000020:  52 43 45 20 46 4f 52 43  45 20 46 4f 52 43 45 20
50000030:  46 4f 52 43 45 20 46 4f  52 43 45 20 46 4f 52 43
50000040:  45 20 46 4f 52 43 45 20  46 4f 52 43 45 20 46 4f
50000050:  52 43 45 20 46 4f 52 43  45 20 46 4f 52 43 45 20
50000060:  46 4f 52 43 45 20 46 4f  52 43 45 20 46 4f 52 43
50000070:  45 20 46 4f 52 43 45 20  46 4f 52 43 45 20 46 4f
50000080:  52 43 45 20 46 4f 52 43  45 20 46 4f 52 43 45 20
50000090:  46 4f 52 43 45 20 46 4f  52 43 45 20 46 4f 52 43
500000a0:  45 20 46 4f 52 43 45 20  46 4f 52 43 45 20 46 4f
500000b0:  52 43 45 20 46 4f 52 43  45 20 46 4f 52 43 45 20
```

```
500000c0:  46 4f 52 43 45 20 46 4f  52 43 45 20 46 4f 52 43
500000d0:  45 20 46 4f 52 43 45 20  46 4f 52 43 45 20 46 4f
500000e0:  52 43 45 20 46 4f 52 43  45 20 46 4f 52 43 45 20
500000f0:  46 4f 52 43 45 20 46 4f  52 43 45 20 46 4f 52 43
More (cr) ?
PowerBoot>
```

3

Getting Started with PowerBoot

Getting Started

The PowerBoot command line interface (CLI) provides an interface to the firmware for a system. From a serial console you can use commands to load, test, and debug software in system memory.
The PPMC-280 implementation of the PowerBoot CLI supports:

- Generic PowerBoot commands (BF, BM, BS, BT, BV, FERASE, FVERIFY, FPROG, FRCEEPROMREAD, FRCEEPROMWRITE, GO, GOLINUX, HELP, FREAD, M, MD, NETLOAD, and NETSAVE)
- Board-specific commands described in Chapter 2 of this manual (BUSSHOW, RESET, SETBOOT, SETMAC, GETMAC)

This chapter details how to get started with PowerBoot.

Setting Up the System for PowerBoot

To use the PowerBoot CLI, you need to connect the system to a console device. The console device can be a video terminal connected with a serial line port or a PC or workstation connected to the system through the serial port.

For information on installing serial-line or network cables, see the appropriate hardware installation guide.

Once you have connected the system to a console device, set up the device to use the following parameters:

- Send/receive 115200 baud
- Eight (8) bit data word
- No parity
- One (1) stop bit
- No flow control

PowerBoot Features

PowerBoot firmware is stored in your module's onboard boot flash memory and features the following:

- Power-on tests (POTs) that can be disabled
- Simple CLI and command set
- Utilities for configuring boot parameters and system components, such as PCI devices
- Set of basic binary tests for testing main memory
- Networking capabilities for loading binary images into memory
- Supports scanning of the PCI bus

PowerBoot Ethernet Support

PowerBoot supports two on-board MGI Ethernet ports for download of:

- Linux Kernel/Ramdisk
- VxWorks image
- Firmware for upgrade purposes.

Entering PowerBoot

The PowerBoot CLI comes up automatically when the power-on tests (POTs) are completed during system startup. Upon entering the CLI, the system displays the following prompt:

```
PowerBoot>
```

The system also enters PowerBoot when:

- You generate a soft reset with the PowerBoot RESET command.

Exiting PowerBoot

The PowerBoot CLI exits when:

- You issue the GO/GOLINUX command to start a binary image that has been loaded into main memory
- Auto booting is enabled and the system automatically jumps to the user-specified boot address

Getting Online Help

To display online help, enter the HELP command. This command displays each PowerBoot command with a brief description, as shown below:

```
PowerBoot> help

 --- Command words ---

       BATCH - Stores/Executes a set of commands
        BAUD - Changes Console Baudrate
          BF - Fills Block of Memory Region
          BM - Copy Block of Memory Region
          BS - Searches a Pattern in a Block of Memory
          BT - Tests the Specified Memory Region
     BUSSHOW - Displays PCI Devices (PCI Scan)
          BV - Verify Block of Memory Region
 CHANGE_BOOT - Changes Bootimage Filename in NVRAM
   CONFIG_RD - Performs a PCI Bus Configuration Read Cycle
   CONFIG_WR - Performs a PCI Bus Configuration Write Cycle
FACTORY_DEFAULTS - Sets Setboot parameters to factory defaults

Page 1 : Enter CR for more Commands


      FERASE - Erases Boot/User Flash Memory
       FPROG - Programs Boot/User Flash Memory
FRCEEPROMREAD - Reads data from EEPROM Device
FRCEEPROMWRITE - Writes data to EEPROM Device
       FREAD - Reads User Flash Memory
     FVERIFY - Verify Boot/User Flash Memory
      GETMAC - Get the MAC Address
          GO - Executes a Binary Images located in the Memory
     GOLINUX - Loading Linux using RAMDISK
        HELP - Displays Help Messages
LINUXCMDLINE - Edits Linux command line
           M - Modifies the Memory Contents

Page 2 : Enter CR for more Commands


          MD - Displays Memory Contents
      MEMMAP - Displays Memory Map of the System
     NETLOAD - Loads a Binary Image into memory through Network
     NETSAVE - Saving Data through Network into File
    PROBEPCI - Displays PCI Devices
       RESET - Does a warm reboot
     SETBOOT - Edits Auto Boot Parameters
      SETMAC - Set the MAC Address
     VERSION - Displays version Information


  "Help <command>": Displays Syntax and Usage of the individual
commands


PowerBoot>
```

Command Overview

The PowerBoot CLI consists of commands for managing the operation of your system, running diagnostics, and verifying the integrity of your system design. Refer to the “PowerBoot Instruction Set” chapter and the the “PowerBoot Differences for PPMC-280” chapter of this manual for generic and platform-specific commands in detail. Table 4 lists the types of operations you can perform by using the commands.

Table 4:*Summary of PowerBoot Operations*

Operation	**Command**
Displaying Online Help	
Display online help	HELP
Configuring Boot Parameters	
Configure boot parameters	SETBOOT
Monitoring a PCI Bus	
Scan a PCI bus segment and display information about devices found	BUSSHOW
Initiating a Soft Reset	
Initiate a soft reset	RESET
Managing the Contents of Main Memory	
Displaying the contents of an area of memory	MD
Fill an area of memory with constant values	BF
Copy the contents of an area of memory to another area of memory	BM
Compare the contents of two memory locations	BV
Search an area of memory for constant values	BS
Modify the contents of memory	M
Load a binary image into memory over the network	NETLOAD
Test memory	BT
Managing Flash Memory Devices	
Erase the contents of a flash memory device or an area of a flash memory device	FERASE
Program a flash memory device	FPROG

Table 4:*Summary of PowerBoot Operations (cont.)*

Operation	Command
Compares bytewise contents of specified flash bank with the contents of a specified memory location	FVERIFY
Reads contents of specified flash device into RAM	FREAD
Running Diagnostic Tests	
Test memory	BT
Performing Network Operations	
Load a binary image into memory over the network	NETLOAD
Starting Binary Images	
Start a binary image that resides in main memory	GO
Start the binary image from the two memory locations (Kernel image and the Ramdisk Image)	GOLINUX
Configuring Onboard Ethernet Port's MAC Address	
Sets the MAC address for the two onboard Ethernet ports	SETMAC
Dumps the MAC addresses for the two onboard Ethernet ports.	GETMAC

Most commands require that you specify arguments. In some cases, you can also specify options that give you a finer level of control over the command's execution. Options appear in syntax as keywords separated with a vertical bar (|) character. When specifying options, you must separate the option from the command and arguments with a comma. For example, you must specify **M *address*,B**. If you enter **M *address* B**, an error message appears.

Command Line Requirements and Restrictions

Table 5 lists PowerBoot command line requirements and restrictions.

Table 5:*PowerBoot Command Line Requirements and Restrictions*

Requirement or Restriction	Explanation
Length	A command line consists of a fixed number of characters, not including the terminating carriage return or any characters that you delete as you enter the command line. The maximum command line length differs among the various commands.
Case	A command line can consist of upper- or lowercase characters. Characters display in the case in which you enter them.
Options	You can adjust the behavior of some commands by specifying them with options. See individual command descriptions for examples.
Numbers	Numbers specified as command parameters are interpreted as hexadecimal values.
ASCII strings	Specify ASCII strings within double quotes (" ") followed by the option P.
File names	File names must include absolute pathnames and cannot exceed 128 characters.
Ethernet addresses	Specify Ethernet addresses with the notation *xx*:*xx*:*xx*:*xx*:*xx*:*xx*.
Internet addresses	Specify Internet Protocol (IP) addresses with the standard Internet 4-decimal position "." notation — *nnn.nnn.nnn.nnn*.
No characters	A command line with no characters is a null command. PowerBoot takes no action and does not issue an error message; it returns the PowerBoot prompt.
Spaces or tabs	PowerBoot compresses and treats multiple adjacent spaces and tabs as a single space and ignores leading and trailing spaces.

Special Keys and Characters

Table 6 lists special keyboard keys and characters that perform specific PowerBoot command operations.

Table 6:*Special Keys and Characters for PowerBoot Operations*

Key or Character	Operation
" "	Denote an ASCII character string.
=	When used with the PowerBoot M command to access a memory location, provides access to the current address.
–	When used with the PowerBoot M command to access a memory location, provides access to the current address minus one times the specified size (*current-address* – 1 x *size*).
–count	When used with the PowerBoot M command to access a memory location, provides access to the current address minus *count* times the specified size (*current-address – count* x *size*).
+	When used with the PowerBoot M command to access a memory location, provides access to the current address plus one times the specified size (*current-address* + 1 x *size*).
+count	When used with the PowerBoot M command to access a memory location, provides access to the current address plus *count* times the specified size (*current-address* + *count* x *size*).
#address	When used with the PowerBoot M command to access a memory location, provides access to the specified address.
.	Exits the current command and returns to the PowerBoot prompt.
Backspace	Deletes one character.
Enter	Terminates a command line. No action is taken on a command until you terminate it. If you do not enter any characters before pressing this key, the PowerBoot re-displays the prompt. When used with the PowerBoot M command to access a memory location, provides access to the current address plus 1 (*current-address* + 1). When used with the PowerBoot MD command, displays the next screen of memory contents.
Return	Terminates a command line. No action is taken on a command until you terminate it. If you do not enter any characters before pressing this key, PowerBoot re-displays the prompt.

Command Line Characteristics

PowerBoot command line characteristics are:

- The command interpreter is not case-sensitive. Lowercase ASCII characters *a* to *z* are treated as uppercase characters.
- The console does not provide type-ahead buffer support.

Index

Product Error Report

Product:	Serial No.:
Date Of Purchase:	Originator:
Company:	Point Of Contact:
Tel.:	Ext.:
Address:	
Present Date:	
Affected Product: o Hardware o Software o Systems	Affected Documentation: o Hardware o Software o Systems
Error Description:	

This Area to Be Completed by Force Computers:

Date:

PR#:

Responsible Dept.: o Marketing o Production
Engineering ‡ o Board o Systems

+ Send this report to the nearest Force Computers headquarters, as listed on the address page.