U-Boot v2010.06em8

Bootloader Manual

Rev6 / 22.06.2011

emtrion GmbH

Revision: **6 / 22.06.2011**

Rev	Date/Signature	Changes
1	16.02.10/pk	updated to U-Boot v2009.08em3
2	24.06.10/pk	updated to U-Boot v2009.08em6
3	01.02.11/sz	updated to U-Boot v2010.06em2 and extended document validity to all core modules
4	07.06.11/sz,rr	Updated to U-Boot v2010.06.em7 and display table 2.6, added mmc update, mmc boot chapter and description of the mmc command, display description
5	16.06.11/ft	Add section “Using Windows Embedded Compact”

1 Inhalt

2 Introduction

This document describes the bootloader on CORE_MODULE. It is a porting of U-Boot with some enhancements regarding updating and booting the operating systems emtrion provides for the target.

2.1 Some conventions in this manual

2.1.1 Path names

All of the given path names are given as relative paths to the BSP root directory. If the given path is outside the BSP directory structure, it is written as an absolute path. For example

```
$> cd CORE_MODULE/rootfs
```

means

```
$> cd $BSP/CORE_MODULE/rootfs
```

In some examples the BSP is given as directory CORE_MODULE in home directory of user 'hico'. For example:

```
/home/hico/emdist/CORE_MODULE/rootfs
```

2.1.2 Naming Conventions

Target	the CORE_MODULE hardware, i.e. dimm-sh7724 or hico7723
Host	The Windows or Linux system the target is connected by serial and Ethernet

2.1.3 IP Addresses

For the sake of clarity, all the code examples in this manual are written with real ip addresses where:

Host ip address (aa.bb.cc.dd):	192.168.105.3
Target ip address (ee.ff.gg.hh):	192.168.2.20
Gateway ip address:	192.168.0.254
Subnet mask:	255.255.0.0

Always remember to replace these addresses to correspond your network setup.

3 Getting Started

3.1 Serial Setup

The target has a bootloader console on serial port UART_A. Connect it to the host and configure it for these settings.

```
baud rate               115200
data bits               8
parity                  none
stop bits               1
hardware flow control   no
```

3.2 Target Configuration

Ensure that the switch SW1-4 (only available for hico7723 and dimm-sh7724) on the target is set to off so that the bootloader uses the console.

3.3 Booting the target

3.3.1 Boot Messages

After the target has been powered up, it will print the bootloader version, some board information like RAM and flash memory sizes, the version and revision of the display table, a list of the supported displays, the current date & time and finally the prompt for command input

```
U-Boot 2010.06em7-svn851 (Jun 03 2011 - 15:16:17)

CPU:   SH4A @ 500 MHz
Reset: Manual
BOARD: DIMM-SH7724
SDRAM: 256 MiB
FLASH: 8 MiB
Display Table Version: 2.6
Display Table Revision: 2
Supported displays:
NL6448 TX14 NLC640T57D480CTMK19 EPSL5S30947P00 TX16 LW700AT9399 AM800480E4TMQW
GLYNC0283QGLC GLYNC0283QGLCI TX26D12 EPSL5F31024T00 EPSL5F31024T01 M170EG01
NLC320T57D240CTYK9 NL6448BC33 UMSH8272 AUO_G104VN01
LCD:   UMSH8272
Date:  1900-01-01 (Sunday) 06:20:53
Base:  c311
Net:   sh_eth
```

3.3.2 LED Status

Immediately after power-on the red LED D2 goes on. When U-Boot is ready to accept commands or is booting an operating system, the green LED D2 is lightened. If there was no error, the red LED is turned off.

These errors can be empty battery or checksum failures. If the error is not fatal, the booting into the operating system will continue.

If there is an error in the lowlevel board initialization, e.g. RAM error, the bootloader will flash “SOS” (three times short, three times long, and three times short).

4 Using Network

The network is used for downloading operating system and filesystem images, either to store them into flash memory or execute them right ahead.
The download protocols are TFTP and NFS.

4.1 Setup Target

The target is configured with these settings:

dhcp	Yes
serverip	192.168.105.3 (for TFTP/NFS)
ipaddr	192.168.2.20 (only when dhcp is turned off)
netmask	255.255.0.0
gatewayip	192.168.0.254
dnsip	0.0.0.0

The actual values can be displayed with:

```
CORE_MODULE # printenv dhcp serverip ipaddr netmask gatewayip
dhcp=y
serverip=192.168.105.3
ipaddr=192.168.2.20
netmask=255.255.0.0
gatewayip=192.168.0.254
```

They can be changed with:

```
CORE_MODULE  # setenv dhcp n
CORE_MODULE  # setenv ipaddr 192.168.105.71
CORE_MODULE  # setenv serverip 192.168.105.1
```

Beware that there is no "=" between variable name and value

Network connectivity can be tested like this:

```
CORE_MODULE  # ping $serverip
host 192.168.105.3 is alive
CORE_MODULE  # ping 192.168.105.3
host 192.168.105.3 is alive
```

At the moment, they are used only until the next boot. To make the values persistent, enter

```
CORE_MODULE # saveenv
Saving Environment to NVRAM...
Writing Parameters #13 to NVRAM
```

4.2 Setup Host

Your host with $serverip needs at least a TFTP server running. The connectivity can be tested with

```
CORE_MODULE  # tftp $loadaddr <some_dummy_file>
```

To configure the DHCP Server, the target's own ethernet address might be needed. It can be retreived by

```
CORE_MODULE  # printenv ethaddr
ethaddr=00:1C:1E:9F:FF:FE
```

The value is fixed and can't be changed with setenv.

5 Using Linux

5.1 Booting From Network

To boot linux from an NFS server, use the "bootx linux nfs" command. Similar, to boot linux from a TFTP server, use the "bootx linux tftp" command.

```
CORE_MODULE # bootx linux nfs
BOOTP broadcast 1
DHCP client bound to address 192.168.1.17
File transfer via NFS from server 192.168.105.3; our IP address is 192.168.1.17
Filename '/nfsroot/hico7723/rootfs/boot/uImage-hico7723'.
Load address: 0x8c000000
Loading: #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         ############################################################
done
Bytes transferred = 1946054 (1db1c6 hex)
linux will be booted now
## Booting kernel from Legacy Image at 8c000000 ...
   Image Name:   Linux-2.6.29.4em2-svn607
   Created:      2009-11-10  13:54:12 UTC
   Image Type:   SuperH Linux Kernel Image (gzip compressed)
   Data Size:    1945990 Bytes = 1.9 MB
   Load Address: 88001000
   Entry Point:  88002000
   Verifying Checksum ... OK
   Uncompressing Kernel Image ...
   SD Firmware: OK

Starting kernel ...

Linux version 2.6.29.4em2-svn607 (ny@em-srv3-kaha.intern.emtrion.de) (gcc
version 4.2.2 (Gentoo 4.2.2 p1.0)) #11 PREEMPT Tue Nov 10 14:54:09 CET 2009
```

These actions will be triggered with the "bootx linux" command.

- if DHCP is enabled, a DHCP request is made to determine target's IP address
- the image referenced in $image.linux is downloaded from network to the memory at $loadaddr
- the kernel image is checked for correctness
- the variable bootargs_linux is expanded and stored in bootargs and given to linux as kernel command line
- the kernel will mount the filesystem by NFS specified in $nfs
- linux is started.

5.2 Updating To Flash

To store linux in flash, at least two images need to be placed in flash. The linux kernel and the root filesystem. There can be an additional application filesystem for further files that might also be stored in flash.

Note that variable loadaddr points usually to the middle of the RAM so the operating system can be relocated to the start of RAM. Yet loadaddr_update points usually to the beginning of RAM, so about 125MB can be downloaded and updated with U-Boot.

5.2.1 Kernel Update

The kernel is updated with "update linux nfs" or "update linux tftp"

```
CORE_MODULE  # update linux nfs
File transfer via NFS from server 192.168.105.3; our IP address is
192.168.105.72
Filename '/nfsroot/CORE_MODULE/rootfs/boot/uImage-CORE_MODULE'.
Load address: 0x8c000000
Loading: #################################################
#################################################
#################################################
#################################################
##################
done
Bytes transferred = 1422957 (15b66d hex)
Calculated data checksum = 0x5c8ea1c3 (OK)
Erasing:   complete
Writing:   complete
Verifying: complete
Update successful
```

These actions will be triggered with the "update linux" command.

- if DHCP is enabled, a DHCP request is made to determine target's IP address
- the image referenced in $image.linux is downloaded from network to the memory at $loadaddr_update
- the kernel image is checked for correctness and it's checksum is printed.
- the kernel is written to the linux kernel partition named "linux"
- after writing, a byte-to-byte verification is performed to being correctly written

To use a different file than the default, either set the name in image.linux or append the name on the command line as in "bootx linux tftp uImage-myboard".

5.2.2 Root Filesystem Update

The root filesystem is updated with "update rootfs nfs" or "update rootfs tftp".

- if DHCP is enabled, a DHCP request is made to determine target's IP address
- the image referenced in $image.rootfs is downloaded from network to the memory at $loadaddr_update
- the image is written to the filesystem partition "rootfs"
- after writing, a byte-to-byte verification is performed to being correctly written

For filesystems that are JFFS2, additional actions are triggered.

- the last sector is padded when necessary
- if the partition is on a NAND chip, the JFFS2 cleanmarkers are written to the NAND OOB

For filesystems that are UBIFS, additional actions are triggered.

- The erase counter of each block is resetted to 0. Therefore it is recommended to update ubifs only once in the production and then update it in the field within linux

As the filesystem image doesn't include any checksum, the correctness of the image is not checked.

5.2.3 Root Filesystem Update

The root filesystem is updated with "update rootfs nfs" or "update rootfs tftp".

- if DHCP is enabled, a DHCP request is made to determine target's IP address
- the image referenced in $image.rootfs is downloaded from network to the memory at $loadaddr_update
- the image is written to the filesystem partition "rootfs"
- after writing, a byte-to-byte verification is performed to being correctly written

For filesystems that are JFFS2, additional actions are triggered.

- the last sector is padded when necessary
- if the partition is on a NAND chip, the JFFS2 cleanmarkers are written to the NAND OOB

For filesystems that are UBIFS, additional actions are triggered.

- The erase counter of each block is resetted to 0. Therefore it is recommended to update ubifs only once in the production and then update it in the field within linux

As the filesystem image doesn't include any checksum, the correctness of the image is not checked.

5.2.4 Application Filesystem Update

The application filesystem is handled like the root filesystem with three exceptions.

- "update appfs nfs" is the command
- the image $image.appfsis downloaded
- the image is stored in the partition named appfs

5.3 Booting From Flash

Once the kernel and the root filesystem are in flash, they can be booted with "bootx linux" or "bootx linux flash".

```
CORE_MODULE # bootx linux flash
Reading:   complete
linux will be booted now
## Booting kernel from Legacy Image at 8c000000 ...
   Image Name:   Linux-2.6.29.4em2-svn607
   Created:      2009-11-10  13:54:12 UTC
   Image Type:   SuperH Linux Kernel Image (gzip compressed)
   Data Size:    1945990 Bytes = 1.9 MB
   Load Address: 88001000
   Entry Point:  88002000
   Verifying Checksum ... OK
   Uncompressing Kernel Image ...
   SD Firmware: OK

Starting kernel ...
```

Booting from flash differs in a few aspects from network booting.

- No network is accesses, no DHCP transfer is performed.
- The kernel will mount the root filesystem as specified in $mtdroot

5.4 Updating to eMMC

For updating Linux and the RootFS on a eMMC there are three simple commands provided. All commands require a running network connection and a NFS-server to read images. The size of the images is limited to the RAM-size because of the update procedure which is described in more detail in 5.2.

5.4.1 Partition table update

To setup the initial eMMC partition table the command “run mmc_update_part_table” must be run.

```
CORE_MODULE  # run mmc_update_part_table
Using sh_eth device
File transfer via NFS from server 192.168.105.12; our IP address is
192.168.1.44
Filename '/nfsroot/dimm-sh7724/rootfs/boot/mmc_partition_table-dimm-
sh7724.bin'.
Load address: 0x88000000
Loading: ##
done
Bytes transferred = 8192 (2000 hex)

MMC write: dev # 0, block # 0, count 16 ... 16 blocks written: OK
```

5.4.2 Kernel update

To update the kernel the command “run mmc_update_kernel” must be run.

```
CORE_MODULE  # run mmc_update_kernel
Using sh_eth device
File transfer via NFS from server 192.168.105.12; our IP address is
192.168.1.44
Filename '/nfsroot/dimm-sh7724/rootfs/boot/uImage-dimm-sh7724'.
Load address: 0x88000000
Loading: ##################################################################
         ##################################################################
         ##################################################################
         ##################################################################
         ##################################################################
         ##################################################################
         ##################################################################
         ##################################################################
         ##########################
done
Bytes transferred = 2782987 (2a770b hex)

MMC write: dev # 0, block # 16, count 15664 ... 15664 blocks written: OK
```

5.4.3 RootFS update

To update the rootfs the command “run mmc_update_rootfs” must be run.

```
CORE_MODULE  # run mmc_update_rootfs
Using sh_eth device
File transfer via NFS from server 192.168.1.54; our IP address is 192.168.1.44
Filename '/nfsroot/dimm-sh7724/rootfs/boot/rootfs-dimm-sh7724-128.ext2'.
Load address: 0x88000000
Loading: ##################################################################
         ##################################################################
         [...]
```

```
         #################################################################
         #########################
done
Bytes transferred = 251658240 (f000000 hex)

MMC write: dev # 0, block # 15680, count 500096 ... 500096 blocks written: OK
```

5.5 Booting From eMMC

Once the kernel and the root filesystem are in eMMC flash, they can be booted with "bootx linux mmc".

```
CORE_MODULE # bootx linux mmc

MMC read: dev # 0, block # 16, count 15664 ... 15664 blocks read: OK
linux will be booted now
## Booting kernel from Legacy Image at 8c000000 ...
   Image Name:   Linux-2.6.35em1+
   Created:      2011-03-31  15:35:23 UTC
   Image Type:   SuperH Linux Kernel Image (gzip compressed)
   Data Size:    2784821 Bytes = 2.7 MiB
   Load Address: 88001000
   Entry Point:  88002000
   Verifying Checksum ... OK
   Uncompressing Kernel Image ...
Starting kernel ...
```

5.6 Setting Kernel Command Line Arguments

The kernel command line can be extended with

"setenv bootargs_linux_pre <value>" or "setenv bootargs_linux_post <value>"

5.7 Configuring For Auto-Booting

To boot linux automatically each time the module is powered-up or resetted, add

```
CORE_MODULE  # setenv bootcmd bootx linux flash
CORE_MODULE  # setenv bootdelay 0
CORE_MODULE  # saveenv
```

6 Using Windows Embedded Compact (formerly Windows Embedded CE)

6.1 Image name

Please regard that the examples in this section are made using our DIMM-SH7723 core module. Therefore certain examples will show the image name for DIMM-SH7723. Please change the name of the image according to your device.

6.2 Booting From Network

To boot Windows Embedded Compact from a TFTP server, use the "bootx wce tftp" command.

```
CORE_MODULE # bootx wce tftp
BOOTP broadcast 1
DHCP client bound to address 192.168.1.55
Using dm9000 device
TFTP from server 192.168.105.3; our IP address is 192.168.1.55
Filename 'wce-hico7723'.
Load address: 0x8c000000
Loading: ################################################################
         ################################################################
         ################################################################
         ################################################################
         ################################################################
         ############################################################
done
Bytes transferred = 26019396 (18d0644 hex)
Calculated data checksum = 0x49afc594 (OK)
wce will be booted now
## Booting kernel from Legacy Image at 8c000000 ...
   Image Name:   DevKit_CE600 v4 /licensed
   Created:      2011-01-21  11:08:43 UTC
   Image Type:   SuperH WinCE Kernel Image (uncompressed)
   Data Size:    26019332 Bytes = 24.8 MiB
   Load Address: 88200000
   Entry Point:  88200004
   Verifying Checksum ... OK
   Loading Kernel Image ... OK
Clear hive based registry when starting WinCE now

Starting with existing hive registry

Starting WinCE ...
```

These actions will be triggered with the "bootx wce" command.

- if DHCP is enabled, a DHCP request is made to determine target's IP address
- the image referenced in $image.wce is downloaded from network to the memory at $loadaddr
- the kernel image is checked for correctness
- the variable bootargs_wce is expanded and stored in bootargs and given to Windows Embedded Compact as kernel command line
- Windows Embedded Compact is started.

6.3 Updating To Flash

To store Windows Embedded Compact in flash, at least one image need to be placed in flash. There can be an additional application filesystem for further files that might also be stored in flash.

Note that variable loadaddr points usually to the middle of the RAM so the operating system can be relocated to the start of RAM. Yet loadaddr_update points usually to the beginning of RAM, so about 125MB can be downloaded and updated with U-Boot.

6.3.1 Kernel Update

The kernel is updated with "update wce tftp"

```
CORE_MODULE  # update wce tftp
BOOTP broadcast 1
DHCP client bound to address 192.168.1.55
Using dm9000 device
TFTP from server 192.168.105.3; our IP address is 192.168.1.55
Filename 'wce-hico7723'.
Load address: 0x88000000
Loading: ##################################################################
         ##################################################################
         ##################################################################
         ##################################################################
         ##################################################################
         ###########################################################
done
Bytes transferred = 26019396 (18d0644 hex)
Calculated data checksum = 0x49afc594 (OK)
Erasing:   complete
Writing:   complete
Verifying: complete
Update successful
```

These actions will be triggered with the "update wce" command.

- if DHCP is enabled, a DHCP request is made to determine target's IP address
- the image referenced in $image.wce is downloaded from network to the memory at $loadaddr
- the kernel image is checked for correctness and it's checksum is printed.
- the kernel is written to the Windows Embedded Compact kernel partition named "wce"
- after writing, a byte-to-byte verification is performed to being correctly written

To use a different file than the default, either set the name in image.wce or append the name on the command line as in "bootx wce tftp wce-myboard".

6.4 Booting From Flash

Once the kernel is in flash, they can be booted with "bootx wce" or "bootx wce flash".

```
CORE_MODULE # bootx wce
Reading:   complete
Calculated data checksum = 0x49afc594 (OK)
wce will be booted now
## Booting kernel from Legacy Image at 8c000000 ...
   Image Name:   DevKit_CE600 v4 /licensed
   Created:      2011-01-21  11:08:43 UTC
   Image Type:   SuperH WinCE Kernel Image (uncompressed)
   Data Size:    26019332 Bytes = 24.8 MiB
   Load Address: 88200000
   Entry Point:  88200004
```

```
   Verifying Checksum ... OK
   Loading Kernel Image ... OK
Clear hive based registry when starting WinCE now

Starting with existing hive registry

Starting WinCE ...
```

Booting from flash differs in a few aspects from network booting.

- No network is accesses, no DHCP transfer is performed.

6.5 Cleaning the Windows CE registry

Windows CE knows to kinds of registries:

- Hive-based registry
- RAM-based registry

The Hive-based registry is located in the filesystem stored in the NAND flash. The RAM-based registry can be copied to the NOR flash partition named "registry" using a special API function under Windows Embedded Compact. The API function has the name RegFlushKey.

The choosen kind of registry is loaded by the Windows Embedded Compact during the startup. To start with the default registry, the bootloader have to erase the corresponding flash partition (for RAM-based registry). For the hive-based registry the bootloader sets a flag which introduce Windows Embedded Compact to use the default registriy.

6.5.1 Cleaning the Hive-based registry

The information for Windows Embedded Compact to start with or without the Hive-based registry is passed from the bootloader through the boot args. If you want to start without the Hive-based registry you should set the environment variable "erase_hive_registry" to "y".

```
CORE_MODULE # setenv erase_hive_registry y
CORE_MODULE # bootx wce
Reading:   complete
Calculated data checksum = 0x49afc594 (OK)
wce will be booted now
## Booting kernel from Legacy Image at 8c000000 ...
   Image Name:   DevKit_CE600 v4 /licensed
   Created:      2011-01-21  11:08:43 UTC
   Image Type:   SuperH WinCE Kernel Image (uncompressed)
   Data Size:    26019332 Bytes = 24.8 MiB
   Load Address: 88200000
   Entry Point:  88200004
   Verifying Checksum ... OK
   Loading Kernel Image ... OK
Clear hive based registry when starting WinCE now

Erasing hive based registry

Starting WinCE ...
```

```
Remark:
You should not call saveenv after you have set "erase_hive_registry" to
"y" because in this case the OS starts every time with a clean hive
registry until you explicitly reset "erase_hive_registry" to "n".
Reason: the value of the environment variable is stored to NVRAM when
saveenv is called and loaded again during bootloader startup.
```

6.5.2 Cleaning the persistent copy of the RAM-based registry

You can erase the persistent copy of the RAM-based registry using the bootloader command "erase_pt":

```
CORE_MODULE # erase_pt registry
Erasing:   complete
CORE_MODULE #
```

6.6 Supported File Formats

The bootloader handles a files which contains the raw binary output from the Platform Builder (nk.nb0) extended by a special header. The special header contains all the information needed by the bootloader, like length, start address etc.

6.7 Setting Kernel Command Line Arguments

The kernel command line can be extended with

"setenv bootargs_wce_pre <value>" or "setenv bootargs_wce_post <value>"

6.8 Configuring For Auto-Booting

To boot Windows Embedded Compact automatically each time the module is powered-up or resetted, add

```
CORE_MODULE  # setenv bootcmd bootx wce flash
CORE_MODULE  # setenv bootdelay 0
CORE_MODULE  # saveenv
```

6.9 Using the bootmenu (deprecated)

The bootloader contains a bootmenu which is shown either the command "bootmenu" is executed from the console or if SW1-4 is on:

```
emtrion GmbH
Bootloader for HiCO.DIMM7723 with Revision 0 or higher
U-Boot 2010.06em3-svn764 (Feb 11 2011 - 12:39:29)
Copyright (c) 2008
All rights reserved
Bootloader menu

1.) Execute stored Image
4.) Download Image via onboard Ethernet controller and store in Flash
5.) Download Image via onboard Ethernet controller, store in SDRAM and execute
6.) Extended functionality


>
```

With item “1” the currently stored image can be executed manually. This item executes the “bootx wce flash” command.

With item “4” and “5” an image can be downloaded via tftp and either stored to the NAND (command “update wce tftp”) or to the RAM (command “bootx wce tftp”). The name of the image is defined by the environment variable $image.wce. The default name of the image for this Developer Kit is set to “wce-hico7723”.

Choosing item “6” the extended menu gets open:

```
emtrion GmbH
Bootloader for HiCO.DIMM7723 with Revision 0 or higher
U-Boot 2010.06em3-svn764 (Feb 11 2011 - 12:39:29)
Copyright (c) 2008
All rights reserved
Bootloader menu

1.) Self test
2.) Clean WinCE Persistent Registry
e.) Return to main menu
q.) Quit and return to command prompt


>
```

With item “2” you can manually clean the Hive-based registry or the persistent copy of the RAM-based registry of Windows Embedded Compact. This should always be done at any time before downloading a new image.

7 Basic Command Help

7.1 bdinfo

Prints some board information like address ranges.

7.2 Bootmenu (deprecated)

Users accustomed of the former emtrion bootloader might use “bootmenu” to have the same GUI.

Bootmenu will automatically execute a stored image when SW1-4 is on.

7.3 bootx <os> [flash|tftp|nfs|usb|mmc] [file]

The extended boot command for downloading and booting images from various sources (NFS,TFTP or USB when available).

Described in detail in chapter 5.1

7.4 chkimg

Performs a CRC32 check of U-Boot in flash to ensure correctness of U-Boot. It may be concatenated with other commands like

```
CORE_MODULE  # chkimg && echo "OK"
CRC32: match: 0xe0a9d63a
```

7.5 clocks

Displays the system and the various peripheral clock frequencies.

7.6 date [MMDDhhmm[[CC]YY][.ss]]

Without argument it displays the current date/time.

With argument the date/time is changed in the RTC.

7.7 erase_pt <partition_name>

Erases a flash partition. Critical partitions like bootloader or NVRAM can’t be erased.

7.8 envreset

Resets all “standard variables” and “dynamic variables”. The NVRAM variables (network configuration) are not affected.

7.9 flinfo

Displays information about the NOR flash. The type of chip, it’s specification and the status of each sector, whether it is readonly (RO), empty (E) or locked-down (LO). Locked-down is more secure than RO

```
CORE_MODULE  # flinfo

Bank # 1: CFI conformant FLASH (16 x 16)  Size: 8 MB in 67 Sectors
  Intel Extended command set, Manufacturer ID: 0x89, Device ID: 0x20
  Erase timeout: 4096 ms, write timeout: 1 ms
  Buffer write timeout: 2 ms, buffer size: 64 bytes

  Sector Start Addresses:
```

```
  A0000000   RO   A0008000   RO   A0010000   RO   A0018000   RO   A0020000   RO
  A0040000 E RO   A0060000 E RO   A0080000        A00A0000
```

7.10 flpart

Used to partition the flash. Described in detail in chapter 0

7.11 help [cmd]

Displays a list of all commands or a specific help for <cmd>

7.12 mmc <read|write|list>

"mmc list" is used to display all known mmc devices.

"mmc read 0 0x8C000000 10 100" is used to load 100 blocks of data since the 10. Block from the 0. Mmc device to memory address 0x8c000000.

"mmc write 0 0x8C000000 10 100" is used to write 100 blocks of data since the 10. Block of the 0. Mmc device which is read from memory address 0x8c000000.

7.13 nand <info|bad|stats>

nand info" is used to display the NAND chip information.

"nand bad" displays the list of bad sectors that are skipped when accessing NAND by reading or writing.

"nand stats" prints ECC statistics

7.14 reset/reboot

Performs a software reset. "reboot" is an alias to reset so Linux users can use the same command in U-Boot and Linux

7.15 ping <ipaddr>

Sends an ICMP message to the host <ipaddr>. DNS names are not allowed.

7.16 printenv [<var …]

Displays either all or only the selected environment variables set with "setenv". The most important ones are listed in chapter 8**Fehler! Verweisquelle konnte nicht gefunden werden.**

- Dynamic variables are marked with a "*"
- Variables shadowing the contents of NVRAM are marked with "+"
- All other variables are not prefixed.

7.17 pstatus

Displays the power-on-self-test status which is also reflected on the red LED D2.

It may be concatenated with other commands like

```
CORE_MODULE  # pstatus && echo "OK"
POST status: warning
CORE_MODULE  # pstatus || echo "Failed"
POST status: warning
Failed
```

7.18 setenv name [value]

Without argument a variable is deleted. Dynamic variables are resetted to default.

With argument, the contents is overwritten or when the variable does not yet exists is overwritten.

7.19 saveenv

Stores all variables to the persistent NOR flash. If the NOR has been configured to be write-protected with “setenv nor_wp y”, the user is asked to confirm writing the variables.

```
CORE_MODULE # saveenv
Saving Environment to NVRAM...
Flash is write protected. Continue? (y/n)
```

7.20 sleep <seconds>

Waits for the specified number of seconds.

7.21 update <os|partition> <nfs|tftp|usb> [<filename>]

Updates an operating system or any partition. OS can be "linux", "wce", "rootfs" etc. For "linux", "wce", "rootfs" default filenames already exist. For other partitions the name must be given.

7.22 update_cfg <os_configuration> <nfs|tftp|usb> <filename>

Updates an operating system specific region or the display table in the NVRAM.

Os_configuration can be “Linux”, “WinCE”, “Unknown”, “Application”, “display_table” or “sd_firmware”.

8 Environment Variables

A list of all known environment variables can be retrieved with the command “printenv”

8.1 Dynamic Variables

A few variables (marked with “*” in printenv) are dynamic, that means they are automatically calculated each time they are used. For example, $mtdparts contains the partition table in a format suitable for the linux kernel command line. If the partition is changed, this variable is also automatically adjusted.

Variable	Used For	Influenced By
board_cfg	Reports state of switch SW1. Can be used to trigger different types of booting.	Switch SW1
loadaddr	Download to RAM for booting by TFTP/NFS/USB	RAM Size
loadaddr_update	Download to RAM for updating by TFTP/NFS/USB	RAM Size
nand_wp	NAND is write-protected if set to "y" and saved. Effective immediately. Default is “n” to bad block table can be maintained. When booting, U-Boot will report that the NOR is write protected.	-
nand_skip_loading	NAND is not initialized when booting. Use it to improve boot speed when needed.	-
nor_wp	NOR is write-protected if set to “y” and saved. Effective immediately. Default is “y” to ensure system stability. Setting it back to “n” requires the command “protect off all” to be executed when manually trying to update the flash with “cp” commands. “protect off” is not required for the update command. Use “saveenv” to make the change persistent. When booting, U-Boot will report that the NOR is write protected.	-
hw_base_board_code	Operating system is able to initialize and load drivers depending on the base board the module is installed. Some baseboards identify	baseboard used

	themselves. If they don't, the code needs to be entered manually. Details are explained in chapter 15	
hw_board_ident	Board identification given to the OS	Board serial number and revision
display_table_version	The version of the display table	Loaded display table
display_table_revision	The revision of the contents of the display table	Loaded display table
display	Selecting display/Bootlogo	Attached display when display adapter ATA_TFT is connected. Otherwise must be manually set
display_calibration_linux	Default touch calibration for linux	Selected display
display_calibration_wce	Default touch calibration for WinCE	Selected display
display_configuration	Default display configuration data set	Selected display
display_phys_width	OS Display Driver Initialization of display's width in [mm]	Selected display
display_phys_height	OS Display Driver Initialization of display's height in [mm]	Selected display
usb_speed	Limits the speed of the USB Host port. Can have the values "fullspeed" and "highspeed". Is given to the Linux/WinCE kernel.	-
image.uboot	U-Boot image	module name, RAM Size
image.linux	Linux kernel image	module name
image.rootfs	Root filesystem image	module name block erase size on partition type of flash (NOR/NAND) filesystem type
image.appfs	Application filesystem image	module name block erase size on partition type of flash (NOR/NAND) filesystem type
image.bootlogo	Bootlogo image filename	display name
bootargs_linux	Linux kernel command line	-
console	Linux console	same as U-Boot is using
ip	Linux network parameters	module name
root	Linux root device	booting linux by TFTP/NFS

mtdparts	Linux mtd partition table	partition table set with flpart
quiet	Linux quiet booting	Switch SW1-4 setting
nfspath	Linux NFS Filesystem path	module name
nfspath_boot	NFS Path for downloading images with update/bootx	-
nfs	used as root when booting Linux by NFS/TFTP	-
mtdroot	Linux root device	partition marked for rootfs filesystem for partition
bootargs_wce	WinCE kernel command line	-
wp_wce_registry	When set to "y", the WinCE registry is write protected. When set to "auto", the value of "nor_wp" is used. When set to "n", the WinCE registry is never write protected	-
image.wce	WinCE kernel image	module name

8.2 NVRAM Variables

Variables (marked with "+" in printenv) that provide easy access to contents that is stored in the OS independent NVRAM area, see chapter 10. These settings can be used by the operating system as well. They are type-checked, invalid parameters are not stored.

Variable	**Used For**
serverip	TFTP/NFS Download
gatewayip	TFTP/NFS Download
ethaddr	TFTP/NFS Download. This variable is readonly
ipaddr	TFTP/NFS Download
netmask	TFTP/NFS Download
dnsip	not used by U-Boot. But will be used in operating system
dhcp	On TFTP/NFS Download.
Hw_product_type	HiCO.DIMM7723
hw_serial_nr	Serial number (is currently empty, use MAC address for identification)
hw_revision	Hardware revision of the HiCO.DIMM7723 module

hw_patch_level	Normally 0 but might have other values if the hardware is the module has been patched.

8.3 Standard Variables

These are standard u-boot variables adjusting the behavior of U-Boot commands.

Variable	Used For
bootcmd	command is executed immediately after reset
bootdelay	execution of bootcmd is delayed for amount of seconds. Even with bootdelay=0 the command can be executed with Ctrl-C
verify	Activates or deactivates (y/n) the CRC check of an image before starting it. The default value is y (activated). Setting to n speeds up the start of an image.

Many more are available in the vanilla U-Boot documentation.

8.4 Variables bootargs_wce and its flags

U-Boot provide WinCE kernels with various boot options defined in bootargs_wce. They are

Variable	Used For
disable_nand	when defined, WinCE won't load the NAND driver nor will use the NAND flash filesystem
erase_hive_registry	When set to y Windows Embedded Compact starts without the Hive-based registry and uses the default registry.

9 Flash Partition Table

The bootloader features flash partitioning which is then used by "update" or "bootx" commands. It can be display or modified with the command "flpart".

Changing the partition table doesn't change the data on the partition until "erase_pt", "update" or "bootx" are executed.

The partition table can't be changed when the NOR partition is write protected.

9.1 Partition Table

```
CORE_MODULE # flpart
Nr | Name     | Chip |  Start     | Size       | Type          | FS    | Flags
-------------------------------------------------------------------------------
 0 | uboot    |  0   | 0x00000000 | 0x00080000 | U-Boot        |       | fixed
 1 | NVRAM    |  0   | 0x00080000 | 0x00040000 | NVRAM         |       | fixed
 2 | bootlogo |  0   | 0x000c0000 | 0x00040000 | Splash-Screen |       |
 3 | linux    |  1   | 0x00000000 | 0x00300000 | Linux-Kernel  |       |
 4 | rootfs   |  1   | 0x00300000 | 0x02000000 | Filesystem    | jffs2 | rootfs
 5 | appfs    |  1   | 0x02300000 | 0x0dd00000 | Filesystem    | jffs2 |
Commands:
   a) Append partition
   d) Delete partition
   m) Modify partition
   p) Print partition table
   r) Reset partition table
   q) Quit
Cmd (? for help)>
   a) Append partition
   d) Delete partition
   m) Modify partition
   p) Print partition table
   r) Reset partition table
   q) Quit&Save
Cmd (? for help)>
```

Variable	Used For
Nr	index of the partition in the table
Name	name of the partition as used with bootx/update/erase_pt
Chip	when more than one flash memory chip is present, this field is displayed. 0 is always the boot flash chip.
Start	start address of partition relative to the chip
Size	the size of the partition
Type	type of contents that will be put into partition. Possible values are U-Boot itself, NVRAM, Splash-Screens, operating systems or filesystems
FS	If the partition holds a filesystem it is the type of the filesystem
Flags	• fixed: the partition start and size is not changable. Only required for partitions necessary when booting. • readonly: the partition can't be erased or updated • mounted readonly: a filesystem is mounted readonly and

	therefore files can't be changed. Direct access to the flash partitions, e.g. /dev/mtd are still possible. • rootfs: This partition is mounted as rootfs.

9.2 Resetting Table

There are preconfigured setups of the partition table as they are used on the "Starterkits"/development boards.

By pressing "r" the list of presets is displayed. "n" for "None" selects the minimum requirements necessary for the bootloader. "l" for "Linux" selects a partition table suitable for Linux and "w" for "WinCE" selects a partition table suitable for Windows Embedded CE.

Partition table suitable for Linux:

```
CORE_MODULE  # flpart
[...]
Cmd (? for help)> r
  OS Types:
   n) None
   l) Linux
   w) WinCE
OS (? for help)> l
Partition table reset
Nr | Name     | Chip |  Start     | Size       | Type          | FS    | Flags
-------------------------------------------------------------------------------
 0 | uboot    |  0   | 0x00000000 | 0x00080000 | U-Boot        |       | fixed
 1 | NVRAM    |  0   | 0x00080000 | 0x00040000 | NVRAM         |       | fixed
 2 | bootlogo |  0   | 0x000c0000 | 0x00040000 | Splash-Screen |       |
 3 | linux    |  1   | 0x00000000 | 0x00300000 | Linux-Kernel  |       |
 4 | rootfs   |  1   | 0x00300000 | 0x02000000 | Filesystem    | ubifs | rootfs
 5 | appfs    |  1   | 0x02300000 | 0x0dd00000 | Filesystem    | ubifs |
Cmd (? for help)>
```

Partition table suitable for Windows Embedded Compact:

```
CORE_MODULE  # flpart
[...]
Cmd (? for help)> r
  OS Types:
   n) None
   l) Linux
   w) WinCE
OS (? for help)> l
Partition table reset
Nr | Name     | Chip |  Start     | Size       | Type          | FS      | Flags
-------------------------------------------------------------------------------
 0 | uboot    |  0   | 0x00000000 | 0x00080000 | U-Boot        |         | fixed
 1 | NVRAM    |  0   | 0x00080000 | 0x00040000 | NVRAM         |         | fixed
 2 | bootlogo |  0   | 0x000c0000 | 0x00040000 | Splash-Screen |         |
 3 | registry |  0   | 0x00780000 | 0x00080000 | WinCE-Registry|         |
 4 | wce      |  1   | 0x00000000 | 0x03000000 | WinCE-Kernel  |         |
 5 | wcefs    |  1   | 0x03000000 | 0x0d000000 | Filesystem    | Unknown|
Cmd (? for help)>
```

9.3 Appending Partitions

Before experimenting with this command you might at first reset the table to "None".

By pressing "a" a partition can be appended. It requires a describing name, a chip number (only present when multiple chips are installed), the start address, size and at least the partition type. Start and Size can be set to 0, so they are automatically calculcated.

```
CORE_MODULE  # flpart
[...]
Cmd (? for help)> a
Adding partition # 3
  Name  (): test
  Chip  (0): 0
  Start (0 for auto) (0x00000000): 0x00000000
   --> Set to 0x00100000
  Size  (0 for auto, 0x00700000 max) (0x00000000): 0x00000000
   --> Set to 0x00700000
Partition Type (U-Boot, ? for help)> ?
   e) WinCE-EBoot
   f) Filesystem
   l) Linux-Kernel
   n) NVRAM
   r) WinCE-Registry
   s) Splash-Screen
   u) U-Boot
   w) WinCE-Kernel
   F) FPGA
   0) Unknown
Partition Type (U-Boot, ? for help)> f
  Fixed  (n): n
  Readonly  (n): n
  Filesystem Types
Filesystem (, ? for help)> ?
   j) jffs2
   c) cramfs
   i) initrd
   f) FlashFX
   y) yaffs
   r) romfs
   u) ubifs
   0) Unknown
Filesystem (, ? for help)> c
  Root-FS  (n): n
  Mount Readonly  (n): y
Partition 3 added
Nr | Name     | Chip |  Start     | Size       | Type          | FS     | Flags
-----------------------------------------------------------------------------
 0 | uboot    |  0   | 0x00000000 | 0x00080000 | U-Boot        |        | fixed
 1 | NVRAM    |  0   | 0x00080000 | 0x00040000 | NVRAM         |        | fixed
 2 | bootlogo |  0   | 0x000c0000 | 0x00040000 | Splash-Screen |        |
 3 | test     |  0   | 0x00100000 | 0x00700000 | Filesystem    | cramfs |
mounted readonly
Cmd (? for help)>
```

9.4 Deleting Partitions

To delete a partition, press "d" followed by the partition number. Partitions marked as fixed can't be deleted, their fixed flag must be first removed with modify.

```
CORE_MODULE  # flpart
[...]
Cmd (? for help)> d
Delete which partition? 3
Partition 3 deleted
Nr | Name     | Chip |  Start     | Size       | Type          | FS   | Flags
```

```
-------------------------------------------------------------------------
 0 | uboot    |  0   | 0x00000000 | 0x00080000 | U-Boot        |      | fixed
 1 | NVRAM    |  0   | 0x00080000 | 0x00040000 | NVRAM         |      | fixed
 2 | bootlogo |  0   | 0x000c0000 | 0x00040000 | Splash-Screen |      |
Cmd (? for help)>
```

9.5 Modifying Partitions

This is essentially the same as appending. Only the partitions need to be selected first.

```
CORE_MODULE  # flpart
[...]
Cmd (? for help)> m
Modify which partition? 2
  Name  (bootlogo): bootlogo
  Chip  (0): 0
  [...]
```

9.6 Saving Partition Table

The partition table is saved when pressing "q" and confirming with "y".

```
CORE_MODULE # flpart
[...]
Cmd (? for help)> q
Partition table has been modified. Save? (y):
```

If the partition table has not been modified and there is no need to save it, it's reported with

```
CORE_MODULE # flpart
[...]
Cmd (? for help)> q
Parititon table NOT changed!
```

The modifications can be cancelled at any time by pressing "Ctrl-C" or resetting the board.

10 NVRAM

Board configuration like ethernet address, network settings, partition table and operating and application specific parameters are stored in a flash partition called NVRAM.

This partition has some unique features

- it is operating system independent. The bootloader, linux or WinCE can access and modify it, e.g. setting the IP address persistent in userspace
- there is a redundancy management. On NOR flash, two erase sectors are allocated for it, each having a copy of the contents. If one copy is defect when booting (e.g. by powerfail while doing "saveenv"), it is automatically repaired on next booting.

```
U-Boot 1.3.4em2-svn231 (Dec  8 2008 - 23:02:48)

CPU:   SH4A @ 400 MHz
BOARD: CORE_MODULE (SW1=0x00)
SDRAM: 128 MB
FLASH: 8 MB
NVRAM: Original Flash is BAD
NVRAM: Original Critical is BAD
NVRAM: Original 0 and Mirror 17 don't match
NVRAM: Using Mirror
NVRAM: Repairing Original
NAND:  256 MB
Net:   DM9000
Date:  1900-01-01 (Sunday) 00:46:10
CORE_MODULE # pstatus
POST status: failure
CORE_MODULE # reboot


U-Boot 1.3.4em2-svn231 (Dec  8 2008 - 23:02:48)

CPU:   SH4A @ 400 MHz
BOARD: CORE_MODULE (SW1=0x00)
SDRAM: 128 MB
FLASH: 8 MB
NAND:  256 MB
Net:   DM9000
Date:  1900-01-01 (Sunday) 00:46:32
CORE_MODULE # pstatus
POST status: OK
```

- On NAND chips, the partition spans 4 erase sectors and is able to handle bad sectors in that partition.
- Typically, only 2kb are used for the configuration parameters. The remaining space is available to operating system and applications, so they can benefit from redundancy management for critical configuration as well. The display table with the list of all supported displays is kept here and updated with update_cfg.

11 Retrieving Board Information/Hardware Revision

```
CORE_MODULE # printenv hw_product_type hw_revision hw_patch_level
hw_serial_nr
hw_product_type=HiCO.DIMM7723
hw_revision=r3
hw_patch_level=0
hw_serial_nr=
```

12 Hardware Switches

These switches of the target are used by U-Boot:

SW1-4	when on, disables any output until a key is pressed. Sets the linux kernel command line quiet.
SW1-3	when on, U-Boot will not assume to run on the configured baseboard and restrict itself to the module features. A Bootlogo will not be displayed.

The current setting of SW1 is immediately reported by the environment variable "board_cfg". '0' means all are off and 'f' when all are on.

"board_cfg" can be utilized to boot different operating systems depending on SW1-1.

```
CORE_MODULE # setenv boot_wce test \$board_cfg = 1 \&\& bootx wce flash
CORE_MODULE # setenv boot_linux test \$board_cfg = 0 \&\& bootx linux flash
CORE_MODULE # setenv bootcmd run boot_linux\; run boot_wce
CORE_MODULE # saveenv
```

13 Bootlogo

When a display is connected, U-Boot can display a bootlogo while booting. These steps must be performed to have a bootlogo.

A display must be selected by setting the environment variable "display". If an unsupported display is given, U-Boot will report the supported displays when booting. The operating system (Linux/WinCE) will also be informed what display is selected.

A bootlogo image file needs to be programmed into the "bootlogo" partition. This file is required in a special format .uzb created by the tool "emBootlogo" that can be downloaded from the support page.
If the bootlogo is in a resolution not matching the display, a default bootlogo will be displayed.

The switch SW1-3 needs to be off. If the switch is on, the display will not be initialized.

```
CORE_MODULE  # setenv display NL6448
CORE_MODULE  # saveenv
CORE_MODULE  # update bootlogo tftp bootlogo_640x480x565.uzb
CORE_MODULE  # reset
```

14 Auto-Reset

When a pure CPU reset, e.g. because of faulty code, is detected that doesn't affect external hardware, a hardware reset is automatically initiated so that the periphery is in a known and identical state each boot.

```
U-Boot 1.3.4em2-svn231 (Dec  8 2008 - 23:02:48)

CPU:   SH4A @ 400 MHz
Reset: Manual
Doing hard reset due to former soft reset


U-Boot 1.3.4em2-svn231 (Dec  8 2008 - 23:02:48)
```

15 hw_base_board_code Table

The baseboard code is a 8bit/16bit identifier for describing the base board, its revision and patchlevel.

Bits	Name	Comment
0…3	Patchlevel	Defines the patchlevel of a board (changes on the printed PCB). Patchlevel "a" is 0x0, Patchlevel "b" is 0x1 etc.
4…7	Revision	PCB Revision. Revision "1" is 0x10, Revision "2" is 0x20 etc. 0x00 is reserved
8…15	Baseboard type	The values 0x0000…0xBF00 are intended for customer specific boards. Baseboards developed by emtrion for customers will be assigned numbers beginning with 0x1. Emtrion's own boards have these numbers assigned DIMMECOBASE: 0xC1000 DIMMBASE: 0xC2000

16 hw_revision Code

The hw_revision code is an identifier to describe the exact revision of the core module. This code will be set during the production of the core module and reflects the revision code printed on the label on the core module.

Usually this code doesn't influent the behavior of the bootloader but there are some exceptions. It depends on this code which display will be supported and which default settings for the touch interface will be taken and sent to the OS.

For more information please see the documentation of the display table and the display table tool.

17 Links

- U-Boot Home
- emtrion Support Page