emLinux

Documentation

Rev008/03.09.2014

emtrion GmbH

Rev	Date	Modifications
001	28.09.2012	First revision
002	16.11.2012	Add updates of emSDK 1.9
003	17.01.2013	Add additional instructions for image creation of DIMM-MX53
004	22.01.2013	Fixed Typos in emSDK update chapter
005	12.04.2013	Modfied boot setup and update section of DIMM-AM335X Added section about installing emLinux on a general linux desktop
		Added environment variable netmask for setup of static ip config.
006	14.05.2014	Extend section about installing emLinux on a general linux desktop VM is updated to Debian 8 (Jessie) Update section about Eclipse Add section about QtCreator
007	03.09.2014	Update Link to U-Boot 2010.06 manual
008	27.01.2015	Add some instructions for em_image

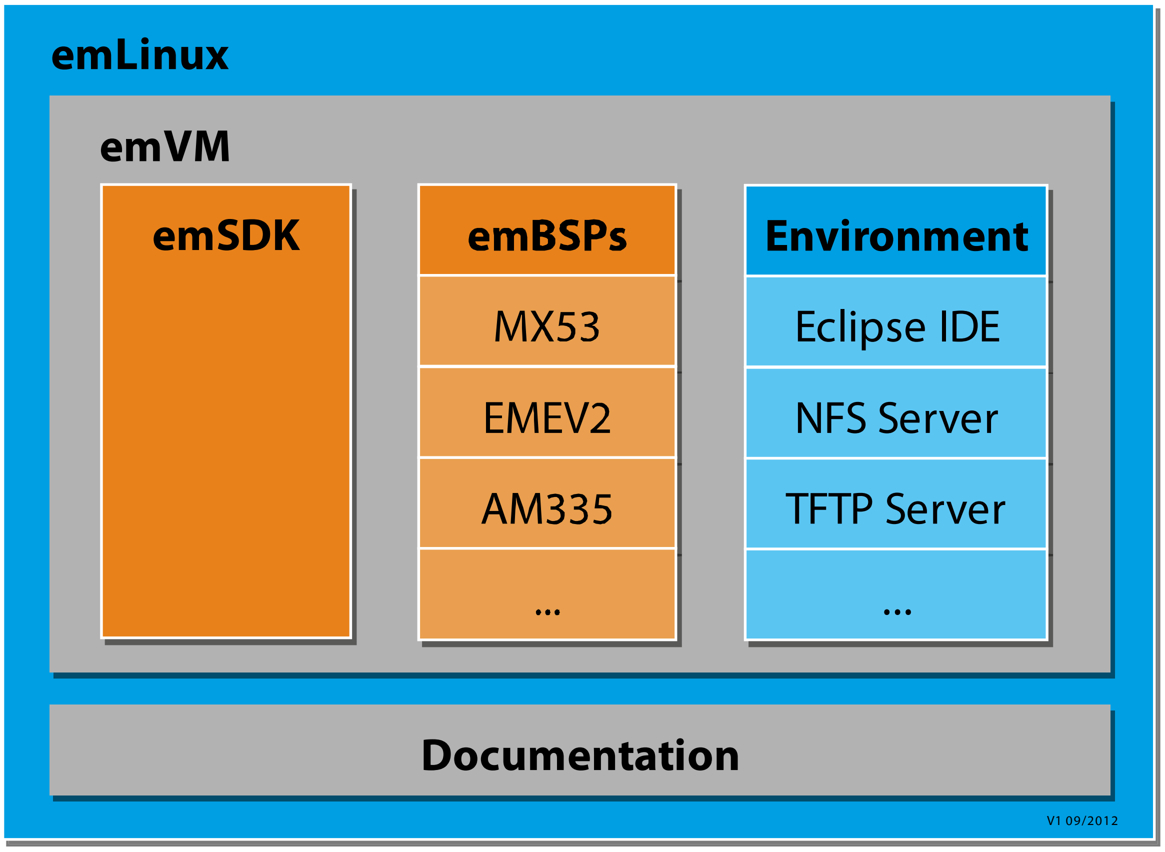
1 Inhaltsverzeichnis

2 Terms and Definitions

Term	Definition
emVM	Emtrion Virtual Machine
emBSP	Emtrion Board Support Package, root file system, toolchain, etc
emSDK	Emtrion System Developer Kit
Dev-Kit/Target	Emtrion Developer Kit
Host	Developer PC
Toolchain	Compiler, Linker, etc.
RootFS	Root file system, contains the basic operating system
Console	Text terminal interface for Linux
TFTP	Trivial file transfer protocol, used to transfer files over network
NFS	Network File System, can share directories over network
U-Boot	Bootloader, starts the operating system
VM	Virtual machine, isolates a guest operating system on the host

3 Introduction

It is assumed that users of emtrion Linux developer kits are already familiar with Linux. General Linux and programming knowledge are out of the scope of this document. emtrion is happy to assist you in acquiring this knowledge. If you are interested in training courses or getting support, please contact the emtrion sales department. This manual documents emSDK version **1.9** and later.



emtrion delivers a DVD accompanying the Linux developer kits. This DVD contains a VMware virtual machine running Debian Linux (emVM) and the VMware player used to start the VM.

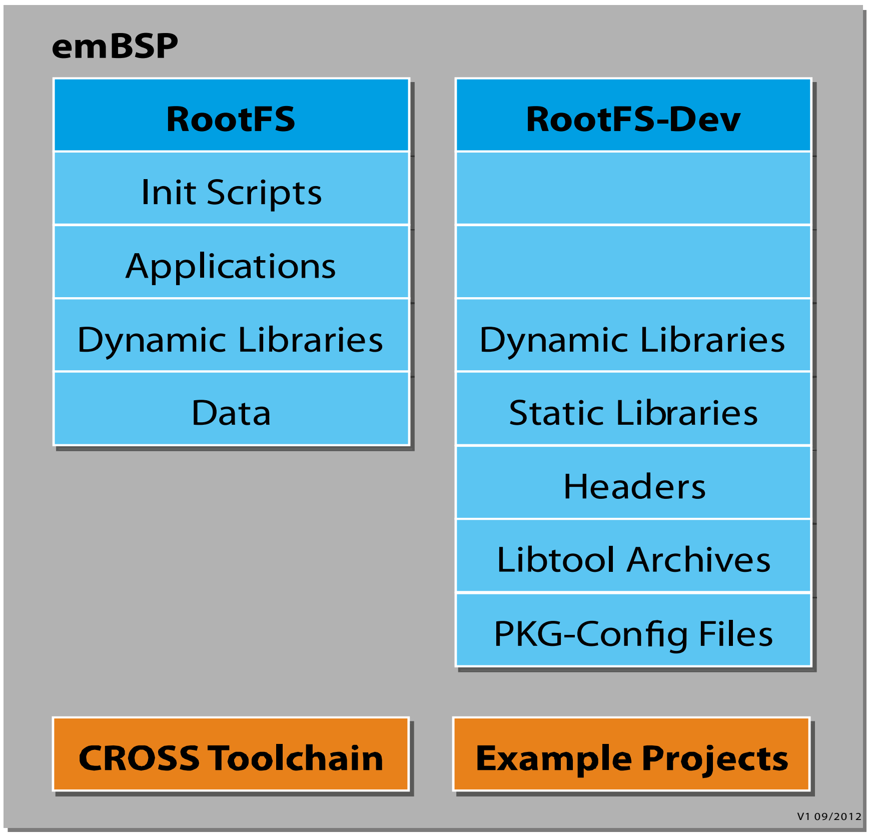
If you have general questions regarding Debian (How do I setup my network connection?) or VMware player (How do I access my hosts' serial port from within the VM?) you can have a look at:

Debian Wiki - https://wiki.debian.org

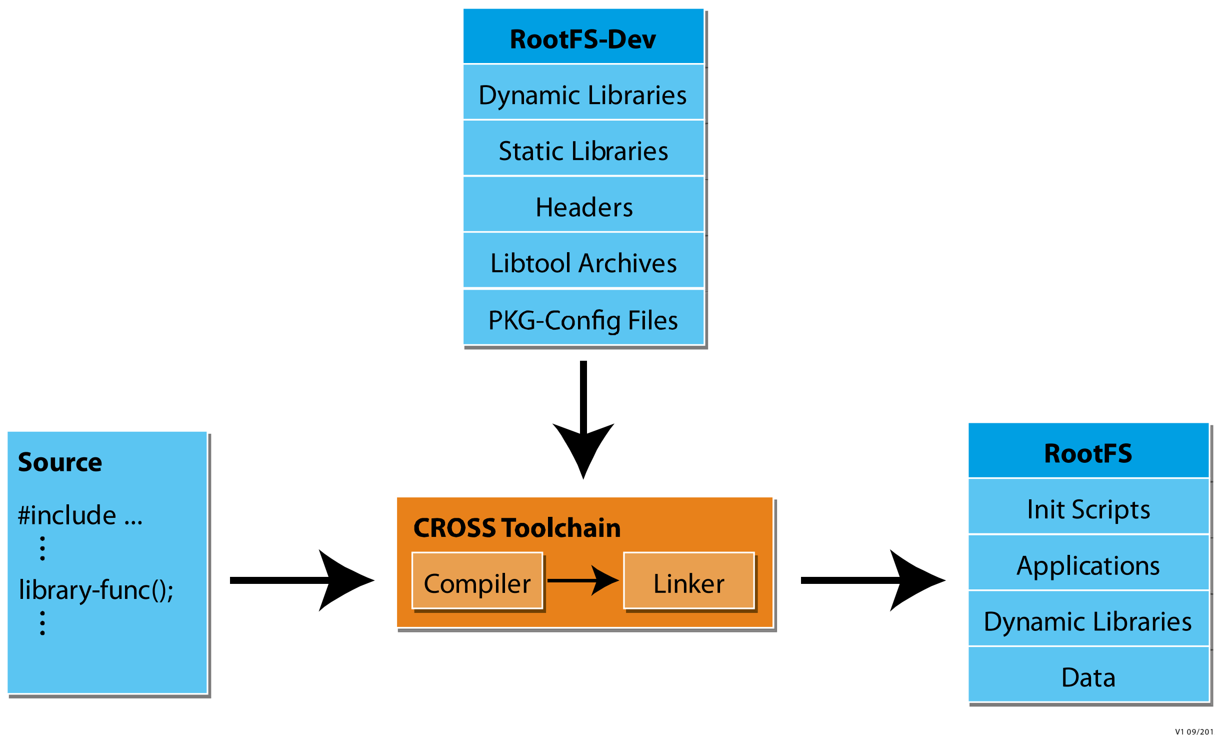
Debian Handbook – http://debian-handbook.info

VMware Documenation - http://www.vmware.com/pdf/desktop/vmware_player50.pdf

The purpose of the virtual machine and its contents is to enable the user of the kit to start an application development as fast as possible. For that reason a complete development environment is set up in the VM, it includes pre-built BSPs (emBSP) for our various Linux kits plus a cross toolchain, a set of helper scripts (emSDK) and general development tools like the Eclipse IDE.



There are two root file systems contained in an emBSP: one used on the target and one used for development only. You compile your applications using the headers and libraries in the RootFS-Dev, the result is copied to the RootFS running on the target. The reason being that things like static libraries and headers are not needed during runtime on the target and we can save space by not including them. The RootFS and RootFS-Dev come pre-built with our Kits.



emLinux is delivered together with emtrion's developer kits. The purpose of such kits is to evaluate, to benchmark, to prototype and to write applications. Therefore, emLinux clearly focuses on application development rather than system development. System development, i.e. modifying the OS or writing drivers, requires quite a lot of knowledge and experience. If your project requires modifying, updating or extending the BSP and you are an experienced Linux system developer, you can get the complete source code for free. If you are not an experienced Linux system developer, emtrion also offers software engineering services. Either way, contact us at sales@emtrion.com.

What you can do with emLinux:

- Use a pre-built, pre-configured BSP on our developer kit
- Develop applications using a completely configured build environment
- Integrate applications/libraries into the BSP for testing or demonstration purposes

The way of achieving this is subject to the following chapters.

4 emVM Overview

The VMware VM comes with an Debian Linux installation.

Some useful facts:

- User: hico, password: hico, it is recommended to change the password
- A TFTP server is preinstalled and running by default, its root directory is /tftpboot.
- Preinstalled BSPs are located in ~/Downloads, the latest BSPs can be downloaded at emtrion.de
- The emSDK installation path is /opt/emtrion
- Serial ports added to the VM appear at /dev/ttySn, USB serial converters at /dev/ttyUSBn, baudrates can be configured using the stty tool, picocom can be used as a terminal
- QtCreator is installed

5 emBSP Overview

An emBSP is a compressed tar.bz2 archive, the naming scheme is:

embsp-**<release date>-<target platform>.tar.bz2**

Currently available target platforms:
dimm-mx6
dimm-mx53
dimm-emev
dimm-am335x
dimm-mx257

The following table describes the structure of an emBSP:

Directory	Purpose
etc	Custom Configuration Files
doc	Documentation
root	emBSP file systems
toolchain	The toolchain for cross-compilation
projects	Example and custom projects

To login on the target use user **root** and empty password.

6 emSDK Overview

emSDK is a set of simple scripts and configuration files to handle some common tasks when evaluating an emtrion processor module running Linux.

These tasks include:

- Installing an emBSP
- Cross compiling an existing application or library and adding the result to the BSP
- Sharing a RootFS over network
- Creating file system images for the target

6.1 emSDK Tools

Term	Definition
em_copy	Unpacks an emBSP to a specified directory
em_env	Sets a platform specific environment, exports the RootFS via NFS
em_devkit_image	Creates an image for direct use on the emtrion Dev-Kit
em_image	Creates images for general use

Every Tool gives a small description of the usage if you pass the –h switch.

6.2 Installing an emBSP – em_copy

After startup the standard BSPs are located at ~/Downloads, the BSPs come as .tar.bz2 archives. em_copy is used to unpack BSPs into a specified directory. Besides unpacking the archive, em_copy takes care of some hardcoded path dependencies, so that you can install the BSP to any location you like.

The general syntax is:

```
em_copy [MODE] [OPTIONS] SOURCE_PATH DESTINATION_PATH
```

Mode	Purpose
--install	emBSP installation
--release	emBSP creation

The release mode is documented in the command help, it is usually not used during evaluation.

Arguments	Purpose
SOURCE_PATH	The path to the emBSP archive
DESTINATION_PATH	The path where the emBSP is to be installed

For example:

```
em_copy      --install      ~/Downloads/embsp-20120918-dimm-am335x.tar.bz2
/home/hico/work
```

installs the BSP in /home/hico/work.

The output will be something like: (you have to enter the hico password)

```
Info: trying to gain super-user privileges
[sudo] password for hico:
Info: successfully gained super user privileges
Info: Fixing rootfs in /home/hico/work/embsp-20120906-dimm-am335x/root/rootfsdev
Info: Fixing toolchain in /home/hico/work/embsp-20120906-dimm-am335x/toolchain
```

After this you can change the working directory to ~/work/embsp-20120906-dimm-am335x/.

6.3 Setting a BSP specific environment – em_env

When developing for the target board it is helpful to have some environment variables at hand which point to commonly used directories or applications.

The general syntax is:

```
em_env OPTION [PLATFORM_CONFIGURATION PLATFORM_ROOT_FOLDER]
```

Option	Purpose
-s	Sets the environment
-p	Prints the environment
-u	Unsets the environment

When the –s option is given you have to specify the PLATFORM_ROOT_FOLDER.

Argument	Purpose
PLATFORM_ROOT_FOLDER	Installation directory of the emBSP

At least the following variables get set:

Variable	Meaning	Example
PLATFORM	Identifies the module SOC	dimm-am335x
PLATFORM_ROOT	The root directory of the BSP	~/embsp-20120906-dimm-am335x
PLATFORM_CFG	Platform configuration file	/opt/emtrion/etc/platform/dimm-am335x.cfg
ARCH	Architecture	ARM
CROSS_COMPILE	Cross Toolchain Prefix	arm-poky-linux-gnueabi-
ROOTFS	Path to RootFS	$PLATFORM_ROOT/root/rootfs
ROOTFSDEV	Path to RootFS-Dev	$PLATFORM_ROOT/root/rootfsdev
MAKE_JOBS	Extra arguments for make	-j4
IMAGE_CONFIG	default image configuration	dimm-am335x.tar-cfg

For example:

Set environment:

```
em_env –s ~/work/embsp-20120906-dimm-am335x/
```

6.4 NFS export/unexport of RootFS – em_nfs

emSDK also provides a tool which automatically handles NFS export of the Root File System. Before using it, you have to setup the environment with “em_env –s”.

The general syntax is:

```
em_nfs OPTION
```

Option	Purpose
-e	Exports the current RootFS via NFS
-u	Unexports the current RootFS
-c	Cleans /etc/exports from all entries created with em_nfs

When using “em_nfs –e” the tool will print the full path of the exported RootFS. This is the path which has to be added in the U-Boot environment variable (See Chapter U-Boot Bootloader).

If booting via NFS is not working try restarting the NFS server:

```
sudo service nfs-kernel-server restart
```

6.5 Compiling and installing an example project

Assuming you installed an emBSP and set up a correct environment you can cross compile one of the example projects.

6.5.1 Hello World

There is a helloworld example in **$PLATFORM_ROOT/projects/helloworld**

```
cd $PLATFORM_ROOT/projects/helloworld           # changes directory to the project
make clean                                      # cleans everything
make ARCH=$ARCH CROSS_COMPILE=$CROSS_COMPILE    # cross compiles helloworld
make DESTDIR=$ROOTFS install                    # installs helloworld to the RootFS
```

Variables in makefiles are different from bash environment variables, that is the reason you have to pass ARCH=$ARCH and so on.

If you have already booted your device via NFS you can now execute the program by typing "helloworld" in a terminal of the device.

6.5.2 Animated Tiles (QT)

This is a QT example. You can find it in **$PLATFORM_ROOT/projects/animatedtiles**

The first step to build a QT application is the execution of qmake. Depending on which of our modules you are using you can either directly use it by typing "qmake" after you have setup your environment or you have to specify it explicitly with the full path:

DIMM-AM335x, DIMM-MX6

```
cd $PLATFORM_ROOT/projects/animatedtiles     # changes directory to the project
qmake                                        # create the Makefile
make ARCH=$ARCH CROSS_COMPILE=$CROSS_COMPILE # cross compiles animatedtiles
make INSTALL_ROOT=$ROOTFS install            # installs animatedtiles to the RootFS
```

DIMM-MX257, DIMM-MX53, DIMM-EMEV

```
cd $PLATFORM_ROOT/projects/animatedtiles     # changes directory to the project
${ROOTFSDEV}/usr/qt-4.6.3/bin/qmake          # create the Makefile
make ARCH=$ARCH CROSS_COMPILE=$CROSS_COMPILE # cross compiles animatedtiles
make INSTALL_ROOT=$ROOTFS install            # installs animatedtiles to the RootFS
```

The executable is installed under /usr/local/bin/animatedtiles. You can execute it from there on your target device.

6.6 Creating a file system image – em_devkit_image

To use the root file system on the target without the need for a network connection or something similar you can create file system images for the flash media on the processor modules. For different platforms different images are needed. Some targets use file systems like ext3 in an MMC, others use UBIFS on raw flash. em_devkit_image uses the default image configuration specified by the environment variable **$IMAGE_CONFIG**. All possible image config configuration files can be found under **$PLATFORM_ROOT/etc**.

The general syntax is:

```
em_devkit_image
```

The image will be put in the current working directory. The resulting image can be brought to the target via the bootloader for example. You can also use an already running Linux.

For example:

```
cd $PLATFORM_ROOT/images
em_devkit_image
```

If you have a “non-standard” developer kit (the standard is equipped with 1GB NAND Flash and has a revision of at most R3A), for example a DIMM-MX53 with 512MB NAND Flash, you may need to use em_image instead of em_devkit_image. It is more flexible, but also more complicated. Alternatively you can change the above mentioned environment variable **$IMAGE_CONFIG.** The configuration files end on "ubi-cfg" and are found in the etc/ directory within the emBSP.

For example:

```
export IMAGE_CONFIG=dimm-mx53-512mb-nand.ubi-cfg
```

6.7 Updating emSDK

You can simply download the new release, remove the current installation and extract the new one.

For example:

```
cd ~/Downloads
wget http://www.support.emtrion.de/emsdk/emsdk-1.9.tar.bz2
sudo rm –rf /opt/emtrion
sudo tar -xf ~/Downloads/emsdk-1.9.tar.bz2 –C /
<open a new terminal so the changes take effect>
```

7 Using QtCreator

QtCreator is the IDE for Qt Development but can be also used as IDE in general. The package is preinstalled in the VM. This section gives you some hints on using QtCreator together with emLinux. Please use the official documentation regarding general issues.

7.1 Running QtCreator

To successfully build for the target, you must run “qtcreator” from a terminal which is setup via em_env. If you want to do a local build of your project in the VM you have to start it normally without the emLinux environment setup.

7.2 Kit Setup

For cross development with QtCreator you have to specify a “kit” under **Tools->Options-> Build&Run**. A kit consists of a target specific Qt build, a cross compiler toolchain and a gdb debugger binary. Select the files below in the kit configuration:

Kit Component	Path
Qt Build	[emBSP_path]/toolchain/usr/bin/qmake
Cross Toolchain	[emBSP_path]/toolchain/usr/bin/arm-poky-linux-gnueabi/arm-poky-linux-gnueabi-g++
GDB	[emBSP_path] /toolchain/usr/bin/arm-poky-linux-gnueabi/arm-poky-linux-gnueabi-gdb

7.3 Device Setup

QtCreator can directly upload a binary to the target and run it via SSH/SCP. To do so, you have to configure a device under **Tools->Options->Device**. We have already preconfigured a generic emLinux device. Running a standard emBSP RootFS on the target you should be able to connect using this setup by only changing the target’s IP address.

8 Using Eclipse

Eclipse can be used as IDE with our processor modules running Linux. You can install it using the package management system of the VM.

On our support pages you will find tutorials that assist you in using Eclipse with our modules:

Eclipse setup for use with emSDK , Debugging:
http://www.support.emtrion.de

Qt Integration:
The Qt Integration in Eclipse is deprecated. Please use QtCreator as IDE for Qt development.

You can also use DS-5 with all our modules but dimm-mx257; the corresponding emBSPs contain the necessary driver and daemon. Currently DS-5 version 5.11 is supported.

9 U-Boot Bootloader

The basic task of U-Boot is to load the operating system from bulk memory into RAM and then start the kernel. You can also use it to initiate an update of the kernel, the RootFS and of U-Boot itself. Furthermore you can configure directly from the medium the operating system is to be booted from, for example MMC, NFS or a serial terminal.

9.1 Basic U-Boot operation

To work with U-Boot, first use a terminal program to connect to the serial line of the board. As soon as the U-Boot prompt appears in the terminal, U-Boot is ready to receive commands. The general U-Boot documentation can be found here: http://www.denx.de/wiki/U-Boot/Documentation

U-Boot has a set of environment variables which are used to store information needed for booting the operating system. Variables can contain information such as IP addresses, but they can also contain a whole script of actions to perform sequentially. The following commands explain the basic handling of environment variables:

U-Boot command	Explanation
printenv [variable]	This shows the value of the specified variable. If no variable is specified, the whole environment is shown.
setenv [variable] [value]	Set a variable to a specific value. If no value is specified, the variable gets deleted.
saveenv	Make your changes permanent, so they remain after power off or reboot.

9.2 Using U-Boot to change boot device or update parts of the system

This chapter describes how U-Boot has to be setup for updating and booting. The exact variable assignment is board specific. DIMM-MX6, DIMM-AM335X and DIMM-EMEV can be used in the same way, but for DIMM-MX53 and DIMM-MX257 the usage is different.

The variable serverip has to be set to the IP-address of the emVM. You can get the IP-address by entering **ifconfig** in the terminal of your emVM.

Take the IP-address of the corresponding network adapter and assign it to the variable serverip in the U-Boot console. The format of [IP-address] is dot decimal notation.

```
U-Boot # setenv serverip [IP-address]
```

9.2.1 Boot setup and update on DIMM-EMEV

9.2.1.1 Update

Updates are generally performed using a TFTP-server, which is running in emVM. It is already set up to run, when you boot it for the first time. The root folder of the TFTP-Server is **/tftpboot**. **Every file which is needed in the update process must be copied into this folder**.

File names:

The files needed in the update process need to have specific file names. These file names are specified in the following environment variables of U-Boot:

Variable name	Description
image.linux	Filename of the linux kernel image
image.uboot	Filename of the U-Boot image
Image.bootstrap	Filename of the Bootstrapper image

Update of the root file system

First you need to copy or move the output of **em_devkit_image** to the TFTP folder. After setting up the "serverip" in U-Boot as described above you can use the following commands to start an update of the root file system:

```
U-Boot # run update_rootfs
```

This starts the update process. Please be patient as the process of fetching the RootFS image via network and decompressing it to the flash storage can take a few minutes.

Update of the linux kernel

To update the kernel you must copy the kernel image to the TFTP directory in emVM. If you take the kernel from the emBSP, you can find the image in $ROOTFS/boot/. The image is called "uImage…" (the exact name depends on the board). It is important to also set up the serverip.

```
U-Boot # run update_kernel
```

Update of U-Boot bootloader and bootstrapper

Attention: If the board is turned off while updating the bootloader and bootstrapper or another error occurs, the board will be rendered unusable to you. Please only update the bootloader if you are explicitly instructed to do so by emtrion.

Copy the bootstrapper and bootloader image to the TFTP folder in emVM. Then you can start the update process with the following command in U-Boot prompt:

```
U-Boot # run update_uboot
```

9.2.1.2 Booting

The default boot device in U-Boot is determined by the variable “bootcmd”. If you want to set up one of the following boot options as a default you have to set “bootcmd” to the command mentioned below.

Boot from on-board flash

This is the default boot option configured when you receive the developer kit from emtrion. To start it manually simply use this command:

```
U-Boot # run flash_boot
```

Boot via network using a NFS share

First you have to export the RootFS in emVM using the command “em_nfs -e”. Then you have to perform the following commands in U-Boot:

```
U-Boot # setenv serverip        [ip-address of emVM]
U-Boot # setenv nfsroot         [path to the RootFS in emVM]
U-Boot # setenv ip-method dhcp
U-Boot # saveenv
U-Boot # run net_boot
```

Now the board should boot via network using the NFS share in emVM.

9.2.2 Boot setup and update on DIMM-MX6 and DIMM-AM335X

9.2.2.1 Update

Updating the kernel or the root file system is done by using the nfs export of emSDK. You must run “em_nfs -e” to set it up. For further information please look into the corresponding chapter of this manual

Updates of the boot loader are performed using a TFTP-server, which is running in emVM. It is already set up to run, when you boot it for the first time. The root folder of the TFTP-Server is **/tftpboot**. **Every file which is needed in the update process must be copied into this folder**.

File names:

The files needed in the update process need to have specific file names. These file names are specified in the following environment variables of U-Boot:

Variable name	Description
image.linux	Filename of the linux kernel image
image.uboot	Filename of the U-Boot image
Image.bootstrap	Filename of the Bootstrapper image

Update of the root file system (uses NFS)

First you need to copy or move the output of **em_devkit_image** to the folder **$PLATFORM_ROOT/images**. After running “em_nfs -e” in you emVM use the following commands to start an update of the root file system:

```
U-Boot # setenv serverip        [ip-address of emVM]
U-Boot # setenv nfsroot         [path to the RootFS in emVM]
U-Boot # setenv ip-method       [dhcp or static]
U-Boot # setenv ipaddr [ip-address for device, only needed for static ip]
U-Boot # setenv netmask [netmask for device, only needed for static ip]
U-Boot # run update_rootfs
```

This starts the update process. Please be patient as the process of fetching the RootFS image via network and decompressing it to the flash storage can take a few minutes.

Update of the linux kernel (uses NFS)

To update the kernel you must copy the kernel image to $ROOTFS/boot/ in your emVM. The image is called “uImage…” (the exact name depends on the board). You also must run “em_nfs –e” in your emVM to be able to update. Now you can use following commands in U-Boot prompt.

```
U-Boot # setenv serverip        [ip-address of emVM]
U-Boot # setenv nfsroot         [path to the RootFS in emVM]
U-Boot # setenv ip-method       [dhcp or static]
U-Boor # setenv ipaddr [ip-address for device, only needed for static ip]
U-Boot # setenv netmask [netmask for device, only needed for static ip]
U-Boot # run update_kernel
```

Update of U-Boot bootloader and bootstrapper (uses TFTP)

Attention: If the board is turned off while updating the bootloader and bootstrapper or another error occurs, the board will be rendered unusable to you. Please only update the bootloader if you are explicitly instructed to do so by emtrion.

Copy the bootloader image to the TFTP folder in emVM. Then you can start the update process with the following command in U-Boot prompt:

```
U-Boot # setenv serverip        [ip-address of emVM]
U-Boot # setenv ip-method       [dhcp or static]
U-Boor # setenv ipaddr [ip-address for device, only needed for static ip]
U-Boot # setenv netmask [netmask for device, only needed for static ip]
U-Boot # run update_uboot
```

9.2.2.2 Booting

The default boot device in U-Boot is determined by the variable “bootcmd”. If you want to set up one of the following boot options as a default you have to set “bootcmd” to the command mentioned below.

Boot from on-board flash

This is the default boot option configured when you receive the developer kit from emtrion. To start it manually simply use this command:

```
U-Boot # run flash_boot
```

Boot via network using a NFS share

First you have to export the RootFS in emVM using the command "em_nfs -e". Then you have to perform the following commands in U-Boot:

```
U-Boot # setenv serverip        [ip-address of emVM]
U-Boot # setenv nfsroot         [path to the RootFS in emVM]
U-Boot # setenv ip-method       [dhcp or static]
U-Boor # setenv ipaddr [ip-address for device, only needed for static ip]
U-Boot # setenv netmask [netmask for device, only needed for static ip]
U-Boot # saveenv
U-Boot # run net_boot
```

Now the board should boot via network using the NFS share in emVM.

9.2.3 Boot setup and update on DIMM-MX53 and DIMM-MX257

The bootloader of DIMM-MX53 and DIMM-MX257 has several emtrion specific features. Please refer to the U-Boot manual available under following link on how to use it to change the boot setup or update the module:

U-Boot 2010.06em8 Support website

This document should also be included on the DVD you received with the Developer Kit.

10 Further Information

10.1 Online resources

Further information can be found on the emtrion support pages.

www.support.emtrion.de

10.2 We support you

emtrion offers different kinds of services, among them Support, Training and Engineering. Contact us at sales@emtrion.com if you need information or technical support.

11 Installing emLinux in a general desktop linux environment

Attention: If you install emLinux in a general desktop linux environment emtrion can not guarantee that this instructions are working for all desktop linux distributions. Furthermore **you do this at your own risk**. Emtrion **can not support** problems occurring with your specific linux installation.
In general Ubuntu derivates are the best starting point since emtrion also uses Ubuntu in its virtual machine.

First you have to install emSDK. Please follow the steps in the section about Updating emSDK.

Now you have to perform following two steps:

1. Add following two lines to **/etc/sudoers**

```
Defaults                      env_reset
Defaults                      !secure_path
```

2. Add following lines to **/home/[your_username]/.bashrc** or the corresponding file if you are not using bash as shell

```
if [ -e /opt/emtrion/lib/libemenv ]; then
  . /opt/emtrion/lib/libemenv
fi
```

3. Add following lines to **/home/[your_username]/.profile**

```
# set PATH to emSDK scripts if it exists
if [ -d "/opt/emtrion/bin" ] ; then
    PATH="/opt/emtrion/bin:$PATH"
fi
```

Now log off and on and you should be able to use emSDK.