F&S i.MX8X Linux

First Steps

Version 1.1
(2022-08-11)

armStone efus NetDCU

PicoCOM PicoCore PicoMOD QBliss

Elektronik
Systeme

© F&S Elektronik Systeme GmbH
Untere Waldplätze 23
D-70569 Stuttgart
Germany

Phone: +49(0)711-123722-0
Fax: +49(0)711-123722-99

About This Document

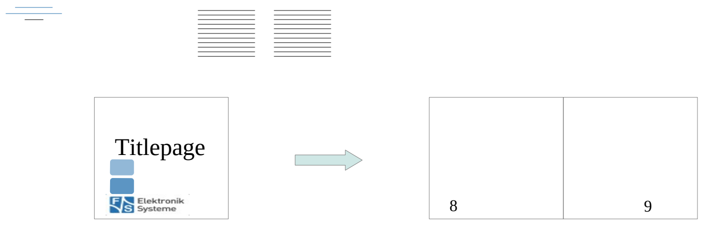
This document shows how to bring up F&S boards and modules under Linux, how to update firmware and how to use the system and the devices. It covers also compiling bootloader, Linux kernel and root filesystem as well as how to build your own applications for the device.

Remark

The version number on the title page of this document is the version of the document. It is not related to the version number of any software release! The latest version of this document can always be found at http://www.fs-net.de.

How To Print This Document

This document is designed to be printed double-sided (front and back) on A4 paper. If you want to read it with a PDF reader program, you should use a two-page layout where the title page is an extra single page. The settings are correct if the page numbers are at the outside of the pages, even pages on the left and odd pages on the right side. If it is reversed, then the title page is handled wrongly and is part of the first double-page instead of a single page.



Typographical Conventions

We use different fonts and highlighting to emphasize the context of special terms:

`File names`

Menu entries

```
Board input/output
```

`Program code`

```
PC input/output
```

```
Listings
```

```
Generic input/output
```

`Variables`

History

Date	V	Platform	A,M,R	Chapter	Description	Au
2020-07-10	1.0	fsimx8x	M	ALL	Based on fsimx8mm V1.0	KM
2022-08-11	1.1	fsimx8x	M	ALL	Modify for new release	KM

V	Version
A,M,R	Added, Modified, Removed
Au	Author

Table of Contents

1 Introduction

1.1 F&S Board Families And CPU Architectures

F&S offers a whole variety of Systems on Module (SOM) and Single Board Computers (SBC). There are different board families that are named NetDCU, PicoMOD, PicoCOM, armStone, QBliss, efus, and PicoCore (see Table 1).

Family	Type	Size
NetDCU	Single Board Computer	80 mm x 100 mm
PicoMOD	System on Module	80 mm x 50 mm
PicoCOM	System on Module	40 mm x 50 mm
armStone	Single Board Computer	100 mm x 72 mm (PicoITX)
QBliss	System on Module	70 mm x 70 mm (Qseven)
efus	System on Module	62 mm x 47 mm
PicoCore	System on Module	40 mm x 35 mm

Table 1: F&S Board Families

Linux is available for all of these platforms. F&S combines releases for platforms with the same CPU – or rather SoC (System on Chip) – as so-called *architecture releases*. All the boards of the same architecture can use the same sources, and the binaries can be used on any board of this architecture. Please note the difference: *board families* are grouped by form factor, *architectures* are grouped by CPU type, i.e. they usually contain boards of differ- ent families.

Table 2 shows all the architectures that are currently supported by F&S.

Architecture	CPU	Platforms
fsvybrid	NXP Vybrid VF6xx	PicoCOMA5, NetDCUA5, armStoneA5, PicoMOD1.2
fsimx6	NXP i.MX6	efusA9, QBlissA9, QBlissA9r2, armStoneA9, armStoneA9r2, PicoMODA9, NetDCUA9
fsimx6sx	NXP i.MX6-SoloX	efusA9X, PicoCOMA9X, PicoCoreMX6SX
fsimx6ul	NXP i.MX6-UltraLite	efusA7UL, PicoCOM1.2, PicoCoreMX6UL
fsimx7ulp	NXP i.MX7ULP	PicoCoreMX7ULP
fsimx8mn	NXP i.MX8MN	PicoCoreMX8MN
fsimx8mm	NXP i.MX8MM	PicoCoreMX8MM
fsimx8x	NXP i.MX8X	efusMX8X

Table 2: F&S Architectures

Remark

In December 2015, the two companies Freescale and NXP merged and both companies are now working under the brand name NXP. The name Freescale will disappear in the future, which is why we only use “NXP” throughout this document now. However some programs still output “Freescale” at some places. We have not touched this output to reflect the situation as it is.

1.2 Scope of This Document

This document describes the *fsimx8x* architecture. That means all F&S boards and modules based on the NXP i.MX8X SoC. The steps in this document will help you getting to know your board and do some basic operations in Linux, so that you can try out all the periphery and do some first tests and comparisons.

The additional document `LinuxOnFSBoards_eng.pdf` explains the more generic ideas and concepts of Linux on F&S boards and modules. So after having become acquainted with the board, you should continue reading that Linux document to get a more in-depth knowledge of the board and software.

2 Setting up the Board

In this chapter we will show how to connect the board to the PC. For a first test of the board functions, we only need a serial connection between PC and board. So as a first step, we will introduce all the boards and Starterkits of the *fsimx8x* architecture and show the location of all connectors, especially the debug port.

2.1 Locate the Connectors on the Starterkit

2.1.1 efusMX8X

The Starterkit includes all components that are required for an initial setup. This includes:

- Cables (Serial, power, USB …).
- Software (source, binaries, install scripts, examples).
- Starterkit carrier board that offers connectivity for most interfaces available in efus-MX8X.
- efusMX8X module.

For basic operation please make sure that power and Serial A debug port are connected correctly.

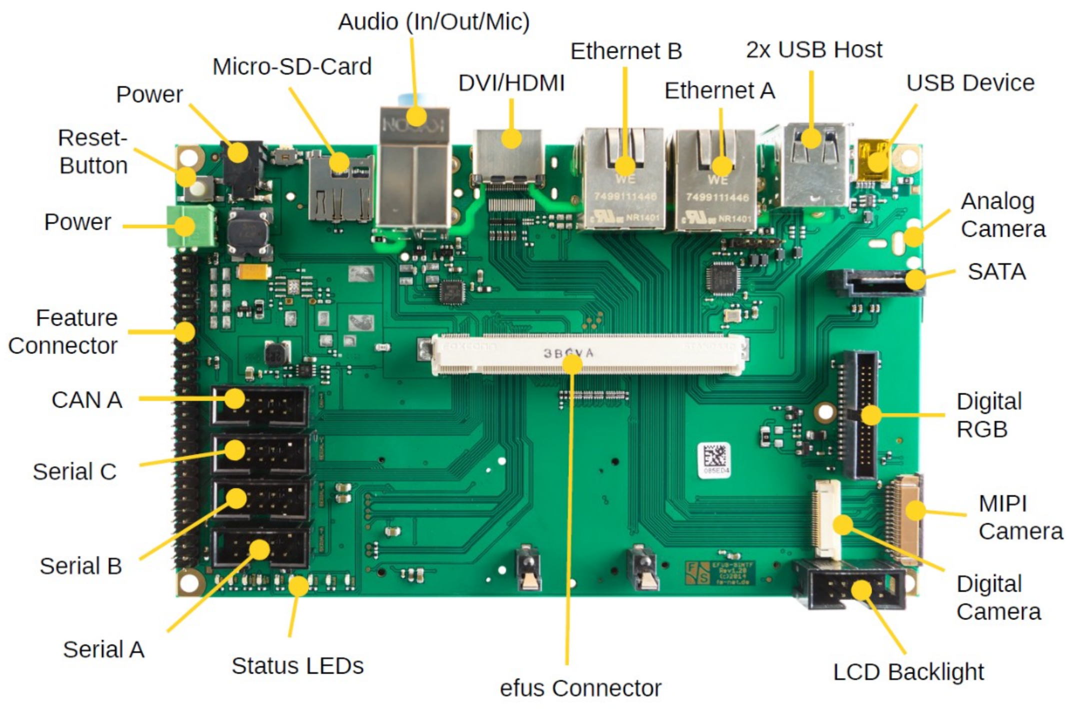


Figure 1: Top side of efusMX8X Starterkit baseboard (efusSKIT)

Figure 1 shows the connectors available on the top side of the efusMX8X SKIT carrier board.

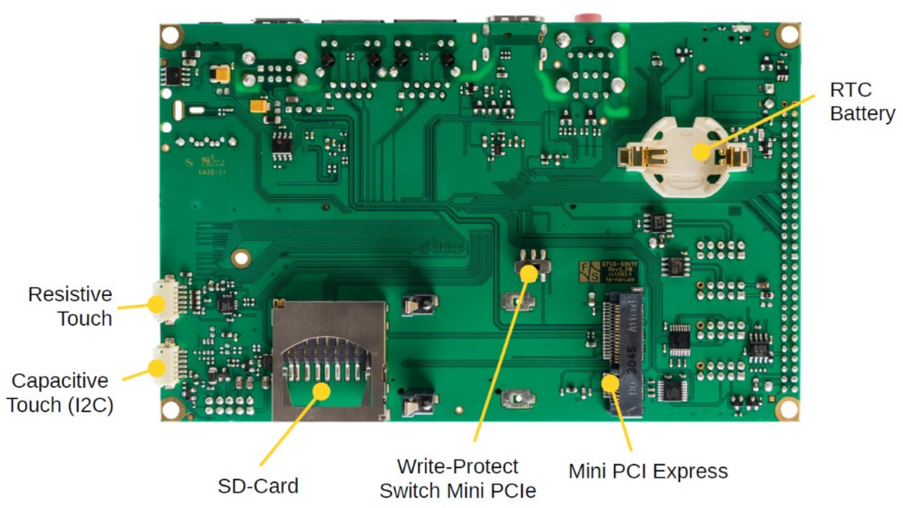


Figure 2: Bottom side of efusMX8X Starterkit baseboard (efusSKIT)

The connections available from the bottom side of the efusMX8X SKIT can be seen in Figure 2. Figure 3 shows the efusMX8X module itself.

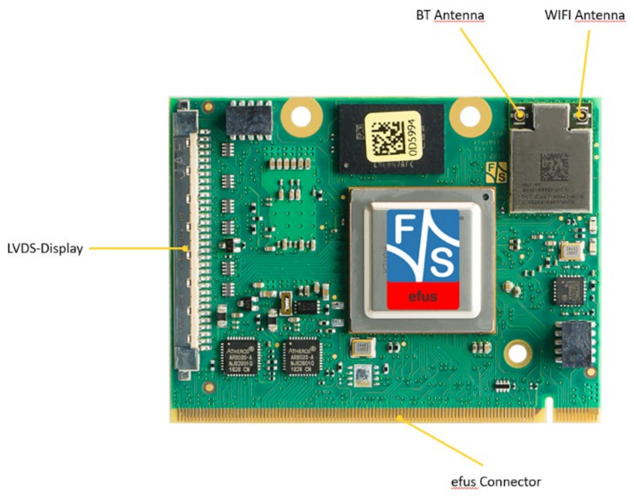


Figure 3: Top side of efusMX8X

2.2 Serial Connection

To work with the board, you need a serial connection with your PC. Use the provided Null-Modem cable and connect the debug port of the board (or Starterkit baseboard) with the serial port of a PC. Please refer to chapter 2.1 for the location of the COM ports. A serial port is mandatory on your PC, because we control the whole board via the serial port. If your PC does not provide a serial port, you have to either use a USB-to-serial adapter or you need to install a PCIe extension card with a serial port.

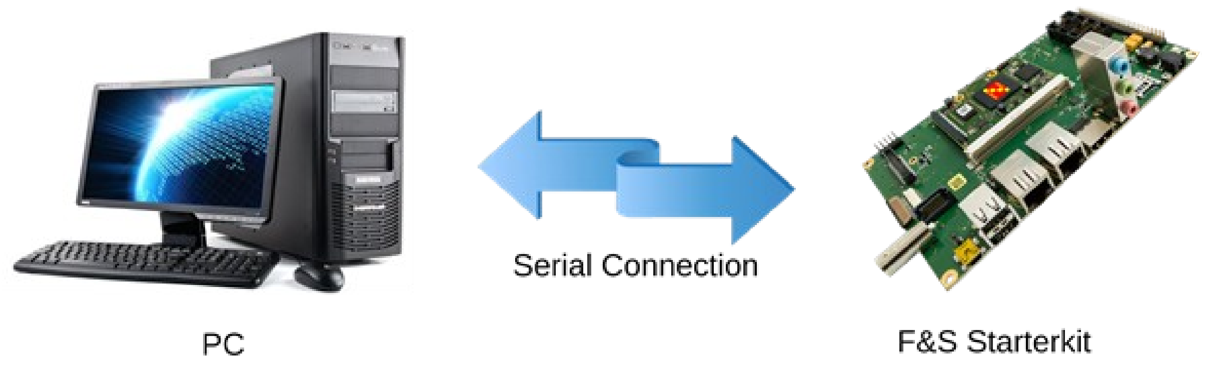


Figure 4: Serial connection from board to PC

For a first test, a Linux PC is not necessarily required. You can also use a Windows PC. But later for development, you definitely need a Linux PC, either native or as a Virtual Machine. With a Virtual Machine, you compile your software in Linux but you can still have the serial connection done in Windows and use tools from Windows. This uses the best of both worlds.

On your PC, start a terminal program and open a serial connection to the board. Use 115200 baud, 1 start, 1 stop bit, no flow control. We recommend a terminal program that supports a 1:1 binary download and also supports ANSI Escape Sequences for color and text highlighting. Examples are:

- `TeraTerm` (Windows)
- `PuTTY` (Windows/Linux, does not support 1:1 download)
- `minicom` (Linux, does not support 1:1 download, but not needed in Linux)

F&S also provides a small terminal program for Windows called `DCUTerm`. You can find `DCUTerm` in the Tools-Section of the Download Area (in My F&S). However `DCUTerm` does not support ANSI Escape Sequences, which means the output of a Linux command like `ls` is nearly unreadable. Instead of different colours for different file types, you will see a mixture of file names and verbatim escape sequences. Also accessing the command history with the up and down arrow keys is not possible in `DCUTerm`. So `DCUTerm` is not suited very well for Linux. However it supports a 1:1 binary download. So `DCUTerm` is actually a good companion for `PuTTY`. Use DCUTerm for serial downloads and `PuTTY` for everything else.

2.3 Start Board

Connect a power supply to the board. Please refer to chapter 2.1 for the location of the power supply pins. You need to supply +5V.

Now switch on the power supply. Quite immediately the terminal program should show boot messages from the booting Linux system. This will go on for a few seconds and then a login prompt should appear.

```
F&S i.MX Release Distro 5.4-zeus fsimx8x ttyLP2


fsimx8x login:
```

Enter `root` to log in. In the default configuration, no password is required.

If everything went well, you can skip the next chapter and proceed with entering Linux commands.

3 Software Installation

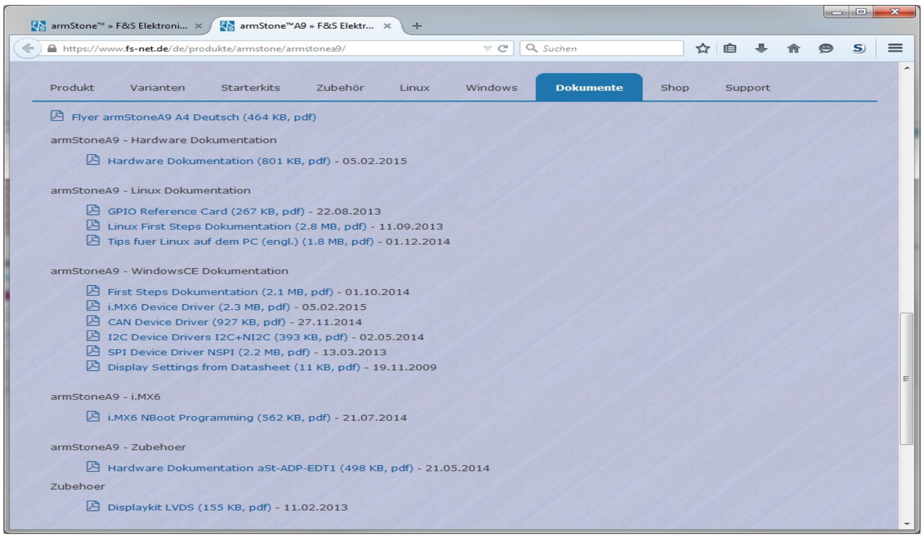
When you get a Starterkit from F&S, the Linux system is usually pre-installed and boots to the Linux login prompt right away. In this case you can skip this chapter. But if you are switching over from a different operating system, if you are upgrading from a previous release, or if your board is empty for some other reason, the following sections describe how to install some standard software on your platform.

Here we will only show a very simple automatic installation procedure using an SD card or USB stick and some pre-compiled images from the F&S website. This is the easiest way to get to a running system. Of course, there are other ways to install software, for example via network (TFTP). However, this would go beyond the scope of this First Steps document.

3.1 Download Images From F&S Website

To download any software, go to the F&S main website

https://www.fs-net.de


armStone™ » F&S Elektroni...
armStone™A9 » F&S Elektr...
https://www.fs-net.de/de/produkte/armstone/armstonea9/
Suchen
Produkt
Varianten
Starterkits
Zubehör
Linux
Windows
Dokumente
Shop
Support

Flyer armStoneA9 A4 Deutsch (464 KB, pdf)

armStoneA9 - Hardware Dokumentation

Hardware Dokumentation (801 KB, pdf) - 05.02.2015

armStoneA9 - Linux Dokumentation

GPIO Reference Card (267 KB, pdf) - 22.08.2013
Linux First Steps Dokumentation (2.8 MB, pdf) - 11.09.2013
Tips fuer Linux auf dem PC (engl.) (1.8 MB, pdf) - 01.12.2014

armStoneA9 - WindowsCE Dokumentation

First Steps Dokumentation (2.1 MB, pdf) - 01.10.2014
i.MX6 Device Driver (2.3 MB, pdf) - 05.02.2015
CAN Device Driver (927 KB, pdf) - 27.11.2014
I2C Device Drivers I2C+NI2C (393 KB, pdf) - 02.05.2014
SPI Device Driver NSPI (2.2 MB, pdf) - 13.03.2013
Display Settings from Datasheet (11 KB, pdf) - 19.11.2009

armStoneA9 - i.MX6

i.MX6 NBoot Programming (562 KB, pdf) - 21.07.2014

armStoneA9 - Zubehoer

Hardware Dokumentation aSt-ADP-EDT1 (498 KB, pdf) - 21.05.2014

Zubehoer

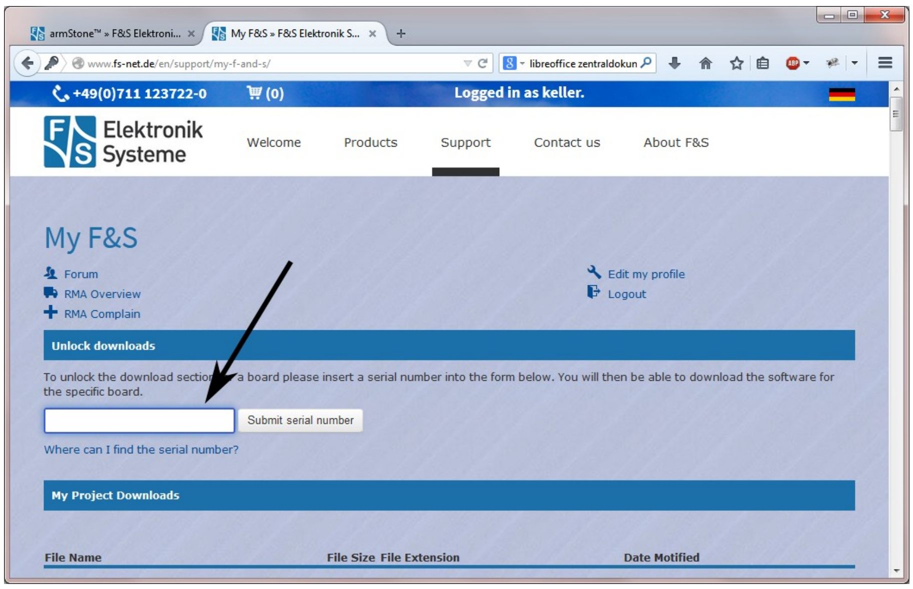
Displaykit LVDS (155 KB, pdf) - 11.02.2013

Figure 5: Register with F&S website

To download any software, you first have to register with the website. Click on *Login* right at the top of the window and on the text "I am not registered, yet. Register now" (Figure 5).

In the screen appearing now, fill in all fields and then click on *Register*. You are now registered and can use the personal features of the website, for example the Support Forum and downloading software.

After logging in, you are at your personal page, called "My F&S". You can always reach this place by selecting *Support → My F&S* from the top menu. Here you can find all software downloads that are available for you. In the top sections there are private downloads for you or your company (may be empty) and in the bottom section you will find generic downloads for all registered customers.



To unlock the download section of a board please insert a serial number into the form below. You will then be able to download the software for the specific board.

Figure 6: Unlock software with the serial number

To get access to the software of a specific board, you have to enter the serial number of one of these boards (see Figure 6). Click on “Where can I find the serial number” to get pictures of examples where to find this number on your product. Enter the number in the white field and press *Submit serial number*. This enables the software section for this board type for you. You will find Linux, Windows CE, and all other software and tools available for this platform like `DCUTerm` or `NetDCUUsbLoader`.

First click on the type of your board, e.g. efusMX8X, then on Linux. Now you have the choice of Buildroot or Yocto. For the first steps here, we will use the newest Yocto release, because this is the software that is also installed on our Starterkits. So click on Yocto. This will bring up a list of all our Yocto releases. Old releases up to 2018 had V<x>.<y> as version identifier, new releases use B<year>.<month>. We will abbreviate this as <v> from now on. Select the newest version, for example *fsimx8x-Y2020.07-pre*. This will finally show two archives that can be downloaded.

When you look at our Linux releases, you will find a list of all our releases and a README text. There are usually two files related to a release.

`fsimx8x-<v>-pre.tar.bz2.`	. This is the prerelease itself containing all sources, the binary images, the documentation and the toolchain.
`sdcard-fsimx8x-<v>.tar.bz2`	Files that can be stored on an SD card or USB stick to allow for easy installation.

For now we will only need the SD card archive. This archive contains some pre-compiled images of bootloaders, Linux kernel, device trees and root filesystem. It is compressed with bzip2. To see the files, you first have to unpack the archive, for example in Linux with

```
tar xvf fsimx8x-<v>-pre.tar.bz2
```

This will create a directory `sdcard` that contains all necessary files. Now copy these files to an SD card or USB stick. We will call this the installation media. It has to be formatted with the FAT filesystem. Do not create any subdirectories, the files have to reside directly in the top directory of the media.

3.2 Enter UBoot

The efusMX8X supports the new NBoot concept for the i.MX8 series. The SPL loads the NBoot, which is a bundle of configurations and firmware images necessary for the board to boot. The NBoot for the I.MX8 series currently has no context menu, so the first bootloader that can be entered is the UBoot. To enter the UBoot it requires the serial setup as explained in Chapter 2.2. After that we can switch on power of the board, you should see something like this (output is taken from efusMX8X, the real messages may vary slightly depending on the software version):

```
U-Boot 2020.04 (Aug 10 2022 - 09:45:27 +0000)

CPU:   NXP i.MX8QXP RevC A35 at 1200 MHz at 37C
Model: efusMX8X
Board: efusMX8X Rev 1.10 (2x LAN, eMMC, 1x DRAM)
DRAM:  1022 MiB
MMC:   FSL_SDHC: 0, FSL_SDHC: 1
Loading Environment from MMC... OK
[*]-Video Link 0 (800 x 480)
        [0] dpu@56180000, video
        [1] lvds-channel@0, display
In:    serial
Out:   serial
Err:   serial
flash target is MMC:0
Net:   eth0: ethernet@5b040000, eth1: ethernet@5b050000
Fastboot: Normal
Normal Boot
Hit any key to stop autoboot:  0
=>
```

You have to hit any key within 3 seconds to enter UBoot.

3.3 Install Kernel, Device Tree and Root Filesystem

Insert the installation device into the board or Starterkit baseboard. The remaining installation is fully automatic and is done by U-Boot. Reboot the system e.g. switch off power, wait a few seconds and switch on power again. This will show something like this:

```
U-Boot 2020.04 (Aug 10 2022 - 09:45:27 +0000)

CPU:   NXP i.MX8QXP RevC A35 at 1200 MHz at 37C
Model: efusMX8X
Board: efusMX8X Rev 1.10 (2x LAN, eMMC, 1x DRAM)
```

```
DRAM:   1022 MiB
MMC:    FSL_SDHC: 0, FSL_SDHC: 1
Loading Environment from MMC... OK
[*]-Video Link 0 (800 x 480)
        [0] dpu@56180000, video
        [1] lvds-channel@0, display
In:     serial
Out:    serial
Err:    serial
flash target is MMC:0
Net:    eth0: ethernet@5b040000, eth1: ethernet@5b050000
Fastboot: Normal
Normal Boot
Hit any key to stop autoboot:  0
=>
```

The number in the last line will count down to zero, then the installation procedure will start. The files are loaded from the installation media and are stored in nand flash on the board. When the installation is over, you will see the following line

```
Installation complete

Please set/verify Ethernet address(es) now and call saveenv
```

3.4 Set MAC Address

When we erased the U-Boot environment including the MAC address for the Ethernet chip. We have to set it again now and save it permanently.

The MAC address is a unique identifier for a network device. Each network device has its own address that should be unique across the whole world. So each network port on each board needs a unique MAC address.

A MAC address consists of twelve hexadecimal digits (0 to 9 and A to F), that are often grouped in pairs and separated by colons. The first six digits for F&S boards are always the same: 00:05:51, which is the official MAC address code for the F&S company. The remaining six digits can be found on the bar-code sticker directly on your board (see Figure 7).

07934B

Figure 7: Barcode sticker

The full MAC address for this example would be 00:05:51:07:93:4B. If your board supports two ethernet ports, you need two MAC addresses. The second one is the first one plus 1, i.e. 00:05:51:07:93:4C.

The following two commands will set the MAC addresses and stores the current environment (including the newly set MAC addresses) in NAND flash. Of course you have to replace `xx:yy:zz` with the six hex digits from the bar-code sticker on your board (and xx:yy:vv with the six hex digits plus 1).

```
=> setenv ethaddr 00:05:51:xx:yy:zz
=> setenv eth1addr 00:05:51:xx:yy:vv
=> saveenv
```

Warning

If you do not set this unique address, a default address is used that is the same for all boards of this type. This will definitely lead to problems in real networking scenarios.

3.5 Restart Board

Installation is complete. To check if everything was done correctly, restart the board. You can either enter U-Boot command, …

```
reset
```

… or press the reset button, or simply switch the power off and on again. Like in chapter 2.3, the terminal program should show boot messages from the booting Linux system. This will go on for a few seconds and then a login prompt should appear.

```
F&S i.MX Release Distro 5.4-zeus fsimx8x ttyLP2


fsimx8x login:
```

Enter `root` to log in. In the default configuration, no password is required. If this is still not working, you should repeat the steps from the whole chapter.

4 Using the Standard System and Devices

By default, the standard root filesystem is mounted read-only. Therefore you cannot create files unless you go to a directory like `/tmp` that is located in a RAM disk. This is to make the system as stable as possible. If the root filesystem is mounted read-only, it is usually no problem to just switch off the power.

If you want to remount the filesystem in read-write mode, just say

```
mount -o remount,rw /
```

Note the slash `/` that is denoting the mount point of the root directory. Now you can create files everywhere. But remember that written data is often buffered in RAM first and is not immediately stored on the media itself. If you simply switch off the power now, data that was still buffered in RAM may be lost. So in this case it is really important to actually shut down the system with ...

```
halt
```

… or restart with:

```
reboot
```

Or you can remount the root filesystem in read-only mode again after applying the changes:

```
mount -o remount,ro /
```

All these commands will force the system to actually write any buffered data to the media.

The `/dev` directory is also built on top of a RAM disk. This allows the kernel to create and remove device entries dynamically. For example if a USB stick is plugged in, a new device entry `/dev/sda1` is automatically created. And when the USB stick is unplugged, the device entry is also automatically removed again.

4.1 Procfs

Linux has a virtual filesystem called Procfs. It is mounted under `/proc` and provides information about the system in general and about each process that is currently active.

Get information about the CPU

```
cat /proc/cpuinfo
```

Show the Linux version

```
cat /proc/version
```

Show the current memory usage

```
cat /proc/meminfo
```

List the supported filesystems

```
cat /proc/filesystems
```

4.2 Sysfs

Sysfs is another virtual file system in Linux. It exports information about devices and drivers from the kernel device model to user space. Which means you can get information about current device settings and some drivers even allow configuring the device at runtime.

Devices that want to share information or want to accept configuration settings, create subdirectories under the `/sys` directory. Then virtual text files are used to pass the information. So for example if a touch panel can accept some sensitivity configuration, it would create a file `sensitivity` there. By reading data from the file, you could query the current setting. And by writing a new value to the file, you could set a new sensitivity value.

For example access the RTC subsystem:

```
cat /sys/class/rtc/rtc0/date
```

Show the CPU core temperature (in 1/1000 °C):

```
cat /sys/class/thermal/thermal_zone0/temp
```

4.3 SD Card

Besides the size (regular SD and Micro SD) there are also two types of SD card slots: slots with and without a Card Detect (CD) signal. Slots without a CD pin can only be used for non-removable media. So the card is detected only if it is present at boot time. If it is inserted later, it is not detected anymore. Slots with a CD pin are meant for removable media. They can detect at runtime when a card is inserted or ejected.

If an SD card is detected in the system, a device `/dev/mmcblk0` is created. This device represents the whole card content. If the device also holds a partition, an additional device `/dev/mmcblk0p1` is created. This device represents this single partition only.

You can mount and unmount the card now. For example to mount the card on directory `/mnt`, you have to issue the following command:

```
mount /dev/mmcblk1p1 /mnt
ls /mnt
```

Now you can work with the device. Later, when you are done, you can unmount it again with

```
umount /mnt
```

4.4 USB Stick (Storage)

If a USB memory stick is inserted, it is available like a standard hard disk. Because there is usually no real hard disk connected, it is found as `/dev/sda`. If you have partitions on your USB stick, you have to access them as `/dev/sda1`, `/dev/sda2` and so on.

You can mount and unmount all the partitions now. For example to mount the first partition on directory `/mnt`, you have to issue the following command:

```
mount /dev/sda1 /mnt
```

To unmount again, issue:

```
umount /mnt
```

Remarks

If a USB storage device contains more than one partition, a device entry will be created for each partition, for example also `/dev/sda2` and `/dev/sda3`. In fact also a device entry `/dev/sda` without a number is available, that refers to the whole device, including all partitions. Some USB storage devices do not contain a partition table at all. Then only `/dev/sda` is created.

If more than one storage device is plugged in (for example via a USB hub), then the second device has the base name `sdb`, the third `sdc`, and so on.

4.5 Qt5

Qt is a cross-platform application framework that is widely used for developing applications, often with a graphical user interface (GUI). The standard root filesystem does not include Qt, but you can use the `fsimx8x_qt5_wayland_defconfig` in Buildroot to build a root filesystem based on Qt5.

4.6 Wayland/Weston

The standard root filesystem automatically starts Wayland. Wayland is a GUI protocol that replaces X11. Wayland itself only manages the protocol and the communication to the clients. It needs an additional component, the so-called compositor. This compositor works like a Window Manager and actually presents the graphical output on the display.

The standard root filesystem does include Wayland. The standard configuration `fsimx8x_std_wayland_defconfig` in Buildroot can be used to build a root filesystem based on Wayland and the Weston compositor.

4.7 Pictures

There is a small image viewer program included called `fbv`. Just call it with the list of images to show. The images will not use a window of the GUI, they are rendered directly into the framebuffer. So `fbv` will also work without a GUI. The following command will show a new image every five seconds.

```
fbv -s 50 /usr/share/pixmaps/*.png
```

To show possible program options use:

```
fbv --help
```

If you want to use a different framebuffer, set the environment variable `FRAMEBUFFER` accordingly. For example to use `/dev/fb2`, use the following command, before calling `fbv`.

```
export FRAMEBUFFER=/dev/fb2
```

4.8 Sound

You can use standard ALSA tools to play and record sound. There is a tool to test the sound output.

```
speaker-test -c2 -p 1 -t wav
```

This will say "Front left" and "Front right" on the appropriate line out channel. If you have a WAV file to play, you can use this command:

```
aplay <file.wav>
```

If you have an MP3 file, you can play it back with:

```
madplay <file.mp3>
```

To record a file from microphone in (mono), just call

```
arecord -c 1 -r 8000 -f s16_le -d <duration> <file.wav>
```

To record a file from line in, you first have to switch recording from microphone to line in. This can be done with

```
amixer sset 'Capture Mux' LINE_IN
```

Then record the stereo file with high quality with

```
arecord -c 2 -r 48000 -f s16_le -d <duration> <file.wav>
```

To see what other controls are available, call `amixer` without arguments:

```
amixer
```

For a rather rudimentary graphical mixer representation of all the mixer controls, call

```
alsamixer
```

You can use the left and right cursor keys to select the control and then up and down cursor keys to change the appropriate value. Some switches also need key *M* to toggle mute mode. Key *H* shows help for all available commands.

4.9 Multimedia (gstreamer support)

Modules based on fsimx8x support hardware accelerated video processing in `gstreamer`. Table 3 holds a list of all iMX8X specific gstreamer1 plugins that use hardware acceleration.

Plugin name	Description
`v4l2src`	Video4Linux source (e.g. camera)
`aiurdemux`	Multi-format demultiplexer for several video formats

Plugin name	Description
`imxvideoconvert_g2d`	IMX g2d video converter (pixel format, rotation)
`imxcompositor_g2d`	IMX g2d video compositor (combine several videos, add background)
`v4l2sink`	Video4Linux video sink (overlay support)
`waylandsink`	Wayland video sink
`beepdec`	Audio decoder for several audio formats
`vpu_enc_h264`	IMX VPU-based AVC/H264 video encoder
`vpuenc_vp8`	IMX VPU-based VP8 video encoder
`vpudec`	IMX VPU-based video decoder

Table 3: VPU based GStreamer plugins

Of course also a lot of regular gstreamer plugins are availabe. The following command will show all available plugins and supported video and audio formats. The output is rather lengthy.

```
gst-inspect-1.0
```

To get more information about a specific item, just add it after the command. For example to get information about the v4l2sink, enter:

```
gst-inspect-1.0 v4l2sink
```

Check if the video path is working:

```
gst-launch-1.0 videotestsrc ! fbdevsink
```

This command renders a kind of test image directly into the framebuffer. Use `v4l2sink` to render into a screen overlay. You can modify the test image by adding `pattern=<n>` to `videotestsrc`, where `<n>` is a number of 0 to 24. For example …

```
gst-launch-1.0 videotestsrc pattern=18 ! fbdevsink
```

… will show a moving ball on the screen.

Check if the audio path is working:

```
gst-launch-1.0 audiotestsrc ! alsasink
```

This will output a tone made of a sine curve. You can modify the output by adding `wave=<n>` to `audiotestsrc`, where `<n>` is a number of 0 to 12. This will use a different curve or use noise instead. For example …

```
gst-launch-1.0 audiotestsrc wave=9 ! alsasink
```

… will output White Gaussian Noise.

The easiest way to play a video or audio file is by using the `playbin` filter.

```
gst-launch-1.0 playbin uri=<name>
```

You have to give the file name in URI-style, i.e. prepend `file:` and use the absolute path. For example if you are working as user `root`, who has his home directory in `/root`, and you have an MPEG-2 video called `video.mpg` there, then call:

```
gst-launch-1.0 playbin uri=file:/root/video.mpg
```

`playbin` will automatically select a suitable set of filters and plugins to play the file.

Please note that most video sinks do not accept interlaced videos. If your video is in interlaced format, you have to add a flag to tell playbin to deinterlace the video when playing. The flags value is a combination of several flags. The default value is 0x17, the flag for deinterlacing is 0x200. This is the final command:

```
gst-launch-1.0 playbin uri=file:/root/video.mpg flags=0x217
```

You can also give your own list of filters, which is slightly more complicated. For example to just play the video part `video.mpg` this would be:

```
gst-launch-1.0 filesrc location=video.mpg ! mpegpsdemux \
    ! mpegvideoparse ! mpeg2dec ! deinterlace ! videoconvert \
    ! fbdevsink
```

To play an audio file `sound.wav`:

```
gst-launch-1.0 filesrc location=sound.wav ! wavparse ! Alsasink
```

4.10 Ethernet

To activate the Ethernet port in Linux, you have to configure the network device first. For example to use IP-Address 10.0.0.242, you can use the command

```
ifconfig eth0 10.0.0.242 netmask 255.0.0.0 up
```

Then you can use network commands, e.g.

```
ping 10.0.0.121
```

There is also a DHCP client included. To receive an IP address via DHCP just call:

```
udhcpc
```

If your board supports more than one network interface, you can add option `-i` to specify the appropriate interface. For example to request IP data for interface `eth1`:

```
udhcpc -i eth1
```

4.11 WLAN

Wireless LAN is available on some fsimx8x boards (e.g. efusMX8X). Configure the WLAN adapter for your needs by using the `ifconfig` or `ip` program. Use program `wpa_supplicant` to set up all parameters required for WLAN access, like data encryption and login information.

The data for known WLAN connections is kept in the file `/etc/wpa_supplicant.conf`. If the root filesystem, is in read-write-mode, you can append the access information for a new network to this file. For example if the SSID of the network is `My WLAN` and the passphrase is `My secret access code`, then you can issue:

```
wpa_passphrase "My WLAN" >> /etc/wpa_supplicant.conf
My secret access code
```

From now on, `wpa_supplicant` will automatically connect to this network if it is in range.

Start the `wpa_supplicant`:

```
wpa_supplicant -B -D nl80211 -i mlan0 -c /etc/wpa_supplicant.conf
```

`wpa_supplicant` stays active in the background and handles all WLAN tasks. For example it scans all WLAN networks in range and checks if a known network is among them. After a few seconds, the WLAN connection should be established. Then you can start the network similar to an Ethernet interface, for example with:

```
udhcpc -i mlan0
```

There is a special program called `wpa_cli` that can talk to the running `wpa_supplicant` process. A single command can be given as argument to `wpa_cli`. For example to show the list of currently available WLANs, call:

```
wpa_cli scan_results
```

Calling `wpa_cli` without arguments will go to interactive mode. Now you can enter `wpa_supplicant` commands directly, without having to type `wpa_cli` before each line.

The list of `wpa_supplicant` commands, that you can show by entering `help`, is rather long, way too much to handle it here. When you are done, type `quit` to leave interactive mode again.

4.12 Bluetooth

Bluetooth is available on some modules. Buildroot offers support for the official Linux Bluetooth protocol stack.

Start the Bluetooth daemon:

```
/usr/libexec/bluetooth/bluetoothd --compat &
```

There are two programs that can talk to the HCI of the Bluetooth device: `hciconfig` to configure the Bluetooth settings on the local side and `hcitool` to handle the communication itself. The following command powers up the Bluetooth device:

```
hciconfig hci0 up
```

Now Bluetooth is generally working. For example to scan for Bluetooth devices nearby, you can issue.

```
hcitool scan
```

More information to the BlueZ stack can be found at http://www.bluez.org .

4.13 TFTP

There is a small program to download a file from a TFTP server. This can be rather useful to get some files to the board without having to use an SD card or a USB stick. For example to load a file `song3.wav` from the TFTP server with IP address 10.0.0.121, just call

```
tftp -g -r song3.wav 10.0.0.121
```

4.14 Telnet

If you want to use `telnet` to login from another PC, you have to start the telnet daemon

```
telnetd
```

However as this service is considered insecure, telnetd does not allow to log in as root. So you have to add a regular user, for example called “telnet”.

```
adduser -D telnet
passwd -d telnet
```

The first commands adds the user “telnet” and the second command sets an empty password for this user.

Now you can log in from another PC with username telnet:

```
telnet <ipaddr>
```

4.15 SSH

Buildroot

You can connect to your device by SSH. But then you need to set a password for your root user

```
adduser <username>
```

or create a new user by:

```
adduser <username>
```

Now you can connect via SSH by any host in the network with:

```
ssh <username>@<device-ip>
```

If for some reason the keys are expired you can calculate new ones. Therefore old keys have to be removed.

```
cd /etc
rm ssh_host_*
cd /etc/init.d
./S50sshd
```

Be careful with `rm ssh_host_key_*`. Just remove the files with the word `ssh_host_ key` in it. After that the startup script `S50sshd` should be executed again

```
/etc/init.d/S50sshd restart
```

Note

Please note that date and time must be valid on the board or login attempts with `ssh` will fail.

The script S50sshd will only create new encryption keys if the directory /etc is writable.

So if your rootfs is read-only, you have to remount it as read-write first before calling the script.

Yocto

In Yocto ssh is configured to allow root login and empty-password by default. Just connect via SSH by any host in the network with:

```
ssh <username>@<device-ip>
```

You can change these settings at:

```
vi /etc/ssh/sshd_config
```

4.16 VNC

If you have no display attached or you want to connect by remote you can start the pre-installed `x11vnc` program on your device by:

```
x11vnc
```

On any host in the network install a vnc-viewer program and connect to your board with:

```
vncviewer <device-ip>:0
```

4.17 Serial

On NXP CPUs with low power UARTs, the devices are called `/dev/ttyLP<n>`, where `<n>` is a number starting with 0. One port is usually used as serial debug port where all console messages are sent to. This one port runs at 115200 bit/s. All other ports are at 9600 bit/s by default. Use the `stty` program to change this.

To access a serial port from the command line, you can use input and output redirection.

Show a string on port `ttyLP0`:

```
echo Hello > /dev/ttyLP0
```

Show characters that arrive on port `ttyLP0`:

```
cat < /dev/ttyLP0
```

Usually character input is line buffered, so you will only see information after sending Return.

Remark

The default setting for serial ports in Linux echo each character that arrives. So if you connect one serial port to a second serial port with a Null-Modem cable, sending a single character will result in an endless loop. Each side will echo the character indefinitely. You can use `stty` to change this behaviour.

4.18 I²C

Most devices on an I²C bus are accessed by a device driver in Linux. But you can also have access to additional devices from userspace software. There are even command line tools to do this. The following examples are from an efusMX8.

Show the available I²C buses:

```
i2cdetect -l
i2c-3   i2c            5a830000.i2c                   I2C adapter
i2c-1   i2c            5a810000.i2c                   I2C adapter
i2c-2   i2c            5a820000.i2c                   I2C adapter
i2c-16  i2c            56226000.i2c                   I2C adapter
```

Show the available devices on bus `i2c-1`:

```
i2cdetect -y 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- -
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -
30: -- -- -- -- -- -- -- -- UU -- -- -- -- -- -- -
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -
70: -- -- -- -- -- -- -- --
```

Some devices are handled by a Linux driver, so they can not be accessed from user space (therefore marked as “used” with `UU`). Now you can read and write data with `i2cget` and `i2cset`, or read whole data areas with `i2cdump`.

4.19 SPI

SPI devices are often directly accessed by device drivers from the kernel. However an SPI device can also be made available to user space via the usermode SPI driver called `spi-dev`. Such devices are available as `/dev/spidev<n>.<m>`, where `<n>` is the SPI bus number and `<m>` is the chip select number on this bus.

4.20 GPIO

You can setup and use GPIOs with the Sysfs system.

```
ls /sys/class/gpio
export        gpiochip0@    gpiochip128@ gpiochip160@
gpiochip192@ gpiochip224@ gpiochip32@   gpiochip64@
gpiochip96@   unexport
```

Please refer to the according “GPIO Reference Card“ document to know how the pins of the board correspond with the Sysfs-GPIO system.

Example

In preparation for the example, the device-tree efusmx8x.dts must be modified by uncommenting the line “#define CONFIG_EFUSMX8X_ADC”. Generally there are no standard GPIO pins. To gain access to GPIO pins, interfaces must be disabled by uncommenting their define in the device-tree. All disabled interfaces will be set to GPIO.

On efusMX8X, use pin RESERVED1 / ADC_IN0 on feature connector as output pin. This is pin 36 on the EFUS-SINTF connector J1, also available on pin 15 of the feature connector J22 on the EFUS-SINTF. The “GPIO Reference Card” for efusMX8X in the row for pin 36 shows in column `/sys/class/gpio/gpio#` the number 42. To get this pin into sysfs write:

```
echo 42 > /sys/class/gpio/export
```

This creates a new directory `gpio42` in `/sys/class/gpio` that is used for all further settings of this GPIO.

```
ls /sys/class/gpio/gpio42/
active_low  direction   edge        power       subsystem
uevent      value
```

Set pin as output:

```
echo out > /sys/class/gpio/gpio42/direction
```

Set pin to high level:

```
echo 1 > /sys/class/gpio/gpio42/value
```

Now the pin should have a high level (about 3.3V) which you can measure with a voltmeter. To set pin low again type:

```
echo 0 > /sys/class/gpio/gpio42/value
```

Now pin has a low level. To set a pin as input, write `in` into `direction`. Then you can read the current value with

```
cat /sys/class/gpio/gpio42/value
```

4.21 RTC

Setting date:

```
date "2015-11-29 22:55"
```

Save current date to RTC:

```
hwclock -w
```

The time will automatically be loaded from the RTC at the next boot.

Note

Make sure VBAT is connected to the module. Otherwise the RTC can not keep time.

5 Next Steps

This document only showed a very basic usage of the board and the Linux system. The next logical step is the generic Linux documentation `LinuxOnFSBoards_eng.pdf`. It will show you the ideas and concepts behind the F&S Linux environment and how you can work efficiently with these boards.

5.1 F&S Workshops

F&S also offers several workshops. Especially if you are new to working with embedded boards or even new to Linux, we recommend visiting the workshop “Linux on F&S Modules”. Working with an embedded system is quite different to working with a desktop Linux. This workshop will show you a basic introduction to Linux, how to use NBoot, U-Boot and Linux on an F&S board, how to compile the system software, how to download files to the board, and how to write your own programs. The workshop lasts four hours and takes place in Stuttgart at the F&S company building. It may save you many hours of reading, trying, and even frustration.

Additional workshops are available for working with Buildroot, Asymmetric Multiprocessing, Secure Boot, Working with GIT. Please look at our website for any additional offerings.

5.2 Further Information

Many additional resources of information are available on the F&S website.

Document	Description
`AdvicesForLinuxOnPC.pdf`	Explains how to install server software and tools on a Linux development PC that is used with F&S Linux boards.
`*-GPIO-Reference-Card_eng.pdf`	Lists all pins of the board and which GPIO number needs to be used in Linux
`*_Hardware_eng`	Hardware documentation; there are separate documents for each board and also for the Starterkit baseboards. F&S also offers Eagle layout files for some of our Starterkits.

Table 4: Important documents, available on the F&S website

We do not include all these documents in the release to make sure that you always get the newest version when you start. The following sections give direct links to important places like documentation and add-ons.

A good source for information is also our internet forum. If you have any questions or specific problems, please feel free to go to: https://forum.fs-net.de/.

5.2.1 Resources for efusMX8X

Hardware documentation for efusMX8X module itself, Starterkit baseboard, including schematics:

https://www.fs-net.de/en/embedded-modules/computer-on-module-efus/efusmx8x-with-nxp-i-mx-8x-processor#panel-6

Available accessories, adapters and extensions:

https://www.fs-net.de/en/embedded-modules/computer-on-module-efus/efusmx8x-with-nxp-i-mx-8x-processor#panel-4

6 Appendix

List of Figures

List of Tables

Listings

Important Notice