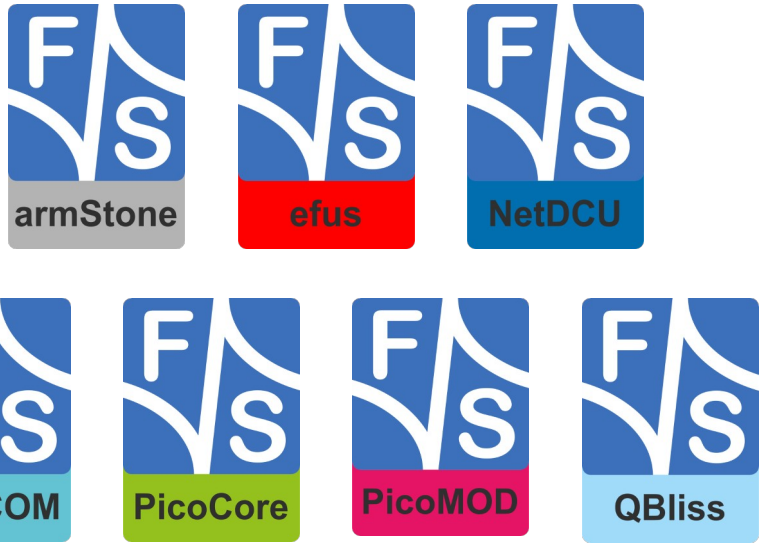
F&S i.MX6 Linux

First Steps

Version 4.3
(2022-02-01)



Elektronik Systeme

Untere Waldplätze 23
D-70569 Stuttgart
Germany

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

About This Document

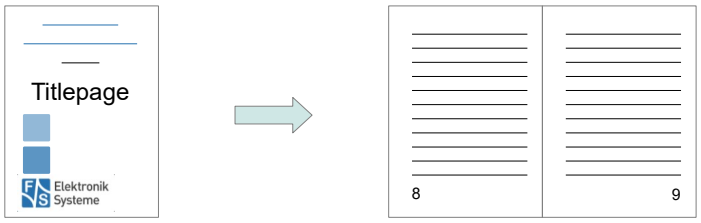
This document shows how to bring up F&S boards and modules under Linux and how to use the system and the devices. It is intended to help you get to know the board so that you can take the first steps to try out and test all the peripherals.

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
2012-09-05	0.1	*	A,M,R	*	Initial Version	MK
2012-11-13	0.2	*	A,M,R	*	Additional modifications. First public version	MK
2012-11-12	0.3	*	A,M,R	*	Buildroot description added. Rootfs and Kernel flashing de-scribtion added.	MK
2013-01-29	0.4	QBA9	M	8	Toolchain path corrected	MK
2013-03-15	0,5	QBA9	A, M	5.2, 11	MTD partitions updated. Chapter about oading and saving firmware added.	MK
2013-03-20	0.6	QBA9	M	10	Kernel build command corrected.	MK
2013-05-31	0.7	QBA9	A	7.9	Bluetooth initialization	MK
2013-06-27	0.8	*	A	5.3.1	Creating a new bootable SD card	MK
2013-09-06	1.0	*	A,M,R	*	armStoneA9 added	MK
2013-12-06	1.1	*	M	4.1.3		JG
2015-03-06	1.2	*	M,R	1-4, 8-11	Updates in 1-4 and 8-11: improved introduction, explain archi-tecture releases, new F&S Download Area, improve manual installation procedure, introduce the new system for switching boot strategies. Modified some names in U-Boot, Kernel and Buildroot. Toolchain name changed to one of current versions.	AD
2015-03-11	1.3	*	M	8	Make sub directory “arm” added. Definition of “CROSS_COM-PILE” environment added.	AD
2015-04-14	1.4	*	A,M,R	2,4,5,6,7	Added PicoMODA9, Infos for new NBoot, new UBoot. Boot strategies.	AZ
2015-04-23	2.0		M	*	Converted to new F&S document layout, large parts rewriten	HK
2015-04-24	2.1		M	7.1	Fix typo in PATH for toolchain	HK
2016-06-20	3.0				Add device tree info, Move generic explanations about the Linux system from install instructions to general introduction, update images, reduce similar/redundant explanations	PH
2016-08-08	3.0		A, M	2.1.3, 2.3, 3.3.2, 7.9	Add armStoneA9r2 image, complete list of files, move erasing the flash to a separate sub-chapter, update gstreamer sup-port, many small changes.	HK
2016-08-10	3.1		M	8.4	Fix device tree name confusion	HK
2016-08-19	3.1	*	A,M,R	*	Address differences between multiple documents	PH
2016-09-14	3.1		A, M	2.3, 3.3.1, 3.3.6, 7.15, 8.2, 8.5	Change NBoot upgrade info, formatting, add mkimage tool in-stallation	PH
2017-06-28	3.2	*	A,M,R	7.*, 8	Add chapter 8 ‘Graphical Environment’ merge similar subchpters and remove them	PH
2017-11-14	3.2	*	M	6.2.* 9.3	Command should be nand erase.part Explain new version of install-sources.sh with option --dry-run	PH HK
2018-05-25	3.3	*	M M M A M M M A	1 2 2.1.5 2.1.6 2.2, 2.3 3.3 9.1 9.2	Add NetDCU platform Add reference to F&S development machine Use same description type on PicoMODA9 as for other boards Add NetDCUA9 Mention Yocto releases, add Yocto file names to content list Change manual installation procedure to use file names from sdcard directory; they are the same for Buildroot and Yocto Collect Buildroot build process chapters under 9.1 Add chapter 9.2 with Yocto build process	HK HK HK HK HK HK HK HK
2018-05-30	3.4	*	M M	1 9	Add PicoCOM1.2 to available platforms; have nicer pagebreak Fix typos, minor improvements	HK HK
2018-09-11	3.4	QBlissA9/ QBlissA9r2	A	5.4	Show workarounds for slow network connections on QBlissA9 and QBlissA9r2 due to SKIT not capable of Gigabit speeds	HK
2019-03-22	3.4	fsimx6	M A R	1 1.1 – 1.3 2.3	Add PicoCoreMX6UL and PicoCoreMX6SX to platforms, drop fss5pv210 and picocom4 architectures, add fsimx7ulp Split into chapters, add chapter with links to F&S website Remove hardware documents from list of contents	HK HK HK

			M M	2.2, 2.3, 3.2, 9.1 2.3, 3.3, 5.1, 5.3, 6.6, 9.1, 9.2	Use generic <v> for versions in most places, explain old & new version numbers Several replacements: uImage → zImage, fs-toolchain-5.2.0 → fs-toolchain-7.4, u-boot-2014.07 → u-boot-2018.03, linux-4.1.5 → linux-4.9.88, buildroot-2016.05 → buildroot-2019.02, 10800000 → 11000000, 11000000 → 12000000	HK HK
2019-07-25	4.0	fsimx6	M	*	Move large parts to LinuxOnFSBoards. Reduce to startng the board, automatic installation if software is not installed and using the peripherals. Use new board-logos on title page.	HK
03/20/20	4.1	fsimx6	M	4.18	Add distinguish between Yocto and Buildroot	PG
10/15/21	4.2	fsimx6	M	1.1 4.19	Add new architectures and boards Add Infobox	PG
02/01/22	4.3	fsimx6	M A	1.1 2.1.5	Add efusA9r2 Add efusA9r2	PJ

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 different 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, efusA9r2, QBlissA9, QBlissA9r2, armStoneA9, armStoneA9r2, PicoMODA9, NetDCUA9
fsimx6sx	NXP i.MX6-SoloX	efusA9X, PicoCOMA9X, PicoCoreMX6SX
fsimx6ul	NXP i.MX6-UL/ULL	efusA7UL, PicoCOM1.2, PicoCoreMX6UL, PicoCoreMX6UL100
fsimx7ulp	NXP i.MX7ULP	PicoCoreMX7ULP
fsimx8mm	NXP i.MX8M-Mini	PicoCoreMX8MM
fsimx8mn	NXP i.MX8M-Nano	PicoCoreMX8MN
fsimx8mp	NXP i.MX8M-Plus	PicoCoreMX8MP

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 *fsimx6* architecture. That means all F&S boards and modules based on the NXP i.MX6 SoC (Solo, DualLite, Dual, Quad). 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 this 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 *fsimx6* architecture and show the location of all connectors, especially the debug port.

2.1 Locating the Connectors on the Starterkit

2.1.1 QBlissA9/QBlissA9r2

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

- Cables (ethernet, serial, power, USB, …).
- Software (source, binaries, install scripts, examples).
- Starterkit carrier board that offers connectivity for most interfaces available in QBlissA9.
- QBlissA9 module.

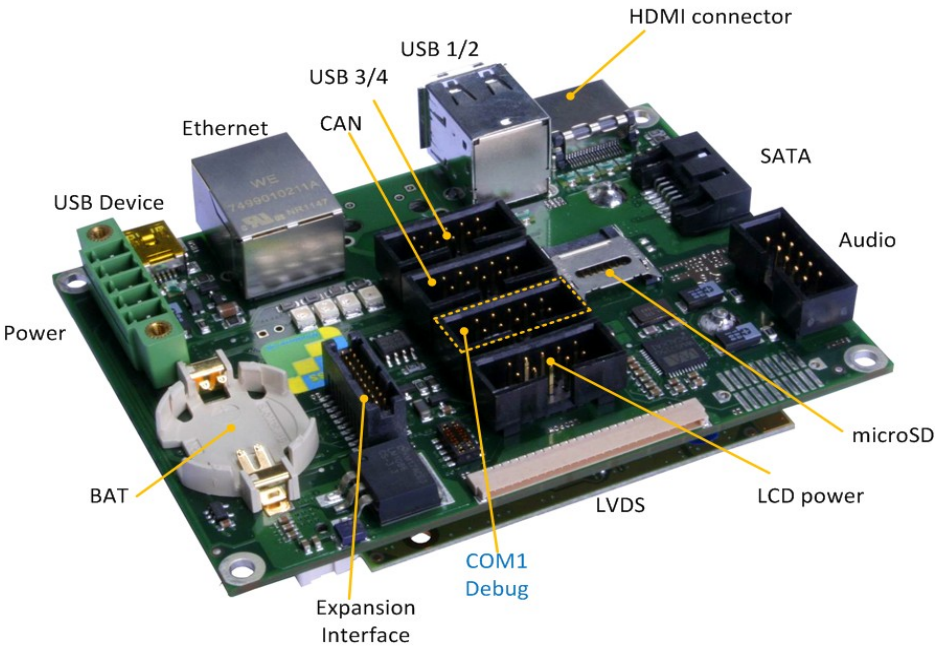


Figure 1: QBliss Starterkit top

For basic operation please make sure that power and COM1 debug port are connected correctly. Additionally it is advisable to add a battery in the designated holder.

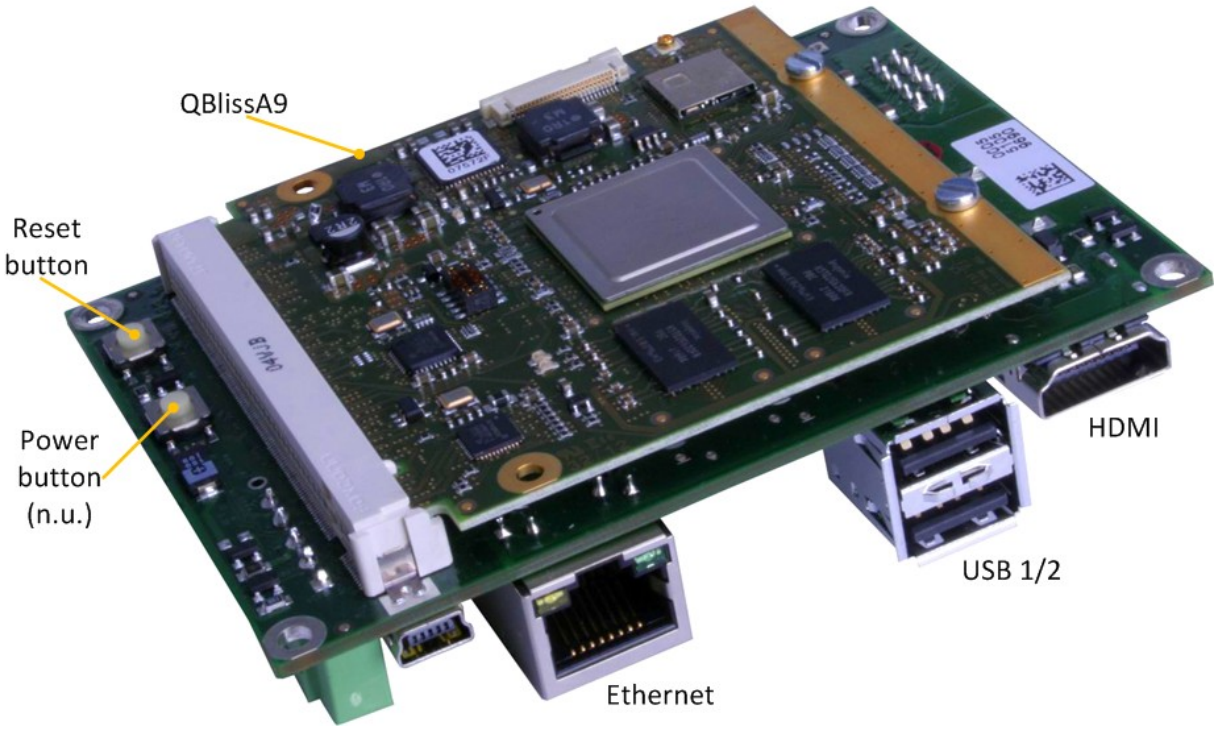


Figure 2: QBliss Starterkit bottom

More detailed information about connectors available on QBlissA9 can be found in the “Hardware documentation for QBliss Startintf”.

2.1.2 armStoneA9

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

- Cables (ethernet, serial, power, USB, …).
- Software (source, binaries, install scripts, examples).
- armStoneA9 module.

For basic operation please make sure that the Serial Debug Port and power are connected correctly.

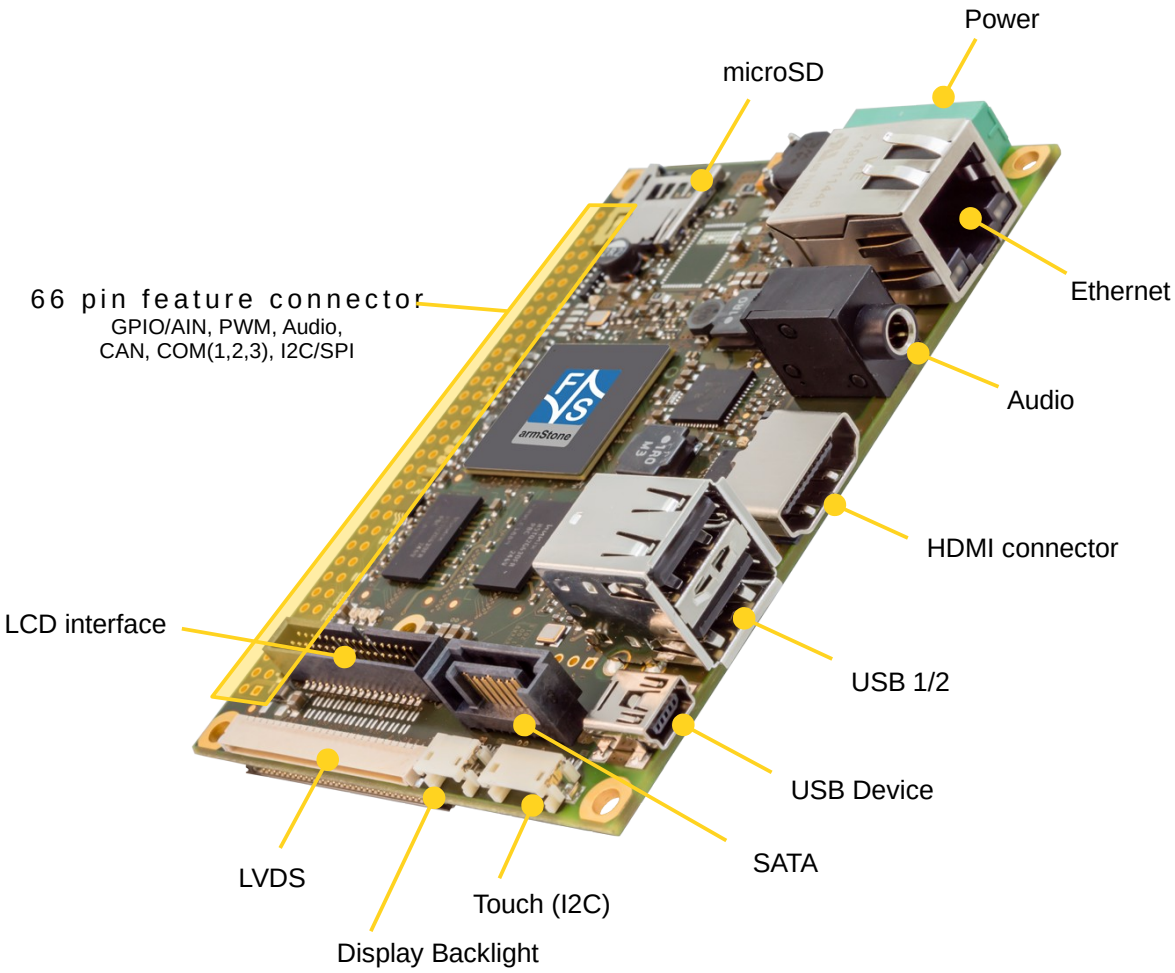


Figure 3: armStoneA9 (top view)

2.1.3 armStoneA9r2

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

- Cables (ethernet, serial, power, USB, …).
- Software (source, binaries, install scripts, examples).
- armStoneA9r2 module.

For basic operation please make sure that the Serial Debug Port and power are connected correctly.

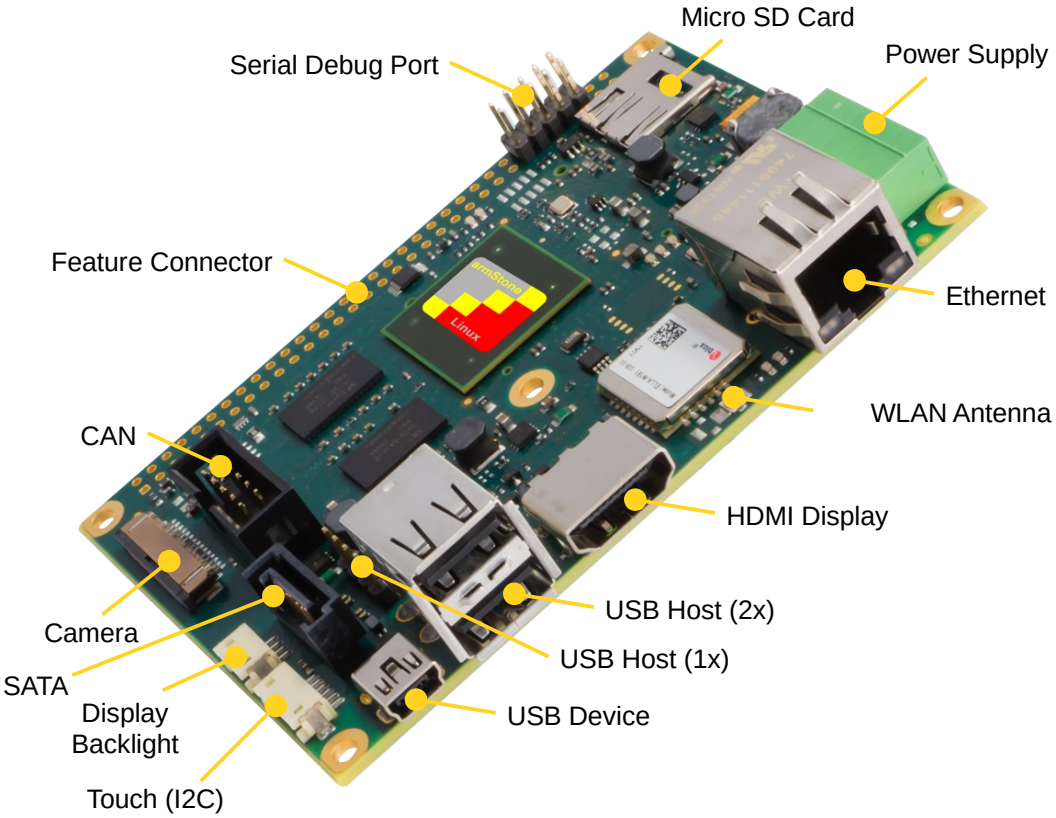


Figure 4: armStoneA9r2 (top view)

2.1.4 efusA9

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

- Cables (Ethernet, serial, power, USB, …).
- Software (source, binaries, install scripts, examples).
- Starterkit carrier board that offers connectivity for most interfaces available in efusA9.
- efusA9 module.

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

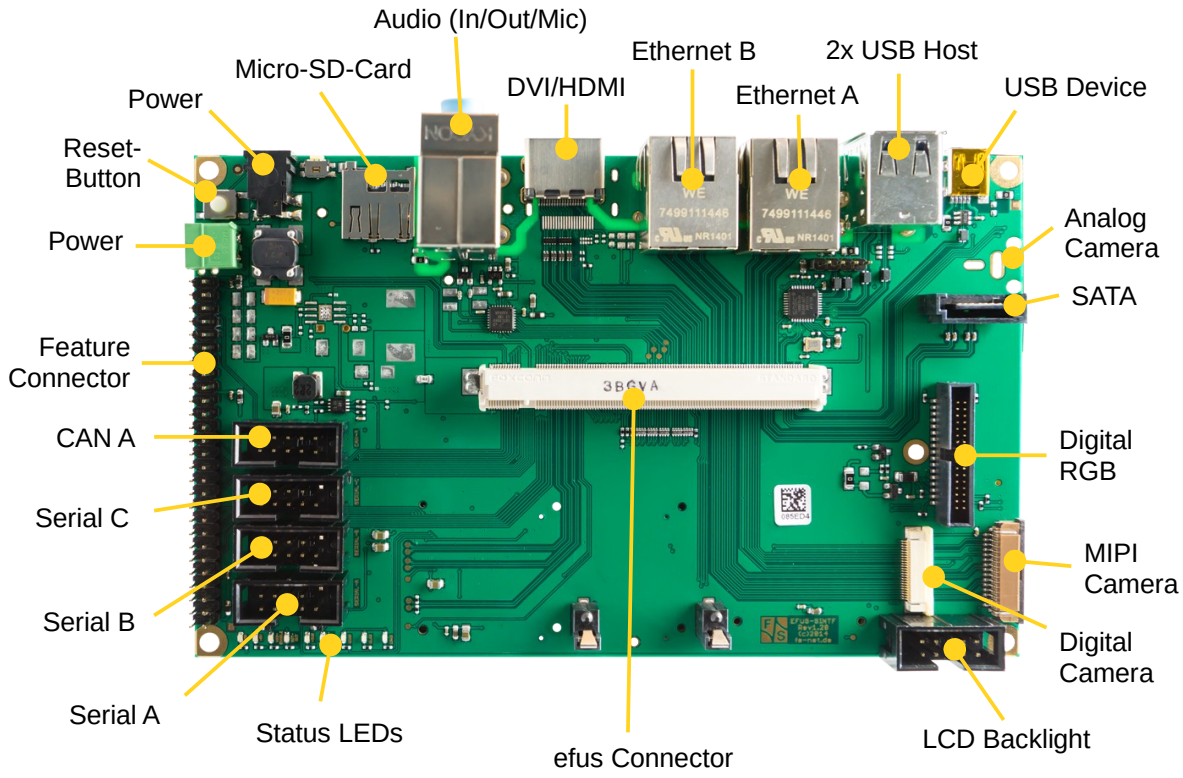


Figure 5: efusA9 StarterKit top

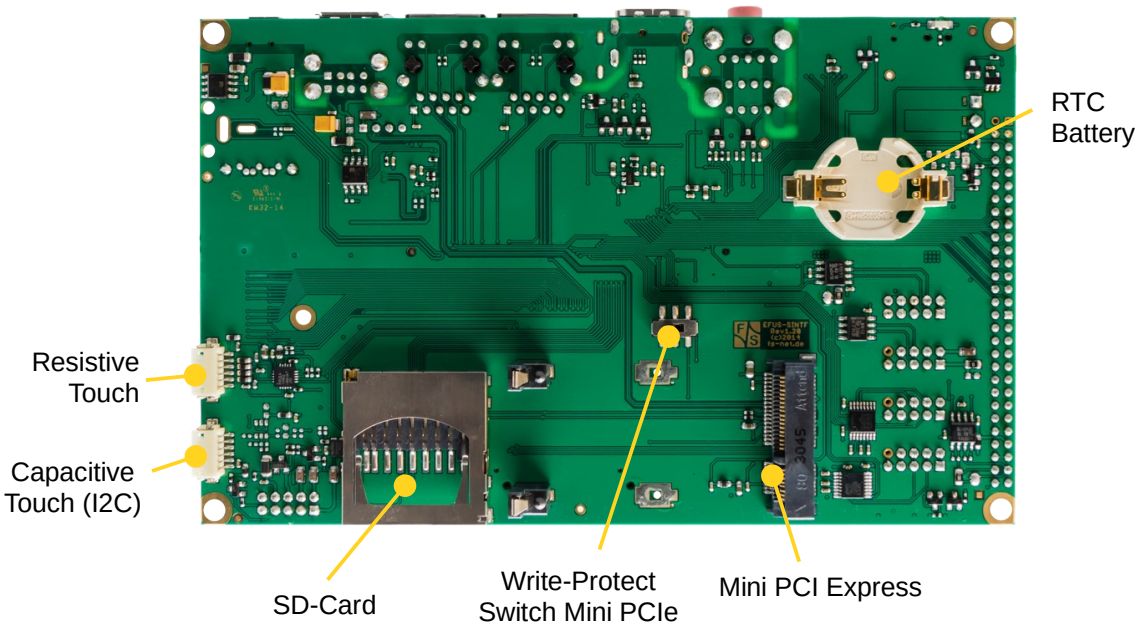


Figure 6: efusA9 StarterKit bottom

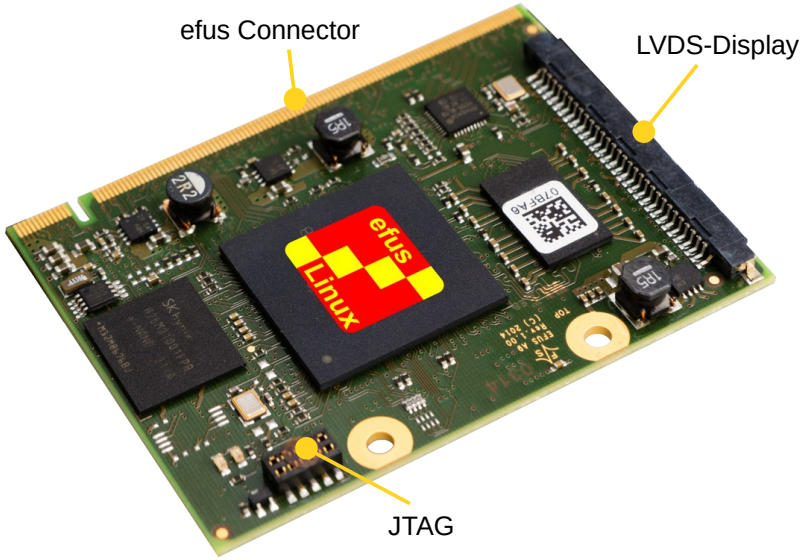


Figure 7: efusA9 module

More detailed information about connectors available on efusA9 can be found in the “Hardware documentation for efus Startinterface”.

2.1.5 efusA9r2

The Starterkit of efusA9r2 is same as efusA9 (above). It includes all components that are required for an initial setup. This includes:

- Cables (Ethernet, serial, power, USB, …).
- Software (source, binaries, install scripts, examples).
- Starterkit carrier board that offers connectivity for most interfaces available in efusA9.
- efusA9r2 module.

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

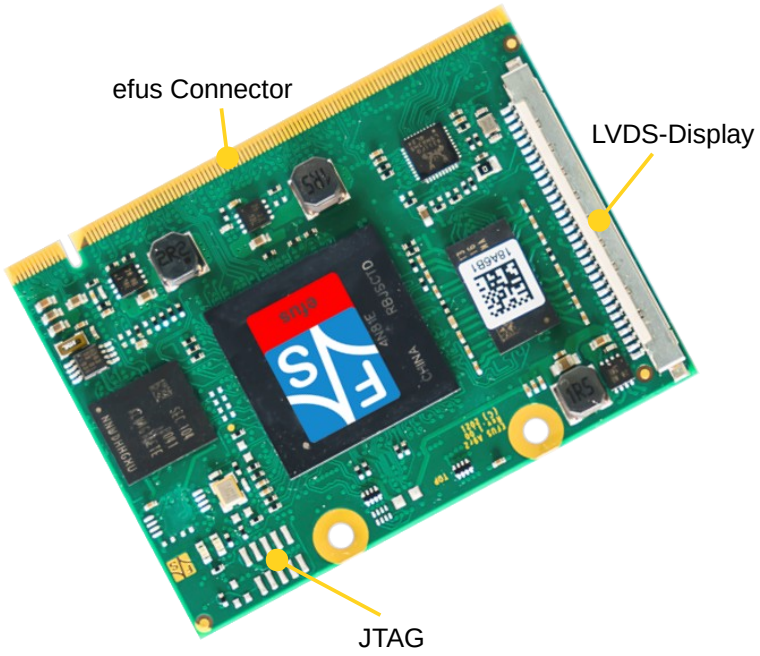


Figure 8: efusA9r2 module

More detailed information about connectors available on efusA9r2 can be found in the “Hardware documentation for efus Startinterface”.

2.1.6 PicoMODA9

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

- Cables (ethernet, serial, power, USB, …).
- Software (source, binaries, install scripts, examples).
- Starterkit carrier board that offers connectivity for most interfaces available in PicoMODA9.
- PicoMODA9 module.

For basic operation please make sure that power and Serial Debug Port (UART_A) are connected correctly.

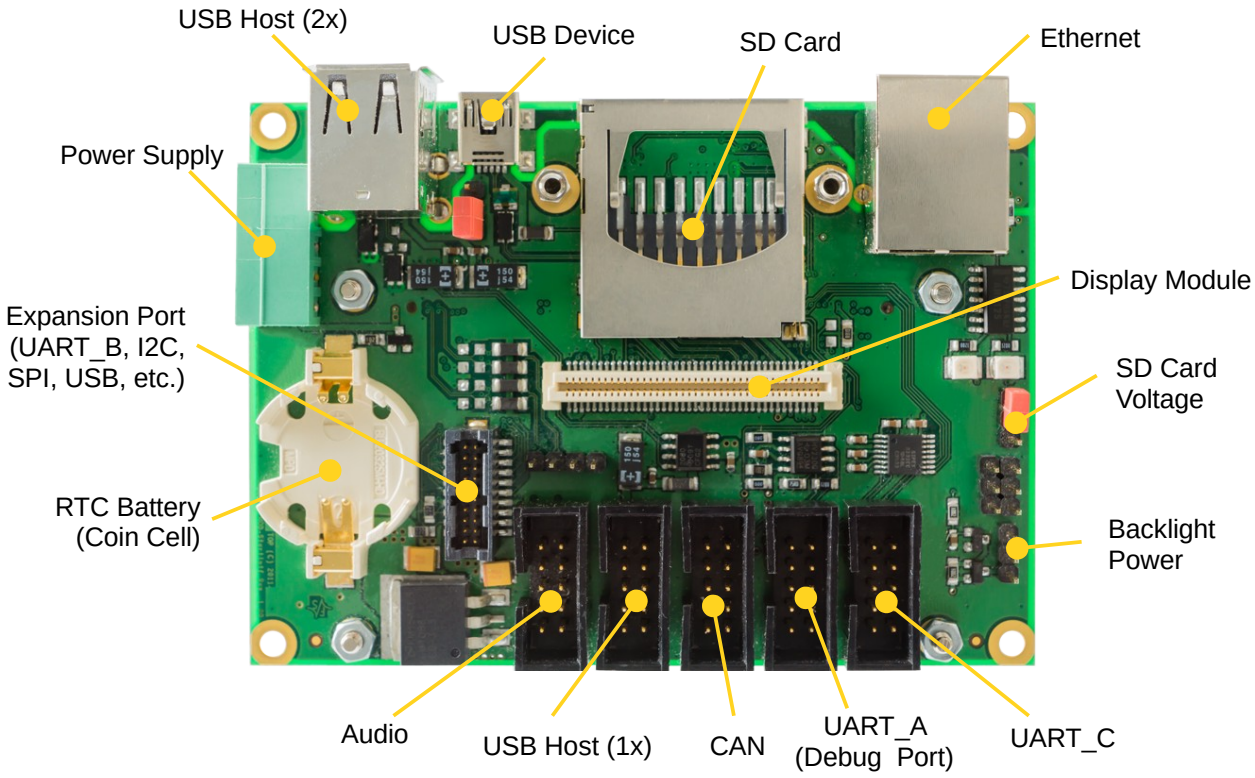


Figure 9: PicoMODA9 Starterkit, top side

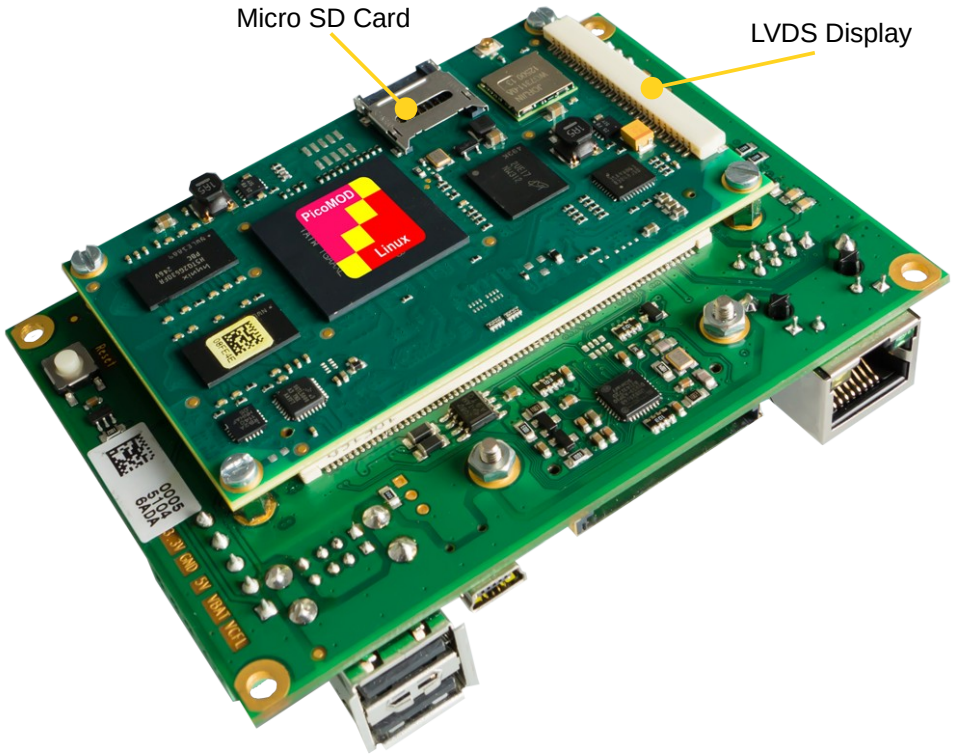


Figure 10: PicoMODA9 Starterkit, bottom side, with PicoMODA9 module

More detailed information about connectors available on PicoMODA9 can be found in the “Hardware documentation for PicoMOD Startinterface”.

2.1.7 NetDCUA9

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

- Cables (ethernet, serial, power, USB, …).
- Software (source, binaries, install scripts, examples).
- Starterkit carrier board that offers connectivity for most interfaces available in NetDCUA9.
- NetDCUA9 module.

For basic operation please make sure that power and Serial Debug Port are connected correctly.

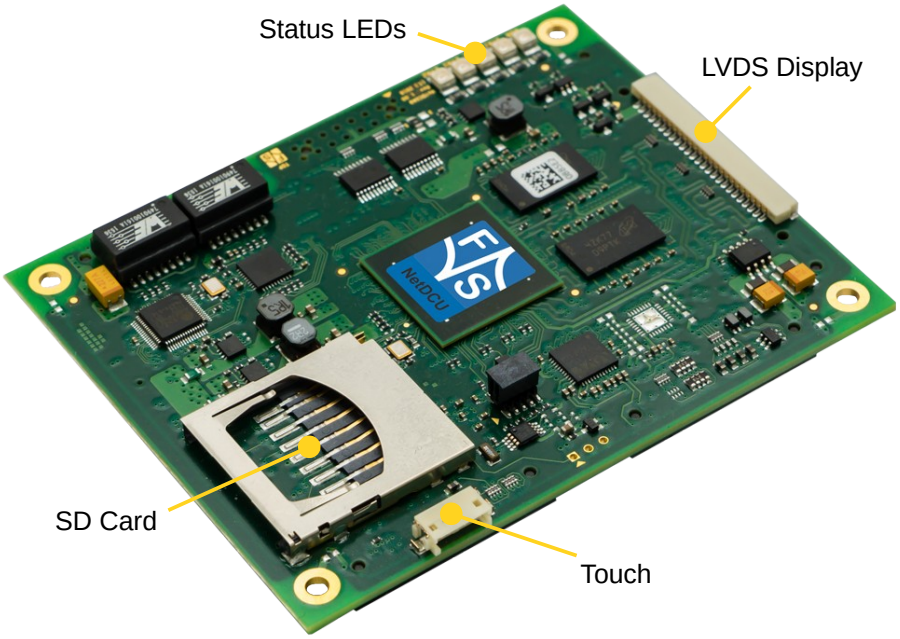


Figure 11: NetDCUA9

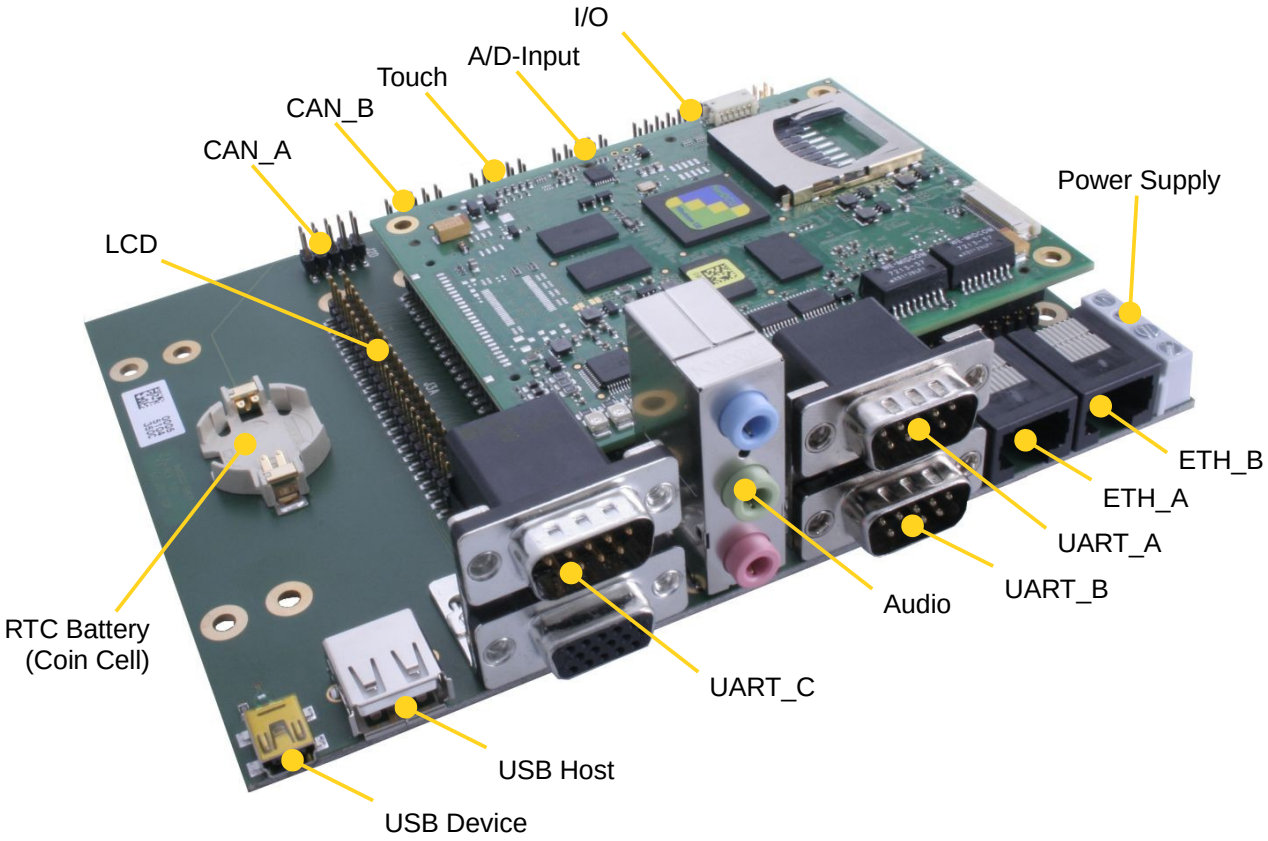


Figure 12: Starterkit with NetDCU

More detailed information about connectors available on NetDCUA9 can be found in the “Hardware documentation for NetDCU-SINTF-14”.

2.2 Serial Connection

To work with the board, you need a serial connection with your PC. Use a 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.

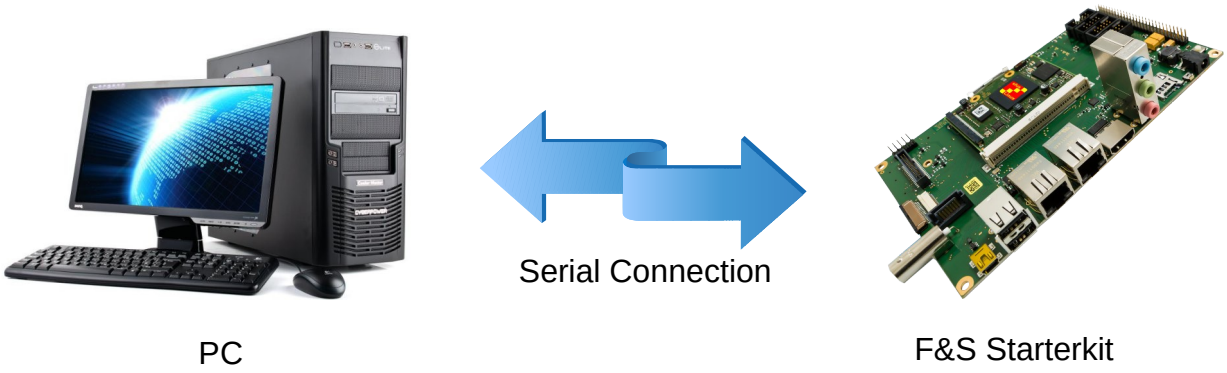


Figure 13: Serial connection from board to PC

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. However it is completely sufficient to have a Virtual Linux Machine. So you still can have a Windows based PC and use a Virtual Machine (VM) software like VirtualBox. This combination allows to use software from both worlds, for example have the serial connection done in Windows and only compile the software in the Linux VM.

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 colour 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.

```
Welcome to F+S i.MX6
fsimx6 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 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

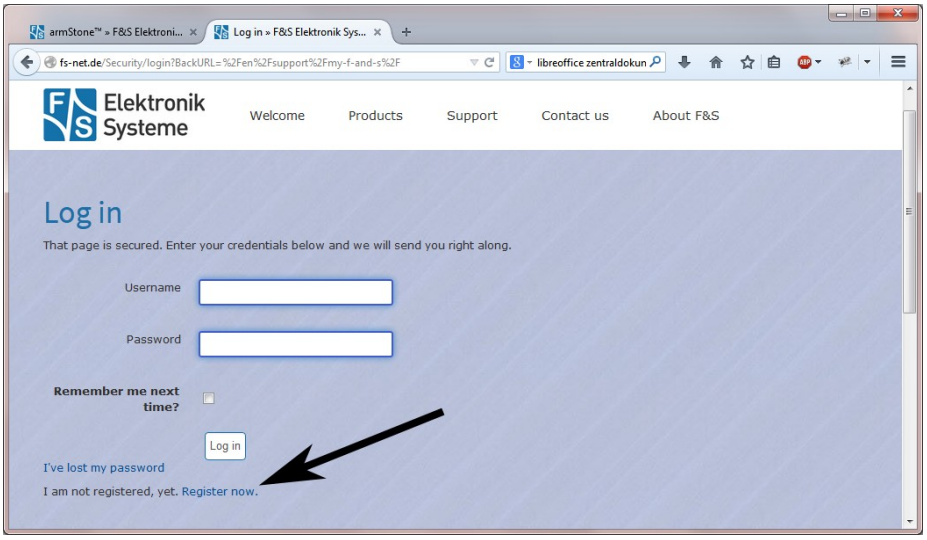


Figure 14: Register with F&S website

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 14).

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, like 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.

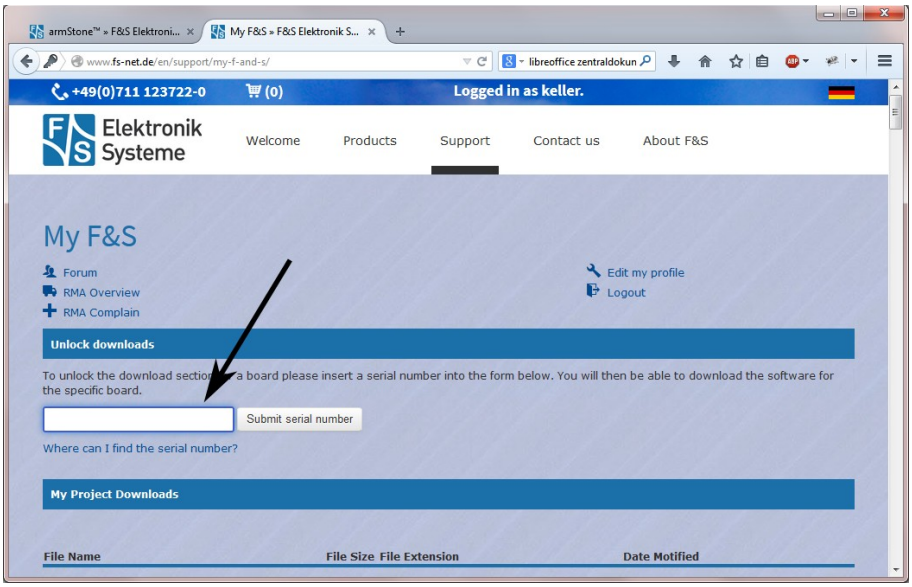


Figure 15: Unlock software with the serial number

To get access to the software of a specific board in the bottom section, you have to enter the serial number of one of these boards (see Figure 15). 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. armStone*A9*, then on *Linux*. Now you have the choice of *Buildroot* or *Yocto*. For the first steps here, we will use the newest Buildroot release, because this is the software that is also installed on our Starterkits. So click on *Buildroot*. This will bring up a list of all our Buildroot 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 *fsimx6-B2019.07*. This will finally show two archives that can be downloaded.

`fsimx6-<v>.tar.bz2`.....This is the main release itself containing all sources, the binary images, the documentation and the toolchain.

`sdcard-fsimx6-<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 sdcard-fsimx6-<v>.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 NBoot

NBoot is a small first-level bootloader that is running before the main bootloader. It is the same for Linux and Windows CE and always remains on the board, even if the whole flash memory is erased. As long as NBoot is available on the board, it is always possible to bring up the whole system again without the need for any special hardware or software tools.

Normally, NBoot is completely invisible and just loads and starts the main bootloader. Then the first output on the serial port is from the main bootloader itself. But now we will use NBoot to erase any old content and download the correct U-Boot image. This means we have to stop the boot process right at the beginning and enter NBoot.

This requires the serial setup as explained in Chapter 2.2. And we need a 1:1 download option, which means PuTTY is not suited for this. So use one of the other terminal programs. Open the serial connection, then press and hold key *s* (lower case S). While holding this key, switch on power of the board (or press the reset button). This should bring you into NBoot. You should see something like this (output is taken from armStoneA9, the real messages may vary slightly depending on the software version):

```
F&S Nand Loader VN29 built Jul 16 2016 16:17:44
armStoneA9 Rev. 1.10

...

Please select action

'd' -> Serial download of bootloader
'E' -> Erase flash
'B' -> Show bad blocks

Use NetDCUUsbLoader for USB download
```

3.3 Erase Flash

To erase any old content, simply press *E* (upper-case e). This removes everything that was on the board before. Don't be afraid, this won't erase NBoot itself.

3.4 Download and Save U-Boot

Unfortunately NBoot for fsimx6 is not capable of accessing files on an SD card or USB stick. Which means you have to download the U-Boot image via the serial line. This is the part where the 1:1 download comes in. While still in NBoot, press *d* (lower-case D). This will show some message similar to this:

```
Waiting for bootloader...
```

Now the steps are different, depending on the terminal program that you use.

- In `DCUTerm`, go to File → *Transmit Binary File...* and open the file `ubotmx6.nb0` from the `sdcard` directory. This starts the serial download.
- In `TeraTerm`, go to *File* → *Send File...* and activate the *Binary* option checkbox. This is important! Then open the file `ubotmx6.nb0` from the `sdcard` directory. This starts the serial download.
- In Linux, you can copy the U-Boot image `ubotmx6.nb0` from the `sdcard` subdirectoy directly to the serial port device. This is possible even if the terminal program does not support a 1:1 download function. For example if `/dev/ttyS0` is your serial port, just use a separate shell and enter:

  ```
  dd if=ubotmx6.nb0 of=/dev/ttyS0.
  ```

  Just note that you must not enter any characters in the terminal program while download is in progress. The character would also be sent to the serial port and would be inserted at an arbitrary position in the sequence of bytes. This would shift the remaining file content and would result in a damaged and unusable download.

Note

After pressing *d*, you have exactly 60 seconds to start the download. Then the download command times out and the menu is shown again. This means if you start to send the file too late, every byte of the file is interpreted as an own NBoot command, which is definitely not what you want.

During download, progress is shown by an increasing number of dots and the number of transmitted bytes from time to time.

```
.......................................................... 65536 Bytes
.......................................................... 131072 Bytes
.......................................................... 196608 Bytes
.......................................................... 262144 Bytes
.......................................................... 327680 Bytes
.......................................................... 393216 Bytes
.......................................................... 458752 Bytes
.......................................................... 524288 Bytes
Success, checksum: 0xaa7d

>>> U-Boot image loaded (524288 bytes) <<<

Please select action
'f' -> Save image to flash
'x' -> Execute image
'd' -> Serial download of bootloader
'E' -> Erase flash
'B' -> Show bad blocks

Use NetDCUUsbLoader for USB download
```

When download is complete, you see the menu again, which now has additional entries. Save U-Boot by pressing *f* (lower case F). This should show

```
Saving U-Boot...Success
```

3.5 Install Kernel, Device Tree And Root Filesystem

Now insert the installation device into the board or Starterkit baseboard. The remaining installation is fully automatic and is done by U-Boot. As U-Boot is still available in RAM from the previous step, you can directly start it by pressing *x* (lower-case X). This will show something like this:

```
U-Boot 2018.03 (Mar 22 2019 - 18:57:38) for F&S

CPU:   Freescale i.MX6SOLO rev1.2 at 792 MHz
Reset: POR
Board: armStoneA9 Rev 1.10 (4x DRAM, 1x LAN, 1x CAN)
DRAM:  1 GiB
NAND:  128 MiB
MMC:   FSL_SDHC: 0
In:    serial
Out:   serial
Err:   serial
Net:   FEC [PRIME]
Hit any key to stop autoboot:  3
```

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
```

3.6 Set MAC Address

When we erased the flash content in Chapter 3.3, we also 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 16).

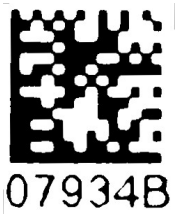


Figure 16: Bar-code sticker

The full MAC address for this example would be 00:05:51:07:93:4B.

The following two commands will set the MAC address and stores the current environment (including the newly set MAC address) 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.

```
setenv ethaddr 00:05:51:xx:yy:zz
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.

3.7 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.

```
Welcome to F+S i.MX6
fsimx6 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 can not 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 back to read-only after applying the changes with

```
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 attached, a device `/dev/sda1` is automatically created. And when the USB stick is unplugged, the device 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 sub-directories 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, we could query the current setting. And by writing a new value to the file, we 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
```

Show board specific information:

```
cat /sys/bdinfo/board_revision
```

4.3 Serial

On NXP CPUs, the devices are called `/dev/ttymxc<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 `ttymxc2`:

```
echo Hello > /dev/ttymxc2
```

Show characters that arrive on port `ttymxc2`:

```
cat < /dev/ttymxc2
```

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.

4.4 CAN

The CAN driver uses Socket CAN, i.e. the CAN bus is accessed as a network device, similar to an ethernet card. If the driver is available, you can find a `can0` device when issuing the command

```
ifconfig -a
```

But better use the newer `ip` program as the older `ifconfig` does not know anything more detailed about CAN controllers.

```
ip link
```

Before you can activate the CAN device, you have to set the baud rate. This requires the `ip` program. For example to set 125000 bit/s for CAN and activate, use this command:

```
ip link set can0 up type can bitrate 125000
```

Now you can create sockets that access the CAN device. Some examples are provided in the package `can_utils` in Buildroot (`can_tx.c`, `can_rx.c`, `candump.c`, `cansend.c`).

4.5 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 eth0
```

4.6 WLAN

Wireless LAN is available optionally on some fsimx6 boards (e.g. armStoneA9r2).

If the MAC address is not set automatically (depends on WLAN chipset), enter the following command to set it manually.

```
ifconfig wlan0 hw ether <HW address>
```

After that you might 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 connections, data encryption and login information.

4.7 Qt5

Qt is a cross-platform application framework that is widely used for developing applications mostly with a graphical user interface. Then standard root filesystem does include Qt5, but you can use the qt5_defconfig in Buildroot to build a root filesystem with Qt5 support.

4.8 Multimedia (gstreamer support)

Modules based on fsimx6 support hardware accelerated video processing in `gstreamer`. Table 3 holds a list of all i.MX6 specific `gstreamer1` plugins that use hardware acceleration.

Plugin name	Description
`imxv4l2src`	Video4Linux source (e.g. camera)
`aiurdemux`	VPU-based demultiplexer
`vpudec`	VPU-based video decoder
`vpuenc_jpeg`	VPU-based JPEG encoder
`vpuenc_h263`	VPU-based H.263 video encoder
`vpuenc_h264`	VPU-based H.264 video encoder
`vpuenc_mpeg4`	VPU-based MPEG4 video encoder
`imxv4l2sink`	Video4Linux video sink
`imxeglvivsink`	GPU based video sink
`beepdec`	Audio decoder
`imxvideoconvert_g2d`	G2D-based video converter
`imxvideoconvert_ipu`	IPU-based video converter

Table 3: VPU based GStreamer plugins

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 a file `video.mp4` there, then call

```
gst-launch-1.0 playbin uri=file:/root/video.mp4
```

`Playbin` will automatically select a suitable set of filters to play the file.

You can also give your own list of filters, which is slightly more complicated.

```
gst-launch-1.0 filesrc location=video.mp4 ! video/quicktime \
        ! aiurdemux ! vpudec ! imxv4l2sink
```

Please note that the Video4Linux video sink (`imxv4l2sink`) will show the video content in an overlay window.

For an MP3 audio file:

```
gst-launch-1.0 filesrc location=file.mp3 typefind=true \
        ! id3demux ! beepdec ! alsasink
```

4.9 SPI

This device can be used with the usermode SPI driver (also called spidev).

4.10 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/mmcblk0p1 /mnt
ls /mnt
```

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

```
umount /mnt
```

4.11 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
```

4.12 RTC

Setting date:

```
date "2015-04-22 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.

4.13 GPIO

You can setup and use GPIOs with the Sysfs system.

```
ls /sys/class/gpio
export       gpiochip128  gpiochip64   unexport
gpiochip0    gpiochip32   gpiochip96
```

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

Example:

Configure for example function pin COL0 (armStoneA9: J12 / PTD2) as output pin. The pin number on J12 is 3. The “GPIO Reference Card” shows in row `/sys/class/gpio/gpio#` the number 147. To get this pin into sysfs write:

```
echo 147 > /sys/class/gpio/export
```

This creates a new directory `gpio147` in `/sys/class/gpio` that is used for all further settings of this GPIO.

```
ls /sys/class/gpio/gpio147/
active_low  direction   edge        power       subsystem
uevent      value
```

Set pin as output:

```
echo out > /sys/class/gpio/gpio147/direction
```

Set pin to high level:

```
echo 1 > /sys/class/gpio/gpio147/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/gpio147/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/gpio147/value
```

4.14 Sound

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

```
speaker-test -c 2 -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>
```

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
```

4.15 Pictures

There is a small image viewer program included called `fbv`. Just call it with the list of images to show. This will show a new image every ten seconds.

```
fbv -s 10 /usr/share/directfb-examples/*.png
```

To show possible program options use:

```
fbv --help
```

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

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

4.16 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
```

4.17 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>
```

4.18 SSH

Buildroot

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

```
passwd
```

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_*`. Just remove the files with the word `ssh_host_` and `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
```

4.19 VNC

Note

This is only supported in native X11 environments.

If you have no display attached or you want to connect by remote you can start the pre-installed (with the standard Buildroot root filesystem) 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
```

4.20 Bluetooth

Bluetooth is available on some modules. Buildroot offers support for the official Linux Bluetooth protocol stack (bluez). To attach the bluetooth device you must first create a hci device. Physically bluetooth device is available on `ttymxc0`:

```
hciattach ttymxc0 texas 38400 flow
```

After powering up the device, bluetooth is working generally.

```
hciconfig hci0 up
```

To scan for bluetooth devices you can use the hcitools.

```
hcitool scan
```

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

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 QBlissA9

Hardware documentation for QBlissA9 module itself, Starterkit baseboard, including schematics:

https://www.fs-net.de/en/products/qseven-qbliss/qblissa9/#panel-7

Available accessories, adapters and extensions:

https://www.fs-net.de/en/products/qseven-qbliss/qblissa9/#panel-4

5.2.2 QBlissA9r2

Hardware documentation for QBlissA9r2 module itself, Starterkit baseboard, including schematics:

https://www.fs-net.de/en/products/qseven-qbliss/qblissa9r2-with-nxp-i-mx-6-cpu/#panel-7

Available accessories, adapters and extensions:

https://www.fs-net.de/en/products/qseven-qbliss/qblissa9r2-with-nxp-i-mx-6-cpu/#panel-4

5.2.3 Resources for armStoneA9

Hardware documentation for armStoneA9:

https://www.fs-net.de/en/products/armstone/armstonea9/#panel-7

Available accessories, adapters and extensions:

https://www.fs-net.de/en/products/armstone/armstonea9/#panel-4

5.2.4 Resources for armStoneA9r2

Hardware documentation for armStoneA9r2:

https://www.fs-net.de/en/products/armstone/armstonea9r2-with-nxp-i-mx6/#panel-7

Available accessories, adapters and extensions:

https://www.fs-net.de/en/products/armstone/armstonea9r2-with-nxp-i-mx6/#panel-4

5.2.5 Resources for efusA9

Hardware documentation for efusA9 module itself, Starterkit baseboard, including schematics and Eagle-Layout files:

https://www.fs-net.de/en/products/efus/efusa9/#panel-7

Available accessories, adapters and extensions:

https://www.fs-net.de/en/products/efus/efusa9/#panel-4

5.2.6 Resources for PIcoMODA9

Hardware documentation for PicoMODA9 module itself, Starterkit baseboard, including schematics:

https://www.fs-net.de/en/products/picomod/picomoda9/#panel-7

Available accessories, adapters and extensions:

https://www.fs-net.de/en/products/picomod/picomoda9/#panel-4

5.2.7 Resources for NetDCUA9

Hardware documentation for NetDCUA9, Starterkit baseboard, including schematics:

https://www.fs-net.de/en/products/netdcu/netdcua9/#panel-6

Available accessories, adapters and extensions:

https://www.fs-net.de/en/products/netdcu/netdcua9/#panel-3

6 Special Hardware Notes

This chapter lists some supplementary notes that only apply to special hardware configurations based on fsimx6.

6.1 QBliss SKIT

The F&S Qseven boards QBlissA9 and QBlissA9r2 have Gigabit ethernet. Therefore the software is also configured for Gigabit Ethernet. However the QBliss SKIT baseboard is *not* capable of Gigabit Ethernet. This means that it may take some while until the Ethernet auto-negotiation detects that it can not use a Gigabit connection. which in turn may result in timeouts when booting over network. In this case you can either connect the board to a 100 Mbit/s Ethernet switch or add a command to the boot sequence in U-Boot that disables Gigabit auto-negotiation.

```
setenv preboot mdio write FEC 9 0
saveenv
```

In the same manner you may need to limit the link speed in Linux. This can be done by adding the line

```
max-speed = <100>;
```

to the `ethphy0` node of the device tree (`qblissa9qdl.dtsi` or `qblissa9r2qdl.dtsi` respectively).

Please note that all this is no problem at all if your final baseboard is compatible to Gigabit Ethernet.

7 Appendix

List of Figures

List of Tables

Listings

Important Notice

The information in this publication has been carefully checked and is believed to be entirely accurate at the time of publication. F&S Elektronik Systeme assumes no responsibility, however, for possible errors or omissions, or for any consequences resulting from the use of the information contained in this documentation.

F&S Elektronik Systeme reserves the right to make changes in its products or product specifications or product documentation with the intent to improve function or design at any time and without notice and is not required to update this documentation to reflect such changes.

F&S Elektronik Systeme makes no warranty or guarantee regarding the suitability of its products for any particular purpose, nor does F&S Elektronik Systeme assume any liability arising out of the documentation or use of any product and specifically disclaims any and all liability, including without limitation any consequential or incidental damages.

Products are not designed, intended, or authorised for use as components in systems intended for applications intended to support or sustain life, or for any other application in which the failure of the product from F&S Elektronik Systeme could create a situation where personal injury or death may occur. Should the Buyer purchase or use a F&S Elektronik Systeme product for any such unintended or unauthorised application, the Buyer shall indemnify and hold F&S Elektronik Systeme and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, expenses, and reasonable attorney fees arising out of, either directly or indirectly, any claim of personal injury or death that may be associated with such unintended or unauthorised use, even if such claim alleges that F&S Elektronik Systeme was negligent regarding the design or manufacture of said product.