

PicoCOM4 First Steps Linux

Documentation about how to download PicoCOM4 bootloader, kernel image and root filesystem.

Version 1.11 Date: 2013-08-21

© by F & S Elektronik Systeme GmbH 2013

F&S Elektronik Systeme GmbH
Untere Waldplätze 23
D-70569 Stuttgart
Fon: +49(0)711-123722-0
Fax: +49(0)711-123722-99



History

Date	V	Platform	A,M,R	Chapter	Description	Au
2011-02-07	1.0	PC4	A	*	First Version	DK
2011-04-08	1.1	PC4	A	3,4	Setup display parameters	DK
2011-04-29	1.2	PC4	A,M	1,*	Rel-0.1.1	DK
2011-05-02	1.3	PC4	M	*	New naming convention	DK
2011-05-31	1.4	PC4	M	4.1.1	CROSS_COMPILE and ARCH	DK
2011-08-09	1.5	PC4	M	5.2	Updated	DK
2012-02-01	1.6	PC4	A,M	*	V1.0	DK
2012-02-08	1.7	PC4	A,M	1,5,12	V1.1	DK
2012-07-03	1.8	PC4	M	1,3,4	V1.2	DK
2012-12-12	1.9	PC4	M	1,2,6	New Root FS , new Bootloader and new Toolchain	DK
2012-12-12	1.10	PC4	M	1,2	V1.3	DK
2013-08-21	1.11	PC4	M	1, 4, 6	V1.4	HK

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



About this document

Documentation about how to download PicoCOM4 bootloader, kernel image and root filesystem.



Table of Contents

1	Change Log.....	1
2	Development Environment.....	2
2.1	Download area	3
3	Quickstart.....	4
3.1	Download U-Boot.....	4
3.2	Download Kernel Image.....	4
3.3	Download Root File System.....	5
3.4	Setup Root Filesystem load process.....	6
3.5	Download N-Boot.....	6
4	Customize Linux Kernel.....	7
4.1	Installing Toolchain.....	7
4.1.1	Setup environment.....	7
4.1.2	Build an Hello World application.....	7
4.2	Unpack Linux Kernel Source.....	8
4.3	Build Linux Kernel Image.....	8
4.4	Setup display parameters.....	9
5	Features.....	11
5.1	Mp3 Playback.....	11
5.2	Can Bus	11
5.3	MMC.....	11
5.4	USB Host.....	11
5.5	UART.....	11
5.6	TOUCH.....	11
5.7	I2C.....	12
5.8	GPIO.....	12
5.9	PWM.....	12
5.10	RTC.....	12
5.11	Analog Inputs (HWMON).....	12
5.12	SPI.....	12
6	Customize Root-Filesystem.....	13
6.1	Adding features.....	13
7	Appendix.....	14
	Important Notice.....	14

1 Change Log

V1.4

pc4boot (V26)

Supported Boards: PicoCOM4

- No changes

U-Boot picocom4-V1.4 (20.08.2013, based on u-boot-1.3.4)

Supported boards: PicoCOM4 (+ PicoCOM5, PicoMOD6, PicoMOD7, QBlissA8)

- Support new toolchain fs-toolchain-4.7.2-armv4t
- Fix serial configuration on s3c2416
- Disable DVS (Dynamic Voltage Scaling); now the PicoCOM4 is 3x faster!
- Remove unnecessary initialization that is already done in NBoot

Linux Kernel picocom-V1.4 (21.08.2013, based on linux-2.6.38)

Supported boards: PicoCOM4

- Add parameters for FT5X06 touch controller to SYSFS
- Improve sound: use codec TLV320 as clock/frame master, add capture feature
- Fix backlight control
- Use new S3C touch driver from F&S
- Remove some debug messages
- Fix SD card issues, some SD cards reported timeouts
- Clean up UART configuration, add alternative configurations
- Fix SPI driver (shift value for rx_lvl_offset)
- Add default gain, thold and offs for FT5X06 touch
- Switch to minimal defconfig
- Make default configuration similar to other F&S boards:
 - Include IPV6, EXT4, CIFS, NFSV4 and squashfs fix into kernel
 - Use SLUB memory allocator
 - Announce USB devices when they are connected
 - Remove support for devtmpfs, initrd, serial USB devices, NTFS, a.out binary format, legacy keyboard and mice, legacy PTYS, swap and SM501 graphic controller as it is usually not needed
 - Remove SMDK2410 eval-kit from included platforms
 - Remove many unused modules, e.g. many codepages and joysticks
- Switch to F&S Tux logo
- Add generic LCD configuration
- Touch: Fix bug where buttons are only recognized after second touch
- Use new toolchain fs-toolchain-4.7.2-armv4t

BuildRoot picocom4-V1.4 (21.08.2013, based on buildroot-2013.02)

Supported Boards: armStoneA5, armStoneA8, NetDCU14, NetDCUA5, PicoMOD7A, PicoCOM4, PicoCOMA5

- Rebase multiplatform-linux version to buildroot-2013.02
- Merge multiplatform-linux and fsybrid versions
- Create link for lib/ld-linux.so.3 if required



- Add libmcc (MCC: Multi Core Communication) for Vybrid
- Add mxboot for Vybrid to start applications on Cortex-M4
- Add mcc-pingpong example (Linux side) for Vybrid
- Configure all targets for new toolchains fs-toolchain-4.7.2-*
- Add ttymxc3/4/5 to /etc/securetty to allow root logins on these lines
- Combine armStoneA8, NetDCU14 and PicoMOD7A to common fss5pv210
- New script /etc/init.d/S01fssetup to set up the F&S environment
- Use login_tty from F&S environment to start a getty on
- Use gdb 7.4.x
- Make STD and MIN defconfigs as similar as possible on all platforms
- Fix some typos, indendations and wrong comments

Toolchain

- New toolchain fs-toolchain-4.7.2-armv4t based on binutils-2.23.1, gcc-4.7.2 and eglibc-2_17

Documentation

- Improved first steps docu PicoCOM4_Linux_V1.4.pdf
- AdvicesForLinuxOnPC.pdf to help configuring your PC for embedded development

V1.3

- **New Kernel Image**, *zImage_PicoCOM4_V1.3*
- **Updated Kernel Source**, Linux 2.6.38, *PicoCOM4-Linux_V1.2_2.6.38.tar.bz2*
 - o 2.6.38-00040-g2332296
 - Display: for default display set VEEK to 0

V1.2

- **New Kernel Image**, *zImage_PicoCOM4_V1.2*
- **Updated Kernel Source**, Linux 2.6.38, *PicoCOM4-Linux_V1.2_2.6.38.tar.bz2*
 - o 2.6.38-00039-ga1f1568
 - CAN: if not enabled, request SPI_CS0_CAN as output w/ pull-up!
 - LCD: new default display for SKIT (320x240).
 - Touch: Cap. Touch EP0XXM0M06 added.
- **New UBoot**, *uboot_picocom4_V1.2.nb0*
- **New Root FS (jffs2)**, *rootfs_PicoCOM4_V1.2.jffs2*
- **New Root FS configuration**, *picocom4_std_defconfig* and *picocom4_min_defconfig*
 - o Integrated into multiplatform-linux-f+s-V2.0 Buildroot.
- **New Toolchain**, *fs-toolchain-4.6.3-armv4t.tar.bz2*

V1.1

- **New Kernel Image**, *zImage_PicoCOM4_V1.1*
- **Updated Kernel Source**, Linux 2.6.38, *PicoCOM4-Linux_V1.1_2.6.38.tar.bz2*
 - o 2.6.38-10429-gc91588b
 - new SPI driver (from S3C6410)

V1.0

- **New Kernel Image**, *zImage_PicoCOM4_V1.0*
- **Updated Kernel Source**, Linux 2.6.38, *PicoCOM4-Linux_V1.0_2.6.38.tar.bz2*
- **Updated GPIO-Referenreccard**, *PicoCOM4-GPIO-ReferenceCard_eng.pdf*



V0.2:

- **New NBoot**, *pc4boot_22.32k_V0.2.bin*
- **New UBoot**, *uboot_picocom4_V0.2.nb0*
- **New Kernel Image**, *zImage_PicoCOM4_V0.2*
- **Updated Kernel Source**, *PicoCOM4-Linux_2.6.37.rc_V0.2.tar.bz2*

V0.1:

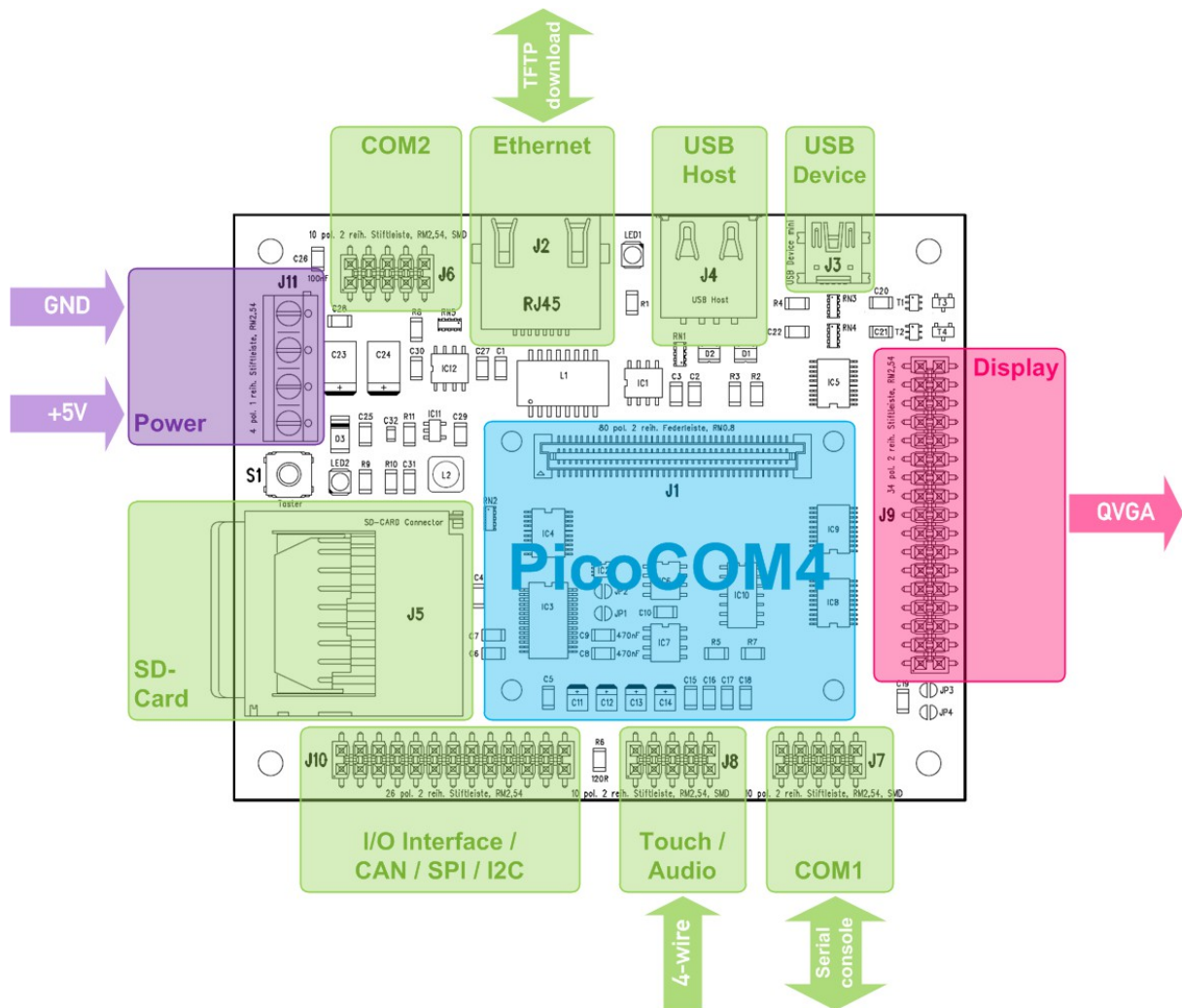
- **NBoot**, *pc4boot_19.32k_V0.1.bin*
- **UBoot**, *uboot_picocom4_V0.1.nb0*
- **Kernel Image**, *zImage_PicoCOM4_V0.1*
- **Kernel Source**, *PicoCOM4-Linux_2.6.37.rc_V0.1.tar.bz2*
- **Toolchain**, *PicoCOM4_toolchain_V0.1.tar.bz2*
- **Root FS (ext2)**, *rootfs_PicoCOM4_V0.1.ext2*
- **Rootf FS (jffs2)**, *rootfs_PicoCOM4_V0.1.jffs2*
- **Root FS configuration**, *PicoCOM4_buildroot-config_V0.1*



2 Development Environment

We use

- VirtualBox (oracle) running Fedora 14 as development machine.
- DCUTerm (terminal program) to download U-Boot.
- Putty (terminal program) connected to PicoCOM4s' serial console.
- TFTP32 (by Philippe Jounin) as TFTP Server to download kernel image and root filesystem to the device (or Linux based TFTP).



2.1 Download area

Starting with V1.4, you will find the whole release packed as one tar.bz2 file `picocom4-Vx.y.tar.bz2`. Download this file and unpack.

```
tar jxvf picocom4-Vx.y.tar.bz2
```

It will create a directory `picocom4-Vx.y` that contains the following subdirectories.

<code>/binaries</code>	Images that can directly be loaded to the board (boot loader, kernel image, root filesystems)
<code>/doc</code>	Documentation for the PicoCOM4 (Hardware, Starterkit, Linux information, ...)
<code>/examples</code>	Some examples
<code>/sources</code>	The source files that can be used to build the binaries (U-Boot, Linux Kernel, BuildRoot)
<code>/toolchain</code>	The toolchain to build binaries and programs

Tools:

Terminalprogram:
(update u-boot) DCUTerm.exe

Terminalprogram:
(serial console) Putty.exe

TFT Server:
(bootloader and
kernel image) TFTPD32.exe



3 Quickstart

Connect power supply (+5V and GND) to power connector of your PicoCOM4 device. Plug Ethernet connector and a serial cable to COM1 connector.

3.1 Download U-Boot

Use the DCUTerm terminal program to connect to the serial debug port of your PicoCOM4 device. Enter Nboot (NAND bootloader) by holding 's' while powering the device. You will see output like:

```
F&S Nand Loader C422 built Apr 28 2011 10:56:19
unknown
64 MB RAM 1 chips 64 MB FLASH

Please select action
'd' -> serial download
'c' -> download eboot.nb0 from SDCard
'E' -> erase flash
use NetDCUusbLoader for USB download
```

Now press 'd' to start serial download. You will see message:

```
waiting...
```

Go to the File menu and select *“Transmit Binary File...”*. Then change to the folder where uboot.nb0 is located (in case of release 0.1 it is named uboot_picoCOM4_V0.1.nb0) and confirm by open button.

You will see download progress by some dots. After download finished (transmit messagebox disappears) you will see output like:

```
Checksum: b335

Action = 0x00000000

Type 'f' to save
      'x' to start
```

Press 'f' to save u-boot and then re-power the device.

3.2 Download Kernel Image

Use DCUTerm or Putty as terminal program to connect to the serial debug port of your PicoCOM4 device. Enter U-Boot by pressing space. Then run the commands to download the Linux kernel image to NAND flash.

```
PICOCOM4 # mtdparts default
PICOCOM4 # setenv ipaddr 10.0.0.27
PICOCOM4 # setenv serverip 10.0.0.47
PICOCOM4 # saveenv
PICOCOM4 # tftpboot 31000000 zImage_PicoCOM4_V1.2

Using ax88796c device
TFTP from server 10.0.0.153; our IP address is 10.0.0.27
Filename 'zImage_PicoCOM4_V1.2'.
Load address: 0x31000000
Loading: #####
          #####
```



```

#####
done
Bytes transferred = 2184548 (0x215564)

PICOCOM4 # nand erase clean Kernel

NAND erase: device 0 offset 0x100000, size 0x300000
Erasing at 0x3fc000 -- 100% complete. Cleanmarker written at 0x3fc000.
OK

PICOCOM4 # nand write.jffs2 31000000 Kernel $(filesize)

NAND write: device 0 offset 0x100000, size 0x215564

Writing data at 0x315400 -- 100% complete.
2184548 bytes written: OK

PICOCOM4 # setenv KernelSize $(filesize)
PICOCOM4 # saveenv

Saving Environment to NAND...
Erasing Nand...
Erasing at 0x7c000 -- 100% complete.
Writing to Nand... done

PICOCOM4 #

```

When re-power the device the linux kernel image will be extracted to RAM and started by u-boot. You can re-enter u-boot by pressing space while powering on the device.

3.3 Download Root File System

Downloading the root filesystem to NAND flash is similar to the kernel image download.

```

PICOCOM4 # tftpboot 30000000 rootfs_PicoCOM4_V0.1.jffs2
Using ax88796c device
TFTP from server 10.0.0.153; our IP address is 10.0.0.27
Filename 'rootfs_PicoCOM4_V0.1.jffs2'.
Load address: 0x30000000
Loading: #####
#####
#####
#####
#####
#####
#####
#####
#####
done
Bytes transferred = 7098296 (0x6c4fb8)
PICOCOM4 # nand erase clean TargetFS

NAND erase: device 0 offset 0x400000, size 0x3c0000
Erasing at 0x3ffc000 -- 100% complete. Cleanmarker written at 0x3ffc000.
OK
PICOCOM4 # nand write.jffs2 30000000 TargetFS $(filesize)

NAND write: device 0 offset 0x400000, size 0x6c4fb8
Writing data at 0xac4e00 -- 100% complete.
7098296 bytes written: OK
PICOCOM4 # saveenv
Saving Environment to NAND...

```



```
Erasing Nand...
Erasing at 0x7c000 -- 100% complete.
Writing to Nand... done
PICOCOM4 #
```

3.4 Setup Root Filesystem load process

You can either load root filesystem from NAND flash or by network (NFS). Therefore you can Setup u-boot by short commands:

```
bootlocal=setenv bootargs console=ttySAC1,38400 ${mtdparts} init=linuxrc
root=/dev/mtdblock1 rw rootfstype=jffs2 ethaddr=${ethaddr}

bootnfs=setenv bootargs console=ttySAC1,38400 ${mtdparts} init=linuxrc
root=/dev/nfs rw nfsroot=/rootfs
ip=${ipaddr}:${serverip}:${gatewayip}:${netmask} ethaddr=${ethaddr}
```

To load rootfs from NAND flash run:

```
PICOCOM4 # run bootlocal
PICOCOM4 # saveenv
```

To load rootfs from network run:

```
PICOCOM4 # run bootnfs
PICOCOM4 # saveenv
```

Note: You need to modify the serverip, ipaddr, netmask and gatewayip values to meet your environment. I.E to change serverip you do:

```
PICOCOM4 # setenv serverip 10.0.0.154
PICOCOM4 # saveenv
```

3.5 Download N-Boot

In case of a newer N-Boot or when switching from Windows Embedded CE to Linux you need to download N-Boot. This is done similar to download the U-Boot. But in all cases an already running N-Boot is required to download a newer version.

Please contact support@fs-net.de for more information.



4 Customize Linux Kernel

4.1 Installing Toolchain

Unpack toolchain (fs-toolchain-4.7.2-armv4t.tar.bz2) to **/usr/local/arm**

4.1.1 Setup environment

```
# export PATH=$PATH:/usr/local/arm/fs-toolchain-4.7.2-armv4t/bin
# export CROSS_COMPILE=arm-linux-
# export ARCH=arm
```

Check for the right GCC :

```
# which arm-linux-gcc
/usr/local/arm/fs-toolchain-4.7.2-armv4t/bin/arm-linux-gcc
```

4.1.2 Build a Hello World application

```
# mkdir helloworld
# cd helloworld/
# vim main.c

#include <stdlib.h>
#include <stdio.h>

int main (void) {
    printf("Hello PicoCOM4... \n\n");
    return 1;
}

# arm-linux-gcc -o hellopc4 main.c
# cp hellopc4 <TFTP Server folder>
```

On the device copy *hellopc4* via tftp and run it:

```
# ifconfig eth0 10.0.0.27 up
# tftp -g -r hellopc4 10.0.0.153
# chmod a+x hellopc4
# ./hellopc4
Hello PicoCOM4...
```



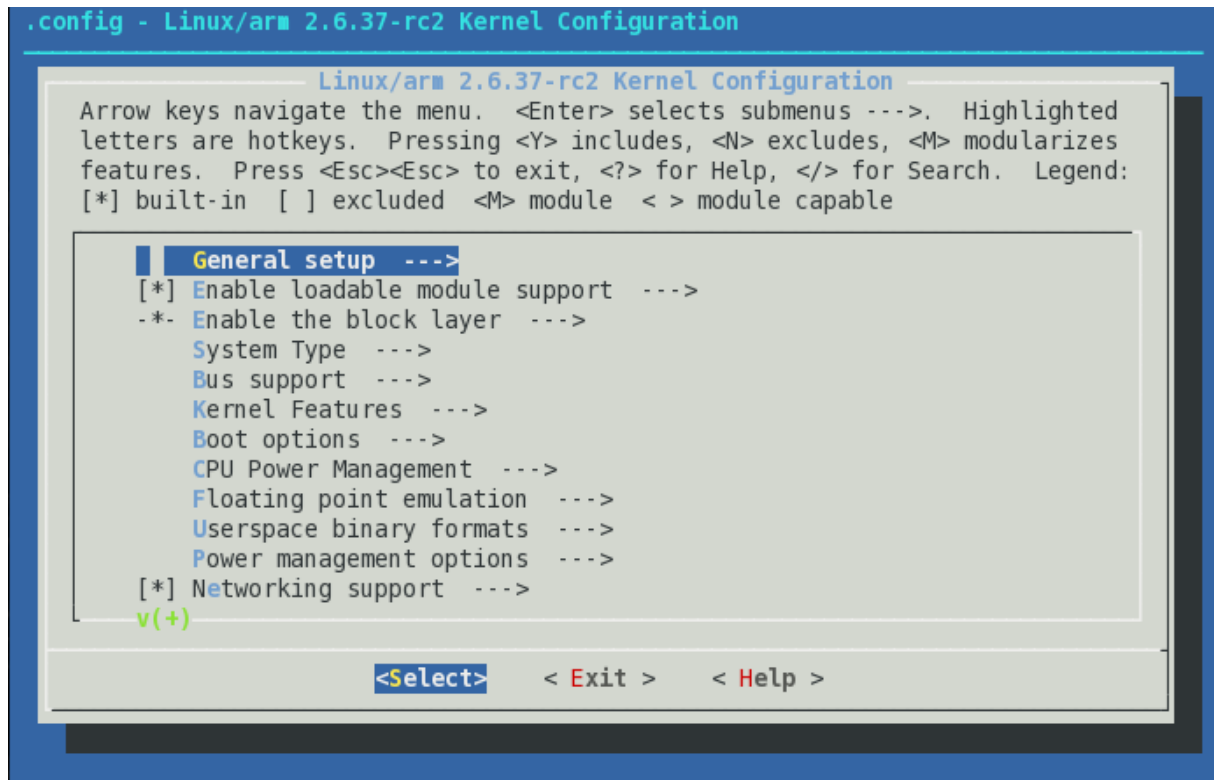
4.2 Unpack Linux Kernel Source

Unpack linux kernel source tree (linux-2.6.38-picocom4-Vx.y.tar.bz2) to i.e. ~/linux.

4.3 Build Linux Kernel Image

Change to ~/linux/linux-2.6.38-picocom4-Vx.y kernel source tree folder.

```
# make picocom4_defconfig
# make menuconfig
```



The screenshot shows the 'Linux/arm 2.6.37-rc2 Kernel Configuration' menu. At the top, it says 'Linux/arm 2.6.37-rc2 Kernel Configuration'. Below that, it provides instructions: 'Arrow keys navigate the menu. <Enter> selects submenus --->. Highlighted letters are hotkeys. Pressing <Y> includes, <N> excludes, <M> modularizes features. Press <Esc><Esc> to exit, <?> for Help, </> for Search. Legend: [*] built-in [] excluded <M> module < > module capable'. The 'General setup --->' submenu is highlighted in blue. It contains the following options: [*] Enable loadable module support --->, -* Enable the block layer --->, System Type --->, Bus support --->, Kernel Features --->, Boot options --->, CPU Power Management --->, Floating point emulation --->, Userspace binary formats --->, Power management options --->, and [*] Networking support --->. At the bottom of the menu, there are three buttons: <Select>, < Exit >, and < Help >.

```
# make
```

The new kernel image is located in *arch/arm/boot* as *zImage*.



4.4 Setup display parameters

For the moment to switch to a different display configuration you have to recompile the kernel. Just run “make menuconfig” and go to Device drivers → Graphics support → Generic LCD configuration. There you can set all the display parameters (resolution, timing, signal polarity). Then recompile the kernel by running “make”. You will find the updated *zImage* binary file in *arch/arm/boot*.



5 Features

5.1 Mp3 Playback

After these steps you can download a mp3 file to your PicoCOM4 device and play it with:

```
# ifconfig eth0 10.0.0.27 up
# eth0: link up, 100Mbps, full-duplex

# tftp -g -r BeHealthy.mp3 10.0.0.153
# madplay BeHealthy.mp3
MPEG Audio Decoder 0.15.2 (beta) - Copyright (C) 2000-2004 Robert Leslie et
al. mapped channel 10 to 0
```

5.2 Can Bus

```
# cd /usr/sbin/
# ./ip link set can0 up type can bitrate 125000
# cansniffer can0
# cansend can0 123#deadbeef
```

5.3 MMC

Note: PicoCOM4 has no card detect pin. Hence the mmc card must be plugged when booting. When using MMC Linux modules the mmc card must be plugged when loading modules by modprobe.

```
mmc0: new SD card at address f474
mmcblk0: mmc0:f474 SU02G 1.84 GiB
mmcblk0: p1

# mount /dev/mmcblk0p1 /mnt/
# umount /mnt
```

5.4 USB Host

You can connect USB HID (mouse, keyboard, etc) and Mass Storage Devices (USB Stick) to your PicoCOM4 device.

5.5 UART

Note: COM2 connector is ttySAC0.

You can find simple demo source code (write/read to/from UART) located in download area of PicoCOM4 named *testUart_PicoCOM4.tar.bz2*.

5.6 TOUCH

You can connect a 4-wire touch to connector J8 on your PicoCOM4 base board.

Note: export TSLIB_TSDEVICE=/dev/input/event0 must be set.

You can use *ts_calibrate* to calibrate your touch. Then you can test your touch with the *ts_test* program.



5.7 I2C

On connector J10 on your PicoCOM4 base board you find SCL and SDA.

You can find simple demo source code (write/read to/from I2C-0) located in download area of PicoCOM4 named *testI2C_PicoCOM4.tar.bz2*.

5.8 GPIO

Using GPIO as Input-/Output-Pins from your application you can access SYSFS system to configure and to read/write.

Please find a simple demo source code located in download area of PicoCOM4 named *testGPIO_PicoCOM4.tar.gz*.

5.9 PWM

On connector J10 on your PicoCOM4 base board you find PWM3 pin. You can access PWM by SYSFS system.

Please find a simple demo source code located in download area of PicoCOM4 named *testPWM_PicoCOM4.tar.gz*.

5.10 RTC

Setting date :

```
date -s '2012/01/25 10:18:00' +%Y/%M/%d %T
```

Save current date to RTC :

```
hwclock --systohc
```

Access RTC :

```
/sys/class/rtc/rtc0 and /proc/driver/rtc
```

*Note : connect 3.3V to VBAT on your PicoCOM4 Startinterface.

5.11 Analog Inputs (HWMON)

To use AIN6-9 you need to disable Touch in Linux Kernel configuration.

Access by SYSFS system :

```
cat /sys/class/hwmon/hwmon0/device/driver/s3c-hwmon/in6_input
```

5.12 SPI

On J10 (PicoCOM4-SKIT) you find MISO,MOSI,SPCK and PCS0 SPI lines. A simple demo application can be found in the download section of PicoCOM4.



6 Customize Root-Filesystem

6.1 Unpack BuildRoot Source

Unpack buildroot source tree (buildroot-2013.02-picocom4-Vx.y.tar.bz2) to i.e. ~/buildroot.

6.2 Build Root Filesystem

Change to ~/buildroot/buildroot-2013.02-picocom4-Vx.y source tree folder.

```
# make picocom4_std_defconfig
# make
```

This will download all required packages and then build the standard root filesystem that is also included with the release. The root filesystem images are available in directory output/images. We always provide two different versions of the filesystem, once as jffs2 image to store on the board and once as ext2 image to be mounted as a NFS share.

Please note that building may take quite a while, maybe over an hour, depending on the power of your PC. And you need to have installed the toolchain in /usr/local/arm. Otherwise you have to change the toolchain settings (make menuconfig → Toolchain → Toolchain Path) before being able to compile the root filesystem.

There is a second target available to build a minimal root filesystem with only busybox included. This can be used as a starting point for your own root filesystem for your own application.

```
# make clean
# make picocom4_min_defconfig
# make
```

6.3 Adding Features

If you want to add features just run

```
# make menuconfig
```

There you can add new packages, modify the settings of existing packages or remove unneeded packages. Then call

```
# make
```

again. If you have built a version before, this now will work much faster. However in some cases, especially when removing packages, you may have to call

```
# make clean
```

to get a working root filesystem, resulting again in the long build process.



7 Appendix

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 authorized 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 unauthorized 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 unauthorized use, even if such claim alleges that F&S Elektronik Systeme was negligent regarding the design or manufacture of said product.

