

Überblick

Hans Buchmann FHNW/IME

12. Dezember 2017

Die Herstellung von einem GNU/Linux eine komplexe Angelegenheit

Gegeben

- ▶ Files von überall her
 - ▶ meistens Sourcefiles

Gesucht

- ▶ GNU/Linux , UNIX
 - ▶ Betriebssystem

Betriebssystem

Sammlung von Programmen auf einem Rechner

- ▶ wenig Programme
- ▶ viele Programme
- ▶ kleiner Rechner
- ▶ grosser Rechner

Unser Betriebssystem und Rechner

- ▶ Target
 - ▶ **BeagleBoneBlack**
 - ▶ GNU/Linux eher klein
- ▶ *Host*
 - ▶ UNIX eher gross

BeagleBoneBlack

Was haben wir ?

- ▶ Hardware mit WLAN
- ▶ Debian im Flash
- ▶ SD-Card Fassung
 - ▶ für eigenes GNU/Linux
- ▶ `doc/beaglebone-black/BBB_SRM.pdf`

Debian:Verbindungen

Siehe 3-network

- ▶ USB/Internet
- ▶ WLAN/Internet
- ▶ `ssh` Secure Shell
- ▶ **BeagleBoneBlack** - WAN

GNU/Linux :von Grund auf Image auf SD-Karte

- ▶ Bootloader
- ▶ Kernel
- ▶ RootFS

The Big Picture

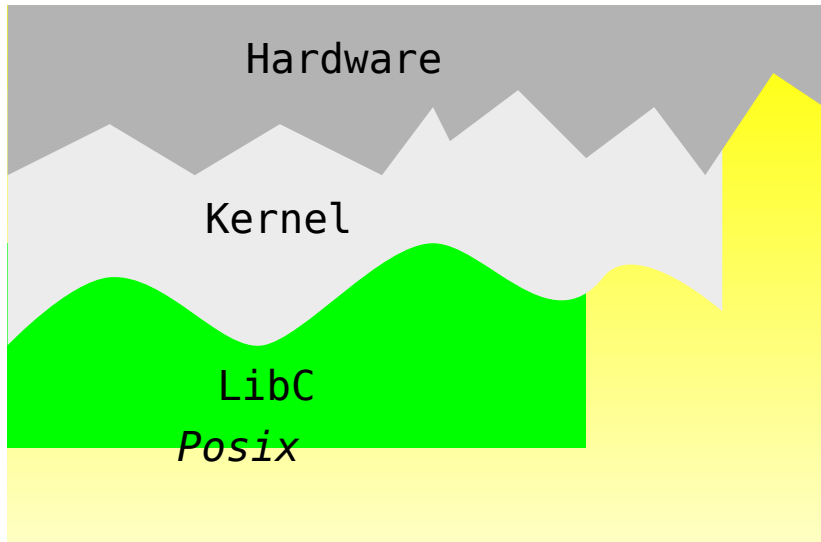


Image: 2 Partitionen

Siehe 3.5-partitions

Partition 1: **p1** `vfat` Bootloader, kernel

Partition 2: **p2** `ext4` RootFS

Befehle auf dem *Host*

- ▶ `fdisk`
- ▶ `mkfs.vfat`
- ▶ `mkfs.ext4`

Herstellung

Teil 1: wenig files

toolchain-bare: Siehe 17-build/tools/{binutils.sh | gcc-bare.sh}

u-boot: Siehe 4-uboot

- ▶ ML0
- ▶ u-boot.img

kernel: Siehe 5-kernel

- ▶ zImage
- ▶ am335x-boneblack-wireless.dtb

Herstellung RootFS

Host - SD-Card - Target(BBB)

Host

path-to-target-root

- ├ bin
- ├ dev
- ├ etc
- ├ home
- ├ lib
- ├ linuxrc
- ├ made-yyyy-mm-dd
- ├ proc
- ├ sbin
- ├ share
- ├ sys
- └ usr

SD-Card

path-to-sd-card

- ├ bin
- ├ dev
- ├ etc
- ├ home
- ├ lib
- ├ linuxrc
- ├ made-yyyy-mm-dd
- ├ proc
- ├ sbin
- ├ share
- ├ sys
- └ usr

BBB

/

- ├ bin
- ├ dev
- ├ etc
- ├ home
- ├ lib
- ├ linuxrc
- ├ made-yyyy-mm-dd
- ├ proc
- ├ sbin
- ├ share
- ├ sys
- └ usr

Transport

Host → *SD-Card*: `rsync -av path-to-target-root path-to-sd-card`

SD-Card → **BBB**: einstecken

SD-Card → *Host*: `rsync -av path-to-sd-card path-to-target-root`

Herstellung

Teil 2: RootFS

LibC Siehe `17-build/tools/glibc.sh`

toolchain `17-build/tools/gcc.sh`

UNIX Verschiedene *flavours*

- ▶ **minimal:** Siehe `19-minimal`
- ▶ **busybox:** Siehe `17-build/tools/busybox.sh`

Ausbau

`ssh` Siehe `17-build/tools/openssh.sh`

`WLAN` Siehe `6-assembly`

`init` Siehe `6-assembly`

Toolchain

Die *toolchain* kommt überall vor und wird per *link* in die einzelnen Verzeichnisse importiert `ln -s`.

- ▶ Machen Sie einen eignen Abschnitt *toolchain-bare*. Benutzen Sie dazu
 - ▶ `17-build/tools/binutils.sh`
 - ▶ `17-build/tools/gcc-bare.sh`

Kernel: in Partition 2

Die beiden Files: *KernellImage DeviceTree* können auch auf Partition 2 liegen.

- ▶ Legen Sie das *KernellImage/DeviceTree* auf Partition 2
- ▶ Passen Sie `u-boot` an
 - ▶ Umgebungsvariable `kernel`

LibC

Neben der `glibc` gibt es noch weitere `libc`'s. www.musl-libc.org/ ist eine kleine `libc`.

- ▶ erzeugen Sie an Stelle von `glibc` `musl`
(Siehe `17-build/tools/musl.sh`)

Toybox

Neben `busybox` gibt es noch ein kleineres UNIX:

www.landley.net/toybox.

- ▶ erzeugen Sie an Stelle von `busybox` `toybox`
(Siehe `17-build/tools/toybox.sh`)

SD Karte

Das Target RootFS auf der SD-Karte kann nur als `root` modifiziert werden

- ▶ passen Sie Ihrem *Host* so an, dass Sie als (gewöhnlicher) *user* das RootFS modifizieren können