

Zusammenbau Assembly

Hans Buchmann FHNW/IME

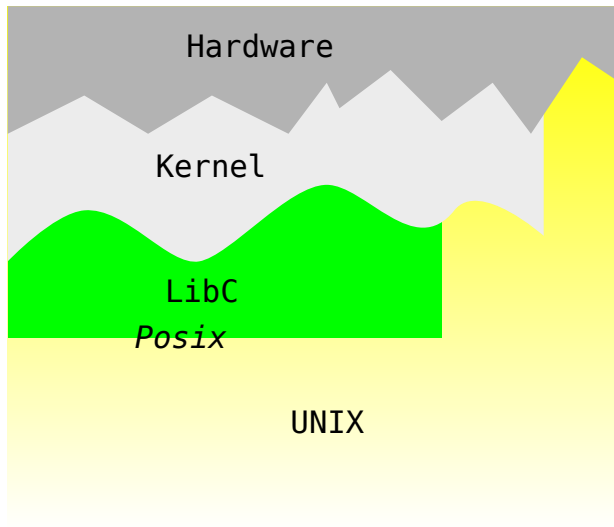
7. November 2017

Um was geht es ?

Ein erstes vollständiges System

- ▶ Bootloader U-Boot
- ▶ **kernel**
- ▶ UNIX

Die Schichten



Das Ziel für BBB

Nach dem Reset:

1. U-Boot startet **kernel**
2. **kernel** startet UNIX
3. UNIX
 - ▶ konfiguriert *ethernet über USB*
 - ▶ startet `ssh` Server

Was wir schon haben

Toolchain: download

U-Boot : selber gemacht

kernel: selber gemacht

root Filesystem: download

▶ libc/UNIX

Die Partitionen und Filesysteme

p1 bootfs:vfat $\approx 20\text{MiB}$

- ▶ U-Boot
 - ▶ MLO
 - ▶ u-boot.img
 - ▶ uEnv.txt Konfiguration
- ▶ **kernel**
 - ▶ zImage
 - ▶ am335x-boneblack-wireless.dtb

p2 rootfs:ext4 $\approx 200\text{MiB}$

- ▶ etc/init.d/rcS init-script

U-Boot

Wichtige Befehle

- ▶ `boot` startet `bootcmd`
- ▶ `fatload mmc 0 addr file`
- ▶ `setenv key value`
- ▶ `run script`

Remark: Siehe www.denx.de/wiki/view/DULG/UBootCmdGroupEnvironment

U-Boot

Wichtige Variablen

- ▶ `bootcmd` für U-Boot `boot`
- ▶ `bootargs` für den **kernel**

U-Boot

Wichtiger File

- ▶ `uEnv.txt` setzt:
 - ▶ `bootcmd`
 - ▶ `load-script` für den **kernel**
 - ▶ `bootargs`

Konfiguration

USB-Gadget Support

```
buchmann@buchmann:~/fhnw/edu/tinL/5-kernel
.config - Linux/arm 4.14.0-rc4 Kernel Configuration
→ Device Drivers → USB support → USB Gadget Support

USB Gadget Support

Arrow keys navigate the menu. <Enter> selects submenus ---> (or empty
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 [ ]

--- USB Gadget Support
[ ] Debugging messages (DEVELOPMENT)
[ ] Debugging information files (DEVELOPMENT)
[*] Debugging information files in debugfs (DEVELOPMENT)
(500) Maximum VBUS Power usage (2-500 mA)
(2) Number of storage pipeline buffers
USB Peripheral Controller --->
< > USB Gadget functions configurable through configs
<*> USB Gadget precomposed configurations (Ethernet Gadget (wit
Ethernet Gadget (with CDC Ethernet support)
[*] RNDIS support
[*] Ethernet Emulation Model (EEM) support

<Select> <Exit> <Help> <Save> <Load>
```

Init Script

`target-root-version.tar.gz`

- ▶ `/etc/init.d/rcS` das *Init-Script*
- ▶ `ifconfig` für Internet
- ▶ `sshd` Server für Verbindung

Workflow

Notationen

sd-card die Partition vom rootfs auf der SD Karte

target-root-V.tar.gz das heruntergeladene rootfs

target-root das rootfs von **BBB** auf dem *Host*

Workflow

schrittweise Verbesserung

1. Initialer Download `target-root-V.tar.gz`
2. `target-root`
 - ▶ `tar -xf target-root-V.tar.gz -C target-root`
3. Transfer auf *sd-card*
 - ▶ `rsync -av target-root/ sd-card/`
 - ▶ `sync`
4. Test/Konfiguration auf dem **BBB**
5. Update auf dem *Host*
 - ▶ `rsync -av sd-card/ target-root/`
6. → 4

Die Files

Partition 1: vfat

- ▶ MLO
- ▶ u-boot.img
- ▶ zImage
- ▶ am335x-boneblack-wireless.dtb

Partition 1: ext4

- ▶ rootfs auf dem *Host*
- ▶ `rsync -av target-root/ sd-card/`
- ▶ `sync`

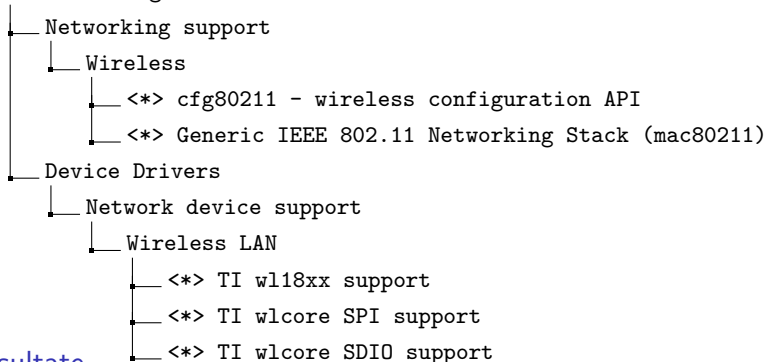
Ziele

wi-fi

- ▶ Konfiguration: **kernel**
- ▶ Neues root-fs: download
- ▶ Konfiguration: wi-fi Zugang
- ▶ schrittweises Vorgehen

Kernel Konfiguration

Kernel Configuration



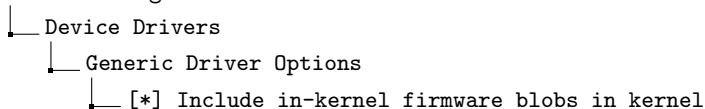
Resultate

► `dmesg | grep wl`

Firmware

Weitere Konfiguration

Kernel Configuration



Resultate binary

- ▶ `dmesg | grep wl`
- ▶ `ip link set wlan0 up`
- ▶ `iw wlan0 scan`
- ▶ Weitere Konfiguration

wpa

- ▶ wpa_passphrase
- ▶ eduroam Versuch

```
network={  
    ssid="eduroam"  
    key_mgmt=WPA-EAP  
    identity="hans.buchmann@fhnw.ch"  
    domain_suffix_match="welcome.fhnw.ch"  
    phase2="auth=MSCHAPV2"  
    password="----"  
}
```

Aufgabe

kernel Ethernet über USB

UNIX Automatisches starten: `/etc/init.d/tcS`

- ▶ Internet: `ifconfig`
- ▶ ssh Server: `sshd`

wi-fi ▶ **kernel** firmware
▶ UNIX wpa

Ein paar tools

- ▶ `touch` - change file timestamps
- ▶ `chown` - change file owner and group

sshd

- ▶ sshd re-exec requires execution with an absolute path
- ▶ Privilege separation user sshd does not exist
- ▶ create group root
 - ▶ `addgroup -g 0 -S root`
- ▶ create user root
 - ▶ `adduser -h /home/root/ -s /bin/sh -G root -S -u 0 root`
- ▶ create group/user sshd
 - ▶ `addgroup sshd`
 - ▶ `adduser -D -H -G sshd sshd`
- ▶ create key
 - ▶ `ssh-keygen -t rsa -f /etc/ssh_host_rsa_key`
- ▶ File `/var/empty` gehört root
- ▶ File `/etc/sshd.config`
 - ▶ `PermitRootLogin yes`