

Kernel

Hans Buchmann FHNW/IME

28. Oktober 2015

Ziele

Neuer **kernel** auf **BeagleBoneBlack**

- ▶ Download
- ▶ Setup
- ▶ Konfiguration
- ▶ Kompilation
- ▶ Installation

The Big Picture

grosses Projekt

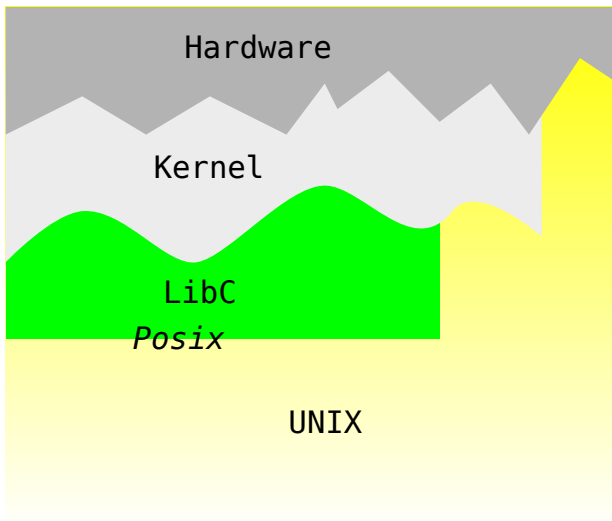
Gegeben Eine grosse Anzahl *source* Files

Gesucht ein einziger File: das **Image**

Lösung Ähnlich wie in **4-devel**

- ▶ Toolchain
- ▶ Makefile

Die Schichten



Kernel Grosses Projekt

Was ist einfach ?

- ▶ **kernel** hängt nicht von anderen Software Komponenten ab
 - ▶ stand alone
- ▶ Braucht nur `make` und *toolchain*

Was ist schwierig ?

- ▶ Konfiguration
 - ▶ Wahl der richtigen *source* Files für das **Image**

<https://github.com/beagleboard/linux>

Mehrere Möglichkeiten

- ▶ das ganze git repository
- ▶ nur die letzten n Versionen `--depth= n`
- ▶ zip File

Tools

Siehe 4-devel

`toolchain` <https://sourceforge.net/projects/fhnw-tinl/files>

- ▶ `beaglebone-black-toolchain-64bit.tar.bz2`
- ▶ Prefix: `arm-linux-gnueabihf-`
 - ▶ beschreibt:
 - ▶ Architektur: `armv7`
 - ▶ **A**pplication **B**inary Interface: `gnueabihf`

`make` Normales `make`

- ▶ **kernel** Herstellung:
 - ▶ `make cmd`

Wo ist was ?

```
tinL
├── 5-kernel
│   ├── build ..... generated kernel files
│   │   ├── .config ..... die aktuelle Konfiguration
│   ├── tools ..... for making
│   │   ├── kernel.sh ..... wrapper to kernel Makefile
│   ├── config
│   │   ├── config.sh ..... for kernel.sh
│   │   └── kernel.config ..... 'gute' kernel Konfiguration
└── resources
    ├── beaglebone-black
    └── linux ..... the source tree
```


Erste Konfiguration

```
sh kernel.sh help
```

- ▶ `sh tools/kernel.sh bb.org_defconfig`
 - ▶ Vordefinierte Konfiguration
- ▶ `sh tools/kernel.sh.sh menuconfig`
 - ▶ Anpassung der Konfiguration

Kompilation

- ▶ `sh tools/kernel.sh zImage`
 - ▶ erzeugt `build/arch/arm/boot/zImage`
- ▶ `sh tools/kernel.sh dtbs`
 - ▶ erzeugt `build/arch/arm/boot/dts/am335x-boneblack-wl1835mod.dtb`
Devicetree

Remark: *Devicetree* später behandelt

Installation auf SD-Card

- ▶ Kopiere

 - `Image` `build/arch/arm/boot/zImage`

 - `Devicetree` `build/arch/arm/boot/dts/am335x-boneblack.dtb`

auf

 - ▶ SD-Card *boot-partition*

Startup

Bootloaders bei eingebetteten Systemen

Reset →	fbl	<i>first stage bootloader</i>
	sbc	ev. weitere bootloader <i>second stage bootloader</i>
	u-boot	Hier haben wir Zugriff
	kernel	Konfiguration/Parametrisierung
	linux	

Startup

Bootloader beim *Host*

Reset →	BIOS	<i>first stage bootloader</i>
	grub	<i>second stage bootloader</i>
	kernel	Konfiguration/Parametrisierung
	linux	

<http://www.denx.de/wiki/U-Boot/WebHome>
ein typischer Bootloader für eingebettete Systeme

- ▶ Kommandozeilen
- ▶ Verbindung zum *Host* via RS232/USB
 - ▶ *Host*: `minicom -D /dev/ttyUSB N , $N = 0, 1..$`
 - ▶ 115200 Baud 8N1
 - ▶ **NO** Handshaking
- ▶ Kopiert Daten von
 - ▶ SD-Karten
 - ▶ Netz

in das Memory vom **BBB**

Ein paar typische Befehle

- ▶ `help`
- ▶ `printenv` Zeigt die Umgebung
- ▶ `md addr` Memory display
- ▶ `fatls mmc p vfat sd-card partition p`
- ▶ `fatload mmc p memAddr file`
- ▶ `tftpboot [loadAddress] [[hostIPaddr:]bootfilename]`
- ▶ `bootz kernelAddr - fdt`

U-Boot Bedienung

Siehe `5-kernel/tools/u-boot.cmd`

► copy paste

Remark: kann sich ändern

Übergang U-Boot-Kernel

Bootargs

```
bootargs  
'root=/dev/mmcblk0p2 rw rootdelay=1 init=linuxrc console=tty00,115200n8'
```

▶ `kernel-source/Documentation/kernel-parameters.txt`

Die Kernel messages

Starting kernel ...

Booting Linux on physical CPU 0x0

Initializing cgroup subsys cpuset

Initializing cgroup subsys cpu

Initializing cgroup subsys cpuacct

...

Kernel command line: root=/dev/mmcblk0p2 rw rootdelay=1 init=linuxrc console=tt

...

Waiting 1 sec before mounting root device...

EXT4-fs (mmcblk0p2): couldn't mount as ext3 due to feature incompatibilities

EXT4-fs (mmcblk0p2): couldn't mount as ext2 due to feature incompatibilities

EXT4-fs (mmcblk0p2): mounted filesystem with ordered data mode. Opts: (null)

VFS: Mounted root (ext4 filesystem) on device 179:2.

Workflow

schrittweise Herstellung

0 Setup der Toolchain

1 Default Konfiguration (falls vorhanden)

- ▶ `sh tools/kernel.sh bb.org_defconfig`

2 Herstellung

- ▶ `tools/kernel.sh zImage`

3 Transfer/Start/Test auf **BeagleBoneBlack**

- ▶ U-Boot

4 (Re)Konfiguration

- ▶ `sh tools/kernel.sh menuconfig`

→ 2 `ev. cp build/.config config/kernel.config`

Start auf BeagleBoneBlack

- ▶ u-boot
 - ▶ UART-USB Kabel
 - ▶ Befehle in `scripts/u-boot.cmd`

Übung: kernel

- ▶ **BeagleBoneBlack** *default* Konfiguration
 - ▶ herstellen
 - ▶ auf SD-Karte
 - ▶ ausprobieren
- ▶ Die *default* Konfiguration ändern:
 - ▶ nur eine CPU
 - ▶ keine ALSA Soundkarte
 - ▶ ...