

Ein kleines UNIX

Hans Buchmann FHNW/ISE

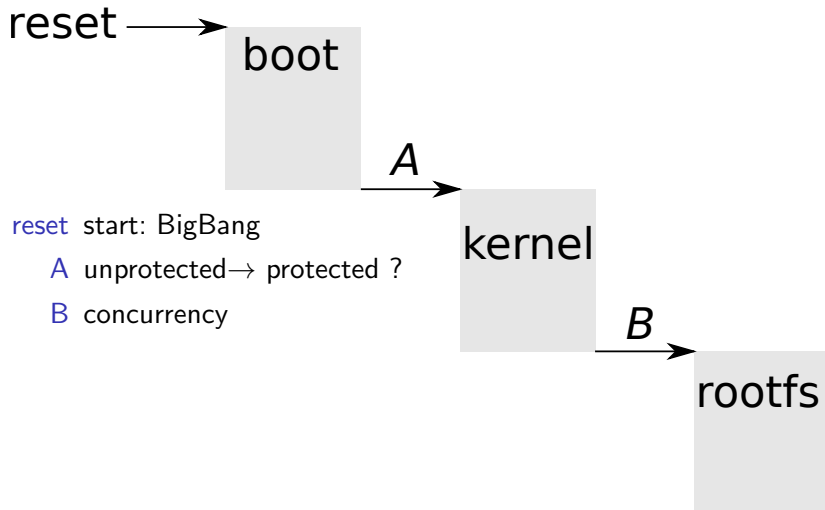
20. November 2019

Ein kleines UNIX

- ▶ die einzelnen Komponenten
- ▶ für Test
- ▶ übersichtlich
- ▶ grosse Systeme haben ähnliche Komponenten

Die grossen Blöcke

Übergänge



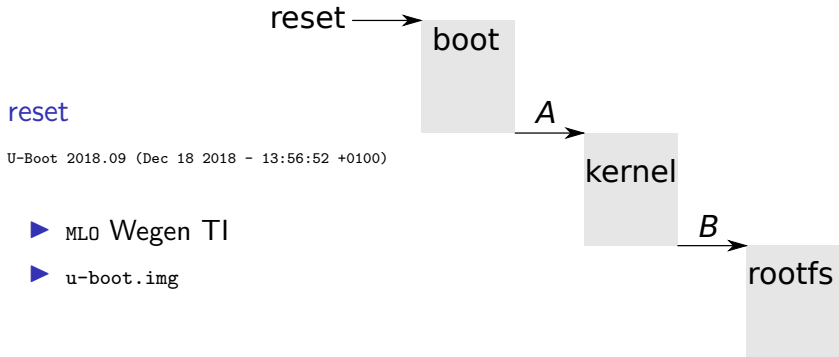
reset start: BigBang

A unprotected → protected ?

B concurrency

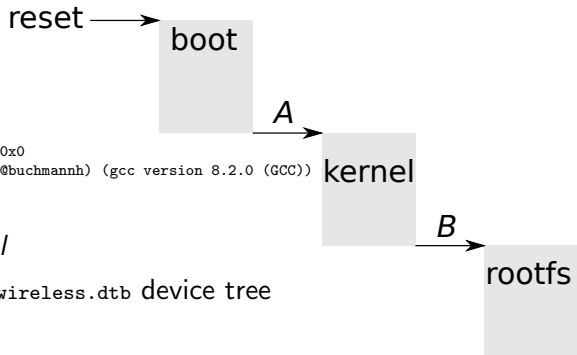
Boot:u-boot

Zwei Files



Kernel

Zwei Files



A

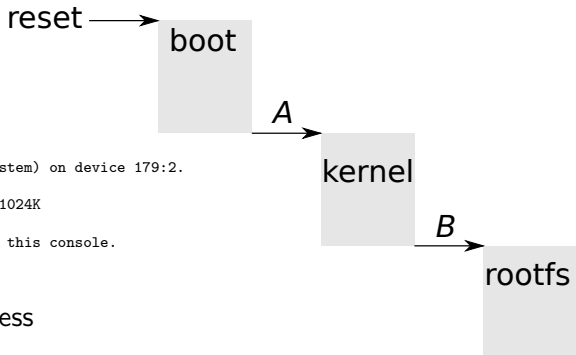
Booting Linux on physical CPU 0x0

Linux version 4.19.5 (buchmann@buchmannh) (gcc version 8.2.0 (GCC))

- ▶ zImage *Der kernel*
- ▶ am335x-boneblack-wireless.dtb device tree

RootFS: Viele Files

Unser Interesse



B

```
VFS: Mounted root (ext4 filesystem) on device 179:2.  
devtmpfs: mounted  
Freeing unused kernel memory: 1024K
```

Please press Enter to activate this console.

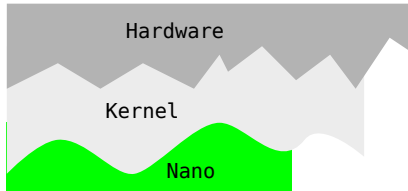
▶ linuxrc Init-Process

▶ ...

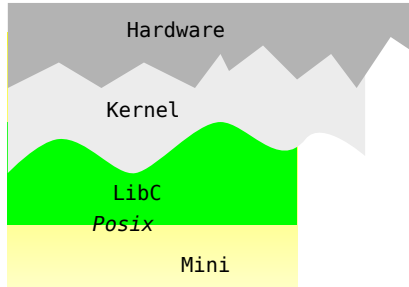
RootFS

Flavours

- ▶ nano
 - ▶ Assembler ohne *libraries* `s-nano.S`
 - ▶ **C** fast ohne *libraries* `c-nano.c`
- ▶ mini
 - ▶ *libraries*
 - ▶ static
 - ▶ dynamic
- ▶ full
 - ▶ busybox
 - ▶ ssh
 - ▶ ...



- ▶ `config/Makefile`
- ▶ `src/s-nano.S`
 `target-root` nichts
- ▶ `src/cc-nano.cc`
 `target-root` `libc.a` wegen `syscall`

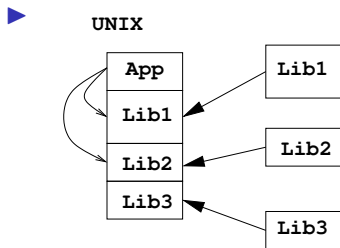


- ▶ `config/Makefile`
- ▶ `src/cc-mini.cc`
 - ▶ `static`
`target-root libc.a`
 - ▶ `dynamic`
`target-root libc.so, loader`

Statische/Dynamische Bibliothek

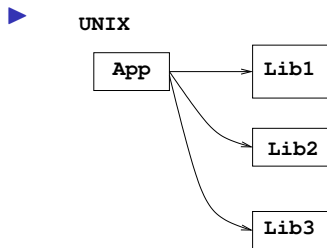
Kopie vs. Referenz

Static



► frühes Binden

Dynamic



► spätes Binden

—► **copy**
—> **reference**

Dynamische Bibliothek:der Loader

am Beispiel `mini.c`

- ▶ wir sind in `work`
- ▶ `gcc ../src/mini.c -o mini`
- ▶ `file mini`
- ▶ start loader
- ▶ start loader mit *executable*

Verzeichnisstruktur

Siehe *17-build*

Aufgaben

- ▶ Erzeugen Sie ein:
 - ▶ **s-nano**
 - ▶ **cc-nano**
 - ▶ **cc-mini-static**
 - ▶ **cc-mini-dynamic**auf dem **BBW** RootFS
- ▶ ändern Sie den `Makefile` so ab, dass er **s-nano**, **cc-nano**, **cc-mini-static** und **cc-mini-dynamic** auf den *Host* erzeugt