

# Admin

Hans Buchmann FHNW/IME

19. September 2018

```
git gitlab.fhnw.ch/edu/tinL.git
```

```
▶ git clone https://gitlab.fhnw.ch/edu/tinL.git
```

Prüfung mündliche MSP

- ▶ Alles öffentlich zugängliche Material zu dieser Vorlesung unterliegt der *GNU GENERAL PUBLIC LICENSE*, auch wenn das in den einzelnen Dokumenten nicht explizit angegeben ist.
- ▶ [www.gnu.org/copyleft/gpl.html](http://www.gnu.org/copyleft/gpl.html)

- ▶ GNU/Linux auf dem **BeagleBoneBlack**
- ▶ Einblick in die Mechanismen
- ▶ Umgang mit verschiedenen Tools
- ▶ Weniger programmieren, mehr konfigurieren
- ▶ Schrittweises Vorgehen: (fast) immer lauffähiges System

- ▶ Führen Sie ein Laborbuch bzw. Laborfile

# The Big Picture

Hans Buchmann FHNW/IME

19. September 2018

# The Big Picture

- ▶ GNU/Linux ist:
  - ▶ Software mit klassischen Methoden hergestellt
  - ▶ gross
  - ▶ komplex nicht kompliziert
- ▶ Darum:
  - ▶ Die grundlegenden Mechanismen beachten
  - ▶ Übersicht bewahren
  - ▶ Verzeichnisstrukturen: wo ist was.

## Ein paar Daten: zum GNU/Linux (Kernel)

- ▶  $\approx 17M$  SLOC (Source Lines of Code)
- ▶  $\approx 4.3K$  Verzeichnisse
- ▶  $\approx 62K$  Files davon
  - ▶  $\approx 46K$  {c|h}-Files
  - ▶  $\approx 6K$  Assembler Files
  - ▶  $\approx 2K$  Makefiles
  - ▶ Rest: Makefile, Scripts etc.

### Remark(s):

- ▶  $M = 10^6$   $K = 10^3$
- ▶ Gemacht mit `sloccount`
  - ▶ Siehe `tools/sloc-create.sh` und `tools/sloc-analyze.sh`



# Die ProgrammierSprachen

C Unabhängig von Rechnerarchitektur, Hauptsprache für  
*Bootloader, Kernel, libc*

Assembler Für kleine Anpassungen

Skript Für Routineaufgaben

Makefile Für den Zusammenbau

## Die wichtigsten Werkzeuge

Compiler `gcc gcc.gnu.org`

binutils Sammlung von Programmen<sup>1</sup>  
([www.gnu.org/software/binutils](http://www.gnu.org/software/binutils))

Assembler `as`

Linker `ld`

Maker `make` [www.gnu.org/software/make](http://www.gnu.org/software/make)

---

<sup>1</sup>Liste nicht vollständig

# Die Komponenten

**BootLoader** *reset* Handler, SingleUser

**Kernel** Prozessverwaltung, Treibersammlung

**libc** Normierte (POSIX) Schnittstelle, Kernel-UNIX

**UNIX** Filesystem, Sammlung von Programmen und Daten

## Die Komponenten:Eigenschaften

Komponenten lassen sich:

- ▶ einzeln herstellen
- ▶ kombinieren
- ▶ austauschen

## Die Komponenten:Wo sind sie ?

BootLoader nicht flüchtiger Speicher: z.B. Flash

Kernel RAM

libc RAM

UNIX RAM,Harddisk,MemoryCard,Netz

# Entwicklungsumgebung

Hans Buchmann FHNW/IME

19. September 2018

# Aufbau

0-intro Diese Folien

1-setup **BeagleBoneBlack** in Betrieb nehmen

▶ Verbindung mit dem *Host*

2-unix-use UNIX aus Benutzersicht: Host und Target

...

## Verzeichnisstruktur

```
tinL
├── i-topic .....git versioniert
│   ├── doc
│   └── src
├── devel .....development: git versioniert
├── resources .....binaries nicht versioniert
├── ziele.txt .....pro Woche
├── tools.txt .....wichtige UNIX Befehle
└── ...
```



# Begriffe

**Host** Entwicklungsrechner, GNU/Linux Betriebssystem

**Target** **BeagleBoneBlack**

## Verbindungen: Host ↔ Target

RS232 u-boot *shell*, GNU/Linux console

Ethernet IP, TFTP etc. IP Stack

MemoryCard u-boot, Kernel, UNIX

# Tools

- ▶ UNIX Befehle
- ▶ (Unvollständige) Liste im File `tools.txt`