- Umgebung aufsetzen
  - Linux
  - git repo
- **BeagleBoneWireless** verbinden mit *Host*
  - per USB
    - serielle Schnittstelle
    - Internet
  - per WLAN
- sich auf dem **BeagleBoneWireless** zurechtfinden

- Alles ist ein File
  - `sshfs` mount
  - `mount` SD-Karte
- Netzwerk
  - Host als Proxy
  - Host als Gateway/Router
  - **BBW** via WiFi

- **BBW** kleines Image auf SD-Karte
- Zugriff via serielle Schnittstelle
- via USB am lokalen Netz
  - ssh & sshfs
- via Wi-Fi am Internet

- init script für **BBW**
- 5-kernel
  - basic config
  - USB Gadget
  - WiFi & firmware

- 6-crossdevelopment
  - Programme in `src` auf *Host* & **BBW**
  - Vergleich Zeit von `primes` auf *Host* & **BBW**

- 6-crossdevelopment Zugriff auf die Hardware
  - via Skript `/sys/class/gpio`
    - `led-enable.sh`, `led-blink.sh`
  - via C++ `/sys/class/gpio`
    - `led-enable.cc`, `led-blink.cc`
  - direkt `src/mem.h|cc`
    - `led-direct-0.cc`, `led-direct-1.cc`

- 6-crossdevelopment Zugriff auf die andere Hardware
  - SWITCH $\rightarrow$ LED
- 4-u-boot
  - Herstellung
  - Installation

- HTTP Server `lighttpd`
- CGI Common Gateway Interface

- bestehendes *rootfs* nicht verlieren
- einfache toolchain *)
- kernel *)
- test mit `src/s-bare-init.s`
- glibc *POSIX*
- volle toolchain *)
- test mit `src/cpp-hello-world.cc`
- busybox
- ssh
- wpa

*) fakultativ

- build
  - ssh
  - wifi
- roofs flavours
  - nano
  - mini
  - full

- einen Teile auf dem *Host* bzw. **BeagleBoneWireless**
  - einfaches C-Programm
  - einfaches C++-Programm
  - komplexes C++-Programm

- Konsolidierung
  - aktuelle Toolchain
  - aktueller Kern
    - mit WiFi
  - aktueller rootfs
    - ssh
    - WiFi
- Kernel-Userspace
  - hotplug
  - socket NETLINK KOBJECT UEVENT