Configure

Hans Buchmann FHNW/IME

25. November 2014

Um was geht es ?

Herstellung von Software aus den Quellen

- drei Schritte:

  `configure` für das **RaspberryPi**

  `make` die *binaries* aus den *Quellen* im *Host*

  `install` auf dem **RaspberryPi**

- die Schwierigkeiten:
  - *Host*& **RaspberryPi** sind verschieden

Die Quellen

C/C++ Code

- meistens als
  - **C** Code
  - *tar.gz* File:
    - **name**-*version*.tar.gz
- es gibt aber noch andere Möglichkeiten

Beispiel
rsync-3.1.1

Wichtige Files

- README
- INSTALL
- *.c die Sourcen
- für die Herstellung:
  - Scripts
  - Makefile(s)
- configure: unser Thema

Die Sourcen

Beispiel `main.c`

der Präprozessor `cpp`

- `#define`
- `#ifdef ... #endif`
- `#include`

lässt Code zur Compilation

- zu

  oder
- nicht zu

Von den Quellen zum Programm

Gegeben

- die Quellen
- *Host*
- *target* (**RaspberryPi**)

Gesucht

- das in den Quellen beschriebene Programm lauffähig auf dem *Target* (**RaspberryPi**)
- gemacht auf dem *Host*

Remark: Der einfachere Fall:

- *Host* = *Target*

Das Problem
die Vielfalt

- es gibt viele verschiedene *Host*'s
- es gibt viele verschiedene *target*'s
  - verschiedene Architekturen
    - *ARM*
    - *Intel*

Das Skript `configure`

erzeugt auf dem *Host*

- einen `Makefile`

  für die

- *binaries* auf dem *Target*

Remark: Der einfachere Fall:

- *Host* = *Target*

Host- Target
die Verzeichnisse

Host	*Target*
`somewhere`	`/` ..... root
├── `tc` ..... toolchain	└── *`install`* .. somewhere
├── `sources`	
├── `build`	
├── `target-root` ... libraries	
│ include	
└── *`install`*	

Verbindung

- *Host*:`install` – *Target*:`install`

per `sshfs`, `ftp`, manueller SD-Card transfer

Die option `--prefix` von `configure`

--prefix

- gibt an wo die *binaries* installiert werden sollen

Ziel

- ohne `root` Privilegien auf dem *Host*

Eine mögliche Verzeichnisstruktur
auf dem Host

```
tinL
├── resources ..................................readonly
│   └── sources
└── 8-configure
    ├── tc ..................................... toolchain
    ├── src ..................................... scripts
    ├── target-root ..............................link
    ├── work ..where to install, connected with RaspberryPi
    └── build-source .....................where to build
```

Die Scripts
in `src`

- *sources*`/configure` in
  - `build-`*source*

  aufrufen
- `make`, `make install` in
  - `build-`*source*

  aufrufen
- resultat in
  - `work`

Aufgaben
auf dem *Host*

- Die *sourcen*:
  - `http://rsync.samba.org`
  - `http://www.lighttpd.net`
  - `http://sox.sourceforge.net`
  - `http://www.openssh.com`

  Remark: die richtigen *sourcen*
- auf dem *Host*
  - unter `work`

Remark(s):
- Ohne `root` Privilegien
- Testen speziell `lighttpd` und `sox`

Aufgaben
auf dem **RaspberryPi**

- Die *sourcen*:
  - `http://rsync.samba.org/`
  - `http://www.lighttpd.net/`
  - `http://sox.sourceforge.net`
  - `http://www.openssh.com`

  Remark: die richtigen *sourcen*
- auf dem **RaspberryPi**
  - unter `work`

Remark(s):
- Ohne `root` Privilegien
- Testen speziell `lighttpd` und `sox`

Auf was ist zu achten

- alle erzeugten Daten in `build-`*`source`*
- Es kann sein, dass im Kernel Treiber fehlen
  - Kernel neu anpassen