Eigene Distribution für Raspberry Pi Mini-Computerboard bauen mit dem yocto Framework

(Bitte vorher Foliensatz „**yocto**“ durchlesen!)

1 Linux-System starten (z.B. von Knoppix-USB-Stick booten)

…

WICHTIG: Auf einem System mit 64bit Kernel, aber 32bit Userspace muss in der Shell, in der mit yocto gearbeitet wird, die Ausgabe von „uname -a“ auf 32bit angepasst werden, sonst versucht yocto, 64bit Binaries für das Buildsystem zu erzeugen, was auf dem 32bit Host-System schief geht,

```
setarch i386 /bin/bash
```

(gilt nur für die laufende Shell). Ausgabe von „`uname -a`“ vorher und nachher vergleichen! Nachher sollte „i686“ statt „x86_64“ in der Ausgabe auftauchen.

2 yocto installieren

Vorbereitet auf USB-Stick befindet sich ein komprimiertes `tar`-Archiv mit den Daten, die in einen normalen Benutzerordner ausgepackt werden (oder dort, wo Platz ist).

Die Installation der Overlays (zusätzliche meta-Verzeichnisse und Konfiguration zu yocto) erfolgte nach der Vorlage unter

http://git.yoctoproject.org/cgit/cgit.cgi/meta-raspberrypi/about/

Ins Home-Verzeichnis wechseln:

```
cd
```

Daten auspacken, z.B.:

```
tar zxvf /media/sdb1/yocto-mit-openembedded+raspi.tar.gz
```

→ Es wird ein Verzeichnis „`poky`“ erzeugt, in dem alle Vorlagen für **yocto** vorhanden sind. Die Raspberry-Pi-Vorlage ist auch schon integriert.

Mit „`ls -l`“ sollte nun das „`poky`“-Verzeichnis sichtbar sein.

3 Anpassen für das „Target“ Raspberry Pi

Kommando:

```
source poky/oe-init-build-env
```

Was passiert hierbei?
Es werden Verzeichnisse für den Build-Prozess angelegt und eine neue Konfiguration erstellt, die modifiziert werden kann.

In der Datei conf/bblayers.conf sollten Zeilen für die raspberry pi spezifischen Overlay stehen:

```
  ${TOPDIR}/../meta-raspberrypi \
```

Bitte das abschließende " in der letzten Zeile nicht aus Versehen löschen!

Ergänzen in der Datei `conf/local.conf` (richtige Stelle suchen!), falls noch nicht vorhanden:

```
  MACHINE ??= "raspberrypi2"
```

Alternativ: `raspberrypi0` oder `raspberrypi2` oder `raspberrypi3`

`raspberrypi2 sollte auch auf Modell 3 laufen.`

Bevor es mit dem Bauen los geht, muss oft auf dem Host-System noch das Paket `chrpath` und `python2.7` installiert werden. Beispiel für Debian:

```
sudo apt-get update
sudo apt-get install chrpath python2.7
```

4 Los geht‘s

```
bitbake rpi-hwup-image
```

→ In ca. 2 Stunden sehen wir uns wieder…. ;-)

Der Bauvorgang darf jederzeit durch Steuerung-C unterbrochen werden (und bricht auch automatisch ab, falls etwas nicht korrekt heruntergeladen oder compiliert wird), und kann mit dem gleichen Kommando fortgesetzt werden.

Mehr Features bieten die Targets `rpi-basic-image` und `rpi-test-image.`

5 Fertig

Im Verzeichnis `tmp/deploy/images/raspberry/` landen die fertigen Images, die wie gewohnt auf SD-Karte geflasht werden können.