Embedded Linux

mit dem BeagleBone Black

Autor: Michael Schäferling
Datum: 2015-03-18

1. Aufbau des "rt-bone"

Alle Komponenten des "rt-bone" sind über einen zentralen 4Port-USB-Hub mit dem Host-PC (Entwickler-PC) verbunden. Zur Stromversorgung des BeagleBone-Black (BBB) ist dieses über die USB-Buchse (mini-USB 'P4') am USB-Hub angeschlossen. Weiterhin ist am USB-Hub noch ein USB-Seriell-Wandler (FT232) angeschlossen, der mit den Pins der seriellen Schnittstelle ('J1') des BBB verbunden ist. Für Echtzeit-Versuche ist zudem ein Logic-Analysator mit dem Hub und vier GPIOs des BBB verbunden.

Für den Zugang zum BBB stehen somit generell die serielle Schnittstelle und Netzwerk (LAN) zur Verfügung.
Parameter dieser Schnittstellen:

- Seriell: → 115200, 8N1, keine HW/SW Flusskontrolle
  → am Host meist "/dev/ttyUSB0"
- LAN: → IP z.B. über DHCP des Host-Rechners
  → Liste aktiver Knoten am Host ermitteln: '`nmap -sP 192.168.0.1/24`'
  (für Rechner rt1 - rt10 im RT-Labor)

2. U-Boot

Zunächst muss eine Konsole zur seriellen Schnittstelle geöffnet werden (z.B. '`minicom -D /dev/ttyUSB0`').
Nach Start/Reboot des BBB: Enter-Taste betätigen bis Boot-Prompt ('U-Boot#') erscheint, dann:

1. Setzen der Client und Server IPs, lädt auch automatisch das Kernel-Image via TFTP.
   `dhcp`

2. Holen des Kernels via TFTP -> wird schon mit "dhcp" ausgeführt
   `(tftp 0x82000000 uImage)`

3. Holen des Device-Tree-Blobs (DTB) via TFTP:
   `tftp 0x88000000 am335x-boneblack.dtb`

4. Setzen der Boot-Argumente:
   `setenv bootargs console=ttyO0,115200n8 root=/dev/mmcblk0p2 ro rootfstype=ext4 rootwait`

5. Booten des Kernels mit dem DTB:
   `bootm 0x82000000 - 0x88000000`

3. Linux

Verbinden auf das BBB mittels seriellem Zugang oder LAN (siehe Punkt 0). Über LAN steht ein SSH-Zugang bereit (je nach Distribution).
Zugangsdaten:

- root / kein Passwort

4. Logic-Analysator, Echtzeit-Versuche

Am BBB ist ein "Saleae Logic 4" Logic-Analyzer angeschlossen (GPIOs 30, 31, 48, 60). Dieser wird mit dem Programm „`Logic`“ betrieben (auf den Rechnern rt1 - rt10 installiert, ggf. muss vorab die Arbeitsumgebung mit „`source /opt/env_all.sh`“ vorbereitet werden).
Im Home-Verzeichnis von „root“ stehen Echtzeit-Demos im Ordner „rt-demos“ bereit.

5. Anhang

5.1. Linux – Erstellen der Distribution mit Yocto

Eine passende Linux-Distribution für das BBB kann z.B. mit Hilfe des Yocto-Projekts erstellt werden. Auf dem Rechnert `rt10` liegt im lokalen Account `rtlabor` unter `elinux/yocto-poky` bereits eine für das BBB angepasste Kopie des Projektes im Ordner `build`.

Hilfreiche Links zum Anpassen und Erstellen der Distribution:

- Wiki des Yocto-Projekts:
  https://wiki.yoctoproject.org
- HowTo zum Erstellen und zur Inbetriebnahme:
  http://android.serverbox.ch/?p=1273

5.2. TFTP im RT-Labor

In der Entwicklung kann es oft gewünscht sein, dass U-Boot das Kernel-Image und den DTB (siehe Kap. 2 „U-Boot “) über TFTP vom Entwicklungsrechner holt (statt wie üblich von der SD-Karte).

Dazu steht auf den Rechnern rt1 – rt10 ein TFTP-Server (wird durch dnsmasq bereitgestellt) auf den zusätzlichen Netzwerkkarten für lokalen Labor-Betrieb zur Verfügung. Nähere Informationen hierzu (z.B. genutzter IP-Adressereich) im Dokument „**Internet-Zugang für Embedded-Boards**, 1. LAN“.

Das Verzeichnis für die per TFTP bereitgestellten Dateien lautet auf rt1 – rt10 „/srv/ftpd“.