The embedded Linux quick start guide lab notes

Embedded Linux Conference Europe 2010

Date: Tuesday 26th October
Location: DeVere University of Arms Hotel, Cambridge
Room: Churchill Suite
Presenter: Chris Simmonds, chis@2net.co.uk

1. Overview

These notes are to help you get up and running with embedded Linux on the LPC3250 stick. They were tested using Ubuntu 10.04 Desktop i386: they will probably work on other configurations as well, with a little tweaking...

At this point, you should have

- A laptop with a version of Linux installed, for example Ubuntu 10.04, with these packages
  - patch
  - ncurses-devel
  - nfs-kernel-server
  - tftpd-hpa
  - minicom
- 1 Ethernet cross-over cable

Set the IP address of the Ethernet port to be 192.168.1.1.

You should have downloaded these files:

http://www.embedded-linux.co.uk/downloads/elce-2010-linux-quickstart.tar.bz2

and either:
http://www.angstrom-distribution.org/toolchains/angstrom-2010.4-test-20100422-i686-linux-armv5te-linux-gnueabi-toolchain-qte-4.6.2.tar.bz2

or:

http://www.angstrom-distribution.org/toolchains/angstrom-2010.4-test-20100421-x86_64-linux-armv5te-linux-gnueabi-toolchain-qte-4.6.2.tar.bz2

2. Install the tool chain

First you need to install a tool chain for the target processor, which is an ARM926JE so you need a compiler that will generate armv5te instructions.

For 32-bit Linux install it like so (assuming that the file is in your ~/Downloads directory):

```
cd /
sudo tar xjf ~/Downloads/angstrom-2010.4-test-20100422-i686-linux-armv5te-linux-gnueabi-toolchain-qte-4.6.2.tar.bz2
```

For 64-bit Linux:

```
cd /
sudo tar xjf ~/Downloads/angstrom-2010.4-test-20100421-x86_64-linux-armv5te-linux-gnueabi-toolchain-qte-4.6.2.tar.bz2
```

Add this line to .profile or .bash_profile

```
PATH=/usr/local/angstrom/arm/bin/:$PATH
```

Then log out and back in to pick up the change in your current session.

Test that it works

Compile a small program, called hello-arm.c in this example, to make sure everything is set up correctly.

```
arm-angstrom-linux-gnueabi-gcc hello-arm.c -o hello-arm
```

Then:

```
$ file hello-arm
hello-arm: ELF 32-bit LSB executable, ARM, version 1 (SYSV), dynamically linked (uses shared libs), for GNU/Linux 2.6.16, not stripped
```

3. Compile the kernel

Extract the files for the target board into your home directory:

```
cd ~
tar xjf ~/Downloads/elce-2010-linux-quickstart.tar.bz2
```

Extract the kernel and apply the board support package:

```
cd ~
tar xjf ~/elce-2010-linux-quickstart/linux-2.6.34.tar.bz2
cd linux-2.6.34
patch -p 1 < ~/elce-2010-linux-quickstart/linux-2.6.34-lpc3250-bsp.patch
```

Copy the mkimage utility that is needed to create the U-Boot header

```
sudo cp ~/elce-2010-linux-quickstart/mkimage /usr/local/angstrom/arm/bin
```

Select the default configuration for the target and build a kernel with the U-Boot header:

```
export ARCH=arm
export CROSS_COMPILE=arm-angstrom-linux-gnueabi-
make lpc3250stick_defconfig
make oldconfig
make uImage
```

Check:

vmlinux size = 36145312

uImage size = 1672876

Later on you will load the kernel onto the target using the tftp daemon that you (should have) installed earlier, so copy uImage from arch/arm/boot to the top-level directory of the tftp server. If you are using Ubuntu and the tftpd-hpa package, that will be /var/lib/tftpboot so the command will be

```
sudo cp arch/arm/boot/uImage /var/lib/tftpboot
```

Other tftp daemons or Linux distributions may use a different directory. Typing "ps ax | grep tdtpd" should give you a clue.

4. Create the root file system

Finally, extract the Ångström root file system

```
cd ~
mkdir rootdir
cd rootdir
sudo tar xjf ~/elce-2010-linux-quickstart/rootdir-lpc3250-stick.tar.bz2
```

You will be using NFS to mount it; you should have installed the nfs server earlier. Now you need to export the root directory for the project, so edit the /etc/exports (you will need root permissions) and add the path as shown below, replacing [your user name] with your actual user name:

```
/home/[your user name]/rootdir *(rw,sync,no_subtree_check,no_root_squash)
```

Restart the server

```
sudo /etc/init.d/nfs-kernel-server restart
```

5. Boot up!

Load the FTDI USB-serial driver with the vendor and product IDs used by the LPC3250 stick:

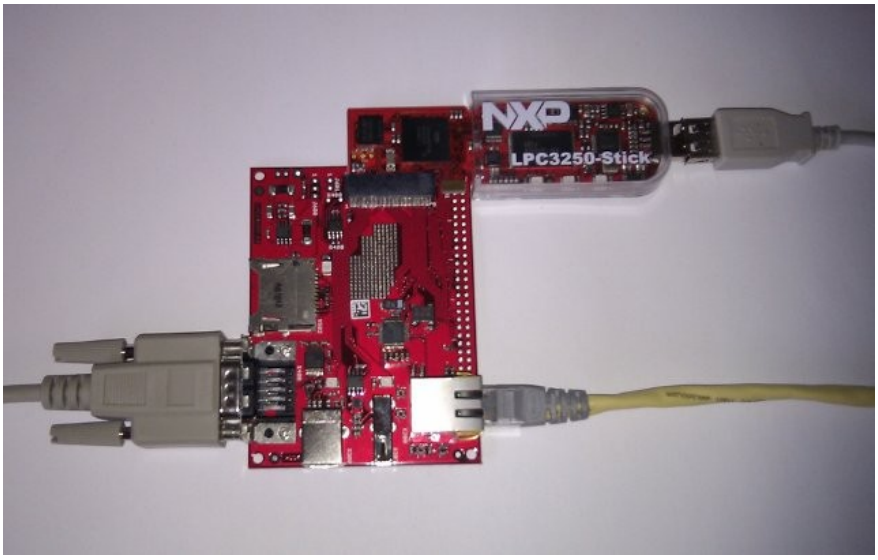
```
sudo modprobe ftdi_sio vendor=0x0640 product=0x0026
```

Type `minicom -s` to start it in setup mode. Select "serial port setup" and press 'a' to change the Serial Device to /dev/ttyUSB1. Press 'e' and select speed 115200 no parity

8 data bits, one stop bit (1152008N1) and press enter. Select No hardware flow control and no software flow control.

Press enter, then select “Save setup as dfl” and press enter again. Finally “Exit from Minicom”.

Plug the LPC stick into the COM board and plug your Ethernet cross-over cable between the COM board and your laptop, as shown:



Plug the LPC stick into a USB port, wait for a couple of seconds to allow udev to create the device node /dev/ttyUSB1 and then start minicom with line-wrap turned on using the command “minicom -w”. You should see these messages:

```
LPCStick 3250 Board
Build date: Oct  8 2010 09:53:50


U-Boot 2009.03-rc1 (Sep 24 2010 - 05:09:44)

DRAM:  64 MB
NAND:  128 MiB
In:    serial
Out:   serial
Err:   serial
Hit any key to stop autoboot:  3
```

Configure U-Boot to load uImage and boot Linux, mounting the root file system over NFS:

```
setenv ipaddr 192.168.1.101
setenv serverip 192.168.1.1
setenv npath /home/[your name]/rootdir
setenv console console=ttyS0,115200n8
setenv bootargs ${console} root=/dev/nfs rw nfsroot=${serverip}:${npath}
ip=${ipaddr} ethaddr=${ethaddr}
saveenv
```

Load the kernel to RAM address 0x80100000 and boot it:

```
tftp 80100000 uImage
bootm 80100000
```

If all goes well, you should see the kernel load, boot and mount its root and finally print a login prompt on the console.

Note: there seems to be a problem logging in on the serial console in the current version of the BSP. Instead, log on over the network using ssh:

ssh root@192.168.1.101

Log on

Log on as root, no password, and have a look round, for example:

- What is the kernel command line? (cat /proc/cmdline)
- What type of CPU? (cat /proc/cpuinfo)
- What processes are running? (ps)
- How much free memory? (free)
- What type of file systems are mounted? (cat /proc/mounts)

On your laptop, copy your hello-arm program to ~/rootdir/usr/bin and run it on the target.

6. Put the kernel into flash

On your laptop, copy uImage to ~/rootdir

On the target, install the mtdutils package, which includes the flash_eraseall command that you will use next:

```
opkg install /ipk/mtd-utils_1.3.1-r0.5_armv5te.ipk
```

Then,

```
flash_eraseall /dev/mtd0
nandwrite -p /dev/mtd0 /uImage
reboot
```

In U-Boot

```
setenv bootcmd nand read 80100000 1000000 200000\;bootm 80100000
saveenv
boot
```

The stick should boot up as before, but this time without loading the uImage file via tftp.

7. Put the root file system into flash

On your laptop

```
cd ~/rootdir
sudo tar cf rootdir.tar *
```

On the target, from a Linux shell

```
flash_eraseall /dev/mtd1
mount -t jffs2 mtd1 /mnt
cd /mnt
tar xf /rootdir.tar
reboot
```

Then in U-Boot,

```
setenv bootargs console=ttyS0,115200n8 root=/dev/mtdblock1 rw rootfstype=jffs2
setenv bootcmd nand read 80100000 1000000 200000\;bootm 80100000
saveenv
boot
```

Now the stick should boot up entirely from flash: you should be able to remove the Ethernet cable and boot it by typing reboot.

8. Other things to try

Using the Angstrom package feeds

You will need to connect your LPC Stick to the Internet, which is as simple as connecting to a network with a DHPC server because, as you will see if you look at /etc/network/interfaces, eth0 is defined as "iface eth0 inet dhcp". This rule is *not* executed if you mount the root file system over NFS, but once you have put the root into flash memory it will be.

Then you need to load the package meta information with:

```
opkg update
```

And then you can list all packages and those installed with

```
opkg list
opkg list_installed
```

Some packages that you may find useful

gdbserver	For remote debugging using gdb
oprofile	System profiling tools
procps	Full versions of process utilities, including ps
strace	System trace utilty
thttpd	A small web server