LPC-2478-STK Software Quickstart Guide

By default, the board executes the U-Boot bootloader located in the internal flash. It's menu is displayed on the serial port of the board. The default settings of the bootloader are:

Com port: 115200 bps, 8N1
LAN settings:
IP address: 192.168.0.158
Netmask: 255.255.255.0
Server IP: 192.168.0.240
MAC: 00:de:ad:b0:05:03
A NULL-modem cable is needed to connect the board to a host PC

Programming u-boot to the board:

If the u-boot image gets corrupted for some reason, the user can easily restore it by using the free FlashMagic utility (available from http://www.flashmagictool.com). Select the correct processor type and frequency and under "Step 2 - Erase" select "Erase all Flash+Code read protect". Then load the "u-boot-bin.hex" file available in the "Images" section on the CD. Connect the board and close the "ISP_E" and "RST_E" jumpers . Then press the start button at the utility and, after the programming has completed, the u-boot image should be restored to the factory default.

Remember to use a correct DB9 RS232 cable. Here is an example wiring:
cross wires 2 and 3
cross wires 4 and 6
cross wires 7 and 8

Running uClinux:

As the board does not have a sufficiently large amount of non-volatile memory on board, the uClinux kernel and root filesystem must be transferred to the board every time the system starts up. There are several ways to do that - by using an USB flash drive, by using an SD/MMC card, by using a network TFTP server or through the serial port.

If you wish to use an USB flash drive, copy the files to it, the plug it in the board and issue the following commands

```
lpc-2478-stk # usb start
...
lpc-2478-stk # fatls usb 0
...
 <size of your file> <filename>
lpc-2478-stk # fatload usb 0 <load_address> <filename>
...
```

If you use an SD/mmc card, substitute "***usb***" in the above example with "***mmc***"

You can also use a TFTP server on the network to load the images. This saves time during e.g. development, because the step of transfering the file to an intermediate medium is elliminated. You need to make sure that the "Server IP" option is set correctly for your network (the address can be changed with the "***setenv serverip 192.168.0.240***", "***saveenv***" commands). The command is:

```
lpc-2478-stk# tftpboot <load_address> <filename>
```

If the above methods are unavailable to you for some reaseon, you can still load a file through the serial line. The syntax is:

lpc-2478-stk# ***loady <load_address> 115200***

Then send the file through your terminal program using the ymodem protocol.

As an example of the above, to run uClinux from an USB flash drive:

1. copy the files vmlinux.bin and romfs_5.img to the flash drive
2. plug it in the board, apply power and interrupt the boot process
3. issue the following commands:

```
lpc-2478-stk# usb start
lpc-2478-stk# fatload usb 0 0xa0800000 romfs_5.img
lpc-2478-stk# fatload usb 0 0xa0008000 vmlinux.bin
lpc-2478-stk# go a0008000
```

Compiling u-boot

The u-boot image has been built under a cygwin host using the GNUARM-gcc_3.4.3 available under the "Utils" folder. After you have your buld enviroenment set up correctly (basically installing cygwin and GNUARM under windows or installiing an arm gcc cross compiler under linux), extract the sources from the "U-boot" directory and issue:

```
$ make lpc_2478_stk_config
$ make
```

After a while you should have a new u-boot image ***u-boot-bin.hex*** .

Recompiling uClinux

You need to use a linux host in order to build the uClinux kernel due to some cross-compiling issuses. You also need to use the Snapgear arm-linux toolchain available from their ftp site (also included on the CD).

First extract the toolchain somewhere:

```
$ cd ~
$ mkdir -p lpc-2478-uclinux/snapgear-cross
$ cd lpc-2478-uclinux/snapgear-cross
$ tar zxf ~/arm-linux-tools-20061213.tar.gz
```

Add the toolchain to your PATH variable:

```
$ PATH=$PATH:~/lpc-2478-uclinux/snapgear-cross/usr/local/bin
```

Extract the sources:

```
$ cd ~/lpc-2478-uclinux/
$ tar zxf uClinux-dist-lpc_2478_stk-20081007.tgz
```

Enter the directory and configure for “Vendor=NXP, Product=LPC2468”

```
$ cd uClinux-dist-lpc_2478_stk
$ make menuconfig
```

Then compile the distribution

```
$ make
```

The ***vmlinux.bin*** kernel image and the ***romfs_5.img*** root filesystem will be stored in the ***uClinux-dist-lpc_2478_stk/images/*** directory.

If you need to make changes to the configuration options, use the command

$ make menuconfig

Developing uClinux applications

It is best to add your custom application to the uClinux build system. See the corresponding documentation file in the uClinux distribution:

uClinux-dist-lpc_2478_stk/Documentation/Adding-User-Apps-HOWTO

The distribution contains a simple framebuffer application that user can use as a starting point. It's located in the following directory:

uClinux-dist-lpc_2478_stk/user/fbtest

It can be selected for inclusion in the uClinux ROMFS image by selecting **"fbtest"** in the **"Miscellaneous Applications"** uClinux user/vendor configuration menu.

On the uClinux board command prompt the frame buffer application can be executed:

$ fbtest

An alternative way to fill the screen with random pixels is to execute the following command

$ dd if=/dev/urandom of=/dev/fb0

Common Questions

Q: What drivers are included in LPC-2478-STK uClinux?
A: The following peripherals have drivers:

- Ethernet
- RS232 (used for Linux console)
- SDRAM
- LCD framebuffer *(no mmap access, though; only file-based through /dev/fb0)*

The following peripherals **lack** support in uClinux:

- Touchscreen
- MP3 decoder
- USB device and host
- IrDA
- PS2
- CAN
- SD/MMC
- Audio

Recent CDs include a framebuffer example for uClinux.

Here is a sample output from the LPC-2478-STK uClinux/busybox shell:

```
# cat /proc/devices
Character devices:
  1 mem
  4 /dev/vc/0
  4 tty
  4 ttyS
  5 /dev/tty
  5 /dev/console
  7 vcs
 10 misc
 13 input
 29 fb
 90 mtd

Block devices:
  1 ramdisk
  7 loop
 31 mtdblock
```

Q: What drivers are included in LPC-2478-STK uBoot?
A: The following peripherals have drivers:

- Ethernet
- RS232 (used for console)
- SDRAM
- USB host
- LCD with boot splash screen
- SD/MMC (**see next question!**)

Q: I have problems using an SD/MMC card under U-Boot.
A: There are known problems with the U-Boot version used on our LPC boards. Here is what our customer Fred Rothganger emailed us:

In case you are curious, the problem has very little to do with the driver, and mostly to do with an improperly relocated interrupt vector block. This is a bug in the version of U-Boot that you distributed on CD, and is triggered any time an interrupt occurs in U-Boot. Since the SD/MMC driver is the first module that actually sets up an interrupt, this is where the crash occurs.

Even more specifically, the bug is in board/lpc_2478_stk/lowlevel_init.c. The first thing the lowlevel_init() function does is (try to) copy the interrupt vectors to the beginning of SRAM. It does this using a GCC inline assembly directive. The problem with the assembly code is that it executes conditioned on NE, rather than unconditionally. Changing it to the unconditional form fixes the problem with crashes during an interrupt. There are several other problems with this code, but I'm too tired now to enumerate them clearly. You should have someone check this code over carefully to ensure it makes sense.

Q. I have problems compiling. What cross compiler should I use?
A. Please check that you are using the following compiler:
http://ftp.snapgear.org/pub/snapgear/tools/arm-linux/arm-linux-tools-20061213.tar.gz
An early batch of CDs contained incorrect compiler. Also triple-check that the snapgear toolchain is the only cross compiler reachable from PATH.

Q: I can't build uClinux. I get strange syntax or link errors.
A: First ensure that you are using the correct cross compiler (the issue is discussed above). You also must have a host compiler, development libraries and tools installed.

Among other libraries and tools you may need your distribution should contain “**zlib1g-dev**” and “**genromfs**” - apt-get these packages if they are not available.

Try to compile with no modification to the source first. The build process has been tested and proves working.

Provide a link to the kernel sources:

```
$ rm -f linux-2.6.x
$ ln -s linux-2.6.24.2-lpc2468-patched linux-2.6.x
```

Clean your uClinux-dist directory:

```
$ make mrproper
```

Get rid of symbolic links left from previous builds:

```
$ rm -f tools/ucfront-gcc
$ rm -f tools/ucfront-g++
$ rm -f tools/cksum
$ rm -f tools/ucfront-ld
$ rm -f lib/libz.a
$ rm -f lib/uClibc
```

Ensure that environment variables are set correctly:

```
$ printenv CROSS_COMPILE
arm-linux-
$ printenv ARCH
arm
```

Finally configure for “Vendor=NXP, Product=LPC2468” and retry the build:

```
$ make menuconfig
$ make
```