published on January 08, 2018 in Tech Tips & Tricks

# Programming the EPM240 devboard on Linux

The supposedly simple task of programming a simple Altera CPLD took me a few days to resolve. Here's what I learned.

There's no Open Source toolchain for Altera. So you must download the multi-GB Quartus Lite software. After installation, I followed the My First FPGA tutorial which mostly worked ok (apart from the 'Megawizard Plug-in Manager' thing). Pin mappings are of course different, but I found the schematic for the EPM240 minimal development board. One can easily see that there's one on-board controlable LED connected through PIN 77.

The big trouble began, when I wanted to flash the design to my devboard. I have a cheap chinese USB Blaster clone. After setting up respective udev rules

```
SUBSYSTEM=="usb", ATTR{idVendor}=="09fb", MODE="0664", GROUP="plugdev"
```

the device becomes visible to the Quartus software, but trying to program it always fails:

```
Error (209040): Can't access JTAG chain
Error (209053): Unexpected error in JTAG server -- error code 44
Error (209012): Operation failed
```

Sometimes jtagd segfaults or otherwise crashes. I couldn't figure out why.

Then I learned more about JTAG and discovered OpenOCD, which also supports the USB Blaster. The minimal thing you need to know is that you start it with some configuration about which device to use

and to which board to connect. It's then accessible through a telnet prompt on port 4444.

Luckily OpenOCD already has fitting configurations for both USB Blaster and the EPM240 CPLD, so I run it as follows:

```
openocd -f interface/altera-usb-blaster.cfg -f cpld/altera-epm240.cfg
```

Apparently it's important that you first switch connect the programmer and only then power up the CPLD, otherwise I'd get errors like:

- Error: JTAG scan chain interrogation failed: all zeroes (CPLD without power)
- USB Blaster connected after board was already powered:

```
Info : JTAG tap: epm240.tap tap/device found: 0xffffffff (mfg: 0x7ff (), part: 0xffff,
Warn : JTAG tap: epm240.tap UNEXPECTED: 0xffffffff (mfg: 0x7ff (), part: 0xffff, ver:
Error: JTAG tap: epm240.tap expected 1 of 1: 0x020a10dd (mfg: 0x06e (Altera), part: 0x
Warn : Unexpected idcode after end of chain: 256 0x03ffffff
Warn : Unexpected idcode after end of chain: 288 0x03030303
Warn : Unexpected idcode after end of chain: 320 0x55030303
Warn : Unexpected idcode after end of chain: 352 0x03030203
Warn : Unexpected idcode after end of chain: 384 0x020a10dd
```

If everything's working the output looked like this:

```
$ openocd -f interface/altera-usb-blaster.cfg -f cpld/altera-epm240.cfg
Open On-Chip Debugger 0.10.0
Licensed under GNU GPL v2
For bug reports, read
http://openocd.org/doc/doxygen/bugs.html
Warn : Adapter driver 'usb_blaster' did not declare which transports it allows; assumi
Info : only one transport option; autoselect 'jtag'
Info : No lowlevel driver configured, will try them all
Info : usb blaster interface using libftdi
Info : This adapter doesn't support configurable speed
Info : JTAG tap: epm240.tap tap/device found: 0x020a10dd (mfg: 0x06e (Altera), part: 0
Warn : gdb services need one or more targets defined
```

In the quartus programmer window you have the option to create an SVF file (File -> Create SVF), this is essentially a recording of JTAG commands which OpenOCD can replay. So I connect using telnet doing exactly this:

```
$ telnet localhost 4444
svf led.svf
```

And voilà, 16 seconds later, the flash process finished and the example runs!

---