Building FPGA Bitstreams with Open-Source Tools

Michael Tretter – m.tretter@pengutronix.de
FOSDEM 2023
About Me

- Michael Tretter
- Embedded Linux developer
- Pengutronix
- Graphics team
Agenda

- Open-source FPGA toolchain
- FPGA-based example system
- Insights and pain points
- Conclusion and next steps
Use Cases for FPGAs

- Real-time requirements
- High data-throughput
- Prototyping
FPGA Toolchain

Verilog → yosys → netlist → nextpnr → config → packer → bitstream
Previous Talks

- The Woos and Woes of Open-Source FPGA-Tools
  - Steffen Trumtrar
  - youtu.be/_0Ipv9Rf1Rc

- Building Open Hardware with Open Software
  - Michael Tretter
  - youtu.be/HgRZpe702JM
# RISC-V Soft-Core CPU

<table>
<thead>
<tr>
<th>VexRiscv</th>
<th>BOOM</th>
</tr>
</thead>
<tbody>
<tr>
<td>(SpinalHDL)</td>
<td>(Chisel)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rocket</th>
<th>CVA6</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Chisel)</td>
<td>(SystemVerilog)</td>
</tr>
</tbody>
</table>
Linux on LiteX

- LiteX as Verilog generator
- Implemented in Migen
- VexRiscv SMP example
How do we add our own custom cores (written in Verilog) to the FPGA bitstream?
Demo System

- LambdaConcept ECPIX-5
- VexRiscv with Linux
- WS2812 LED ring
- CNC handwheel
VexRiscv with Linux

- LiteX platform support → lambdaconcept_ecpix5.py
- VexRiscv core as Verilog created from SpinalHDL
- Wrapped into Migen and Python for LiteX
- Example target with LiteDRAM and LiteSDCard
WS2812 LED Ring

- WS2812 protocol
- LED core in LiteX
- MMIO bus slave
- 4 bytes per LED
CNC Handwheel

- Two signal pulse encoder
- https://shadowcode.io/quadrature-decoder-verilog
- Wrapped in Python for LiteX
- Runs as bus master
Putting it All Together

- handwheel
- encoder.v (custom)
- VexRiscv
- LiteSDCard
- LiteDRAM
- LED ring (custom)
- led.py
- SD card
- RAM
Integration into LiteX

- Create new LiteX target → PtxSoC
- Inherit lambdaconcept_ecpix5.BaseSoC
- Configure and instantiate base SoC
- Reconfigure Pmod I/O pins
- Add WS2812 core
- Add Rotary_ENCODER core
Encountered and Fixed Issues

- Linux failed to access SD card after adding custom cores
- Linux needs to use device tree generated by LiteX

- ROM code changes required rerun of place and route
- Update memory in bitstream after synthesis
Pain Points

- Bugs in Migen are not fixed anymore
- Yocto environment is not as reproducible as expected
- JTAG debugging of VexRiscv cannot be used via ECP5 JTAG
Conclusion

- Adding and customizing LiteX targets is convenient
- Step from “blinky” to SoC is large
- Various system components must be kept in sync
Next Steps

- Kernel CI for Linux-on-LiteX-VexRiscv?
- Linux on VexRiscV boot time?
- Multi-core VexRiscV?
- RISC-V core replacement?
Show me the Source

https://github.com/pengutronix/meta-pxt-fpga
Thank You!

Michael Tretter – m.tretter@pengutronix.de
Steffen Trumtrar – s.trumtrar@pengutronix.de