Open source tools for FPGA development
What is available? What is missing? How can we contribute?

Francesco Robino

FOSS-Sthlm #16
Goal of the talk:
Spread knowledge about FPGAs and introduce open source tools which can be used for the development of FPGA-based projects.

- FPGAs
  - What is an FPGA?
  - What is an FPGA used for?
  - Who is interested in FPGAs?
- FPGA design flow and available open source tools
  - Overview of the design flow
  - Where and how can the FOSS community contribute
  - Available tools today (links)
- Conclusions (and other useful links)
What is a Field Programmable Gate Array (FPGA)?

FPGAs are programmable semiconductor devices composed by a matrix of Configurable Logic Blocks (CLBs).

CLBs are connected through **programmable interconnects**.

IO Buffers (IOBs) are used to communicate with the external world. IOBs can be configured to support different I/O standards (e.g. LVCMOS25).
What is a Field Programmable Gate Array (FPGA)?

"Clean slate" FPGA: programmable **gates** and **routers**

A 3-input, 1-output LUT programmed to compute \((a\&b)\mid c\). Bits \(a, b, c\) are the LUT index, \((a\&b)\mid c\) are the stored values.

Routing \((a\&b)\mid c\) from the LUT computing it to another through **switch boxes**
How to program an FPGA?

- Hardware description languages (HDLs): VHDL, Verilog
- HDL are synthesized (compiled) to a bitstream (the equivalent of binary executable)
- Bitstream is pushed into the FPGA from a configuration memory. This configures the FPGA components with the described functionality
What is an FPGA used for?

FPGAs are often used to:

- accelerate computation (e.g. digital signal processing);
- prototype ASIC designs;
- create autonomous systems through run-time reconfigurability;
- ...much more

Benefits of FPGAs:

- Faster computation using less energy when compared to GPUs/CPUs-based solutions
- Reduce time-to-market and reduce development cost (when ASICs development is too expensive)
- Reliability and maintenance
Who is interested and uses FPGAs

- Companies working with big data and high data bandwidth:
  - reduce power, improve performances of algorithms (e.g. compression and decompression, data encryption) in data centers
  - increasing the number of cores is not the only way to go to improve performances and follow Moore’s law – **heterogeneity** is the key
- Embedded: automotive, telecom
- Military
- Space agencies (run time reconfiguration and fault tolerance)
- Researchers
- ... and many others ...
So, if FPGAs have many advantages, why are they not used more often?

- Expensive when compared to CPUs and DSP based solutions
  - Niche market, small community
- Not easy to program (standard HDLs are not abstract)
- Complex, heavy, closed-source and expensive development tools maintained by few companies

**FOSS development tools for FPGA can help to:**

- Increase number of users and create a larger community
- Get new ideas to simplify the programming methods for FPGAs
- Enable small companies to use FPGAs
- ... [add your own] ...
Design flow

FLOW

CODE

SIMULATE

SYNTHESIS

PACK - MAP

PLACE

ROUTE

BITSTREAM

CODE

Editors: emacs vhdl-mode, vim, ...
Auto documentation: doxygen, vhdocl, vhdl-dot

* ghdl (when the design entry is VHDL)
* iverilog (when the design entry is Verilog)

* Yosys
* Vtr front-end: OdinII - ABC

* Torc: analyze edif and fine grain (architecture)

* Vtr back-end: vpr (theoretical architectures)
* Arachne-pnr (only ice40)

* Torc: analyze xdl and fine grain (physical)

* Vtr back-end: vpr (theoretical architectures)
* Arachne-pnr (only ice40)

* Torc: analyze xdl and fine grain (physical)

* Project icestorm (only ice40)
* Fpgatools (only Spartan6 xc6slx9)
Good editor HDL modes, poor documentation generators, low level design entry

**Tools**

Editors:
- Emacs vhdl-mode, Vim,…

Automatic documentation generator:
- Doxygen, vhdocl, vhdl-dot

**What can we do?**

- New features to the hdl-mode(s) of the editors
- Improve the documentation generators
- Ideas for new abstract programming techniques
Simulate

Good simulators but very few maintainers

Tools

- VhdL: GHDL (written in ADA)
- Verilog: icarus verilog
- GTKWave

What can we do?

- Do not let the projects disappear!
- More maintainers
Check code syntax, analyze the hierarchy of the design and **generate an intermediate hardware description optimized for the selected FPGA family architecture** (netlist).

1. **Generic hardware generation** (HDL to generic HW such as AND/NAND gates)
2. **Logic optimization**: remove redundant logic expressions,...
3. **Binding to FPGA primitives** (e.g. LUTs, MUL,...) of the target FPGA family (e.g. Kintex7)

### Tools
- Yosys (includes ABC), only Verilog
- Vtr project front-end: OdinII + ABC

### What can we do?
- Enable VHDL for Yosys
- Improve logic optimizations
- Extend Yosis to other FPGA families (requires knowledge of primitives for each family)
The netlist of primitives is **mapped into CLB and IOBs** for the specific FPGA (e.g. Xilinx Kintex7 XC7K325T-2FFG900C).

Require detailed knowledge of the specific FPGA architecture (e.g. number of LUTs per CLB, number of LUT inputs,...) Usually represented using a proprietary file (e.g. NCD and XDL files in Xilinx)

**Tools**
- Vtr back-end: vpr (*only theoretical architectures*)
- Arachne-pnr: includes the mapping step (*only iCE40HX*)
- Torc: read and interpret XDL files (physical namespace)

**What can we do?**
- **Document** FPGA architectures when documents are available,
- Use tools like Torc to understand the details of the architectures
- Include those infos in map & pnr tools
Place and route

Placing: decides on the placement of the CLBs and IOBs cells of the target hardware.
Routing: Determines wiring of inputs and outputs of the CLBs and IOBs cells through wiring channels and configuring the configurable switches of the target hardware.
Usually represented using a proprietary file (e.g. NCD and XDL files in Xilinx)

Tools
- Vtr back-end: vpr (only theoretical architectures)
- Arachne-pnr: includes the mapping step (only iCE40HX)
- Torc: read and interpret XDL files (physical namespace)

What can we do?
- Document FPGA architectures when documents are available
- Use tools like Torc to understand the details of the architectures
- Improve exploration algorithms (e.g. constraint programming)
Bitstream generation

Tools
- Project icestorm (only iCE40HX)
- fpgatools (only Spartan 6 xc6slx9)

What can we do?
- **Document** FPGA architectures when documents are available,
- Use tools like Torc, **debit, bitgen** to understand the bitstream formats
- **Connect fpgatools** to arachne-pnr and yosis to have a full toolchain for Xilinx
Available toolchains

Toolchain for Lattice iCE40HX1k
- iVerilog
- Yosis
- arachne-pnr
- icestorm

Toolchain for Xilinx xc6slx9
- iVerilog
- Yosis
- ...
- fpgatools
Where can I start with FPGAs?

- Zynq based boards (Parallella, Zybo)
- Get an iceStick (ca 200 kr) and:

  ```
yosis -p read_verilog example.v; synth_ice40 -blif example.blif
  arachne-pnr -d 1k -p pin_file.pcf -o example.txt example.blif
  icepack example.txt example.bin
  iceprog example.bin
  ```
Links

- vhdl-dot: https://code.google.com/p/vhdl-dot/
  https://github.com/frobino/vhdl-dot-resurrection/
- GHDL: http://sourceforge.net/projects/ghdl-updates/
- iVerilog: http://iverilog.icarus.com/
- Yosys: http://www.clifford.at/yosys/
- Vtr: https://code.google.com/p/vtr-verilog-to-routing/
- Torc: http://torc-isı.sourceforge.net/
- Arachne-pnr: https://github.com/cseed/arachne-pnr/
- Project Icestorm: http://www.clifford.at/icestorm/
- Fpgatools: https://github.com/Wolfgang-Spraul/fpgatools