The Hardware Design Toolchain
Approaches and State of the Art

Fredo Erxleben

September 17, 2014
We will hate the tools       (FCCM 1996 prediction for 2001)
We will still hate the tools (FCCM 1998 prediction for 2003)
We will merely dislike the tools (FCCM 2000 prediction for 2005)
We [will] hate the tools more (FCCM 2007 prediction for 2012)
Outline

Introduction

The Tools

Chaining Tools together

Open Source

Summary
Introduction

Motivation
Some basics first

The Tools

Chaining Tools together

Open Source

Summary
Why bother?

- Complexity of . . .
  - . . . designs
  - . . . the design process

- Tools are . . .
  - . . . rarely known to developers
  - . . . of unknown usefulness
  - . . . (not?) adequate?
HDLs: General

Everything below RTL is often specific to the HW-vendor → HDLs also serve as interfaces

- VHDL
- Verilog
- SystemC, ABEL, JHDL...
HDLs: VHDL and Verilog

Most commonly used HDLs.

VHDL...
- ...originated from Ada
- ...can cover structure, behavior and some testing
- ...is consequently very complex

Verilog...
- ...is extended to SystemVerilog
- ...resembles C
The extended Gajski-Kuhn Chart

Some basics first
Criteria for Tool Evaluation

- The field of application [5].
- The design of human interaction
- Availability and openness
- Tool-chain integration capabilities
Introduction

The Tools

Specification
HLS, HDL generation and editing
Test and Verification

Chaining Tools together

Open Source

Summary
Specification: Why dedicated tools?

Specification is tightly linked to verification. Verification should be done by automated tools. → Specification should be formalized and machine-readable.
Specification: Where are we now?

- State of the art
  - Text, tables, graphics
  - not formalized at all
- Approaches
  - try UML [1]
  - embed spec. in VHDL [11]
- ...purely academic
HLS from High-level Languages

Idea
Write in an HLL, like you do with software.
Let the tools do the rest...

- C/C++ [13]
- Haskell [3][12]
- Java → JHDL
- Matlab [2]
Approach: Evolutionary Algorithms

Idea

Only describe the environment conditions. Let the algorithm figure out the best solution.

- Might work for optimization
- Not useful for new designs
- Purely academic for now. [8]
HLS by Dedicated Tools

Idea
Have a tool for each specific part of the design. Control it by providing some parameters.

FloPoCo[4]
...is a generator for arithmetic cores for FPGAs
HLS from Visual Representations

Idea
Design by placing and connecting components using a GUI. Translate the resulting layout into an HDL.

Show-off
Qucs
Editing Tool Requirements

Editing tools should help to tame the complexity of HDLs.

- Different views on design
- *Code* navigation
- *Code* completions
- Shortcuts for repeated tasks
- *Code* refactoring
- Enforcement of conventions

Show-off

*SigasiPro*
Also: documentation

High design complexity
→ extensive user and developer documentation required
Documentation quality impacts productivity!

Available tools
VHDLDoc, doxygen-VHDL, VHDocl
Considerations on T+V

- No tools focused on HW-design T+V
- Tool-assisted verification requires formalized spec
- Testing very much limited to
  - runtime assertions
  - waveform inspection
  - (JTAG)
State of the Art

Testing
Whatever your HDL offers you...

Verification
SPIN [10], HDL features
Introduction

The Tools

Chaining Tools together

Considerations regarding Toolchains

Exchange formats

Open Source

Summary
Considerations regarding Toolchains

Chaining: Why bother?

One tool can not cover the whole design process.
But...
Considerations regarding Toolchains

Behold!

One-vendor-to-rule-them-all-policies

Figure: Source: wikipedia
Considerations regarding Toolchains

**Difficulties when creating Toolchains**

- Communication between tools in the chain → exchange formats
- Specialized tools rarely support many formats
- Chain needs to be set up manually

**The true chore**
Which tool is the right one?
Does my tool fit in the chain?
Exchange formats

- **EDIF**
  - Attempt to create industry standard (1985)
  - State: abandoned

- **BLIF**
  - Attempt to create academic standard (1992)
  - State: rarely used at best

- some HDL
  - limited to capabilities of the HDL
Introduction

The Tools

Chaining Tools together

Open Source
Why and What

Summary
Open Source: Why bother?

- Tools can get widespread
- Closer interaction Developer ↔ User
- Less effort for customization
- High number of potential developers

Disclaimer
Complex tools need high initial effort
What is out there?

Design  Qucs, fritzing

HLS    JHDL + JHDL-CAD [7]

Simulation  GHDL, FreeHDL

Below RTL  ABC [6], open HW-Platforms [14]
Introduction

The Tools

Chaining Tools together

Open Source

Summary

Improvements

Further reading
Suggested Improvements (by others)

- Formulation (Specification)
- Resource and Performance prediction
- Modeling techniques
- Bridge between formulation and design phase
- Reduced translation and routing times
- also see [9]
Suggested Improvements (by me)

- user interaction, usability
- content awareness
- chaining capabilities
- openness...
Further reading

Some Papers I

Terry Bahill and Jesse Daniels.
Using objected-oriented and uml tools for hardware design: A case study.

Automatic conversion of floating point matlab programs into fixed point fpga based hardware design.

Per Bjesse, Koen Claessen, Mary Sheeran, and Satnam Singh.
Lava: Hardware design in haskell.

Florent de Dinechin and Bogdan Pasca.
Designing custom arithmetic data paths with flopoco.

Douglas Densmore, Roberto Passerone, and Alberto Sangiovanni-Vincentelli.
A platform-based taxonomy for esl design.

Berkeley Logic Synthesis Verification Group.
ABC: A System for Sequential Synthesis.
Some Papers II

A CAD suite for high-performance FPGA design.

Paul Layzell.
A new research tool for intrinsic hardware evolution.

Strategic Challenges for Application Development Productivity in Reconfigurable Computing.

Budi Rahardjo.
Spin as a hardware design tool.

R. Reetz, K. Schneider, and T. Kropf.
Formal specification in VHDL for hardware verification.
Some Papers III

Mary Sheeran.

Hardware design and functional programming: a perfect match.

S. Vernalde, P. Schaumont, and I Bolsens.

An object oriented programming approach for hardware design.

Aaron Weiss.

Open source hardware: Freedom you can hold?
*netWorker, 12(3):26–33, September 2008.*