Some thoughts on HDL licensing

Javier Serrano, Andrew Katz

LLW, Barcelona, 9 April 2014
Outline

1. HDL basics
2. Current HDL licensing practice
3. Issues and open questions
Origins of HDL

Initially created to specify deliverables for the DoD

- Less equivocal than human language.
- Can be treated as a contract.

And then software and electronics evolved:

- Behaviour could be simulated in a computer.
- HDL code could be synthesised into ASICs and FPGA bitstreams.
What FPGAs are made of
Ultra-simplified view
FPGA design flow

Design entry
Using any text editor, as for software development.

Synthesis
Turns your HDL into gates and flip-flops.

Place and Route
Find appropriate places in the FPGA for each piece of logic, with suitable connections among them, using programmable interconnect.

Bitstream generation
Generates the bitstream that will go into the configuration memory of the FPGA.
Circuit example

Fig. 3-37 Circuit to Demonstrate HDL

Courtesy of Richard Nelson
//Circuit specified with Boolean equations
module circuit_bln (x, y, A, B, C);
    input A, B, C;
    output x, y;
    assign x = (A & B) | ¬C;
    assign y = ¬C;
endmodule
Three ways to use HDL sources

Hardware can be described in a number of languages and then be used to:

- feed into a simulator (use case #1).
- generate a bitstream for an FPGA (use case #2).
- design an ASIC (use case #3).
Using software licenses for HDL

These are the most commonly used licenses

- They play out well for case #1.
- Fuzzy for #2 and (especially) #3, particularly copyleft ones.
Using software licenses for HDL

Weak copyleft usually preferred

- The mapping of strong copyleft wording to HDL is unclear.
- Most designs use proprietary cores.
- opencores.org: 502 cores under LGPL, 167 GPL, 46 BSD, 39 other.
Open questions

Questions engineers ask themselves:

- What does *linking* mean for HDL?
- And *dynamic linking*?
- How does one define borders in HDL?
- Can we use licenses which explicitly state they apply to software only?
Current practice is sub-optimal

HDL deserves better! Two possible ways:

- Add explanatory material to help interpret current software licenses.
- Create/evolve a new license for HDL.