High Level Synthesis

An EDA & hardware IP industry perspective

Nigel Topham

The University of Edinburgh

November 2017
Overview

- What does High Level Synthesis (HLS) mean in the EDA industry?
- Languages & Systems for High Level Synthesis
- Example of a commercial HLS tool + use case
- Why is HLS timely and relevant now?
- Summary of research challenges
- Example of learning-based u-arch synthesis
EDA view of High Level Synthesis

- Definitely not the hardware equivalent of Program Synthesis
- HW abstractions have always been lower level than SW
- HLS raises the abstraction
- Synthesis means “compilation”

(a) Traditional HW design
(b) High(er) Level Synthesis
## Languages & Systems for High Level Synthesis

<table>
<thead>
<tr>
<th>Company</th>
<th>HLS Tool</th>
<th>Languages</th>
<th>Application areas</th>
</tr>
</thead>
<tbody>
<tr>
<td>Synopsys</td>
<td>Synphony</td>
<td>M</td>
<td>DSP algorithms</td>
</tr>
<tr>
<td></td>
<td>ASIP Designer</td>
<td>nML</td>
<td>ASIP specification</td>
</tr>
<tr>
<td>Cadence</td>
<td>Stratus HLS</td>
<td>C, C++, SystemC</td>
<td>Home, automotive, mobile</td>
</tr>
<tr>
<td>Xilinx</td>
<td>Vivado HLS</td>
<td>C, C++, SystemC</td>
<td>High-productivity FPGA</td>
</tr>
<tr>
<td>Mentor</td>
<td>Catapult</td>
<td>C, C++, SystemC</td>
<td>DSP, Vision, etc.</td>
</tr>
<tr>
<td>Mathworks</td>
<td>HDL Coder</td>
<td>Matlab</td>
<td>DSP algorithms</td>
</tr>
</tbody>
</table>
Example: Synopsys “ASIP Designer”
ASIP example – Synopsys EV6x CNN Engine

- Requires high MAC throughput
- Non-standard arith. precision
- Specialized ISA, not exposed to user (only needs a graph compiler)
- Embedded black-box behaviour lends itself to ASIP design
- Rapid development can be assisted by HLS...
- ASIP Designer creates Verilog and all software tools
- But is this “synthesis” or just coding at a higher level of abstraction?
Why is Hardware HLS timely and relevant now?

1. Moore’s Law continues for a few more generations…
2. But Denard Scaling ended some time ago
3. Increasing complexity + compressed development schedules
   - Energy efficiency trade-off versus transistor count
   - 10x designer productivity required (c.f. Verilog coding)
   - Reduced verification cost

- DSP designers have led the way (Matlab and current HLS tools)
- Domain experts in other areas can now reap the benefits
  - Vision systems
  - Neural computation
  - Deep learning
  - Robotics
  - Medical
Synthesis vs Translation – my definitions

Translation
- Combination of lowering and optimization, in the classical sense
- E.g. from C++ to x86 assembler, or C++ to Verilog

Synthesis
- Creating a solution based on a formal notion of what constitutes a correct and acceptable solution
- Designer specifies an acceptable solution without implying the precise construction
- Synthesis creates the solution
- It may not be obvious how (or why) the solution actually works!
- E.g. from an application to a synthetic instruction set
- Or, from a formal protocol specification to an implementation (need not be lowered all the way to Verilog, could target an intermediate form in C/C++/SystemC which is then translated by a hardware HLS tool)
Example – ISA and Microarchitecture Synthesis

(a). Algorithms to hardware

(b). Applications to architectures, and then to microarchitecture

Summary of research questions

- What would genuine high-level synthesis look like?
  - Formal specification of (part of) the system?
  - What synthesis techniques can be formulated?

- What pressing problem(s) would it address?

- Who would benefit from it?

- Can we identify a small number of use cases to motivate research?