1 Introduction

We find ourselves caught in the snare of ever-increasing computational complexity. The demands of modern computation can bog down a general-purpose CPU to the point of impracticality[1]. Hardware accelerators are mechanisms used to mitigate such situations by offloading specific computational tasks to hardware. Developing optimized hardware solutions to replace software comes with the trade-off of increased development time due to the introduction of hardware's inherent lower abstraction levels. This increased overhead can often be prohibitive, thus it has been a dream of hardware designers since the 1970s to create tools that synthesize optimized hardware using traditional software development techniques[2]. This dream tool would complete a process known as High Level Synthesis (HLS).

In order to consider the dream realized, HLS must create optimized hardware using an abstract, algorithmic level description of a circuit as the primary input specification[3]. Unfortunately, due to reasons such as standardization[4], reliability[5] and portability[6], the two most ubiquitous languages for constructing hardware descriptions (VHDL and Verilog) do not operate natively at the algorithmic level [7]. This presents a problem: the most oft-used, best supported, languages for specifying synthesizable hardware do not behave like that of traditional compiled software. Instead, these two languages operate at a level of abstraction known as the Register-Transfer Level (RTL), which is one level of abstraction below that of the algorithmic level [8], and HLS seeks to bridge that gap.

The development of HLS tools is not the focus of Homsirikamol and Gaj's study, rather they seek to validate the state of the HLS dream by comparing the performance of hardware generated using an RTL circuit description to that of an algorithmic, HLS, specification. This report will, first, explain the methodology used to conduct the case study and present its findings. Next, an evaluation of their conclusions will be presented alongside alternative methods of comparison, ultimately presenting a more complete picture of whether HLS is, or ever will be, a dream-come-true.

2 Summary of Selected Paper

This section will briefly summarize the claims presented by Homsirikamol and Gaj's study on the state of HLS.

2.1 Background

HLS is the process of converting an algorithmic level description to RTL. Therefore, in this context the term synthesis can be defined as a method for traversing abstraction levels in hardware design. This is carried out by implementing a description found at higher levels of abstraction using only methods found in the lower [6]. Specifically, HLS translates descriptions found at the algorithmic level, often written in a relatively high-level language like C, to a functionally equivalent representation in an RTL language like VHDL or Verilog.

2.2 Problem

Measuring the quality of HLS tools is a problem that requires a high degree of specificity in order to be useful. Quality has often been measured by simply investigating whether a complex circuit synthesized through HLS can achieve a desired functionality[9][10][11][12][13]. Other studies [14][15] seek to measure the quality of HLS by comparing the performance of an HLS circuit to its software implementation. The authors contend that neither of the above metrics for quality (functionality or software comparison) paint a sufficient picture of the state of HLS, rather they claim that the quality of HLS is better assessed when compared to a circuit synthesized through traditional RTL synthesis. Additionally, since computational challenges vary greatly based on functional domain, Homsirikamol and Gaj claim that comparisons between HLS tools and RTL synthesis tools must be domain-specific, i.e., it is not useful to make blanket statements of a tool's ability without, first, specifying the types of computational tasks each method is expected to synthesize. The selected paper chooses to make an HLS-to-RTL comparison in the cryptographic domain, specifically

comparing implementations of the Advanced Encryption Standard.

2.3 Proposal

2.3.1 Methodology

Ultimately, the authors wish to demonstrate the quality of HLS by comparing the performance of circuits synthesized using a software description to circuits synthesized using an RTL description. In order to make a fair comparison, the authors need to make a case that the synthesis process was responsible for the changes in performance. This is accomplished by conducting their evaluations across multiple chip manufacturers and chip families, thus removing the target technology as a possible source of performance variability. The RTL language, FPGA tools and synthesis options were all held constant, leaving HLS as the claimed independent variable in the case study.

2.4 Evaluation

2.5 Conclusion

3 Analysis of Selected Paper

3.1 Paper Strengths

Synthesis tools vary greatly across manufacturers and device families, therefore comparing implementations of synthesized designs becomes a delicate exercise. The authors present a case that their methodology for comparing synthesis performance is a fair one and this section explores that claim.

3.2 Paper Weaknesses

3.3 Critique

3.4 Future Work

4 Conclusion

References

[1] S. Skalicky, C. Wood, M. Lukowiak, and M. Ryan, "High level synthesis: Where are we? a case study on matrix multiplication," in *Reconfigurable Computing and FPGAs (ReConFig), 2013 International Conference on*, Dec 2013, pp. 1–7.

[2] G. Martin and G. Smith, "High-Level Synthesis: Past, Present, and Future," *Design Test of Computers, IEEE*, vol. 26, no. 4, pp. 18–25, July 2009.

[3] M. C. McFarland, A. Parker, and R. Camposano, "The high-level synthesis of digital systems," *Proceedings of the IEEE*, vol. 78, no. 2, pp. 301–318, Feb 1990.

[4] "IEEE standard for verilog register transfer level synthesis," *IEEE Std 1364.1-2002*, 2002.

[5] S. Tosun, N. Mansouri, E. Arvas, M. Kandemir, and Y. Xie, "Reliability-centric high-level synthesis," in *Design, Automation and Test in Europe, 2005. Proceedings*, March 2005, pp. 1258–1263 Vol. 2.

[6] P. P. Chu, *RTL Hardware Design Using VHDL: Coding For Efficiency, Portability, and Scalability.* John Wiley & Sons, Inc., 2006, pp. 1–22. [Online]. Available: http://dx.doi.org/10.1002/0471786411.fmatter

[7] D. M. Harris and S. L. Harris, *Digital design and computer architecture*, second edition ed. Amsterdam, Boston: Morgan Kaufmann Publishers, Cop., 2013.

[8] F. Vahid, *Digital Design with RTL Design, Verilog and VHDL.* John Wiley & Sons, Inc., 2010, p. 247.

[9] F. Burns, J. Murphy, D. Shang, A. Koelmans, and A. Yakorlev, "Dynamic global security-aware synthesis using SystemC," *Computers Digital Techniques, IET*, vol. 1, no. 4, pp. 405–413, July 2007.

[10] M. Ernst, S. Klupsch, O. Hauck, and S. Huss, "Rapid prototyping for hardware accelerated elliptic curve public-key cryptosystems," in *Rapid System Prototyping, 12th International Workshop on, 2001.*, 2001, pp. 24–29.

[11] S. Morioka, T. Isshiki, S. Obana, Y. Nakamura, and K. Sako, "Flexible architecture optimization and ASIC implementation of group signature algorithm using a customized HLS methodology," in *Hardware-Oriented Security and Trust (HOST), 2011 IEEE International Symposium on*, June 2011, pp. 57–62.

[12] S. Ahuja, S. Gurumani, C. Spackman, and S. Shukla, "Hardware Coprocessor Synthesis from an ANSI C Specification," *Design Test of Computers, IEEE*, vol. 26, no. 4, pp. 58–67, July 2009.

[13] J. Davis, D. Buell, S. Devarkal, and G. Quan, "High-level synthesis for large bit-width multipliers on FPGAs: a case study," in *Hardware/Software Codesign and System Synthesis, 2005. CODES+ISSS '05. Third IEEE/ACM/IFIP International Conference on*, Sept 2005, pp. 213–218.

[14] K. Rupnow, Y. Liang, Y. Li, and D. Chen, "A study of high-level synthesis: Promises and challenges," in *ASIC (ASICON), 2011 IEEE 9th International Conference on*, Oct 2011, pp. 1102–1105.

[15] Y. Liang, K. Rupnow, Y. Li, D. Min, M. N. Do, and D. Chen, "High-level Synthesis: Productivity, Performance, and Software Constraints," *JECE*, vol. 2012, pp. 1:1–1:1, Jan. 2012. [Online]. Available: http://dx.doi.org/10.1155/2012/649057