PhD. Qualification Report

Can High-Level Synthesis Compete Against a Hand-Written Code in the Cryptographic Domain? A Case Study [1]

Authors:
Ekawat Homsirikamol and Kris Gaj

Ryan Silva
Boston University
Department of Electrical and Computer Engineering
rjsilva@bu.edu

1 Introduction

In the beginning, there was a dream. That dream was to synthesize reliable hardware using traditional software development techniques [2]. In order to accomplish this, logic synthesis would be required to operate reliably using an algorithmic-level description as the primary input specification [3]. In the case of High-Level Synthesis (HLS), the term "synthesis" can be defined as the process of traversing abstraction levels in hardware design and is carried out by implementing a description found at higher levels of abstraction using only methods found in the lower [4]. The two most ubiquitous languages for creating such a hardware description are VHDL and Verilog [5]. The problem is that these languages do not operate natively on the algorithmic level, i.e., the synthesis of VHDL and Verilog does not resemble that of traditional compiled software. These two languages operate at a level of abstraction known as the Register-Transfer Level (RTL) [6]. The implementation of a hardware design using an abstract description of High-level synthesis (HLS) is a tool for developing [4] [7] [2] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24]

1.1 Introduction

It is difficult to imagine a computer engineer having to create a VLSI layout and fabricate an ASIC every time they wanted to run a C program. While there will always be a place for custom circuit design in the world of digital electronics, the basic tenants of digital design require that it remain very much the exception rather than the rule. Unfortunately for the experimentalist, this is not the case in the field of microfluidics. The creation of custom, "one-off", designs for individual microfluidic experiments, no matter how user-friendly the corresponding CAD software is, could be is what is keeping mLSI from more closely resembling that of its silicon counterpart in terms of productivity. Since Thorsen *et al.* successfully integrated thousands of micromechanical valves in 2002 [25], academic researchers have attempted to manage exponentially greater complexity in microfluidic design via the introduction of new design methodologies that attempt to introduce "top-down" specificity and move away from a "bottom-up" design philosophy [26] [27] [28] yet microfluidic experimentalists still find themselves in front of an oven baking a photoresist until it ceases to be sticky. If the goal is truly experimental automation the costly, in units time and overhead cost, fabrication step must be removed from the work flow for the majority case as it is for electronic computation. Often, academic papers delving into the relm of mLSI begin by presenting an analogy between microfluidic LSI and LSI found in digital electronics. This analogy seems strange as computer engineers are not required to know how to wash chemical from

printed circuit boards (PCBs), use CAD tools to layout application specific integrated circuits (ASICs), nor enlist the aid of experts in fabrication who can in order to be productive. The *in silico* analogy does not hold when applied to the common-use case: that of the individual scientist or engineer. Why is that?

should be well versed in the art of fabricating devices would they ever hope to conduct a microfluidic experiment has served as a catalyst for academic research since . This paper will provide a historical overview of the emergence of digital design and highlight instances where microfluidics has deviated, resulting in the current design topology seen today. It will also analyse various attempts to rectify microfluidic design challenges using various computer-aided tools and the state of microfluidic design and computation alongside a historical analysis of the emergence of digital computation *in silico*. Important deviations will be highlighted and a new approach that better fits the *in silico* analogy is presented. digital electronics and highlight important deviations from . A natural first step in the production of viable microfluidic design rules that draw from those found *in silico* is the definition of specific layers of abstraction. These layers allow the engineer to properly design, build and test complex microfluidic designs at the least complex, or "highest", level of detail possible [29]. Analogies are often made between computation *in silico* versus via a microfluidic platform. At first glance, the analogy is sound in that computation *in silico* provides automation in computation by managing complexity through the introduction of abstraction layers. Abstraction works by placing the user at only the highest level relevant to the computation being performed and masking all underlying details. It can, therefore, be contended that functionally complete automation of microfluidic experiments implies placing the scientist at only the highest levels of abstraction and masking all underlying details. Currently, even the best efforts in microfluidic tools only remove intermediate levels of abstraction, while exposing the scientist to the highest **and lowest** levels. Imagine if the only output of a C program were a circuit schematic that must first be built in order to obtain the result of the program. presented its output in the form of a circuit schematic that must top-down [26] [27]
abstraction layers [29] Furthermore, the successful execution of this research will present emergent capabilities as applied to the field of microfluidics.

1.2 Managing Complexity

Managing complexity is a necessary craft in that it allows the engineer to design complicated systems without becoming overwhelmed by details. The art of managing complexity in digital electronics design is a mature process relative to that found in microfluidics. This is evident by the existence of larger scales of integration *in silico*, such as VLSI, and by efforts to create tools that mirror the design-to-execution workflow found in electronic computing. Examples of such tools are Micado, for automation of control layer routing [30], and BioStream [29], which could serve as the cornerstone of true experimental automation and will be described in the subsequent section. It could serve as a useful exercise to review the methodology computer engineers use to manage complexity and then to contrast these principles with the current state of microfluidic design. There are few better place to find fundamental digital design practices than in an introductory textbook, which lists five key components involved with properly managing complexity: abstraction, discipline, hierarchy, modularity and regularity [5].

1.2.1 Abstraction

Abstraction can be defined as a method for hiding details when they are not important, often through the creation of different working levels of minutia. [5]. Figure 1 is an example of how electronic computing can be broken down into separate working levels of complexity.

The use of the term "working levels" of complexity implies that the person operating within that layer need not concern themselves with the details of a lower layer, as such requirements would ultimately defeat the purpose of abstraction. Lower levels of detail are said to be "abstracted" away when their use is considered automatic. However, designers of systems residing in one particular abstraction layer should have an understanding of how their design decisions affect the layers immediately above and below the working layer, such as a C programmer understanding the nature of an address space [5]. Theis *et al* advocate for the creation of abstraction layers in microfluidics similar to that found in electronic

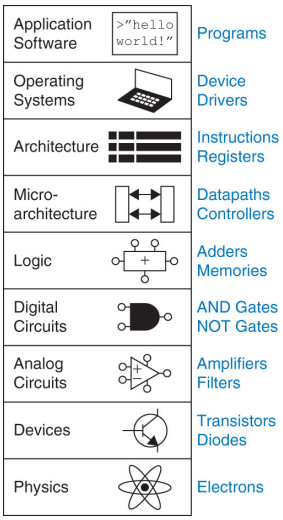

Fig. 1. Abstraction Layers for Electronic Computing (I'll have to make my own. . .)

computing [29]. These layers achieve success by focusing on three basic fluidic operations: mixing, transport and storage. Their BioStream protocol is a brilliant step in decoupling microfluidic architecture from biological computation by providing a common language for describing an experimental protocol. BioStream served initially as a standard language for reporting biological protocols but expanded to an end-to-end system that effectively describes biological protocols within the BioStream Fluidic ISA and executes them at the hardware level independent of microfluidic chip microarchitecture [29]. BioStream, however, does not fully address a functional purpose of abstraction, which is to provide automation, but it does accomplish a very important first step the authors describe as a "division of labor" between the biology and microfluidic experts.

One incredible benefit of BioStream is its ability to operate independently of a standard microfluidic architecture. Unfortunately, this independence could be holding back the biological experimentalist from true top-to-bottom automation. Why would the experimentalist purposely restrict their design decisions by adopting a standard architecture? The answer to this question is found within the second

1.3 The Productivity Gap

There is, and probably always will be, a place in digital electronics for PCB design and ASIC fabrication. However, before an engineer decides to begin the process of building a custom PCB or layout a new ASIC they should first consider how their deisgn decision addresses the productivity gap. In order to proceed, a working definition of the term "productivity" must be presented. Process and requirements engineers [31] have defined productivity strictly in terms of hours saved [32], as a function of on-time delivery [33] or as some measure of quality [34]. This paper will define the productivity of a particular method as the number of hours saved through the implementation of a particular process.

Device fabrication is not a task oft performed by a computer scientist. Rather, a computer scientist spends many hours debugging a program such that it runs reliably and correctly within the confines of a particular ISA. This exemplifies the nature of design discipline. It is well-within the realm of possibility for a computer engineer to give up debugging a program and reach for a CAD tool, which which to build a custom chip designed for their particular purposes. That scenario does not make sense because it creates an extremely large gap in productivity. The amount of lead-time required to design and build a PCB could significantly outweigh the benefits of having a single custom-chip to use only in very specific circumstances and only within that one engineer's lab. Why then is this practice deemed acceptable in microfluidics?

Even attempts to create some framework for flow-based microfluidic design, such as a common microfluidic ISA [30] or predefined software modules [35] are still, fundamentally, design methods for

chip fabrication. Microfluidic chip fabrication is a highly unproductive in that it requires many hours to design and build a device incapable of performing diverse and repeatable experiments. Fortunately for the computer engineer there exists other prototyping options besides PCBs and ASICs, such as the use of a field programmable gate array (FPGA) or microcontroller. The microfluidic experimentalist is left with only one prototyping option that almost always requires some level of device fabrication.

1.4 The Digital Landscape

The computer engineer has three general classes of prototyping methods platforms: ASIC vs FPGA vs Microcontroller. Each has its own general use cases and tradeoffs. Microfluidics has seen many efforts to create device architectures that resemble each of the three general *in silico* platforms (FPGA [36], Microcontroller

1.5 On Deck Refs

MHDL: essentially a Plugin for AutoCAD [37] Micado: It is, in fact, a plugin for AutoCAD [30]

2 Connecting Ideas in Biosensor Design: From the Cell to the Electronic Domain

2.1 Abstract (Arsenic Sensor)

Recent academic research endeavors in the detection heavy metals in drinking water delved into the realm of synthetic biology. This came as a response to the need for a cost-effective mechanism to prevent arsenicosis and other water-born illnesses in developing countries at the point of collection. Researchers at the University of Edinburgh successfully built cell-based arsenic sensors and deployed them to Nepal for field testing. These sensors work by modulating the pH of a culture, and the pH is the interpreted value. Emergent capabilities of this effort include self-contained, self-reporting and self-actuating microfluidic tests, which allow for more rapid processing of experiments. A minor criticism of the technology stems from an inability to properly discern the meaning of the sensor's colorimetric output. This research will combine previous efforts in biosensors and biotransducers in order to translate the output of the Edinburgh arsenic biosensor (pH) into a more readable format (electronic display).

2.2 Abstract (General)

The exchange of information is rooted in the electronic domain. This domain is governed using well-established techniques cultivated from years of engineering development. This research explores the design space that is presented once the microbial world is given access to the electronic domain. We demonstrate how synthetic biology can be an effective tool for communicating with life at the microbial level.

2.3 Introduction

ACCORDING to a 2007 study, arsenic poisoning may affect at least 137 million people in over 70 countries, a disproportionate amount of which happen to be located in the developing world [38]. As a result of this widespread endemic, some research efforts have focused on developing inexpensive and deployable tests for detecting arsenic in drinking water. Many of these tests rely on a visual inspection of color-based reporting mechanisms in order to interpret the test's results. One particular effort to achieve the goal of an inexpensive, deployable arsenic sensor was created by the Arsenic Biosensor Collaboration (ABC) [39]. Their extensive fieldwork concluded that local users preferred "a numeric scale instead of a colour gradient" in their device's readout [40]. This paper will solve the problem by integrating an electrochemical transducer to the output of ABC's arsenic detection device to render a more quantifiable reading, possibly on a mobile phone. Furthermore, the successful execution of this research will present emergent capabilities as applied to the field of microfluidics.

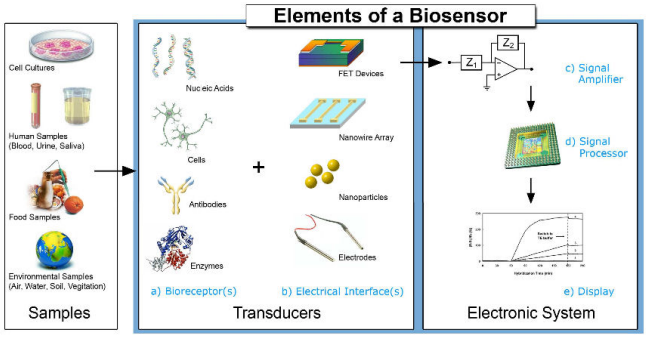

Fig. 2. General Bioelectronics Flow [43]

2.4 Background

The use of biological elements as sensing platforms is not a new idea [41] and tends to follow a design flow similar to that shown in Figure 2. This research will use the ABC's arsenic detector as the bioreceptor and a silicon nanowire field effect transistor, developed at IBM's T.J. Watson Research Center as the electrical interface [42]. The arsenic detecting bioreceptor is able to sense the presence of arsenic in a solution and affect the pH of another (?) solution. The pH is currently being read by a simple (I assume) litmus test.

2.5 Experiment

The initial goal of the experiment is to convert an input signal, in this case the presence of arsenic in solution, to an electrical signal. This goal involves two steps, the first of which is to use a biological sensor to convert the input signal into a format that can be read by an electrical transducer. The chosen format involved the modulation of a solution's pH based on the level of arsenic in solution. The biological sensor involved a **explain genetic circuit**. After correlating the input signal with pH, the next step is to convert pH into an electrical signal using a pH probe The basic elements involved in the experiment involve an input signal (arsenic in solution) invoking an electrical response. This electrical response could be processed by a microcontroller, after which there are many directions to go.

2.6 Design Elements

Explain the different elements that will be involved in this complete circuit.

2.6.1 Biosensor

Explain who made the Biosensor and how it is currently being used. It will be important to include the arsenic to pH curve.

2.6.2 Silicon Nanowire

Explain the background of the silicon nanowire. It will be important to include the IV curve as well as the pH to current (assuming you are sweeping the Vsol) and the pH to Vsol (assuming a constant drain current)

REFERENCES

[1] E. Homsirikamol and K. Gaj, "Can high-level synthesis compete against a hand-written code in the cryptographic domain? A case study," in *ReConFigurable Computing and FPGAs (ReConFig), 2014 International Conference on*, Dec 2014, pp. 1–8.

[2] 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.

[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] 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

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

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

[7] 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.

[8] 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

[9] F. Gruian and M. Westmijze, "VHDL vs. Bluespec System Verilog: A Case Study on a Java Embedded Architecture," in *Proceedings of the 2008 ACM Symposium on Applied Computing*, ser. SAC '08. New York, NY, USA: ACM, 2008, pp. 1492–1497. [Online]. Available: http://doi.acm.org/10.1145/1363686.1364037

[10] J. Cong, B. Liu, S. Neuendorffer, J. Noguera, K. Vissers, and Z. Zhang, "High-Level Synthesis for FPGAs: From Prototyping to Deployment," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 30, no. 4, pp. 473–491, April 2011.

[11] Berkeley Design Technology Inc., "An Independent Evaluation of: High-Level Synthesis Tools for Xilinx FPGAs," *Executive Summary*, 2010. [Online]. Available: http://www.bdti.com/MyBDTI/pubs/Xilinx_hlstcp.pdf

[12] 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.

[13] 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.

[14] 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.

[15] 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.

[16] 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.

[17] National Institute of Standards and Technology, "FIPS PUB 197: Advanced Encryption Standard (AES)," *Federal Information Processing Standards Publications*, vol. 197, Nov 2001.

[18] ——, "SP PUB 800-38A: Recommendation for Block Cipher Modes of Operation," *Special Publications*, no. 38A, Dec 2001.

[19] R. Housley, "Using Advanced Encryption Standard (AES) Counter Mode with IPsec Encapsulating Security Payload (ESP)," *Internet Engineering Task Force*, no. RFC3636 (Proposed Standard), Jan 2004.

[20] S. Shen, Y. Mao, and N. Murthy, "Using Advanced Encryption Standard Counter Mode (AES-CTR) with the Internet Key Exchange version 02 (IKEv2) Protocol," *Internet Engineering Task Force*, no. RFC5930 (Informational), Jul 2010.

[21] P. Barreto and V. Rijmen, "Reference code in ANSI C v2.2," http://www.ktana.eu/html/theRijndaelPage.htm, Mar 2002.

[22] Y. Hara, H. Tomiyama, S. Honda, H. Takada, and K. Ishii, "CHStone: A benchmark program suite for practical C-based high-level synthesis," in *Circuits and Systems, 2008. ISCAS 2008. IEEE International Symposium on*, May 2008, pp. 1192–1195.

[23] K. Gaj, J. Kaps, V. Amirineni, M. Rogawski, E. Homsirikamol, and B. Brewster, "ATHENa - Automated Tool for Hardware EvaluatioN: Toward Fair and Comprehensive Benchmarking of Cryptographic Hardware Using FPGAs," in *Field Programmable Logic and Applications (FPL), 2010 International Conference on*, Aug 2010, pp. 414–421.

[24] K. Gaj, E. Homsirikamol, and M. Rogawski, "Fair and Comprehensive Methodology for Comparing Hardware Performance of Fourteen Round Two SHA-3 Candidates Using FPGAs," in *Cryptographic Hardware and Embedded Systems, CHES 2010*, ser. Lecture Notes in Computer Science, S. Mangard and F.-X. Standaert, Eds. Springer Berlin Heidelberg, 2010, vol. 6225, pp. 264–278. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-15031-9_18

[25] T. Thorsen, S. J. Maerkl, and S. R. Quake, "Microfluidic large-scale integration," *Science*, vol. 298, no. 5593, pp. 580–584, 2002.

[26] W. H. Minhass, P. Pop, J. Madsen, and T.-Y. Ho, "Control synthesis for the flow-based microfluidic large-scale integration biochips," in *18th Asia and South Pacific Design Automation Conference (ASP-DAC 2013)*, 2013, pp. 205–212.

[27] J. Melin and S. R. Quake, "Microfluidic large-scale integration: the evolution of design rules for biological automation," *Annu. Rev. Biophys. Biomol. Struct.*, vol. 36, pp. 213–231, 2007.

[28] W. H. Minhass, P. Pop, J. Madsen, and F. S. Blaga, "Architectural synthesis of flow-based microfluidic large-scale integration biochips," in *Proceedings of the 2012 international conference on Compilers, architectures and synthesis for embedded systems*. ACM, 2012, pp. 181–190.

[29] W. Thies, J. P. Urbanski, T. Thorsen, and S. Amarasinghe, "Abstraction layers for scalable microfluidic biocomputing," *Natural Computing*, vol. 7, no. 2, pp. 255–275, 2008.

[30] N. Amin, W. Thies, and S. Amarasinghe, "Computer-aided design for microfluidic chips based on multilayer soft lithography," in *IEEE International Conference on Computer Design, 2009. ICCD 2009.* IEEE, 2009, pp. 2–9.

[31] D. Damian and J. Chisan, "An empirical study of the complex relationships between requirements engineering processes and other processes that lead to payoffs in productivity, quality, and risk management," *Software Engineering, IEEE Transactions on*, vol. 32, no. 7, pp. 433–453, July 2006.

[32] S. Lauesen and O. Vinter, "Preventing requirement defects: An experiment in process improvement." *Requir. Eng.*, vol. 6, no. 1, pp. 37–50, 2001. [Online]. Available: http://dblp.uni-trier.de/db/journals/re/re6.html#LauesenV01

[33] H. Wohlwend and S. Rosenbaum, "Software improvements in an international company," in *Proceedings of the 15th International Conference on Software Engineering*, ser. ICSE '93. Los Alamitos, CA, USA: IEEE Computer Society Press, 1993, pp. 212–220. [Online]. Available: http://dl.acm.org/citation.cfm?id=257572.257621

[34] J. D. Herbsleb and D. R. Goldenson, "A systematic survey of cmm experience and results," in *Proceedings of the 18th International Conference on Software Engineering*, ser. ICSE '96. Washington, DC, USA: IEEE Computer Society, 1996, pp. 323–330. [Online]. Available: http://dl.acm.org/citation.cfm?id=227726.227791

[35] A. K. Soe, M. Fielding, and S. Nahavandi, "Lab-on-a-chip turns soft: Computer-aided, software-enabled microfluidics design," in *Advances in Social Networks Analysis and Mining (ASONAM), 2013 IEEE/ACM International Conference on*. IEEE, 2013, pp. 968–971.

[36] L. M. Fidalgo and S. J. Maerkl, "A software-programmable microfluidic device for automated biology," *Lab on a Chip*, vol. 11, no. 9, pp. 1612–1619, 2011.

[37] J. McDaniel, A. Baez, B. Crites, A. Tammewar, and P. Brisk, "Design and verification tools for continuous fluid flow-based microfluidic devices." in *18th Asia and South Pacific Design Automation Conference (ASP-DAC 2013)*, 2013, pp. 219–224.

[38] P. Ravenscroft, "Predicting the global distribution of arsenic pollution in groundwater," presented at the Royal Geographical Society Annual International Conference, 2007.

[39] (2009) Arsenic biosensor collaboration. [Online]. Available: http://arsenicbiosensor.org/index.html

[40] (2009) Arsenic biosensor collaboration. [Online]. Available: http://arsenicbiosensor.org/fieldwork.html

[41] de Mora K., "A ph-based biosensor for detection of arsenic in drinking water," *Anal. Bioanal. Chem.*, vol. 400, p. 1031, 2011.

[42] S. Zafar, C. DEmic, A. Afzali, B. Fletcher, Y. Zhu, and T. Ning, "Optimization of ph sensing using silicon nanowire field effect transistors with hfo 2 as the sensing surface," *Nanotechnology*, vol. 22, no. 40, p. 405501, 2011.

[43] D. Grieshaber, "Biosensor system," 2008. [Online]. Available: http://en.wikipedia.org/wiki/Biosensor#mediaviewer/File:Biosensor_System.jpg

Ryan Silva Ryan Silva is a Captain on active-duty in the United States Air Force. He current assignment is at Boston University where is pursuing his PhD in Computer Engineering. After graduating with his degree, Ryan will be assigned to an Air Force unit before returning to teach at the United States Air Force Academy, where he graduated with a degree in Electrical Engineering in 2005.