ATHENa – Automated Tool for Hardware EvaluatioN: Toward Fair and Comprehensive Benchmarking of Cryptographic Algorithms using FPGAs

Kris Gaj, Jens-Peter Kaps, Venkata Amirineni, Marcin Rogawski, Ekawat Homsirikamol
ECE Department, George Mason University
4400 University Drive, Fairfax, VA 22030, USA
Email: {kgaj, jkaps, vamirin1, mrogawsk, ehomsiri}@gmu.edu

***Abstract*—In this talk, we will introduce an open-source environment, called ATHENa for fair, comprehensive, automated, and collaborative hardware benchmarking of cryptographic algorithms. We believe that this environment is very suitable for use in evaluation of hardware performance of SHA-3 candidates from the point of view of speed, resource utilization, cost, power consumption, etc. At this point, the environment supports the evaluation using several families of Field Programmable Gate Arrays (FPGAs) from two major vendors, Xilinx and Altera. The environment is accompanied by a comprehensive website and database of results. We encourage all FPGA benchmarking teams to use our environment in their evaluations of SHA-3 candidates, and share their results, constraint files, testbenches, test vectors, and possibly even source codes using our web site. All results will be displayed in the form of the web based interactive tables permitting searching, filtering, ranking, and organizing multiple sets of data from various groups. All results pertaining to the comparison of the SHA-3 candidates, collected by the time of the conference, will be summarized and highlighted in our talk. This way, the presented ranking of Round 2 SHA-3 candidates will be based on the *best* results available for each candidate, rather than results obtained by a single group. In the future, in collaboration with other groups, we are planning to extend our database of results to cover the results for ASIC and software implementations, and allow the comparison of selected universal performance measures, such as maximum throughput, across multiple technologies.**

***Index Terms*—open-source; performance evaluation; benchmark tool;**

I. Introduction

Starting from the Advanced Encryption Standard (AES) contest organized by NIST in 1997-2000 [1], open contests have become a method of choice for selecting cryptographic standards in the U.S. and over the world. The AES contest in the U.S. was followed by the NESSIE competition in Europe [2], CRYPTREC in Japan, and eSTREAM in Europe [3]. At this point, the focus of attention of the entire cryptographic community is on the SHA-3 contest for a new hash function standard, organized by NIST [4].

Four major criteria are typically taken into account in the evaluation of candidates for a cryptographic standard: security, performance in software, performance in hardware, and flexibility. While security is commonly recognized as the most important evaluation criterion, it is also a measure that is most difficult to evaluate and quantify, especially during the relatively short period of time reserved for the majority of contests. The typical outcome is that, after eliminating a fraction of candidates based on security flaws, a significant number of remaining candidates do not demonstrate any easy to identify security weaknesses, and as a result are judged to have adequate security.

Performance in software and hardware are next in line to clearly differentiate among the candidates for a cryptographic standard. Both criteria are very convenient – they are relatively easy to evaluate and quantify, objective, and of practical importance for the commercial viability (in terms of cost, speed, and energy consumption) of the end products incorporating the standard.

Interestingly, the differences among the cryptographic algorithms in terms of the hardware performance seem to be particularly large, and often serve as a tiebreaker when other criteria fail to identify a clear winner [1], [5].

The difficulties associated with a fair comparison of the hardware performance of cryptographic algorithms can be divided into

- Evaluation Pitfalls: Mistakes that can be quite easily avoided if the person performing comparison is aware of potential dangers, and exercises appropriate caution and fairness; and
- General Objective Difficulties: Objective inherent difficulties that must be comprehensively addressed before a fair comparison is possible.

Examples of evaluation pitfalls include: Taking credit for improvements in technology, choosing a convenient (but not necessarily fair) performance measure, comparing designs with different functionality, comparing designs optimized using a different optimization target (speed, area, cost, power, balanced, etc.), comparing clock frequency after synthesis vs. clock frequency after placing and routing, etc. These mistakes can be most easily described using the phrase "comparing apples and oranges."

Objective difficulties are more challenging to overcome, and include lack of standard interfaces, influence of tools and their options, differences between a stand-alone performance vs. performance as a part of a bigger system, the dependence of the obtained results on the time spent for optimization, etc. [6].

Our project aims to address all aforementioned difficulties by developing an open-source benchmarking environment called ATHENa – Automated Tool for Hardware EvaluatioN [7]. The goal of our project is to spread knowledge and awareness about good performance evaluation practices (and this way eliminate or at least limit the evaluation pitfalls), and to develop the methodology and tools required to overcome objective difficulties.

II. Environment

A. Overview

We have developed a prototype of ATHENa: Automated Tool for Hardware EvaluatioN [7]. At the heart of our tool is a set of scripts written in Perl aimed at an *automated* generation of *optimized* results for *multiple* hardware platforms.

The only software required to run the tool is an interpreter of Perl, which is available for free. The tool also assumes that FPGA design environments are already installed on the system executing the scripts. The users can use either free, educational, or commercial versions of these FPGA design environments.

The general idea of our hardware evaluation environment is shown in Fig. 1.

The ATHENA Server is a focal point of the environment. It hosts the project web site [7], and repository of project scripts and sample configuration files. In the near future, this server is intended to host a large database of results. Each algorithm will be initially represented in the project database by several entries, including algorithm specification (e.g., Federal Information Processing Standard, FIPS) reference implementation in C (or other programming language), and test vectors. In the next step, we will develop and store for each of these algorithms one or more proposed standard hardware interfaces, and the corresponding testbenches.

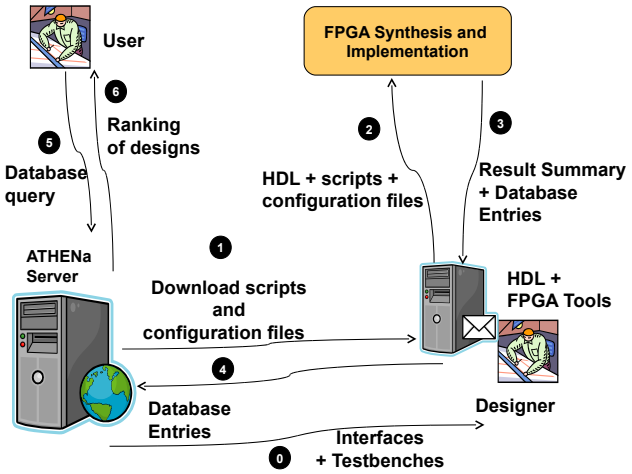

Fig. 1: Data flow within the hardware evaluation environment

A hardware designer can download the aforementioned entries to his local machine, and use them to develop his/her implementation of a given algorithm in the form of Hardware Description Language (HDL) code. The designer can also choose his own interface and develop the corresponding testbench by himself. In this case, the initial download of information from the server is not necessary. After the HDL code is ready, and its functionality verified through simulation, the actual performance evaluation process can begin.

At this point, the user downloads our scripts and sample configuration files to his local machine. He/she modifies configuration files, so they contain proper information about the location of HDL source files, location of tools, target hardware platforms (e.g. Xilinx Virtex 5 and Altera Cyclone III), and other parameters required by the scripts. The user then starts the scripts that run the FPGA implementation in batch mode, and generate the result summary in the form of text files suitable for the designer's review.

In the near future, our environment will be extended with the database of results. The ATHENa scripts will generate the necessary database entries automatically. The designer will be in position to first review the human-friendly result summary, and only afterwards to decide whether to submit the corresponding database entries to the project database.

The important feature of our approach is that all computations are performed on a local machine of the designer, and thus the HDL code never leaves this machine, and is never a subject to interception by any third party, including the project server administrators.

On the other hand, the user must have all FPGA tools and libraries necessary for the evaluation installed on his/her own machine.

B. Features

The main features of our environment include:

1) Running all steps of synthesis, implementation, and timing analysis in batch mode: This is a very important property, as it allows running time-consuming optimizations, without any user supervision, over long periods of time, such as nights, days, or even weeks. ATHENa also tries to speed-up this process by running several instances of the tools concurrently on multi-core computers.

2) Support for devices and tools of two major FPGA vendors: Xilinx and Altera: Xilinx and Altera account for about 90% of the FPGA market. Their FPGA devices differ considerably in terms of the structure of a basic building block: configurable logic block (CLB) for Xilinx, and logic element (LE) for Altera. They also differ in terms of dedicated hardwired units, such as blocks of memory, multipliers, DSP units, etc. As a result, the ranking of algorithms or architectures obtained using devices of one FPGA vendor may not carry to the devices of another vendor.

3) Generation of results for multiple FPGA families of a given vendor, e.g. Xilinx: Spartan 3, Virtex 5; Altera: Cyclone III, Stratix III: Our tool allows specifying as target platforms multiple families of FPGA devices of each of the two major vendors. Every vendor supports over time two or three classes of families, which are optimized respectively for performance, cost and power consumption, and performance to cost ratio. Families belonging to different classes differ significantly, and therefore may produce substantially different results and rankings. Families belonging to the same class also gradually evolve over time. Our tool allows an easy and comprehensive investigation of the dependence of results and rankings on the FPGA families.

4) Automated choice of a device within a given family of FPGAs assuming that the resource utilization does not exceed a certain limit, e.g. 80% of CLB slices or 50% of BRAMs: A maximum clock frequency of a circuit implemented using an FPGA is a function of device resource utilization. When the device utilization reaches 80%–100% in terms of one of the critical resources, such as configurable logic blocks or Block RAMs, the performance degrades. This effect is caused mostly by the difficulties associated with routing in congested circuits. The utilization threshold at which the performance degradation begins is a function of an FPGA family and the implemented circuit. ATHENa supports first determining these thresholds separately for each family of FPGAs and each class of digital circuits. Our environment includes special library files characterizing all devices of a given FPGA family in terms of available resources. The tool is then able to match information from these library files, with the maximum percentage of resources permitted to be used without performance degradation, and select an FPGA device within a given family automatically.

5) Automated optimization of results aimed at one of the three optimization criteria: speed, area, and ratio speed to area: Results generated by the FPGA tools depend highly on the choice of multiple options and the contents of constraint files. Variation of results obtained by changing just a single option may easily exceed 25%.

At this point, ATHENa contains two design space exploration functions: Placement Search and Exhaustive Search.

Placement Search permits the exploration of result dependencies on the starting point of placement. This starting point is determined by the options of the FPGA implementation tools called: Cost Table in Xilinx tools, and Seed in Altera tools. Cost Table can take any integer value between 1 and 100, and Seed any value between 1 and 2^{32}. Both parameters are by default set to 1. Exploring the full range of these parameters may be computationally prohibitive, especially in case of Altera, so a representative subset of the full range needs to be selected.

Exhaustive Search is a superset of Placement Search and extends the set of options to be explored to other options, such as: optimization target (area, speed, or balanced), optimization level, maximum fanout, multiple target clock frequencies, etc. All options are divided into two levels. Level 1 options are changed first, while keeping Level 2 options at their default values. Afterwards, two (or more) sets of Level 1 options are selected and kept constant while Level 2 options are explored.

6) Automated verification of a design through functional simulation, run in batch mode: Our tool has an additional capability of simulating designs in batch mode in order to verify their correct functionality. The verification is based on a testbench utilizing test vectors stored in a file, and providing a binary answer whether the circuit operates correctly or not.

Sample testbenches and hardware interfaces will be provided for the most common cryptographic algorithms (including all NIST standards). One such testbench has already been published at the ATHENa web site. This testbench can be used for the verification of implementations of 14 round-two candidates for the new SHA-3 standard, as well as implementations of current standards SHA-1 and SHA-2.

Designers themselves will be responsible for designing testbenches for any new algorithms, based on generic template files and coding guidelines made available through the project web site. The advantage of simulation in batch mode is that it can be run without any supervision for a long time.

III. Future Work

A. New Features

Several new features of our environment are currently under active development, and are likely to become available during 2010. The release schedule can be found on the Athena webpage [7]. These features include:

1) Additional FPGA vendors: In the near future our environment will be extended to support other FPGA vendors, such as Actel and Lattice Semiconductor.

2) Support for Windows and Linux: The majority of FPGA design environments (including those from Xilinx and Altera) operate under both Windows and Linux. After the initial development of our tool under Windows, its operation will be extended into Linux.

3) Graphical User Interface (GUI): In the current version of the ATHENa environment, the preparation of each evaluation run is done by editing sample configuration files using an arbitrary text editor. In the second phase, a GUI tool will be developed to facilitate the preparation of configuration files, and display of generated results.

4) Extension to ASICs: In collaboration with other groups, in particularly, the Computer Engineering group from Virginia Tech, we will work on extending our environment to benchmarking of ASIC designs. The new environment will support evaluation using multiple libraries of standard cells (from academia and industry), multiple processes (characterized by different feature sizes), multiple gate families and optimization targets, as well as two major integration levels (stand-alone vs. System on Chip).

IV. Conclusions

We have proposed and substantially advanced the development of an open-source tool, called ATHENa, for a fair,

comprehensive, reliable, and practical benchmarking of digital systems using FPGAs from various vendors.

The most important features characterizing our environment are as follows:

- *Comprehensive*: The environment supports evaluation using multiple FPGA devices from several vendors.
- *Automated*: All tools run in batch mode, without the need for any user supervision.
- *Collaborative*: The environment allows and facilitates benchmarking by hundreds of designers from all over the world. As a result the effort on development, debugging, and optimization of codes is shared by a large number of designers, each of which can specialize in a single type of implementation platform and a single set of tools.
- *Practical*: Our environment supports but does *not* require revealing the source codes; as a result it can be safely used by a wide range of designers from academia, industry, and government unable to place their codes in public domain because of intellectual property or export restrictions issues.
- *Distributed*: The majority of the most time consuming computations (including all phases of hardware design and optimization) are performed on local machines of individual designers using tools they already have licenses for and are familiar with.
- *Optimized*: Our scripts will make the best effort to select the best options of tools used for synthesis and implementation in FPGAs. In order to create such scripts, a comprehensive set of computationally intensive experiments will be performed during this project in order to select the best optimization strategy for each available tool and implementation platform.
- *With single point of contact*: Our project server will work as a single point of contact and will contain all information necessary to perform benchmarking, and to share, look up, and compare the results.

The first big test of our environment is its application to the evaluation of candidates submitted to the SHA-3 contest for a new hash function standard, organized and coordinated by NIST. At the time of writing, ATHENa has been already used to collect results for two GMU studies comparing respectively the 256-bit and 512-bit versions of 14 Round 2 SHA-3 candidates. We hope that our environment and the corresponding web site will be helpful for other groups, interested in optimizing, verifying, collecting, publishing, and comparing results of their SHA-3 candidate benchmarking studies.

The environment will continue to serve the cryptographic and FPGA community for years to come, providing comprehensive and easy to locate results for multiple cryptographic standards and other classes of algorithms. Researchers all over the world will benefit from the capability of fairly, comprehensively, and automatically comparing their new algorithms, hardware architectures, and optimization methods against any previously reported work. The designers will benefit from the capability of comparing results of benchmarking the same algorithm using multiple FPGAs from several major vendors, and will be able to make an informed decision about the choice of the implementation platform most suitable for their particular application. Finally, the developers and users of our tools will benefit from the comprehensive comparisons done across tools from various vendors, and from the optimization methodologies developed and comprehensively tested as a part of this project.

References

[1] J. Nechvatal *et al.*, "Report on the development of the Advanced Encryption Standard (AES)," Oct. 2000, http://csrc.nist.gov/archive/aes/round2/r2report.pdf.

[2] B. Preneel *et al.*, "Final report of European project number IST-1999-12324, named New European Schemes for Signatures, Integrity, and Encryption," Apr. 2004, https://www.cosic.esat.kuleuven.be/nessie/Bookv015.pdf.

[3] M. Robshaw and O. Billet, *New Stream Cipher Designs: The eSTREAM Finalists*. Springer, 2008.

[4] "Cryptographic hash algorithm competition," http://csrc.nist.gov/groups/ST/hash/sha-3/.

[5] K. Gaj and P. Chodowiec, "Fast implementation and fair comparison of the final candidates for Advanced Encryption Standard using Field Programmable Gate Arrays," *LNCS 2020, Progress in Cryptology - CT-RSA 2001, Ed. D. Naccache, RSA Conference 2001 - Cryptographers' Track*, pp. 84–99, Apr. 2001.

[6] S. Drimer, "Security for volatile FPGAs," Chapter 5: The meaning and reproducibility of FPGA results, Ph.D. Dissertation, University of Cambridge, Computer Laboratory, Nov 2009, uCAM-CL-TR-763.

[7] "ATHENa Project Website," http://cryptography.gmu.edu/athena/.