4th Mediterranean Conference on Embedded Computing MECO - 2015 Budva, Montenegro

High Performance Pipelined FPGA Implementation of the SHA-3 Hash Algorithm

Lenos Ioannou, Harris E. Michail
Department of Electrical Engineering, Computer Engineering and Informatics, Cyprus University of Technology, Lemesos 3036, Cyprus
{lc.ioannou,harris.michail}@cut.ac.cy

Artemios G. Voyiatzis
SBA Research
Vienna, Austria
avoyiatzis@sba-research.org

Abstract—**The SHA-3 cryptographic hash algorithm is standardized in FIPS 202. We present a pipelined hardware architecture supporting all the four SHA-3 modes of operation and a high-performance implementation for FPGA devices that can support both multi-block and multi-message processing. Experimental results on different FPGA devices validate that the proposed design achieves significant throughput improvements compared to the available literature.**

Keywords-SHA-3, hash algorithm, FPGA, pipeline, high performance.

I. INTRODUCTION (HEADING 1)

Cryptographic hash functions are a fundamental building block of modern network security protocols and infrastructures, such as TLS, SSL, SET, IPSec, and PKI [1-4]. They are used for data integrity verification and also, in their keyed versions, as message authentication codes, for example in the case of the Hash Message Authentication Code (HMAC) [5]. MD5, SHA-1, and SHA-2 are three well-known cryptographic hash functions that are widely used. Advances in cryptanalysis raised concerns on the security level these algorithms offer and will offer in the future. In response, the National Institute of Standards and Technology (NIST) of the USA initiated a worldwide competition for the SHA-3, a new cryptographic hash function that could replace existing ones in case they are broken [6]. In October 2012, the Keccak algorithm was nominated as the SHA-3 and in May 2014 the final details and modes of operation were published as a draft FIPS 202 (Federal Information Processing Standard) for comments [7].

Following the standardization process, it is expected that in the next years the SHA-3 algorithm will be incorporated as an optional or mandatory secure hash algorithm in new or revised versions of network security protocols and standards. The ever-increasing transmission rates of data networks calls for hardware implementations of cryptographic primitives and algorithms to meet the strict timing requirements. The plans to fully adopt IPv6 contribute towards this direction. The IPv6 protocols mandate the use of the IPsec suite for protecting data exchanges; encryption and hashing are the building blocks for implementing the security protocols defined in IPsec.

In this paper, we present a novel, high-performance pipelined implementation of the SHA-3 algorithm on FPGAs. Specifically, we study the performance of one- and two-stage pipeline designs that can handle *both single- and multi-block messages*. The designs are implemented in Virtex FPGA technologies and experimental results of frequency, area, and throughput are reported. The proposed architecture achieves enormous improvements in terms of throughput and throughput/area factors compared to published literature and can serve as a design guideline for the implementers of the SHA-3 algorithm.

The rest of the paper is organized as follows. Section II presents the SHA-3 algorithm and previous works on SHA-3 FPGA implementations. Section III describes our proposed pipelined architecture. Section IV presents the experimental results and draws comparisons with the previous implementations. Finally, Section V provides the conclusions of the paper.

II. THE SHA-3 ALGORITHM AND FPGA PERFORMANCE

The SHA-3 cryptographic hash algorithm departs from the design of previous cryptographic primitives (SHA-2 and AES). The SHA-3 algorithm does not use the Merkle-Damgård construction, as is the case of MD5, SHA-1, and SHA-2 but rather uses the so-called "sponge construction". The algorithm initially *absorbs* input bits into its hash state and then an output of equal length is *squeezed* out of it. The hash state has length of 1,600 bits and can be considered as a $5x5x2^{\ell}$, where $\ell = 6$. An SHA-3 round consists of 24 iterations of five sub-rounds. The sub-rounds are denoted with Greek letters θ, ρ, π, χ, and ι and consist of simple operations, like column parity computation, bitwise rotate operation, word permutation, bitwise row combination, and bitwise exclusive-OR operation with per-round constants.

The FIPS standard defines six functions for the SHA-3 family. Four are cryptographic hash functions called SHA3-224, SHA3-256, SHA-384, and SHA3-512; two are extendable-output functions (XOFs), called SHAKE128 and SHAKE256 [7].

The SHA-3 algorithm was designed to be fast in hardware. In fact, the Keccak algorithm was the second fastest of the 14 algorithms that advanced to the second round of the competition and the fastest of the finalists [8].

A.G. Voyiatzis was supported by COMET K1, FFG - Austrian Research Promotion Agency and EU FP7 COST Action IC1403 CRYPTACUS.

During the competition period, many researchers designed and implemented the then-Keccak algorithm in FPGA technologies [9-20]. Their implementation results in terms of area and throughput are varying for more than two orders of magnitude. For example, 188 slices and 0.077 Gbps are reported in [10], while 1,207 slices and 12.98 Gbps for the same (Virtex-6) FPGA family are reported in [18].

A more recent work reports throughput closer to 20 Gbps using a two-stage *internal* pipeline [21]. In that work, the pipeline is laying within the sub-rounds hence the name "internal". However, we note that the reported throughputs are unrealistic, since the authors assume that the internal pipeline can be fed with the next blocks of the *same* message. This assumption does not hold in the case of multi-block messages because the final output of each block computation must be already available as to start the computations of the next block. Thus, it is more realistic to consider that the attainable throughput is closer to 10 Gbps and the throughput/area ratio is closer to 5-6 Mbps/slice in the case of multi-block messages. This is because the two stages of the pipeline can be fully exploited only in the (rare) case of single-block messages.

III. Proposed Architecture

We describe in the following an architecture that is capable of handling *multi-block* messages and of processing of *multiple* messages. Our architecture supports all the four standardized SHA-3 modes of operation. A high-level dataflow diagram is depicted in Fig. 1. The SHA-3 core functionality is contained in the five sub-round blocks θ, ρ, π, χ, and ι. These blocks realize the sub-round transformations of the state matrix A that has a size of 1,600 bits. A control logic with a 5-bit counter drives the 24 repetitions required for a complete SHA-3 round and the use of the appropriate 64-bit constants at each round. A 2-to-1 multiplexer selects as input the result of the previous round (if the counter is greater than 0) or the next input block of 1,600 bits. The result of each repetition is stored in a register and upon completion, the "byte inverse" and "truncate" operations take place for producing the final 512-bit hash value.

The architecture utilizes a software scheduler performing three functions. The first is to prepare the input by splitting and padding long messages into blocks of 1,600 bits (multi-block messages). The second is to truncate, if necessary, the 512-bit output of the hash computation in the appropriate size of the selected mode of operation: 224, 256, 384, or 512 bits for the SHA3-224, SHA3-256, SHA3-384, or SHA3-512 respectively. Finally, the third is to update the state matrix A in the case of multi-block messages.

The critical path of our architecture is located inside the round transformation, passing through the multiplexer and the five sub-round blocks. It consists of a multiplexer; three XOR units and a cyclic left shift block at the θ sub-round block; a cyclic left shift block at the combined ρ and π sub-round blocks; a XOR unit at the χ sub-round block; and a XOR unit at the ι sub-round block. Since the critical path lies within the round computation, the delay of the software scheduler is accommodated. This design choice is justified since it reduces the overall design's complexity and it does not affect its security level [22].

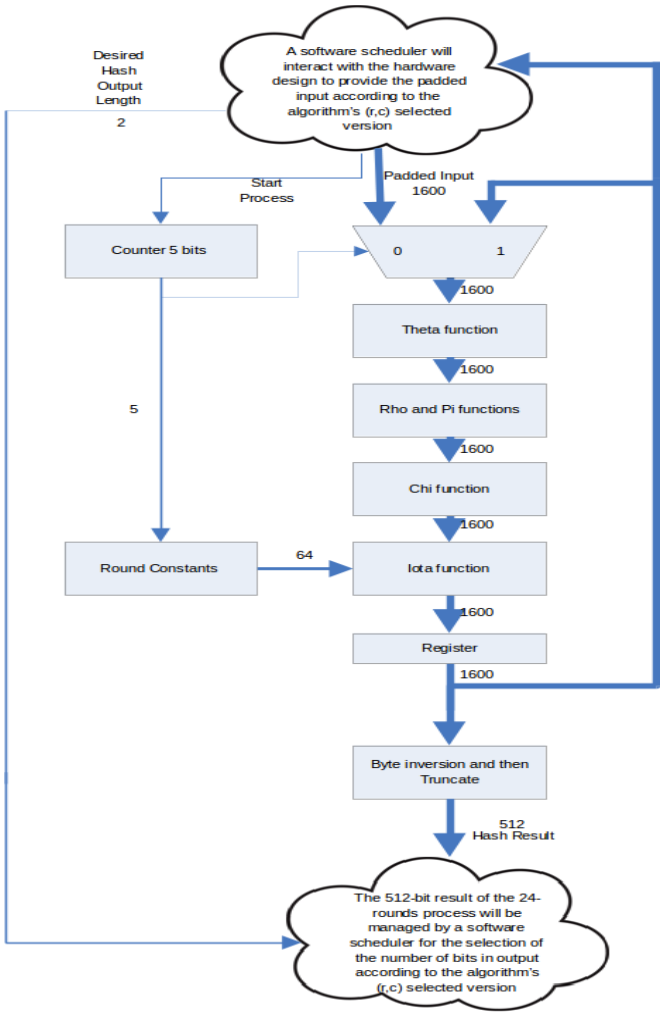

Figure 1. SHA-3 block-level architecture.

The approach described above can be easily extended into a *pipelined design*. Here, we analyze the case of a two-stage pipeline, as depicted in Fig. 2. The output register of the first stage is used to feed the input of the second stage and two 4-bit counters are now used. The design allows outputting a first block after 24 cycles and produces a new block every 12 cycles. This holds under the assumption that the software scheduler handles multiple messages for hashing, as in the case of a network card servicing multiple secure streams of packets that require hashing. In this sense, it is an *external* pipeline: all the five sub-rounds are used at each pipeline stage.

The proposed architecture can fully occupy the pipeline. To see this, assume there are two or more messages of one or more blocks are available and denote with M_i^j the j-th block of the i-th message. During the first 12 cycles, M_1^1 passes through the first stage of the pipeline and the second stage remains empty. During the next 12 cycles, M_1^1 passes through the second stage

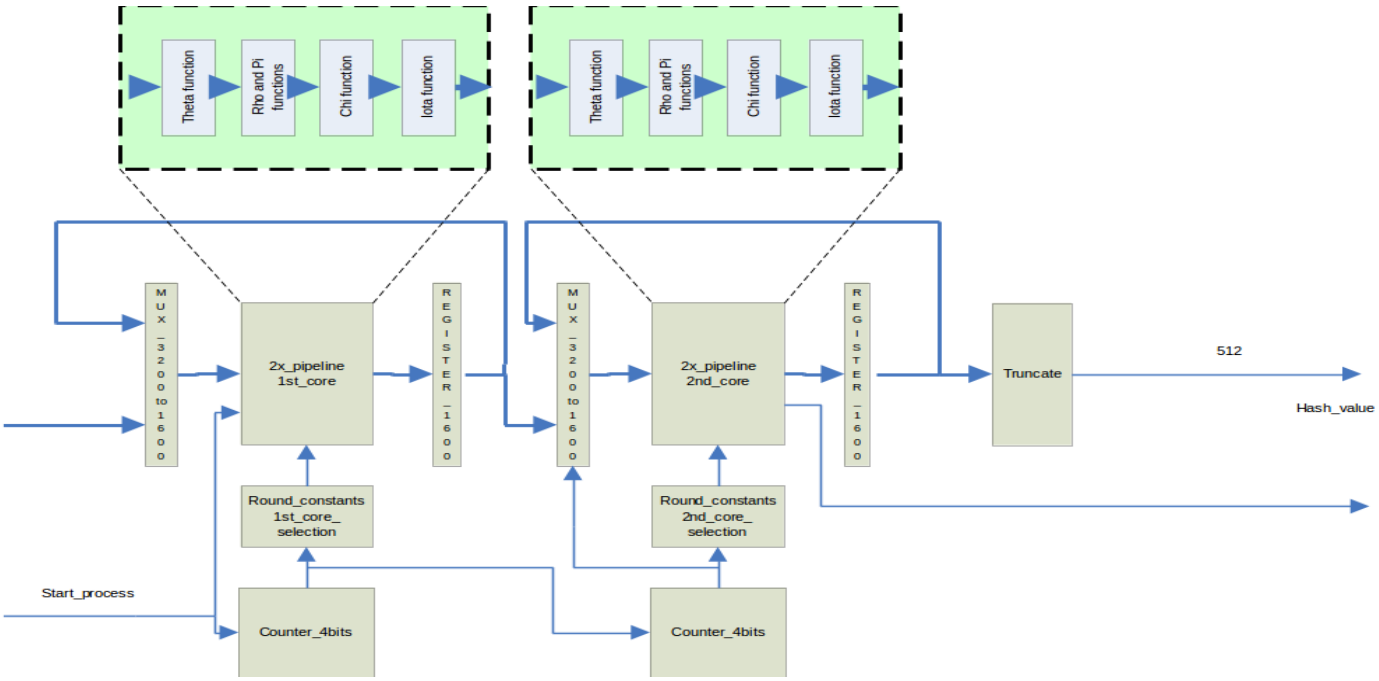

Figure 2. SHA-3two-stage pipeline architecture.

of the pipeline and M_2^1 passes through the first stage. During the next cycles, the next blocks M_1^2,M_2^2, … are fed and advance through the pipeline stages. If no more blocks are available, then the first block of the next message, M_3^1 is fed and the process continues as before. Experimental Results and Discussion

The proposed pipelined architecture was captured in VHDL using the Xilinx ISE Design Suite. Three FPGA boards were used, namely Virtex-4 XC4VLX200, Virtex-5 XC5VLX330T, and Virtex-6 XC6VLX760.

Table I summarizes the performance of one- and two-stage pipelined implementation of the SHA-3 on the three boards and in terms of frequency, area, throughput, and throughput/area ratio. The throughput is calculated as $T = (\#bits\ x\ f)/\ \#cycles$, where $\#bits$ refer to the number of the processed bits, $\#cycles$ corresponds to the required clock cycles between successive messages to generate each message digest, and f is the operating frequency of the design. The reported throughputs are for the SHA3-512 mode, in alignment with the available literature.

While the area does increase significantly (but it is not doubled), the attained throughput is almost the double in all cases. Furthermore, in the case of the Virtex-5, the throughput/area is also increased. In Fig. 3, we summarize and compare the raw throughput measures reported in the literature. Our two-stage pipelined implementation outperforms by more than 30% any published results for the case of Virtex-5 and Virtex-6. In Fig. 4, we summarize and compare the throughput per area ratios reported in the literature. Our two-stage pipelined implementation is second best in Virtex-5 technology and it is by far the best for the case of Virtex-6.

TABLE I. EXPERIMENTAL RESULTS

Pipeline stages	*Frequency (MHz)*	*Area (slices)*	*Throughput (Gbps)*	*Throughput/area (Mbps/slice)*
Virtex-4 XC4VLX200				
1	273	2,365	6.552	2.770
2	269	5,494	12.912	2.350
Virtex-5 XC5VLX330T				
1	382	1,581	9.168	5.799
2	352	2,652	16.896	6.371
Virtex-6 XC6VLX760				
1	412	1,115	9.888	8.868
2	391	2,296	18.768	8.174

IV. CONCLUSIONS AND FUTURE WORK

A novel two-staged pipelined architecture for the SHA-3 cryptographic hash algorithm is proposed in this paper. The novelty of the architecture lies in the fact that it can efficiently serve multiple multi-block messages. The design exploits a software scheduler for offloading message splitting and padding, reducing circuit complexity, and increasing performance. The design can be considered as an *external, round pipeline* that can saturate the two stages of the pipeline. This is a common case in protecting packets of different network streams. The presented implementations in three Virtex families outperform in throughput by at least 30% any published results.

As a future work and based on the promising results, we aim to further explore the design space of deeper pipelines.

Furthermore, to study design trade-offs for area-throughput optimizations and further improve the per-round performance.

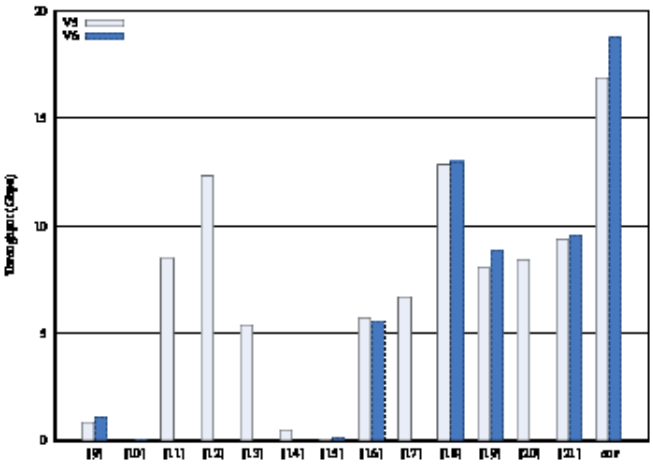

Figure 3. SHA-3 throughput on V5 and V6 (our and [9]–[21]).

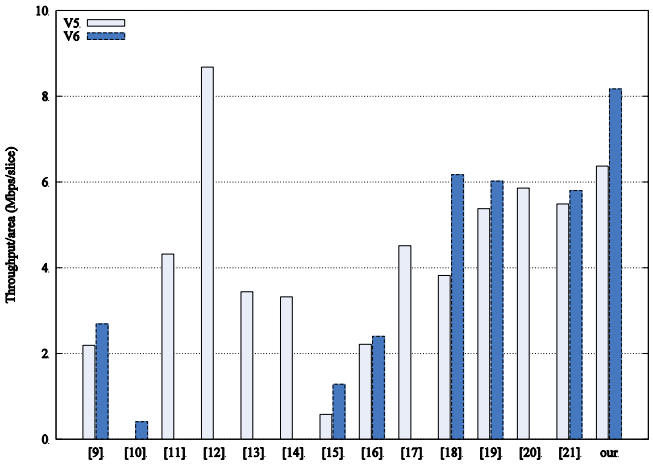

Figure 4. SHA-3 throughput/area on V5 and V6 (our and [9]-[21]).

REFERENCES

[1] S. Thomas, *SSL & TLS Essentials: Securing the Web*. John Wiley and Sons Publications, 2000.

[2] L. Loeb, *Secure Electronic Transactions: Introduction and Technical Reference*. Artech House Publishers, 1998.

[3] P. Loshin, *IPv6: Theory, Protocol and Practice*. Elsevier Publications: USA, 2004.

[4] "SP 800-32, Introduction to public key technology and the federal PKI infrastructure," 2001, NIST Publication, US Dept. of Commerce.

[5] "FIPS 198, the keyed-hash message authentication code (HMAC) federal information processing standard," 2002, NIST Publication, US Dept. of Commerce.

[6] NIST, "Cryptographic hash algorithm competition - SHA-3," 2010, http://csrc.nist.gov/groups/ST/hash/sha-3/index.html.

[7] "Draft FIPS 202, SHA-3 standard: Permutation-based hash and extendable-output functions," May 2014, NIST Publication, US Dept. of Commerce, http://csrc.nist.gov/publications/drafts/fips-202/fips 202 draft.pdf.

[8] X. Guo, S. Huang, L. Nazh, and P. Schaumont, "Fair and comprehensive performance evaluation of 14 second round SHA-3 ASIC implementations," NIST 2nd SHA-3 Candidate Conference, 2010, http://csrc.nist.gov/groups/ST/hash/sha-3/Round2/Aug2010/.

[9] B. Jungk, "Evaluation of compact FPGA implementations for all SHA-3 finalists," NIST 3rd SHA-3 Candidate Conference, 2012, http://csrc.nist.gov/groups/ST/hash/sha-3/Round3/March2012.

[10] S. Kerckhof, F. Durvaux, N. Veyrat-Charvillon, F. Regazzoni, G. M. de Dormale, and F.-X. Standaert, "Compact FPGA implementations of the five SHA-3 finalists," in Smart Card Research and Advanced Applications. Springer, 2011, pp. 217–233.

[11] B. Baldwin, A. Byrne, L. Lu, M. Hamilton, N. Hanley, M. O'Neill, and W. P. Marnane, "FPGA implementations of the round two SHA-3 candidates," in Field Programmable Logic and Applications (FPL), 2010 International Conference on. IEEE, 2010, pp. 400–407.

[12] Y. Jararweh, L. Tawalbeh, H. Tawalbeh, and A. Moh'd, "Hardware performance evaluation of SHA-3 candidate algorithms," Journal of Information Security, vol. 3, p. 69, 2012.

[13] A. Gholipour and S. Mirzakuchaki, "High-speed implementation of the Keccak hash function on FPGA," International Journal of Advanced Computer Science, vol. 2, no. 8, 2012.

[14] I. San and N. At, "Compact Keccak hardware architecture for data integrity and authentication on FPGAs," Information Security Journal: A Global Perspective, vol. 21, no. 5, pp. 231–242, 2012.

[15] J.-P. Kaps, P. Yalla, K. K. Surapathi, B. Habib, S. Vadlamudi, and S. Gurung, "Lightweight implementations of SHA-3 finalists on FPGAs," NIST 3rd SHA-3 Candidate Conference, 2012, http://csrc.nist.gov/groups/ST/hash/sha-3/Round3/March2012.

[16] G. Provelengios, P. Kitsos, N. Sklavos, and C. Koulamas, "FPGA-based design approaches of Keccak hash function," in Digital System Design (DSD), 2012 15th Euromicro Conference on. IEEE, 2012, pp. 648–653.

[17] J. Str¨ombergson, "Implementation of the Keccak hash function in FPGA devices," 2008 http://www.strombergson.com/files/Keccak in FPGAs.pdf.

[18] E. Homsirikamol, M. Rogawski, and K. Gaj, "Comparing hardware performance of round 3 SHA-3 candidates using multiple hardware architectures in Xilinx and Altera FPGAs," Ecrypt II Hash Workshop 2011, 2011, http://www.ecrypt.eu.org/hash2011/proceedings/hash2011 07.pdf.

[19] K. Gaj, E. Homsirikamol, M. Rogawski, R. Shahid, and M. U. Sharif, "Comprehensive evaluation of high-speed and medium-speed implementations of five SHA-3 finalists using Xilinx and Altera FPGAs," Cryptology ePrint Archive, Report 2012/368, 2012, http://eprint.iacr.org/.

[20] K. Kobayashi, J. Ikegami, M. Knezevic, E. X. Guo, S. Matsuo, S. Huang, L. Nazhandali, U. Kocabas, J. Fan, A. Satoh et al., "Prototyping platform for performance evaluation of SHA-3 candidates," in Hardware-Oriented Security and Trust (HOST), 2010 IEEE International Symposium on. IEEE, 2010, pp. 60–63.

[21] G. Athanasiou, G.-P. Makkas, and G. Theodoridis, "High throughput pipelined FPGA implementation of the new SHA-3 cryptographic hash algorithm," in Communications, Control and Signal Processing (ISCCSP), 2014 6th International Symposium on, May 2014, pp. 538-541.

[22] H. E. Michail, G. S. Athanasiou, V. Kelefouras, G. Theodoridis, and C. E. Goutis, "On the exploitation of a high-throughput SHA-256 FPGA design for HMAC," ACM Transactions on Reconfigurable Technology and Systems (TRETS), vol. 5, no. 1, p. 2, 2012.