

High throughput implementation of the new Secure Hash Algorithm through partial unrolling

Konstantinos Aisopos

Athanasios P. Kakarountas

Haralambos Michail

Costas E. Goutis

Dpt. of Electrical and Computer Engineering

University of Patras

Patras, GREECE

Abstract—A design approach to create small-sized high-speed implementation of the new version of Secure Hash Algorithm is proposed. The resulted design can be easily embedded to operate in HMAC IP cores, providing a high degree of security. The proposed implementation does not introduce significant area penalty, compared to other competitive designs. However the achieved throughput presents an increase compared to commercially available IP cores that range from 43%-1830%.

I. INTRODUCTION

Hash functions are common and critical cryptographic primitives. Their primary application is combined use with public-key cryptosystems in digital signature schemes. By far the most widespread hash functions are SHA-1 (Secure Hash Algorithm-1), a revised version of the NIST American federal standard [1], and MD5 (Message Digest) [2]. These two hash functions are widely known for being used in the Keyed-Hash Message Authentication Code (HMAC) [3], which is met in numerous communication applications, to address authentication issues. However, due to the ever increasing demand for security, a new version of the Secure Hash Algorithm [1] has been introduced. The new version of SHA (commonly named SHA-2) satisfies various needs of applications and comes in a variety of configurations. The most widely known is the SHA-512 version, which produces a message digest of 512 bits. The latter hash function is considered for inclusion in many applications, replacing existing SHA-1.

The SHA hash functions were selected for the Digital Signature Algorithm (DSA), as specified in the Digital Signature Standard (DSS) [4], and whenever a secure hash algorithm is required for federal applications. The latter hash functions are used widely in the field of communications, where until nowadays throughput of the cryptographic systems' was not required to be high. However, since the use of the HMAC in the IPsec [5], e-payment and VPN applications, the throughput of the cryptographic system, especially the server, has to reach the highest degree of throughput. The high-speed requirement of the hash value calculation is strongly related to the streamlined

communication of two subscribers of the latter mentioned applications. Especially in these applications that transmission and reception rates are high, any latency or delay on calculating the digital signature of the data packet leads to degradation of the network's quality of service. Software implementations are presenting unacceptable performance for high-speed applications. Additionally, most of the proposed implementations didn't consider that the products introduced to the market tend to be as small as possible.

The latter facts were a strong motivation to propose a novel hardware implementation of the SHA-512, with many differences from existing competitive implementations. The proposed implementation was developed as an IP core, in order to be reused in a variety of applications, allowing integration in FPGA or ASIC technologies. Thus, this paper aims to provide a low-cost design approach, compared to the proposed solutions from both academia and industry, in order to satisfy the requirements of the new communication applications. The proposed implementation introduces a negligible area penalty; increasing the throughput and keeping the area small enough as required by most portable communication devices. The main contribution of this work is the design approach to optimize performance without introducing extra area. Furthermore, power dissipation is kept low in contrast to existing implementations of similar throughput and size.

The rest of this paper is organized as follows. In Section II, the implementation of SHA-512 is presented, as it is proposed by the standard [1]. In Section III the proposed implementation is presented in depth, providing details regarding the architecture, the logic and the modifications to decrease the critical path, and the characteristics of the circuit that are expected. In Section IV the proposed SHA-512 is implemented for an FPGA technology and it is compared to other implementations. Finally, in Section V the paper concludes.

II. EXISTING IMPLEMENTATION OF SHA-512

The Secure Hash Standard [1] describes in detail the SHA-512 hash function. SHA-512 may be used to hash an k -bits message, where $0 \leq k < 2^{128}$. During preprocessing phase

We thank European Social Fund (ESF), Operational Program for Educational and Vocational Training II (EPEAEK II) and particularly the program PYTHAGORAS.

the message is padded and parsed into 1024-bit message blocks, which are used to generate the message schedule W_t 's. SHA-512 requires 80 cycles to produce the 512-bit message digest. Each cycle requires the previous round's results, W_t , as well as constant values K_t .

Following the guidelines of [1], the architecture of a SHA-512 core is formed as illustrated in Fig. 1. Constant's array is a hardwired array that provides constant values K_t and the constant initial hash values. In addition, it includes the W_t generators. MS RAM stores the W_t 's.

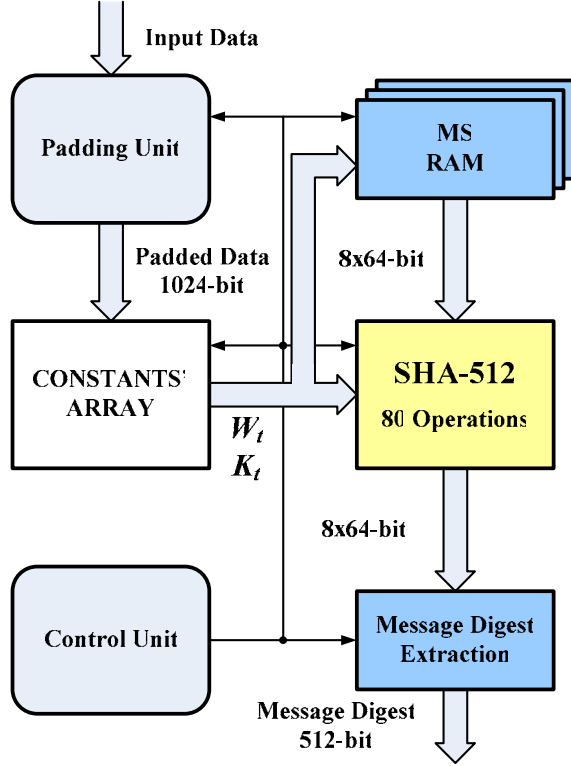


Figure 1. A typical SHA-2 IP core architecture

Each cycle has an 512-bit input and an 512-bit output, both formed by eight 64-bit words: $A-H$. The expressions to calculate each cycle's outputs are given in Eq. 1.

$$\begin{aligned}
 A_{t+1} &= \text{Tmp} + \Sigma_0(A_t) + \text{Maj}(A_t + B_t + C_t) \\
 B_{t+1} &= A_t \\
 C_{t+1} &= B_t \\
 D_{t+1} &= C_t \\
 E_{t+1} &= \text{Tmp} + D_t \\
 F_{t+1} &= E_t \\
 G_{t+1} &= F_t \\
 H_{t+1} &= G_t \\
 \text{Tmp} &= W_{t+1} + K_{t+1} + H_t + \Sigma_1(E_t) + \text{Ch}(E_t + F_t + G_t) \\
 \text{Ch}(x,y,z) &= (x y) \oplus (-x z) \\
 \text{Maj}(x,y,z) &= (x y) \oplus (x z) \oplus (y z) \\
 \Sigma_0(x) &= \text{ROTR}^{28}(x) \oplus \text{ROTR}^{34}(x) \oplus \text{ROTR}^{39}(x) \\
 \Sigma_1(x) &= \text{ROTR}^{14}(x) \oplus \text{ROTR}^{18}(x) \oplus \text{ROTR}^{41}(x)
 \end{aligned} \tag{1}$$

where $\text{ROTR}^y(x)$ stands for rotation of x by y positions to the left. In Fig. 2., two consecutive operation blocks are illustrated.

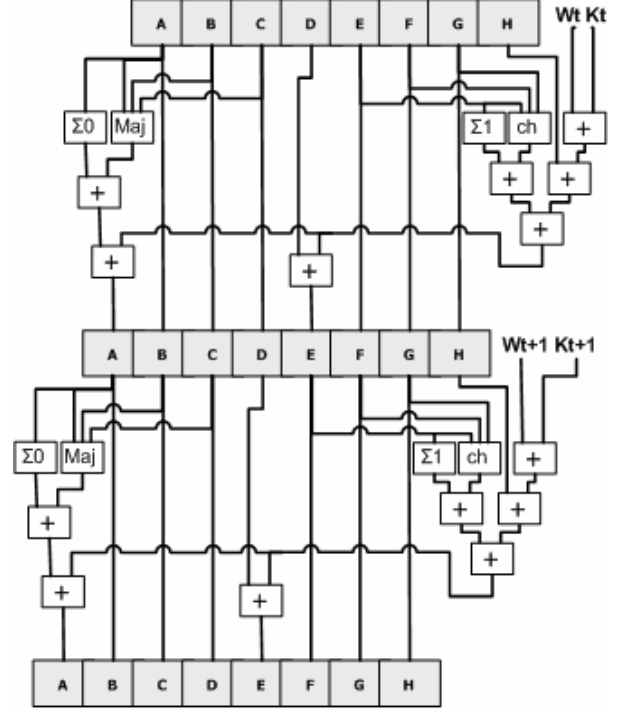


Figure 2. Two consecutive SHA-512 operation blocks

Something that can be easily observed from the latter figure is that the operation blocks (in contrast to the operation block of SHA-1 [1]) are identical to each other during the 80 operations for generating the message digest. This can be exploited to minimize the size by rolling up the algorithm and using any configuration of Z pipelined operations blocks. This approach is followed by most of the existing implementations. Thus, throughput is Z times the throughput of an implementation with a single operation block.

III. PROPOSED SHA-512 IMPLEMENTATION

A. Basic Implementation

Considering the existing architecture, the paths lack of symmetry: Some inputs feed directly some outputs ($B-D$ and $F-H$) while others delay to produce result (A and E). This results to next round's available inputs "waiting" for an unavailable minority of inputs, to proceed to the next round. The proposed optimization is based on unrolling the operations of the algorithm. Specifically the goal is to permit next round's available inputs to calculate immediately an intermediate result. Then, when the required intermediate inputs become available, the internal result will be ready and may be used to produce the output in a shorter time.

The proposed design approach is to condense two cycles ($t+1, t+2$) in one. During the first half of the condensed cycle, which corresponds to the $t+1$ uncondensed cycle, two operations will be working in parallel: the $t+1$ outputs calculation and the $t+2$ internal result calculation based on the $t+1$ outputs provided shortly. As long as $C_t + G_t$ are available, D_{t+1} and H_{t+1} can start calculating next round's internal result at once (Eq. 2)

$$\begin{aligned} \text{Im1} &= W_{t+2} + K_{t+2} + H_{t+1} = W_{t+2} + K_{t+2} + G_t \\ \text{Im2} &= W_{t+2} + K_{t+2} + H_{t+1} + D_{t+1} = W_{t+2} + K_{t+2} + G_t + C_t \end{aligned} \quad (2)$$

So, the second half of the cycle will have a remarkably shorter critical path compared to the first half (Eq. 3)

$$\begin{aligned} A_{t+2} &= \text{Im1} + \Sigma_1(E_{t+1}) + \Sigma_0(A_{t+1}) + \text{Ch}(E_{t+1} + F_{t+1} + G_{t+1}) + \\ &\quad \text{Maj}(A_{t+1} + B_{t+1} + C_{t+1}) \\ E_{t+2} &= \text{Im2} + \Sigma_1(E_{t+1}) + \text{Ch}(E_{t+1} + F_{t+1} + G_{t+1}) \end{aligned} \quad (3)$$

The grey marked areas on Fig. 3 indicate the parts of the proposed SHA-512 operation block that operate in parallel.

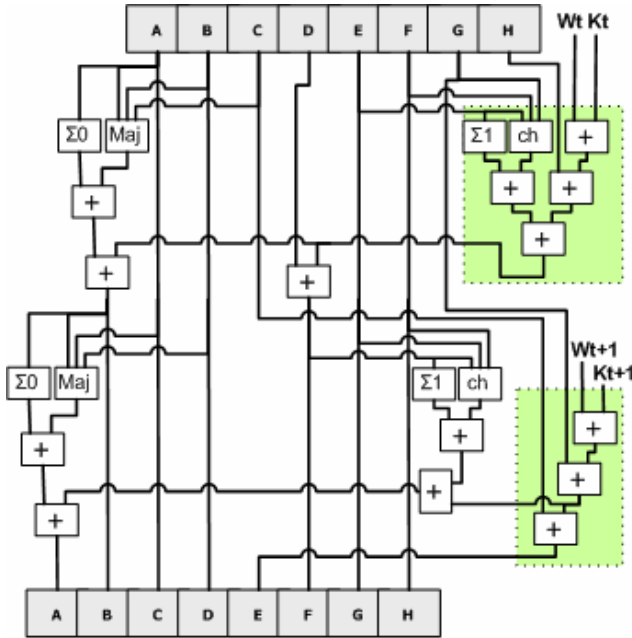


Figure 3. Proposed SHA-512 operation block applying partial unrolling

The selection to unroll two operations and not more can be answered by the analysis of the resulted critical path. A clue is derived from the fact that input A_t and B_t feed directly outputs C_{t+2} and D_{t+2} respectively. If more than two operations were unrolled then the critical path would not only increase by the circuitry needed to calculate output A_t , but it would also be delayed from the parallel circuitry that calculates the intermediate values. For example in the case of four operations unrolling, the critical path is not duplicated but it is further delayed by 25%. After careful analysis on the SHA-512 algorithm, it was derived that the best ratio of unrolled operations per achieved frequency was given for

two operations unrolling. Partial unrolling does not always present the same best fitting point, i.e. two operations unrolled, but it is the algorithm that indicates the best configuration. The SHA-512 properties are appropriate for the proposed unrolling.

B. Additional paths to implement parallelism

The next observation, that may reveal more paths to implement parallelism, is that the path to provide E is remarkably shorter than the path to provide A : Synthesis tools are optimising structures including multilevel CPAs by replacing them with CSAs. Practically, each addition is implemented with a CSA except from the last one when the two remaining inputs cannot be condensed any more. CSA cannot be used when calculating $\Sigma_0(A_{t+1}) + \text{Maj}(A_{t+1} + B_{t+1} + C_{t+1})$ which results to the foresaid calculation waiting for $W_{t+1} + K_{t+1} + H_t + \Sigma_1(E_t) + \text{ch}(E_t + F_t + G_t)$ result to be produced in order to be further condensed. Thus, E_{t+1} will be available before A_{t+1} .

We can take advantage of this delay to produce more intermediate results from E_{t+1} as shown in Eq. 4.

$$\begin{aligned} \text{Im3} &= \text{Im1} + \Sigma_1(E_{t+1}) + \text{Ch}(E_{t+1} + F_{t+1} + G_{t+1}) \\ \text{Im4} &= \text{Im2} + \Sigma_1(E_{t+1}) + \text{Ch}(E_{t+1} + F_{t+1} + G_{t+1}) \end{aligned} \quad (4)$$

This results to an even shorter critical path for the second half of the cycle as shown in Eq. 5.

$$\begin{aligned} A_{t+2} &= \text{Im3} + \Sigma_0(A_{t+1}) + \text{Maj}(A_{t+1} + B_{t+1} + C_{t+1}) \\ E_{t+2} &= \text{Im4} \end{aligned} \quad (5)$$

The new critical path for the condensed cycle increases by $\Sigma_0(A_{t+1}) + \text{Maj}(A_{t+1} + B_{t+1} + C_{t+1})$ compared with the uncondensed one, that is extra two XOR gates and one CPA. Consequently, the maximum operation frequency will decrease by 38%. Although, the throughput is increased significantly because the condensed cycle produces the result that two uncondensed ones would produce as shown in Eq. 6.

$$\begin{aligned} \text{Throughput} &= \frac{\# \text{ bits} \cdot f_{\text{operation}}}{\# \text{ operations}} \\ \text{Throughput}' &= \frac{\# \text{ bits} \cdot f_{\text{operation}}'}{\# \text{ operations}'} = \frac{\# \text{ bits} \cdot 0,62 f_{\text{operation}}}{\frac{\# \text{ operations}}{2}} = 1,24 \cdot \text{Throughput} \end{aligned} \quad (6)$$

C. Power Dissipation

A significant advantage of the unrolling transformation is power dissipation. Application of partial unrolling does increase critical path, but usually it is applied when the increase does not exceed 80% of the sole operation block. Thus, operation frequency is reduced but throughput is increased significantly overcoming the increase of the critical path. Furthermore, using the proposed SHA-512 operation block, the temporal register write operations are reduced by 50%. This means that with the sole operation block there was the need for 80 reads and writes to the

temporal registers. With the partially unrolled SHA-512 operation block reads and writes are reduced to the number of 40. This is a significant power decrease. Additionally, although the required hardware to implement the operation block is doubled, the maximum frequency operation block is reduced, which in turn reduces the overall power dissipation of the SHA-512 core.

Overall, power dissipation is decreased not only by the reduced number of accesses to the temporal registers but also by the decrease of the operation frequency. However, throughput reaches the levels of two sole blocks connected in a pipeline structure. Compared to this structure, power dissipation is near 30% reduced while throughput is kept at the same level. Most works met in the international literature do not consider power dissipation issues, pursuing only the maximum operation frequency hence the overall throughput. In this paper, the throughput of the proposed implementation exceeds in some cases that of the commercially available IPs respecting the needs for low-power operation.

IV. IMPLEMENTATION AND RESULTS

The proposed SHA-512 was captured in VHDL and was fully simulated and verified using the Model Technology's ModelSim Simulator. A large number of test vectors were used to verify the designs' functionality. Some test vectors that were used were adopted from the standards [1],[2],[3] and the rest were randomly created. The XILINX FPGA technologies were selected as the targeted technologies, synthesizing the designs for the VIRTEX-E device family. The selection of the technology was based on the available information from commercial IP cores [8],[9],[10] and research works [6],[7].

The synthesis tool used to port VHDL to the targeted technologies was Synplicity's Synplify Pro Synthesis Tool. Simulation of the designs was also performed after synthesis, exploiting the back annotated information that was extracted from the synthesis tool. Further evaluation of the designs was performed using the prototype board for the Xilinx Virtex-E device family. The FPGA device on the board is an XCV1600EBG560, which was more than sufficient to implement the proposed SHA-512.

Probing of the FPGA's pins was done using a logic analyzer. No scaling frequency technique was followed, selecting one master clock for the system, which was driven in the FPGA from an onboard oscillator. The behaviour of the implementation was verified exploiting the large capacity of the FPGA device. Thus, a BIST unit was developed which was responsible for providing predefined inputs to the HMAC and monitoring of the output response. This allowed additionally to cope with high output throughput.

After successful verification of the SHA-512 functionality, the proposed implementation was compared to existing commercially available IP cores [8],[9],[10]. As it is shown in Table I, only the implementation of [8] is close

to the characteristics of the proposed implementation. However, although the required slices of [8] are less (almost 1%) the achieved operating frequency of the proposed implementation allows throughput to reach a value near the 1,5 Gbps. The rest of the commercially available IP cores are even 60% slower.

In the case of the implementations that have been proposed by the academia, [6] presents a throughput even 18300% lower than the achieved from the proposed implementation. On the other hand [7] is considered a very good implementation compared even to existing commercial IP cores. However, the area overhead is not suitable for small-sized and low-power applications.

TABLE I. CHARACTERISTICS OF VARIOUS IMPLEMENTATIONS OF THE SHA-512 HASH FUNCTION

SHA-512	XILINX Virtex E		
	Area (Slices)	Operation frequency (MHz)	Throughput (Mbps)
[6]	1004	42,9	77
[7]	3441	55,5	670
[8]	2690	68,0	859
[9] ¹	880	77,0	606
[10]	2403	49,5	626
Proposed	2710	58,0	1485

- a. Operation frequencies of the other implementations are given as reported in scientific articles and the datasheets of the companies
- b. ¹ this is a SHA-256 implementation but it is offered for comparison reasons

It has to be mentioned that surprisingly not all the companies that provide cryptographic IP cores support SHA-512. This is the reason that only few are included in Table I. However, due to the wide use of SHA-1 even nowadays, it is expected to appear more SHA-512 IP cores in the near future. This supports more the motivation to present and propose novel implementations of SHA-512. As it was shown in this paper, the proposed implementation is the best performing one and it is the first that provides throughput that exceeds the 1Gbps.

V. CONCLUSIONS

A novel partially unrolled implementation of the SHA-512 was presented in this paper. It was showed that the critical path can be increased without alternating significantly the SHA-512 throughput. This was achieved by exploiting special properties of the hash function. The proposed implementation was expected to present at least 15% higher throughput than any other available implementation (from academia or industry). Significant design effort was paid to keep power dissipation low. The experimental results showed that power dissipation was kept low and comparable to an implementation containing a sole operation block of SHA-512. Finally the design was fully

tested and verified for the Xilinx Virtex-E FPGA family using a prototype board.

ACKNOWLEDGMENT

We thank European Social Fund (ESF), Operational Program for Educational and Vocational Training II (EPEAEK II) and particularly the program PYTHAGORAS. The fund was used to improve the educational material and the educational process in the course and the laboratory exercises of Prof. C.E. Goutis. This allowed us to finalize our work. Also we would like to thank Alma Technologies, for their help when it was requested.

REFERENCES

- [1] National Institute of Standards and Technology (NIST). Secure Hash Standard (SHS). FIPS PUB 180-2 Standard, 2002.
- [2] Rivest, R. L. The MD5 Message Digest Algorithm. IETF Network Working Group, RFC 1321, 1992.
- [3] National Institute of Standards and Technology (NIST). The Keyed-Hash Message Authentication Code (HMAC). FIPS PUB 198 Standard, 2002.
- [4] National Institute of Standards and Technology (NIST). Digital Signature Standard (DSS). FIPS PUB 186-2, 2000.
- [5] IP Security Protocol Charter (IPSEC). Internet Drafts for IPsec. <http://www.ietf.org/html.charters/ipsec-charter.html>.
- [6] S. Dominikus, "A Hardware Implementation of MD-4 Family Hash Algorithms," in Proc. of IEEE International Conference on Electronics, Circuits and Systems (ICECS'02), pp. 1143–1146, 2002.
- [7] T. Grembowski, et al. "Comparative Analysis of the Hardware Implementations of Hash Functions SHA-1 and SHA-512," in Proc. of Information Security Conference (ISC'02). Springer-Verlag, Heidelberg, pp. 75–89, 2002.
- [8] ALMA Technologies. Web page, available at <http://www.alma-tech.com>.
- [9] Helion Technology Ltd. Web page, available at <http://www.heliontech.com>.
- [10] Amphion. Web page, available at <http://www.amphion.com/index.html>.