Networking Data Integrity: High Speed Architectures and Hardware Implementations

Nicolas Sklavos, Epaminondas Alexopoulos, and Odysseas Koufopavlou
Electrical and Computer Engineering Department, University of Patras, Greece

Abstract: *Hash functions are widely used in encryption schemes and security layers of communication protocols (wap, ipsec) for data integrity, digital signature and message authentication codes. In addition to the demanded high security level, the need for high performance is a major factor of the security implementations. In this work, an ultra high speed architecture for the hardware implementation of both md5 and sha-1 is proposed. Both hash functions have been developed with vhdl description language and have been integrated in fpga devices. The introduced md5 implementation performance is equal to 2,1 gbps while sha-1 proposed implementation achieves throughput equal to 2,3 gbps. Both proposed implementations are compared in throughput, operating frequency and in the area-delay product, with other related works. From these comparisons, it is proven that the md5 proposed implementation is better by a factor range from 700% to 1500%. The sha-1 proposed implementation is better by about 800% to 1700% in the term of performance, compared with the other conventional works.*

Keywords: *MD5, SHA-1, hash functions, hardware implementation, cryptography, wireless protocols security*.

Received February 24, 2003; accepted May 5, 2003

1. Introduction

In the last few years, communications have been grown up rapidly, due to the users increased needs and the maximized offered services and applications. In parallel to this growth, is the increased major demand for powerful secure implementations of cryptographic algorithms and encryption schemes. Almost all the communication protocols have specified security layers, which ensure security with high-level strength. In order these special and sensitive needs for cryptography to be satisfied in the desirable level, different categories of encryption algorithms, support the communication protocols and networks defense.

Hash functions belong to one of the most important categories of encryption algorithms, against the external harmful attacks [1]. They are widely used in a great number of security applications and have been included in the specifications of communication protocols like WAP and IPsec. They mainly serve data integrity and message authentication codes (MAC) but they are also used in digital signatures, HMAC and random number generators. The most well know and widely used hash functions are MD5 and SHA-1 [10, 11].

In this paper, an ultra high speed architecture for the VLSI implementation of both MD5 and SHA-1 hash functions is proposed. Both hash functions specifications and operation processes have been studied and a pipeline architecture with four processing levels is proposed. The internal components of the proposed architecture are analyzed and presented for each one of the MD5 and SHA-1 implementations separately. Both MD5 and SHA-1 hash functions have been integrated by using VHDL description language, in FPGA hardware module. The synthesis results are presented in detail. The MD5 implementation throughput reaches the value of 2,1 Gbps while the SHA-1 throughput is equal to 2,3 Gbps. Both proposed implementations are compared with other related works for both MD5 [2, 4], and SHA-1 [3, 4, 6, 9]. In these comparisons, the hardware terms of system performance (throughput), operating frequency and covered area are given, for both proposed and conventional implementations. Furthermore, in order to have a fair a detailed comparison, the Area-Delay products for all the hardware implementations are compared. These comparisons prove that the proposed implementations have ultra high-speed performance compared with the other conventional architectures in all of the cases. Especially the proposed MD5 implementation has better performance with a factor equal from 7 to 15 times, compared with the conventional architectures. The Area-Delay product of the proposed MD5 implementation is still better (less) compared with all the others ones. The SHA-1 proposed implementation performance is superior to all the other conventional works. Especially, it is better at about 8 to 17 times. It is proved that the SHA-1 introduced VLSI integration has better Area-Delay product in all of the cases. The proposed MD5 and SHA-1 implementations could be used efficiently in applications with high performance and minimized covered area demands. They can be flexible solutions for hardware implementations of data integrity, digital signatures and MAC applications. Furthermore, they can substitute

successfully possible existing developments, with better achieved performance and high security offered level at the same time. Especially, they can be used to implement the data integrity specified schemes of wireless protocols such as WAP and IPsec.

This paper is organized as follows: In section 2 both MD5 and SHA-1 hash functions are introduced. In the next section, the proposed system architecture is presented. The internal components of this architecture for both MD5 and SHA-1 implementations are described in detail. The synthesis results for MD5 and SHA-1 VLSI implementations are given in the next section 4. Comparisons with other related works are also presented in the same section. Finally, important conclusions and observations are discussed in section5.

2. Hash Functions

An n-bit hash is a map from arbitrary length messages to n-bit hash values [1]. An n-bit hash function is an n-bit hash which is one-way and collision-resistant. One-way is the function that for a given hash value, it should require work equivalent about 2^n hash computations to find any message that hashes that value. The term collision resistance characterizes the functions that finding two messages, which hash the same value, should require work equivalent to $2^{n/2}$ hash computations. Of course the hash functions architectures ofare public and commonly known. In the hash computation process, there is no secrecy and no keys, public or private, are used at all. The security is based on the one-way operation of each hash function itself. Hash functions are used for digital signature scheme, data integrity, HMAC and other cryptographic purposes (random number generators) [5, 8]. In most of the wireless protocols, such as WAP and IPsec, the widely used hash functions are SHA-1 and MD5.

MD5 is the Message Digest algorithm developed by Ronald Rivest [10, 12]. The algorithm accesses 512-bit message blocks and finally produces a 128-bit hash value (message digest). This hash function is an improved version of MD4 but a more complex design [1]. In MD5 architecture a fourth round has been added, while each transformation step has a unique additive constant. Each step now adds in the result of the previous transformation step. This promotes a faster valance effect compared with MD4.

Furthermore the order of the processed message sub-blocks is changed in transformation rounds 2 and 3 [1]. In spite of these differences, both MD4 and MD5 produce a 128-bit message digest.

SHA-1 is the Secure Hash Algorithm designed by NIST [7, 11]. This hash function is widely used in the Digital Signature Algorithm [8]. The SHA-1 is based on design aspects and mathematical principles similar to the applied to MD4 and MD5. Especially, SHA-1 is almost the same with MD4 with the addition of an expand transformation, an extra round, and better avalanche effect [1].

SHA-1 produces a 160-bit message digest, longer than the generated 128-bit hash value by MD5. This hash function offers high security level and no cryptanalytic attacks have been applied successfully against SHA-1 yet. The 160-bit message digest of SHA-1 makes it more resistant to birthday and brute-force-attacks than the 128-bit hash value of MD5.

3. Proposed System Architecture

3.1. MD5 Hash Function

The proposed system architecture is illustrated in the following Figure 1. This architecture is used for both MD5 and SHA-1 implementation, with the appropriate modifications each time.

The Padding Data Unit pads the input data and converts them to 512-bit blocks (padded data). This operation is characterized of simplicity and it is well defined by the MD5 specifications (for more details see [10]). Every produced padded data block is stored in one of the four used RAM blocks. Each one of the used RAM blocks is equal to 16x32-bit (=512-bit). Four padded data blocks in total can be processed by the proposed system architecture at the same time.

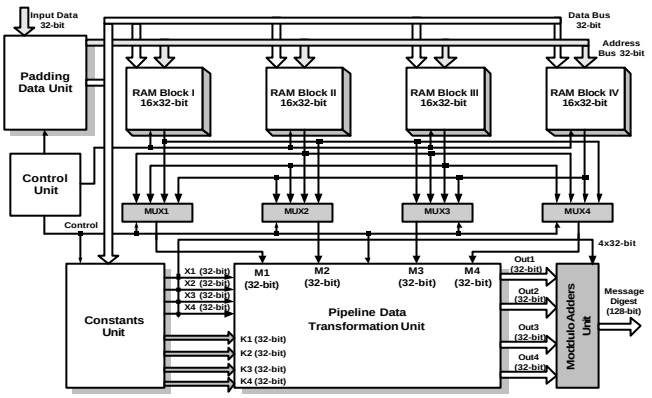

Figure 1. Proposed system architecture.

The necessary data transformation is performed in the Pipeline Data Transformation Unit. The architecture of this unit is shown in Figure 2.

This last unit basically consists of four different in architecture data transformation rounds. Of course, the pipeline applied design technique of this unit (Figure 2) needs four registers between the data transformation rounds. Every round operates on 4 inputs (AIn, BIn, CIn, DIn) plus the message input (Mi) and the constant input (Ki), all equal to 32-bit (Figure 3).

The four rounds are very similar but its one performs a different operation. Each operation is based on a nonlinear function on three of AIn, BIn, CIn, and DIn, inputs. Then, this result is added to the fourth input with the input data block (Mi) and the constant (Ki). That result is rotated to the right and the rotated data output is added with the input (B). There are four different nonlinear functions, one for each round, which are described by the following four equations:

Round 1:
F (X,Y,Z) = (X AND Y) OR ((NOT X) AND Z)

Round 2:
G (X,Y,Z) = (X AND Z) OR (Y AND (NOT Z))

Round 3:
H (X,Y,Z) = (X XOR Y XOR Z)

Round 4:
I (X,Y,Z) = Y XOR (X OR (NOT Z))

where X,Y,Z are equal to 32-bit.

Each round modifies the input data 16 times. These necessary transformations are performed with the use of loop rolling technique (feedback logic).

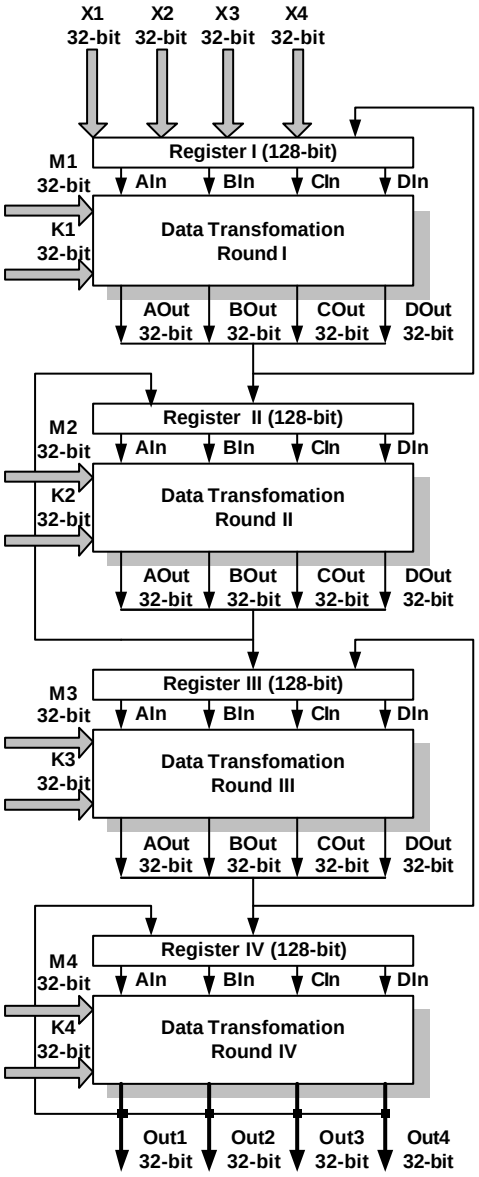

Figure 2. Pipeline data transformation unit.

As it has been mentioned before the data (input message) are processed 16 times in each transformation round and in this way 64 transformations are performed in total. The proposed architecture of Figure 2, processes four different padded data blocks at the same time. When the transformation of the N padded data block is completed in the round unit K, then it is forward to round unit K+1. After that, the N+1 padded data block is entered to the K round unit.

This operation is repeated every 16 clock cycles. Every padded data block needs 64 clock cycles in total to be completely transformed. Then the outputs data (Out1, Out2, Out3, Out4) are loaded from the Modulo Adders Unit (Figure 1). The modulo adders unit consists of 4 modulo adders. In this unit modulo additions 2^{32} are performed, between the input data (Out1, Out2, Out3, Out4) and the four constants (X1, X2, X3, X4). In this way the message digest (128-bit) is finally produced. With the pipeline applied technique a new 128-bit message digest is generated every (16+1) clock cycles. The specified constants for the MD5 operation are stored in the Constants unit. Four 32-bit constants (initial values) have been defined. These values (X1, X2, X3, X4) are called chaining variables. In addition, every one of the four transformation rounds demands 16x32-bit constants to support its operation, according to MD5 hash function specifications [10]. These values are loaded from the four data inputs (K1, K2, K3, K4), one for each round.

3.2. SHA-1 Hash Function

The proposed system architecture (Figure 1) can be used alternatively for the implementation of the other widely used hash function SHA-1. Only the Constants Unit and the Pipeline data transformation unit needs minor modification in order the proposed system architecture (Figure 1) to perform efficiently as SHA-1 hash function. The Data transformation round architecture of SHA-1 is shown in the following Figure 4.

The basic difference with the MD5 transformation round is that SHA-1 round operates on five 32-bit variables (inputs/outputs). It also based on a different nonlinear function. The specified nonlinear functions for each one of the SHA-1 transformation rounds are:

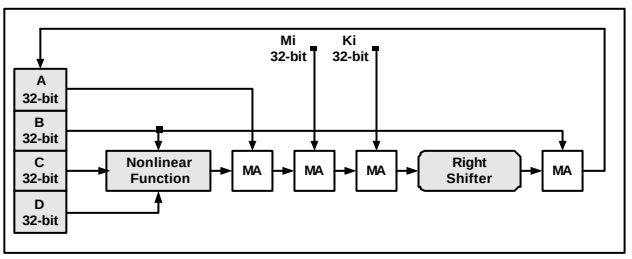

Figure 3. MD5 data transformation round.

Round 1:
F1(X, Y, Z) = (X AND Y) OR ((NOT X) AND Z)

Round 2:
F2(X, Y, Z) = (X XOR Y XOR Z)

Round 3:
F3(X, Y, Z) = (X AND Y) OR (X AND Z)
OR (Z AND Y)

Round 4:
F4(X, Y, Z) = (X XOR Y XOR Z)

where X,Y,Z are equal to 32-bit.

The data are transformed 20 times in each round (80 times in total) and finally a 160-bit (5x32-bit) message
digest is produced.

The proposed architecture of Figure 2 ensures that four 512-bit padded data blocks are processed at the same time and every (20+1) clock cycles a new message digest is generated.

The Constants Unit (Figure 1) in the case of SHA-1 implementation initializes the inputs of SHA-1 pipeline data transformation unit (Figure 2), with five 32-bit specified initial values. In addition four 32-bit constants are used (K1, K2, K3, K4), one for each round for the Pipeline Data Transformation Unit. Of course both initial values and used rounds constants have been specified by the SHA-1 standard. Especially, the initial values are refreshed each time that a message digest is produced (for more details see [11]).

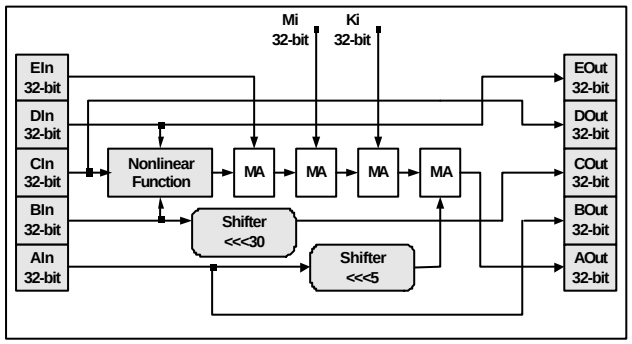

Figure 4. SHA-1 Data transformation round

4. Synthesis Results

The proposed system architecture, (Figure 1) has been captured by using VHDL. All the internal components of the design were synthesized placed and routed using XILINX FPGA device [14]. The system then was simulated again, for the verification of the correct functionality. The test scenarios that are applied to the proposed architectures, in order to verify systems' correct functionality, are provided by the MD5 and SHA-1 standards [10, 11]. In addition, during the test procedure a great number of test vectors were used to verify the right operation of the received FPGA device samples. These test vectors, were mostly selected in a random way, but there have been included some special values of the input data (for example "FFF…FFF", "000…000") to ensure maximum test coverage.

The synthesis results for both MD5 and SHA-1 implementations are illustrated in the next Table 1, in terms of covered area resources, and operating frequency.

Both implementations have almost the same operating frequency. Especially SHA-1 operates up to 72 MHz, while MD5 implementation has frequency equal to 70 MHz. The covered area resources are less for the MD5 integration by about 15-20 % compared with the SHA-1 implementation. Both hash functions designs, fit to the same used FPGA device (v100ecs14). MD5 implementation allocates 1226 FG, 713 CLB slices and 1591 Dffs. SHA-1 FPGA integration uses 1473 FGs, 878 CLB slices, and 1735 Dffs.

Comparisons of the proposed MD5 integration, with previous published implementations of the same hash function are presented in Table 2. The proposed MD5 implementation has very high throughput compared with the other conventional works of both hardware implementations [2, 4, 13] and software developments [3]. It has to be mentioned that work [13] is an estimation of a possible hardware implementation of the MD5 and not a real hardware integration of this certain hash function.

Fpga Device:			**Xilinx V100ecs144**	
Hash Functions	Md5		Sha-1	
Covered Area	Used / Available	Utilization	Used / Available	Utilization
Inputs/Outputs	68 / 94	72 %	68 / 94	72 %
Fun. Generators	1226 / 2400	51 %	1473 / 2400	61 %
Clb Slices	713 / 1200	59 %	878 / 1200	73 %
Dffs Or Latches	1591 / 2988	53 %	1735 / 2988	58 %
Operating Frequency	70 Mhz		72 Mhz	

Table 1. FPGA implementations synthesis results D Flip-Flops (DFFs), Configurable Logic Blocks (CLBs), Function Generators (FGs).

Nevertheless, in this work [13] no estimations about the covered area are presented. In spite of these omissions, the presented estimations results of [5] are very interesting for the readers and they are reported in order to have a fair and detailed comparison.

Furthermore, the proposed MD5 implementation is also compared with the other hardware implementations in the Area-Delay Product.

Especially, and only for the hardware implementations the Area-Delay product can be used as a comparison term. This product is calculated easily according to the equation:

A-D Product = Allocated Area X T_{delay}
where T_{delay} = 1 / Frequency.

The following Figure 5 shows the Area-Delay Product Comparison for the MD-5 implementations. From, the illustrated diagram of Figure 5 it is proven that the proposed MD5 implementation has better (less) Area-Delay product in the case of conventional works related to hardware implementations.

In the following Table 3, previous published implementations of the SHA-1 are presented and are compared with the proposed implementation of this hash function. The used pipeline architecture is proved a

better applied design technique for the SHA-1 implementation.

Table 2. MD5 implementations comparison.

Architecture	Covered Area (CLBS)	Frequency (MHz)	Throughput (Mbps)
Dominikus [4]	1004	43	146
Dobbertin [5]	Software	90	114
Touch [6]	-	300	256
Deepakumara [7]	880 4763	21 71.4	165 354
Proposed MD5	1313	70	2,1 Gbps

Table 3. SHA-1 implementations comparison.

Architecture	Covered Area (CLBS)	Frequency (MHz)	Throughput (Mbps)
Roe [8]	Software	- 133	4.23 41.51
Dominikus [4]	1004	43	146
Dobbertin [5]	Software	90	40
Kitsos [9]	2506	47	300
Proposed SHA-1	1578	72	1,7 Gbps

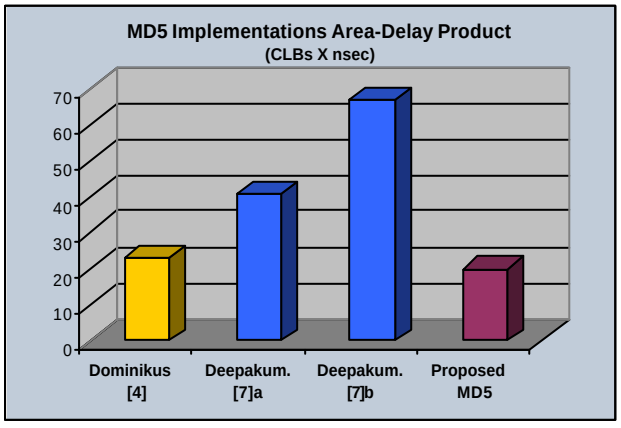

Figure 5. MD5 area-delay product comparison.

The proposed implementation has far high throughput compared with all the others software and hardware implementations. The work [9] is an assembly development in two different kinds of processors. The second processor of [8] operates at 133 MHz while for the first no information is given for the system clock. In the assembly implementation of [3] a 90 MHz processor is used. The hardware implementation of [6] uses a loop rolling technique and has the higher throughput of all the conventional architectures. Nevertheless, the pipeline proposed architecture for the SHA-1 implementation has 8 times better throughput compared with [6]. The work [4] has been designed as a typical processor and needs a great number of clock cycles in order to generate a 160-bit message digest block. Especially, 320 clock cycles are needed for every produced message digest in [4], while our proposed system architecture demands only 21 clock cycles.

The proposed SHA-1 implementation is compared with the other related hardware integrations in the term of Area-Delay product. This comparison is shown in the Figure 6. From the following illustrated diagram it is proven that the proposed SHA-1 implementation has better (less) Area-Delay product compared with the other published work [6] and almost the same with work [4]. It is obvious that comparison by using this implementation factor (Area-Delay product) can be done only in the cases of hardware implementations and not for software developments [3, 9].

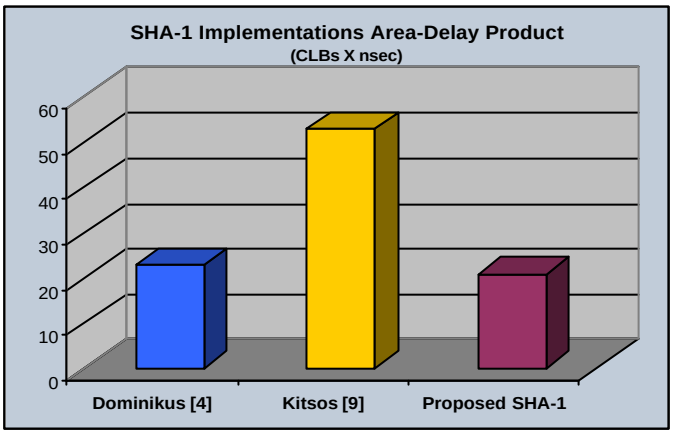

Figure 6. SHA-1 area-delay product comparison.

5. Conclusions

Security has become a very critical issue on the provision of electronic services. In addition to the supported security level, performance is major factor for both hardware and software implementations. The system throughput must not be the bottleneck of the implementation itself.

In this work, an ultra high speed architecture for the VLSI implementation of MD5 is presented. In addition, with minor modifications the proposed architecture can be used for the hardware integration of the SHA-1 hash function. Both hash functions have been implemented by using VHDL in FPGA devices. The synthesis results are illustrated and compared with other related works, published in the technical literature. From the performance comparison it is proven that the MD5 proposed implementation is better by a factor range from 700% to 1500%. In addition, the Area-Delay product of the proposed implementation is better in all of the cases. The SHA-1 proposed implementation is better at about 800% to 1700% compared with the other conventional works. The Area-Delay product comparison proves that the proposed SHA-1 implementation is superior to all the other related implementations. Both MD5 and

SHA-1 proposed implementations offer high-speed performance and support high security level at the same time. They can be used efficiently in all the hash functions applications such us digital signature, data integrity, message authentication and random number generators. Both of them can substitute successfully any existing implementations in the above referenced applications with superior performance. They can also be used successfully in communication protocols such as IPsec and WAP and security schemes in general.

References

[1] Bruce Schneier, *Applied Cryptography–Protocols, Algorithms and Source Code in C*, Second Edition, John Wiley and Sons, New York, 1996.

[2] Deepakumara J., Heys H. M., and Venkatesan R., "FPGA Implementation of MD5 Hash Algorithm," *in Proceedings of IEEE Canadian Conference on Electrical and Computer Engineering (CCECE'2001)*, Toronto, Ontario, May 2001.

[3] Dobbertin H., Bosselaers A., and Preneel B., "RIPEMD-160: A strengthened version of RIPEMD," *in Proceedings of Fast Software Encryption*, LNCS 1039, Springer-Verlag, pp. 71-82, 1996.

[4] Dominikus S., "A Hardware Implementation of MD4-Family Hash Algorithms," *proceedings of IEEE International Conference on Electronics Circuits and Systems (ICECS'02)*, Dubrovnik, Croatia, September 15-18, 2002.

[5] HMAC Standard, National Institute of Standards and Technology, *The Keyed-Hash Message Authentication Code*, http://csrc.nist.gov/ publications/fips/dfips- HMAC.pdf, 2003.

[6] Kitsos P., Sklavos N., and Koufopavlou O., "An Efficient Implementation of the Digital Signature Algorithm," *in Proceedings of IEEE International Conference on Electronics Circuits and Systems (ICECS'02)*, Croatia, vol. 3, pp. 1151-1154, September 15-18, 2002.

[7] Menezes A., Oorchot P., and Vanstone S., *Handbook of Applied Cryptography*, CRC Press, October 1997.

[8] National Institute of Standards and Technology (NIST), *Digital Signature Standard*, FIPS PUB 186-2, http://csrc.nist.gov/publications/fips/fips 186- 2.htm, 2003.

[9] Roe M., "Performance of Block Ciphers and Hash Functions-One Year Later," *in Proceedings of Second International Workshop for Fast Software Encryption '94*, Leuven, Belgium, December 14-16, 1994.

[10] Rivest R., *The MD5 Message-Digest Algorithm*, RFC 1321, MIT LCS and RSA Data Security Inc., April 1992.

[11] SHA-1 Standard, National Institute of Standards and Technology (NIST), *Secure Hash Standard*, FIPS PUB 180-1, www.itl.nist.gov/fipspubs/fip180-1, 2003.

[12] Stinson D. R., *Cryptography: Theory and Practice*, CRC Press LLC, 1995.

[13] Touch J. D., "Performance Analysis of MD5," *in Proceedings of ACM SIGCOMM'95*, Cambridge, Massachusetts, 1995.

[14] Xilinx, Virtex, *2.5 V Field Programmable Gate Arrays*, San Jose, California, USA, www.xilinx. com, 2003.

Nicolas Sklavos received a Diploma in electrical and computer engineering from the University of Patras, Greece, in 2000. He is currently pursuing the PhD degree at Department of Electrical and Computer Engineering, University of Patras, Greece. His research interests include security/cryptography, VLSI and low power design, hardware implementations for wireless communications security and reconfigurable computing architectures. He is an IEEE member and referee of international journals and conferences. He has published many technical papers in the areas of his research.

Epaminondas Alexopoulos is a student of the Department of Electrical and Computer Engineering at University of Patras, Greece. His research includes hardware implementations, mobile computing and security.

Odysseas Koufopavlou received the Diploma of electrical engineering in 1983 and the PhD degree in electrical engineering in 1990, both from University of Patras, Greece. From 1990 to 1994 he was at the IBM Thomas J. Watson Research Center, Yorktown Heights, NY, USA. He is currently an associate professor with the Department of Electrical and Computer Engineering, University of Patras. His research interests include VLSI, low power design, VLSI crypto systems, and high performance communication subsystems architecture and implementation. Dr. Koufopavlou has published more than 80 technical papers and received patents and inventions in these areas. He served as general chairman for the IEEE ICECS'1999. He is IEEE member.