Available online at www.sciencedirect.com
SCIENCE DIRECT®
Computers and Electrical Engineering 30 (2004) 593–604
Computers and Electrical Engineering
www.elsevier.com/locate/compeleceng

64-bit Block ciphers: hardware implementations and comparison analysis

P. Kitsos *, N. Sklavos, M.D. Galanis, O. Koufopavlou

VLSI Design Laboratory, Electrical and Computer Engineering Department, University of Patras, 26500 Rio, Patras, Greece

Received 20 November 2003; received in revised form 13 September 2004; accepted 3 November 2004

Abstract

A performance comparison for the 64-bit block cipher (Triple-DES, IDEA, CAST-128, MISTY1, and KHAZAD) FPGA hardware implementations is given in this paper. All these ciphers are under consideration from the ISO/IEC 18033-3 standard in order to provide an international encryption standard for the 64-bit block ciphers. Two basic architectures are implemented for each cipher. For the non-feedback cipher modes, the pipelined technique between the rounds is used, and the achieved throughput ranges from 3.0 Gbps for IDEA to 6.9 Gbps for Triple-DES. For feedback ciphers modes, the basic iterative architecture is considered and the achieved throughput ranges from 115 Mbps for Triple-DES to 462 Mbps for KHAZAD. The throughput, throughput per slice, latency, and area requirement results are provided for all the ciphers implementations. Our study is an effort to determine the most suitable algorithm for hardware implementation with FPGA devices.
© 2005 Elsevier Ltd. All rights reserved.

Keywords: Cryptography; ISO/IEC 18033-3; 64-bit Block cipher; Hardware implementation; FPGA

* Corresponding author. Tel.: +30 2610 997 323; fax: +30 2610 994 798.
E-mail address: pkitsos@ee.upatras.gr (P. Kitsos).

0045-7906/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.compeleceng.2004.11.001

1. Introduction

The growing requirements for high-speed, high level secure communications forces the system designers to propose the hardware implementation of cryptographic algorithms. However, cryptographic algorithms impose tremendous processing power demands that can be a bottleneck in high-speed networks. Modern applied cryptography in the wireless communications networks, demands high processing rate to fully utilize the available network bandwidth.

FPGA devices are a highly promising alternative for implementing private-key cryptographic algorithms. Compared to software-based implementations, FPGA implementations can achieve superior performance. The fine-granularity of FPGAs matches extremely well the operations required by private-key cryptographic algorithms (e.g., bit-permutations, bit-substitutions, look-up table reads, Boolean functions). As a result, such operations can be executed more efficiently in FPGAs than in a general-purpose computer. Furthermore, the inherent parallelism of the algorithms can be efficiently exploited in FPGAs as opposed to the serial fashion of computing in a processor environment. At the cryptographic-round level, multiple operations can be executed concurrently. On the other hand, at the block-cipher level, certain operation modes allow concurrent processing of multiple blocks of data.

In this paper, a hardware implementation performance comparison of the 64-bit block ciphers is given. There is a large variety of implementations of 64-bit block ciphers both in software running usually in general microprocessors and in hardware. Although we could have presented a detailed comparison analysis of existing implementations, we choose to have a general comparison methodology, which considers two generic architectures (among the VLSI implementations of block ciphers).

For block ciphers a third security level, *normal-legacy*, has been specified, which means a block size of 64 bits instead of a size of 128 bits (AES [1] do not specifies smaller block size than 128-bit). This was suggested by the industry, because the market will still need this block size for compatibility with present applications (e.g., payments with 8-byte personal identification numbers). It is interested in 64-bit block ciphers, which are more secure and efficient than the ones presently used. The block ciphers that are compared are the: Triple-DES (TDES) [2], IDEA [3], CAST-128 [4], MISTY1 [5], and KHAZAD [6]. All the above block ciphers are under consideration from the International Organization for Standardization (ISO/IEC) 18033-3 [7,8] standard in order to provide an international encryption standard for the 64-bit block ciphers. This standard consists of block cipher processing data blocks of 64 bits, using keys of 128-bit or 192-bit. Triple-DES, MISTY1, and CAST-128 are *Feistel* ciphers. IDEA is a semi-Feistel cipher. A Feistel cipher is a type of block cipher that has the effect of modifying only half of the block in each round. KHAZAD is a *Substitution-Permutation network* (*SP-network*) cipher. An SP-network is a type of block cipher that has the effect of modifying the entire data block in each round. Similar works, have been taken place by others research groups in the past, for AES candidates algorithms [9–11].

Two architectures for each block cipher are considered. The Basic Looping Architecture, where only one round is implemented, and the Full Loop Unrolling Architecture, where the rounds are fully unrolled with pipeline stages between the consecutive rounds. The performance metrics that are used for the ciphers comparison are: (1) throughput, defined as the number of bits encrypted (decrypted) in a unit of time, (2) throughput per slice, that measures the hardware resource cost associated with the implementation resulting throughput, (3) latency, defined as the time necessary to encrypt (decrypt) a single block of plaintext (ciphertext), and finally, (4) area requirements.

2. Architectures and VLSI implementations

The general block diagram of the hardware implementation of the symmetric-key block cipher is given in Fig. 1.

All block ciphers operate with 64-bit block size plaintext and accept 128-bit secret key. Also, Triple-DES has the ability to operate with 192-bit secret key. All implementations support on-the-fly round sub-keys generation.

Five modes of operation have been defined for the block ciphers [12]. Three of them are feedback modes, cipher block chaining mode (CBC), cipher feedback mode (CFB), and output feedback mode (OFB). The rest two are non-feedback modes, electronics code book mode (ECB), and Counter mode (CTR). In the feedback modes, it is not possible to start encryption for the next data block until encryption of the previous block is completed. As a result, all blocks must be encrypted sequentially, with no capability for parallel processing. On the other hand, in the non-feedback modes the encryption of each subsequent block of data can be performed independently from the processing of the other blocks. So, many blocks can be encrypted in parallel.

Two architectures of each algorithm are implemented. The basic looping architecture (BLA), and the full loop unrolling architecture (FLUA). Figs. 2 and 3 show the block diagrams of BLA and FLUA implementations, respectively.

In the first one, only one round of each cipher is implemented in order to decrement the required hardware resources. The output of the basic round unit is buffered and one additional register is used for the input plaintext storage. During initialisation the multiplexer chooses the

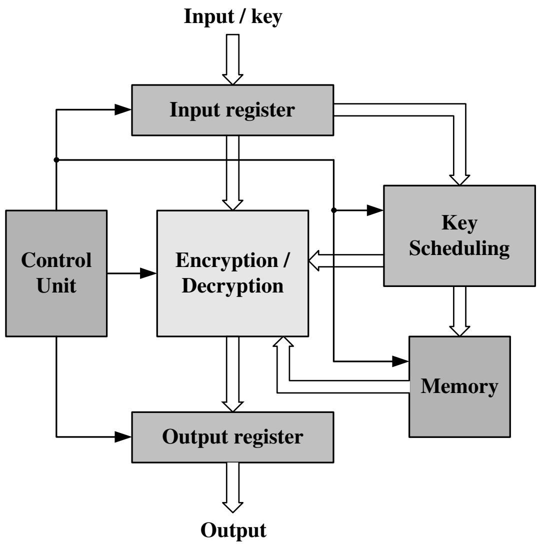


Fig. 1. General block diagram of the hardware implementation.

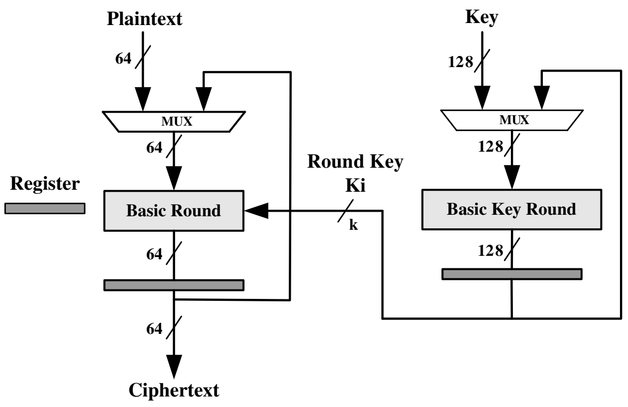


Fig. 2. Block diagram of the BLA implementation.

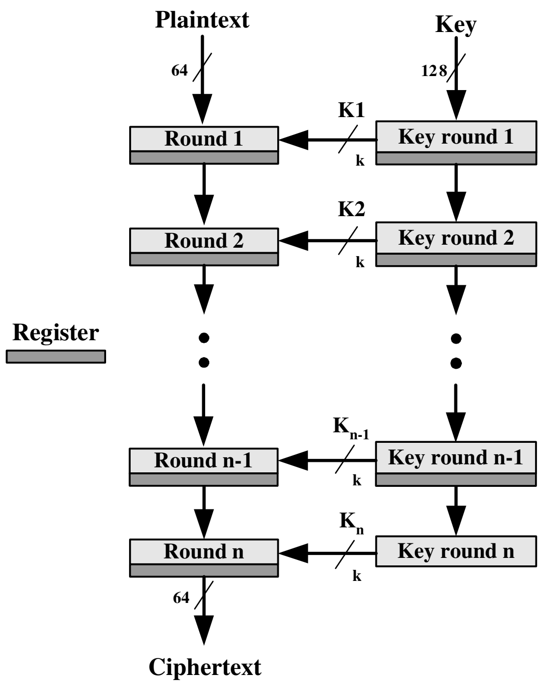


Fig. 3. Block diagram of the FLUA implementation.

plaintext and then chooses the output of the basic round unit. In this architecture the key scheduler consists of one basic round. The produced round sub-key is used both for the data encryption and as input to the next key round. The major disadvantage of this architecture is the requirement of more clock cycles in order to perform the complete cipher. This is because for an n-round cipher, n clock cycle is required to perform encryption. This architecture has low register-to-register delay and so operation frequency is increased.

In a loop unrolling architecture where all n rounds of the data encryption part and the key scheduling part are unrolled and implemented, the required hardware resources are increased. The key scheduling part is implemented with pipeline stages in order to balance the pipelining in each data encryption round. This approach minimizes the number of clock cycles required for encryption and increases the throughput. It also increases the worst-case register-to-register delay for the system and decreases the system frequency. For an n-round cipher, n rounds are unrolled, n pipeline stages are used and it is capable to process n data blocks simultaneously.

In order to make reliable and fair comparisons, between the hardware implementations of the ciphers, the same design considerations are followed. As a result, the hardware implementations of them are straightforward, since their block diagrams are actually their hardware architectures. In this work, we present a general comparison methodology for the hardware implementations of block ciphers, by considering the BLA and FLUA architectures. These two generic architectures can mimic a large amount of existing block ciphers implementations, as it will be made clear in the rest of the paper. Actually, most of the existing implementations only differ in the number of pipeline stages that they consist of.

Small exceptions comprise TDES, MISTY1, and CAST-128 implementations and these are explained below. The proposed TDES_FLUA implementation consists of 48 pipeline stages. Each DES execution starts with the initial permutation IP and ends with the inverse initial permutation IP^{-1}. These two permutations are inverse operations. When three DES are concatenated, the initial permutation of the previous DES follows the inverse initial permutation of the current DES. There is no reason to do either permutation, since the result is no permutation at all [13]. After that, any IP-IP^{-1} pairs of the algorithm can be avoided and only the first IP and the last IP^{-1} need to be executed. As a result, the initial permutations of the second and third DES, and the final permutations of the first and the second DES were not included in the proposed architecture. With this technique a significant performance gain is achieved.

In addition, in MISTY1 the odd rounds are different from the even rounds. The even rounds are the same with the odds, except of the FL/FL^{-1} unit. So, in order to significantly decrease MISTY1_BLA hardware area only one round is implemented (Basic Round). With the usage of multiplexers the appropriate operation (odd or even round) is selected in each time. The multiplexers select either the output or the input of the FL/FL^{-1} unit, when an odd round or an even round is executed, respectively. The FL/FL^{-1} unit has as input the output of the basic round. In addition, the round sub-keys are produced immediately when the cipher key is applied to the cipher. In order to apply the necessary delays for each round sub-key, pipeline registers are used. The same technique for the round sub-keys is applied also for CAST-128 cipher. This is a major design philosophy comparing with others ciphers that are designed with key rounds.

3. Implementation results and comparison analysis

Each one of the block ciphers was captured by using VHDL, with structural description logic. The VHDL codes were synthesized for XILINX (VIRTEX) FPGA devices [14], using the LeonardoSpectrum^{TM} tool [15]. VIRTEX 1600EBG560-6, was used for all the implementations of the specific block ciphers. This device was used in order to fit the implementation of the largest block cipher, which is the FLUA implementation of CAST-128. The necessary by the algorithms S-boxes were designed by using look up tables (LUT).

In Table 1, the implementation results of the algorithms are presented. The reconfiguration time of the FPGA device is not considered in the throughput results, since each block cipher implementation fits in the FPGA device. So, dynamic reconfiguration of the FPGA is not required. This is the reason why the reconfiguration time is not included in the calculation of throughput. Also, the FPGA configuration bitstream is loaded before a block cipher starts to execute, since static reconfiguration of the FPGA is considered. Thus, the place of keeping the configuration bitstream (local or external memory) is not an issue in this work.

Some previous designs for the 64-bit block ciphers have been implemented in FPGAs. Exception is the CAST-128 cipher, since there are no previous implementations for it.

For the TDES block cipher, in [16] three architectures are presented. In the first one, only the basic architecture is implemented and it achieves 45 Mbps at an operation frequency of 35.79 MHz. The second architecture is unrolled in two-rounds in order to decrease the execution clock cycles. This architecture matches a throughput equal to 68.8 Mbps at 29 MHz. Finally, in the third implementation the pipelined technique is used. Therefore, the design consists of 24 consecutive double-iteration rounds with pipeline registers in between. The throughput is 2912 Mbps at 45.55 MHz. In [17], a TDES implementation in Virtex devices [14] that achieves a throughput of 13.3 Gbps at 207 MHz, is presented. This throughput value is achieved using a 144-stage pipeline, whereas in the proposed TDES_FLUA, a 48-stage pipeline is considered. The application note of [18] presents a fully unrolled and pipelined TDES implementation that achieves a throughput of 6.464 Gbps at 101 MHz operation frequency; thus it is a slower implementation than ours.

In [19,20], one basic round of TDES is implemented. The throughput is 116 Mbps at 91 MHz and 83 Mbps at 69 MHz, respectively. The BLA implementation in [21] has a throughput of 917 Mbps, while consumes 604 Control Logic Blocks (CLB) slices. The operation frequency of this design is 165 MHz for a XCV300E-6 device and 258 MHz in a XC2V1000-5 device. However, this implementation encrypts/decrypts a data block every 58 clock cycles compared to our proposed one that encrypts/decrypts a data block every 48 clock cycles. This means that in the implementation of [21], the critical path of the basic round is reduced. On the other hand, in our TDES_BLA implementation, the critical path of the basic round is fixed. A BLA design for Xilinx

Table 1
Block ciphers implementation results

Architecture	Area (CLBs)	Frequency (MHz)	Throughput (Mbps)	Latency (μs)
TDES_BLA	431	86	115	0.56
TDES_FLUA	14240	108	6900	0.44
IDEA_BLA	1852	50	356	0.18
IDEA_FLUA	11700	47	3008	0.19
CAST-128_BLA	2600	55	220	0.29
CAST-128_FLUA	24200	53	3392	0.30
MISTY1_BLA	4820	30	213	0.26
MISTY1_FLUA	13080	26	3328	0.30
KHAZAD_BLA	2250	65	462	0.12
KHAZAD_FLUA	9277	70	4480	0.11

Intellectual Properties (IP) cores [22] gives a throughput of 668 Mbps on a Virtex-II FPGA, consuming 790 CLB slices. However, there are no other details available for this TDES implementation.

A BLA FPGA implementation of IDEA is proposed in [23], where one basic round achieves a throughput of 2.8 Mbps. A 56-stage pipeline IDEA implementation is presented in [24] that achieves throughput of 528 Mbps. Another FLUA implementation of IDEA [25] gives a throughput of 8.3 Gbps at 131.1 MHz, while using 6078 slices of a Xilinx Virtex XCV600 device. However, this implementation is deeply pipelined since it uses 158 pipeline stages, while our IDEA_FLUA design utilizes only a 9-stage pipeline. So, the throughput improvement of [25] is justified by the large amount of pipelining.

Another deeply pipelined IDEA implementation is presented in [26] that has a throughput of 6.78 Gbps at 105.9 MHz and occupies 9075 CLB slices. The work in [27] estimates that a bit-parallel (BLA) implementation of IDEA can achieve a throughput of 5.247 Gbps using 11602 CLB slices of a Xilinx Virtex XCV1000 FPGA, while the bit-serial fully pipelined one (FLUA) can give 2.4 Gbps using 11512 slices. Nevertheless, each round of BLA implementation of [27] has a latency value of 21 clock cycles opposed to our implementation that its latency equals to 1 cycle. This means that the critical path delay of BLA basic round in [27] is reduced. However, the basic round in our IDEA_BLA is implemented without pipeline stages. Also, each round of the FLUA implementation in [27] has a latency of 109 cycles, while our IDEA_FLUA one has a latency of 1 cycle. The overall latency of the bit-serial design of [27] is equal to 923 clock cycles. For the proposed IDEA_FLUA implementation, latency equals to 9 cycles, which is considerably smaller than that of the bit serial in [27].

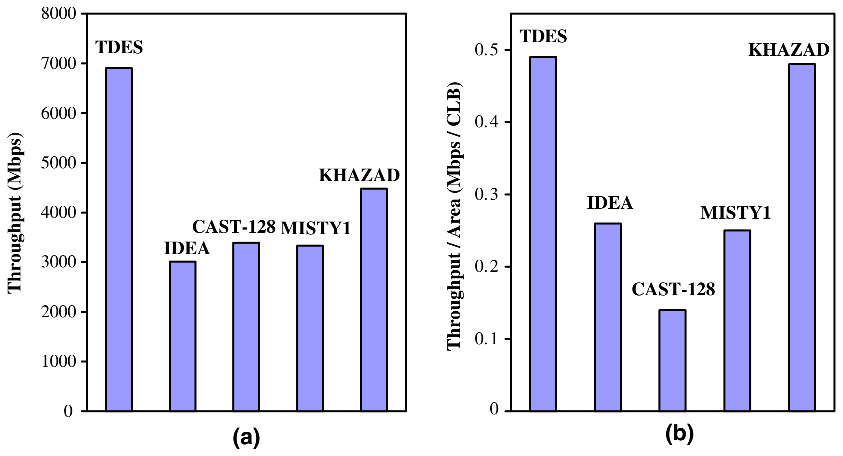
Finally, in [28] efficient implementations of MISTY1 and KHAZAD block ciphers are presented. These implementations support only the encryption mode of operation, compared to our proposed ones that support both encryption and decryption mode. For that MISTY1 block cipher a 208-stage pipelined implementation is presented, compared to the proposed one where 8 pipeline stages are used. A throughput of 8960 Mbps at 140 MHz is achieved. For the KHAZAD block cipher a 62-stage pipelined implementation—compared to the proposed one that uses only 8 pipeline stages—is presented with a throughput of 9472 Mbps and 148 MHz clock frequency.

In Fig. 4, throughput comparisons are presented between the proposed FPGA implementations of the 64-bit block ciphers. The comparisons are made in terms of throughput and throughput-to-area ratio requirements. The throughput results are obtained by the following equation:

$$\mathrm{Thr} = n \cdot \frac{\#\mathrm{bits}}{\#\mathrm{cycles}} \cdot \mathrm{Freq} \tag{1}$$

where #bits is the number of bits at the cipher's input, #cycles is the number of clock cycles that the block cipher needs in order to encrypt/decrypt the 64-bit input and Freq is the operation clock frequency. In the BLA implementation n is equal to 1, while in the FLUA implementation is equal to subsequent data blocks that each cipher can process in parallel. The throughput-to-area ratio reveals the hardware utilization efficiency of each implementation. It is only used in the non-feedback implementations.

For the FLUA implementations, the results show that TDES achieves the best throughput. This is explained by the fact that TDES can process in parallel 48 data blocks. In addition, it appears that its algorithmic philosophy matches better to the FPGA characteristics (due to the high



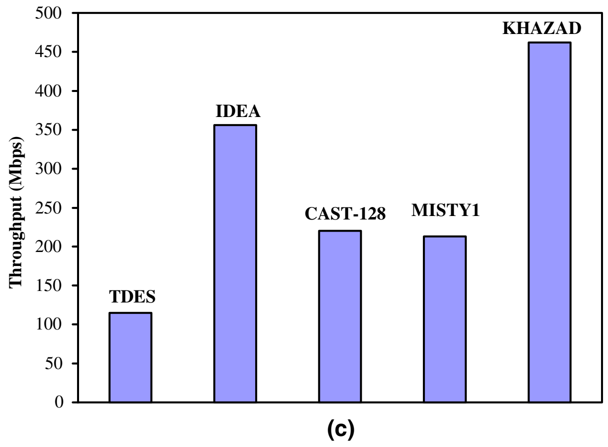


Fig. 4. Throughput comparisons of the FPGA implementations: (a) throughput comparisons between FLUA implementations, (b) throughput/area comparisons between FLUA implementations, (c) throughput comparisons between BLA architectures.

Throughput/Area value). While, KHAZAD implementation has slower clock period and fewer rounds, has similar behaviour to the TDES cipher implementation. This is due to the fact that KHAZAD_FLUA implementation is much smaller than the TDES_FLUA. CAST-128 cipher has the worst fit in the FPGA characteristics, simultaneously with low throughput. In addition, the BLA implementations KHAZAD and IDEA have better performance compared to the other cipher implementations. TDES needs 48 clock cycles per data block, so it has the lowest throughput.

Fig. 5 shows the latency comparisons between the FPGA implementations of the 64-bit block ciphers.

As it is shown in Fig. 5, the behaviour of the cipher is roughly the same in both architecture implementations (FLUA and BLA). TDES cipher needs more time to encrypt (decrypt) a single

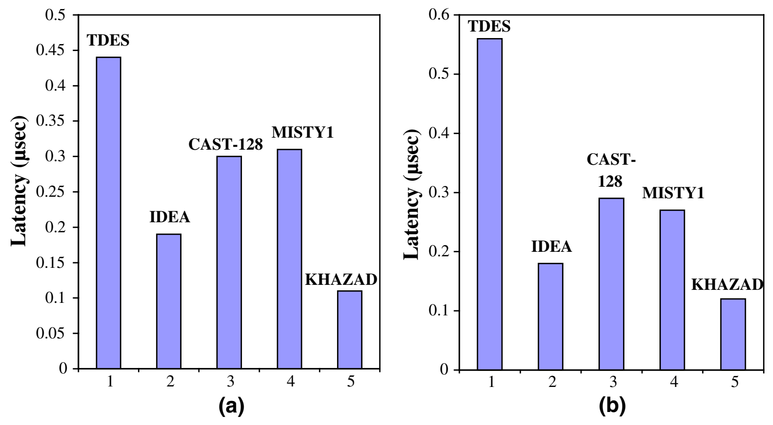


Fig. 5. Latency comparisons of the FPGA implementations: (a) latency comparisons between FLUA implementations, (b) latency comparisons between BLA implementations.

block of plaintext (ciphertext) comparing to the others, but only KHAZAD needs less time. This is due to the fact that TDES 48 has rounds compared to the 8 rounds of the KHAZAD.

The area results for both implementations using the VIRTEX device XCV1600EBG560-6, are summarized in Fig. 6.

For the FLUA implementations CAST-128 cipher requires the most FPGA hardware resources, while KHAZAD and IDEA have the most compact implementations. In addition, for the BLA implementations, TDES has the most compact implementation, and MISTY1 has the higher FPGA area resources utilization. Finally, IDEA and KHAZAD ciphers utilize 288 bytes and 56 bytes RAM blocks, respectively, for the temporary storage of the encryption round sub-keys. These sub-keys are used for the production of the decryption sub-keys.

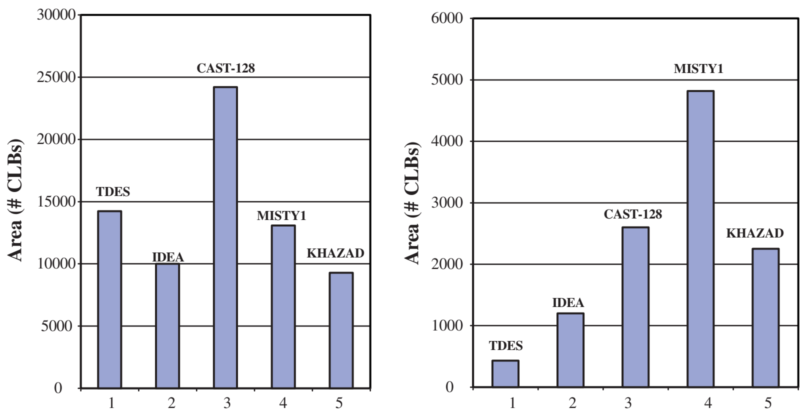


Fig. 6. Area comparisons for the FLUA and BLA implementations.

4. Conclusions

Time performance and area requirements results for 64-bit block ciphers (Triple-DES, IDEA, CAST-128, MISTY1, and KHAZAD) hardware implementations are presented in this paper. All these algorithms are under consideration for the ISO/IEC 18033-3 standard. Two architectures for each cipher are implemented. For the non-feedback implementations Triple-DES and KHAZAD achieve the best performance and meet better the FPGA characteristics. For the feedback implementations KHAZAD and IDEA appears to have better performance. Triple-DES has the highest latency. This is due to the required large number of rounds. KHAZAD cipher appears to have the better overall behaviour. It achieves admirable performance and requires reduced area resources for both (non-feedback and feedback) implementations. The high value of the throughput per slice proves that KHAZAD is the best choice for an FPGA implementation. Considering that the ISO/IEC 18033-3 consists of block ciphers processing data blocks of 64 bits, using keys of 128-bit or 192-bit, Triple-DES has an advantage in order to be selected as standard because is the only cipher that has the possibility to use 192-bit keys. On the other hand, for the ciphers using 128-bit key, KHAZAD seems to be (based on the above measurements) the best choice.

References

[1] Advanced Encryption Standard. Available fom: http://www.nist.gov/aes, 2000.
[2] Data Encryption Standard. Federal Information Processing Standard (FIPS) 46, National Bureau of Standards, 1977.
[3] Lai X, Massey JL. A proposal for a new Block Encryption Standard. Eurocrypt'90, Aarhus, Denmark, May 21–24 1990. p. 389–404.
[4] Adams C, Tavares S. The CAST-128 Encryption Algorithm. The Internet Engineering Task Force, Network Working Group, RFC 2144, Entrust Technologies, May 1997, on line available at: http://www.ietf.org/rfc/rfc2144.txt.
[5] Matsui M. Specification of MISTY1—a 64-bit block cipher. New European Scheme for Signatures, Integrity, and Encryption (NESSIE) Project. On line available at: https://www.cosic.esat.kuleuven.ac.be/nessie/workshop/submissions.
[6] Baretto P, Rijmen V. The KHAZAD legacy-level block cipher. Finalist of the NESSIE Project. On line available at: http://www.cosic.esat.kuleuven.ac.be/nessie/.
[7] Consulting G. On line available at: http://asn-1.com/EncryptionAlgorithms-3.asn.
[8] International Organization for Standardization. ISO/IEC WD 18033-3: information technology—security techniques—encryption algorithms—Part 3: block ciphers, 2002.
[9] Elbirt AJ, Yip W, Chetwynd B, Paar C. An FPGA implementation and performance evaluation of the AES block cipher candidate algorithm finalists. AES Candidate Conference 2000, New York, USA, 2000. p. 13–27.
[10] Dandalis A, Prasanna VK, Rolim JDP. A comparative study of performance of AES final candidates using FPGAs. In: Cryptographic hardware and embedded systems-CHES, August 2000. LNCS, vol. 1965. Worcester, MA, USA: Springer-Verlag; 2000. p. 125–40.
[11] Gaj K, Chodowiec P. Fast implementation and fair comparison of the final candidates for advanced encryption standard using field programmable gate arrays. In: RSA security conference—cryptographer's track, April 8–12, 2001. LNCS, vol. 2020. San Fransisco, CA, USA: Springer-Verlag; 2001. p. 84–99.
[12] Recommendation for Block Cipher Modes of Operation. Methods and Techniques. National Institute of Standards and Technology (NIST), Technology Administration, US Department of Commerce. Special Publication. On line available at: http://csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf.

[13] Feldmeier DC, Karn PR. UNIX password security—ten years later. CRYPTO'89, Santa Barbara, California, USA, 1989. p. 44–63.
[14] Xilinx Inc., San Jose, California, USA, Virtex, 2.5 V Field Programmable Gate Arrays, 2003. Available from: www.xilinx.com.
[15] Mentor Graphics Inc., LeonardoSpectrum^{TM} synthesis tool. Available from: http://www.mentor.com/leonardospectrum.
[16] Hämäläinen P, Hännikäinen M, Hämäläinen T, Saarinen J. Configurable hardware implementation of triple-DES encryption algorithm for wireless local area network. In: International Conference on Acoustics, Speech, and Signal Processing (ICASSP'2001), Salt Lake City, USA, May 7–11, 2001. p. 1221–4.
[17] Xilinx Inc., Pasham V, Trimberger S. High-speed DES and triple-DES encryptor–decryptor. On line available at: http://www.xilinx.com/bvdocs/appnotes/xapp270.pdf, August 2001.
[18] Celoxica Inc. Triple DES hardware design and implementation in one week with only one software engineer. On line available at: http://www.celoxica.com/techlib/files/CEL-W0307171K5J-43.pdf, August 2002.
[19] Chodowiec P, Gaj K, Bellows P, Schott B. Experimental testing of the gigabit IPSec-compliant implementations of rijndael and triple DES using SLAAC-1V FPGA Accelerator Board. In: Information security conference, Malaga, Spain, October 1–3, 2001. LNCS. Springer-Verlag; 2001. p. 220–34.
[20] Kwon O, Seike H, Kajisaki H, Kurokawa T. Implementation of AES and triple-DES cryptography using a PCI-based FPGA board. In: International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC'2002), Phuket, Thailand, July 16–19, 2002. p. 940–3.
[21] Rouvroy G, Standaert F-X, Quisquater J-J, Legat J-D. Design strategies and modified descriptions to optimize cipher FPGA implementation: fast and compact results for DES and triple-DES. In: 13th international conference on field-programmable logic and applications, FPL 2003, Lisbon, Portugal, September 2003. LNCS, vol. 2778. Springer-Verlag; 2003. p. 181–93.
[22] CAST, Inc. Triple DES encryption core. On line available at: http://www.cast-inc.com.
[23] Runje D, Kovac M. Universal strong encryption FPGA core implementation. Design, automation, and test in Europe, Paris, France, February 23–26, 1998. p. 923–4.
[24] Mencer O, Morf M, Flynn MJ. Hardware software tri-design of encryption for mobile communication units. In: International Conference on Acoustics, Speech, and Signal Processing (ICASSP '98), vol. 5, Washington, USA, May 1998. p. 3045–8.
[25] Gonzalez I, Lopez-Buedo S, Gomez FJ, Martinez J. Using partial reconfiguration in cryptographic applications: an implementation of the IDEA algorithm. In: 13th International conference on field-programmable logic and applications, FPL 2003, Lisbon, Portugal, September 2003. LNCS, vol. 2778. Springer-Verlag; 2003. p. 194–203.
[26] Hamalainen A, Tommiska M, Skytta J. 6.78 Gigabits per second implementation of the idea crytpographic algorithm. In: 12th International conference on field-programmable logic and applications, FPL 2002, Montpellier, France, September 2002. LNCS. Springer-Verlag; 2002. p. 760–9.
[27] Cheung OYH, Tsoi KH, Leong PHW, Leong MP. Tradeoffs in parallel and serial implementations of the international data encryption algorithm IDEA. In: 3rd International workshop on cryptographic hardware and embedded systems, CHES 2001, Paris, France, 2001. LNCS, vol. 2162. Springer-Verlag; 2001. p. 333–47.
[28] Standaert FX, Rouvroy G, Quisquater JJ, Legat JD. Efficient FPGA implementations of block ciphers KHAZAD and MISTY1. In: Third NESSIE Workshop, November 2002, Munich, Germany, 2002.

Paris Kitsos received the B.Sc. degree in Physics in 1999 and a Ph.D. in 2004 from the Department of Electrical and Computer Engineering, both at the University of Patras. His research interests include VLSI design, hardware implementations of cryptography algorithms and security protocols for wireless communication systems. Dr. Kitsos is a referee of International Journals and Conferences and is a member of the IEEE and the International Association for Cryptologic Research (IACR). He has published more than 35 technical papers and reports in the areas of his research.

Nicolas Sklavos received the Ph.D. Degree in Electrical and Computer Engineering, and the Diploma in Electrical and Computer Engineering, in 2004 and in 2000, respectively, both from the Electrical & Computer Engineering Dept., University of Patras, Greece. His research interests include Cryptography, Wireless Communications Security, VLSI Design, and Reconfigurable Computing Architectures. He holds an award for his Ph.D. thesis on "VLSI Designs of Wireless Communications Security Systems", from IFIP VLSI SOC 2003. He is a referee of International Journals and Conferences. Dr. Sklavos is a member of the IEEE, the Technical Chamber of Greece, and the Greek Electrical Engineering Society. He has authored or coauthored more than 60 scientific articles, book chapters, and reports, in the areas of his research.

Michalis D. Galanis received B.Sc. in Physics and M.Sc. in Electronics from the Physics Department of University of Patras in 2000 and 2002, respectively. Since October 2002, he has been working towards a Ph.D. degree in the domain of Reconfigurable Computing at the Electrical and Computer Engineering Department of University of Patras. He receives, since 2003, a scholarship from the Alexander S. Onassis Public Benefit Foundation. Until now, he has published 22 research works in international conferences and journals.

O. Koufopavlou received the Diploma of Electrical Engineering in 1983 and the Ph.D. 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 a Professor with the ECE Department, University of Patras. His research interests include VLSI design, VLSI crypto systems, and high performance communication subsystems. Dr. Koufopavlou has published more than 100 technical papers and received patents and inventions in these areas. He served as general chairman for the IEEE ICECS'1999.