ISSN: 1983 7402 ITA, 29 a 30 SET 2020

Design of DES Encryption Algorithm as Bundled-Data Asynchronous Pipeline using FPGA

Diego A. Silva, Duarte L. Oliveira, Gracieth C. Batista
Instituto Tecnológico de Aeronáutica – ITA – São José dos Campos – São Paulo – Brazil

***Abstract* — Currently, digital systems that are able to meet major security restrictions are increasingly being demanded, both in the military and in commercial areas. Data security can be achieved by cryptographic algorithms. An important encryption algorithm known as DES (Digital Encryption Standard) was implemented in Field Programmable Gate Array (FPGA) in different synchronous architectures. In this paper we propose an implementation of the DES algorithm in FPGA, in the asynchronous pipeline style. Comparing with the implementation in FPGA of two different project styles the proposal asynchronous obtained an average increase of 14.9% in throughput and an average reduction of 66.3% in latency time.**

***Keywords*—SCAs, pipeline, data-path, SoC, DPA**

I. INTRODUCTION

In recent decades, there is a strong demand for digital systems that ensure the confidentiality of information, whether in processing or data storage. As examples, we have the purchasing activities on Internet, banking, etc., which require transmission security and sensitive data storage. The design of a digital system, meeting these security restrictions, demands communication protocols and use encryption methods. These methods are based on the arithmetic, and focus on hiding data. Currently, there is also a concern on the inclusion of "traps" in digital SoC (System-on-Chip) systems design, especially for military purposes [1], for example, cryptographic algorithms are intensively applied in software-defined radio, a strategic area in the military sector [2]. We can also mention aerial drones, which are mobile cyber-physical systems, with applications in military operations, package delivery, reconnaissance, etc. There are applications, where aerial drones must be highly targeted, therefore insurances. For example, military applications, where attacks on these drones must be frequent, for that some important information can be extracted [3].

Despite the encryption algorithms, implemented in SoCs, seek to be robust to the attempting of breaching confidential data, there is a number of techniques that demonstrate, through physical properties, that is possible to reveal the secret processed data [4,5]. This class of techniques is known as Side Channel Attacks – SCA, which extracts sensitive information based on physical features, such as power consumption, electromagnetic radiation, processing time, etc., allowing discovering the information protected by encryption.

Diego A. Silva, dasilva@ita.br; Duarte L. Oliveira, Tel +55-12-3947-6813, duarte@ita.br; Gracieth C. Batista, gracieth@ita.br;

These attacks seek to establish a relationship between the analyzed physical features and the processed data.

A cryptographic system typically uses a "word", called secret cryptographic key, which affects its efficiency. In modern cryptographic systems, knowing the key is equivalent to be able to perform operations on the encrypted system. Different encryption algorithms have been proposed to raise the reliability of data security, such as the RSA algorithm (Rivest Shamir Adleman) [6], TEA (Tiny Encryption Algorithm) [7], AES (Advanced Encryption Standard) [8] and DES (Digital Encryption Standard) [9]. The DES algorithm became one of the most popular algorithms in the late twentieth century it was developed by IBM with help from the National Security Agency (NSA) in the 1970s. In 1977 it was adopted as an information processing standard in USA agencies [10,11]. The security of the DES algorithm lies in the size of the key and difficulty in decrypting without knowledge of the key. The operations of DES encrypt and decrypt is publicly owned. The DES algorithm is relatively slow if implemented by software, due to the size of the key and a permutation involving a 64-bit input block.

Different proposals have been made for the implementation of cryptographic systems, aiming at a greater reliability facing to attacks, by hardware. We can cite the implementations of DES algorithm in the synchronous style in FPGA (Field Programmable Gate Array) [12-20] or in VLSI (Very Large Scall Integration) [21,22]. In DSM (Deep-Sub-Micron) MOS technology, used today, the implementation of synchronous circuits causes difficulties related to the global clock signal, for example, clock skew, high electromagnetic emission, low modularity, high noise. Asynchronous style is a promising alternative for solving problems related to the global clock signal. In the asynchronous style, we have the implementation of Zhang et al. [23], which works in the QDI class (quasi delay insensitive) and the work of [24-26] which implements in the GALS style (Global Asynchronous and Locally Synchronous).

This paper proposes a high performance DES cryptographic processor, which is synthesized on asynchronous pipeline architecture and prototyped in FPGA. This proposed architecture consists of eight stages, operates on the two-phase handshake protocol and is bundled-data, so the data-path in each stage is synthesized in the conventional way, that is, single-rail [27]. Comparing with two design styles, which are synchronous pipeline and multi-point GALS of [25], the proposed asynchronous pipeline achieved an average reduction of 66.3% in latency time and an average increase in throughput of 14.9%.

II. Previous Works

In the end of the 1990s, some works concerning attacks by means of physical characteristics of devices running cryptographic algorithms were presented. In 1996, Kocher [28] reported about the weakness of some algorithms related to their timing characteristics, like differences in the encryption time of different keys and plaintexts that would be exploited for attacks in some processor architectures. He suggested some countermeasures, as random delay insertion in the operations and time uniformization of all needed operations, what would turn the timing characteristics of algorithms unfeasible to analysis. The attacks based on timing characteristic were named Timing Attacks [28]. Another work showed the possibility to analyze power consumption and electromagnetic emission from chips based on CMOS technology due the switching characteristics of these devices. This work focused in a class of attacks called Power Analysis, and brought special attention to the Differential Power Analysis (DPA), who is simple to be performed. The authors proposed some countermeasures against DPAs, like device shielding and noise insertion.

In the beginning of 2010 years, the DPA countermeasures were divided in strategies of Uniformization, Randomization and Masking. Soares et al. [24] developed a strategy against SCAs based on randomization. They implemented the DES Algorithm in GALS pipeline architecture, using a two-phase handshaking protocol as an interface protocol between the synchronous islands and a random clock frequency system that at each round feed the islands with random clock frequencies. The handshaking protocol was also used for the communication between islands and its own clock systems. The goal of this architecture was hide the leakage of information by randomization of execution time, provided by the random clocks, and overlapping of current measurements caused by the pipeline. This proposed architecture achieved robustness against SCAs attacks when compared to versions of the same algorithm implemented in full synchronous and asynchronous styles [24].

In 2017, a work showed an energy-based attack flow against the architecture proposed for Soares et al. [24]. This attack was based on current traces time-alignment and subsampling, and achieved success to find sub-keys of 2-stages pipelined GALS architectures. The authors also proved that this attack flow had relative low computational cost compared to other previous trials [29].

III. Cryptographic Algorithm: DES

Figure 1 shows the basic flow of DES Algorithm encryption process. At the beginning of the process, the bits of the plaintext are permuted, by the Initial Permutation *IP* and then divided in two symmetric parts. After this, 16 iterations of a Round Function will be performed, consisting in nonlinear transformations of one side followed by an XOR operation. Each round uses a sub-key of 48 bits generated from an original key of 64 bits in a process called Key Scheduling. In the end, the right and left sides of the word are concatenated and another permutation is applied. The Permutations are simply input bits mapping to predetermined positions, which are defined in the official documentation of the DES Algorithm, by the Federal Information Processing Standard (FIPS). In hardware level, permutations can be implemented by means of wiring. The same structure of Fig. 1 can be used to perform the decryption process, differing from the encryption by the use of the sub-keys in a reverse order of application [30].

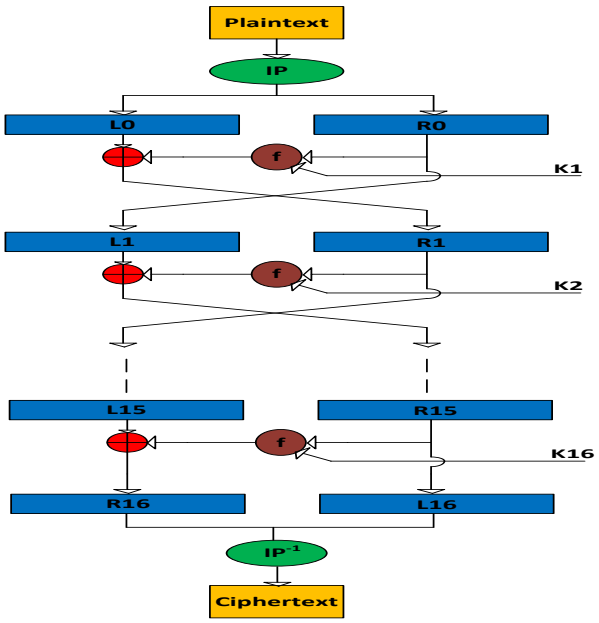

Fig. 1. Sequence of DES Algorithm operations.

The block diagram of the Round Function can be viewed in Fig. 2. This Round Function has four operations: Expansion Bits, XOR operation, SBOXes and Permutation *P*. Like in the case of initial and final permutations, the operations of Expansion Bits, SBOXes and Permutation are all defined in the official standard. In the Expansion Bits operation, a 32-bits word is expanded by a mapping process, generating a word of 48 bits. An operation of XOR between this expanded word and the round sub-key is performed in sequence. Then, the 48-bits word is applied to a set of 8 SBOXes. Each SBOX has 4 rows and 16 columns, where the addresses are defined by a 6-bits input, so the input word is divided in 8 sets of addresses. The output of a SBOX is a 4-bits word defined by the address specified by the input. In this work, the SBOXes will be implemented by Boolean functions in a XOR-SOP form that presented better performance compared to implementations based on Look-Up Table [31].

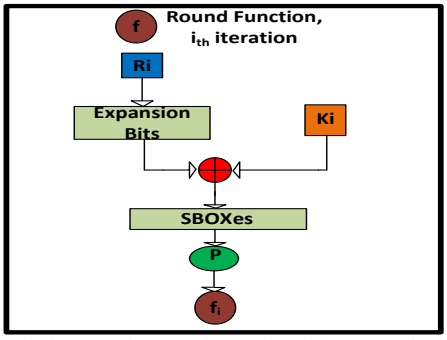

Fig. 2. "Block diagram of a Round Function of the DES Algorithm."

The Key Scheduling is the process where a set of 16 sub-keys for each round are created from an original key, as shown in Fig. 3. The original key of 64 bits is permuted in the beginning of the process, as labeled in *PC1*, and then divided in two sub-sets with 28 bits each one, where 8 parity bits are

discarded. At each round, the sub-sets are exposed to left-shifts, from 1 or 2 positions according to the round number. After shifting, an operation of permutation is used to deliver the sub-key to the round function, as defined by *PC2*. The Permuted Choices PC1 and PC2 are also defined by standard and uses only wiring operations. In a pipeline structure with hardware replication, the left shift operations can be implemented also by means of wirings, and it is the case of this work.

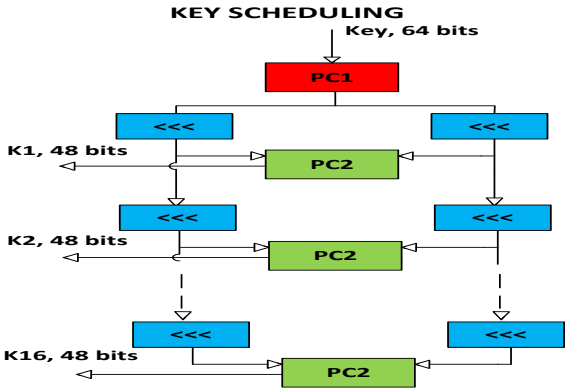

Fig. 3. Sequence of operation DES Algorithm Key Scheduling.

IV. Proposed Asynchronous Pipeline

The proposed bundled-data pipeline architecture is shown in Fig. 4, where the bundled-data style is composed of N + 2 lines, with N lines related to the data and two lines related to the request and acknowledge signals, which are used to carry out the communication. The proposal is made up of flip-flop D-based registers. At each stage there is a data-path that is responsible for processing the data. For each register there is an XNOR port and a control that is an asynchronous finite state machine (AFSM). The XNOR port allows the registers to be activated at both edges of the signal. Between the stages there is a delay element that is defined by the critical path of the data-path, i.e. the propagation time of the data-path. The AFSMs are responsible for the communication between the stages, through the two-phase handshaking protocol, that is, using input and output request signals [*Ri, Ro*] and input and output acknowledgment signals [*Ai, Ao*]. The AFSMs and delay elements ensure synchronization of pipeline operations.

Figure 5 shows the specification of the proposed control that was described in STG (Signal Transition Graph). The STG specification is an interpreted Petri net and was proposed by Chu [32]. The STG of the proposed control the input signals are [*Ri, Ao*] and the output signal is [*Ro*]. The control synthesis involves two steps. In the first step, the state graph (SG) is generated and the property CSC (Complete State Coding) is checked if it is satisfied, if not, state signals must be inserted [33]. An SG satisfies the CSC property if every pair of different states which are assigned the same binary code enables exactly the same set of non-input signals. Figure 6a shows the state graph of the control, which satisfies the CSC property, so it is in conditions to implementation. The second step is to obtain the output equation *Ro*, which must be hazard free.

Figures 6b shows the extraction of the Ro signal through the Karnaugh map. The equation of the signal Ro can be implemented with a C element [33] and an inverter gate, as shown in Fig. 6c, d. Figure 7 shows the proposed asynchronous pipeline architecture.

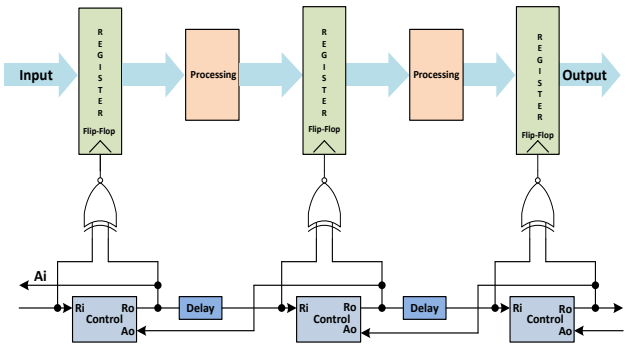

Fig. 4. Proposed linear pipeline architecture.

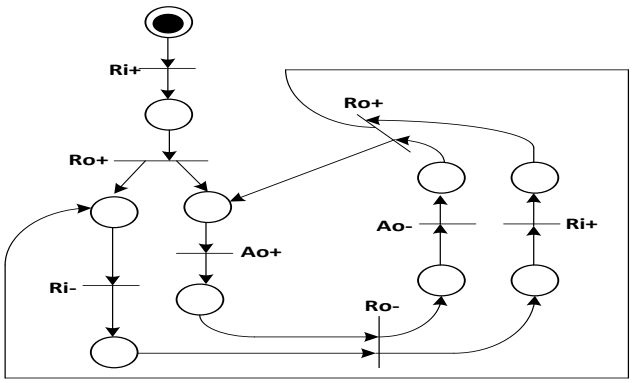

Fig. 5. STG specification of control.

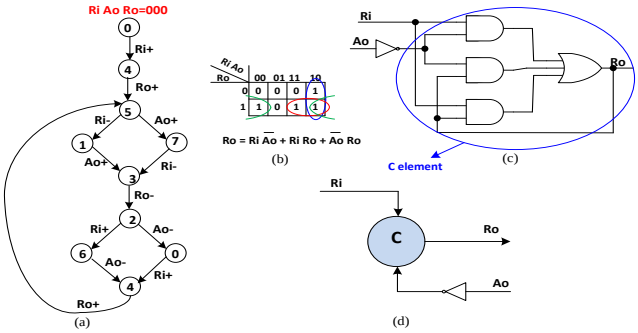

Fig. 6. Synthesis: a) state graph; b) map of Karnaugh for Ro signal and covered; c) logic circuit; d) logic circuit based in C element.

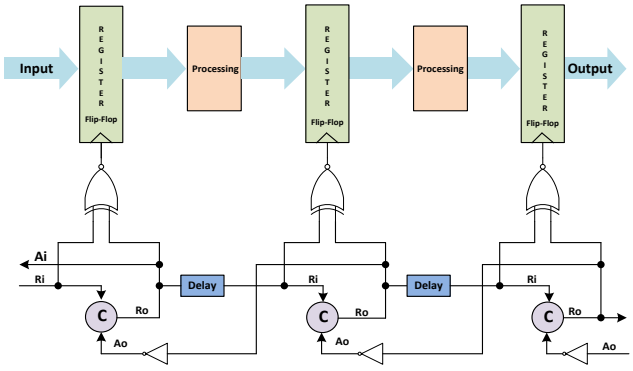

Fig. 7. Proposed linear pipeline architecture.

V. DES Encryption: Asynchronous Hardware Design

The DES_Encryption algorithm was decomposed into eight stages as shown in Fig. 8. The novelty is the Keyblock proposal as shown in Fig. 9. In the case of encryption, the displacements are for left and in the decryption are done to the right, which ensures that the keys are applied in the opposite direction as shown [13]. The keys of the previous

round enter the KeyBlock, suffer displacements, which are basically repositioning the wires, are multiplexed according to the selection signal MODE (0 = Enc, 1 = Dec) and pass through a permutation box called PC2, which also is implemented by means of wire repositioning. From this permutation box comes the key to be applied in the function of the round. Figure 10 shows the eight-stage asynchronous pipeline of the DES algorithm, which follows the proposed decomposition shown in Fig. 8.

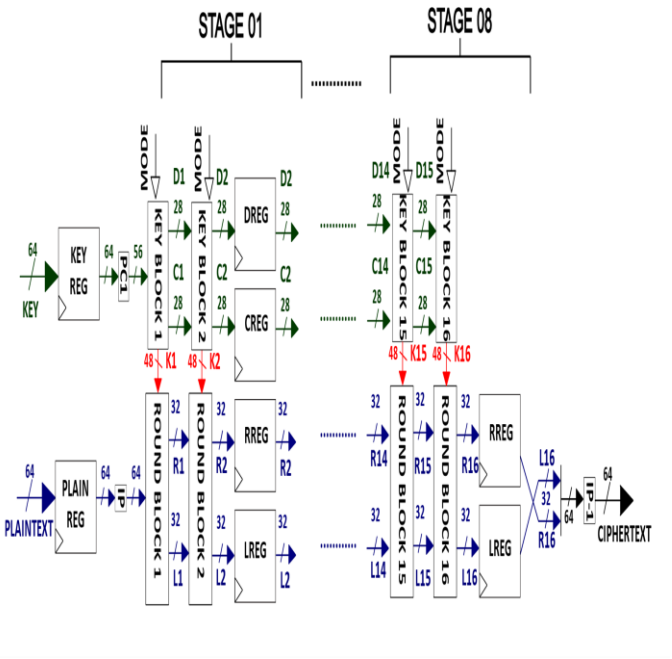

Fig. 8. Decomposition proposal of DES algorithm.

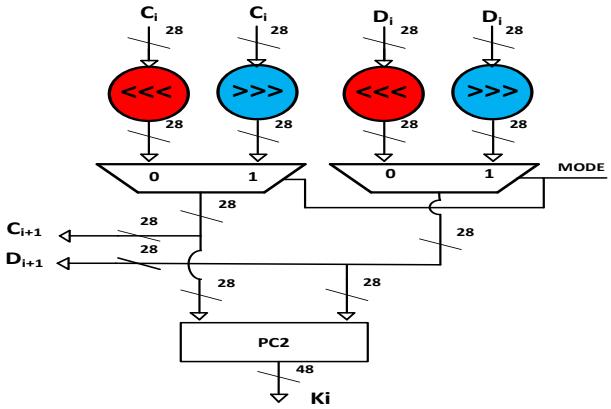

Fig. 9. Proposed scheme for the Keyblock.

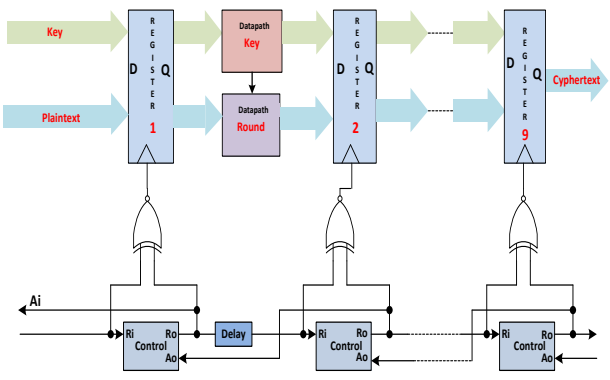

Fig. 10. Proposed bundled-data asynchronous pipeline of eight-stage for DES encryption algorithm.

VI. Simulations & Results

To show the feasibility of the proposed project, the DES encryption algorithm was synthesized in three styles, in this case: *a)* synchronous of four stages; *b)* multi-point GALS of [25] that consists of four modules; *c)* proposed asynchronous pipeline of eight stages. The three projects were described in structural VHDL, compiled and synthesized in post-layout in Intel Altera® tool, Quartus II software, version 9.0, Cyclone III family, in EP3C16F484C6 [34] device.

A. Simulations

Figure 11 shows the simulation of the asynchronous pipeline of encryption DES algorithm (DES_E) with initial data. Figure 10 shows the simulation of DES_E algorithm after eight clock cycles. The waveforms are exactly as expected to the DES_E.

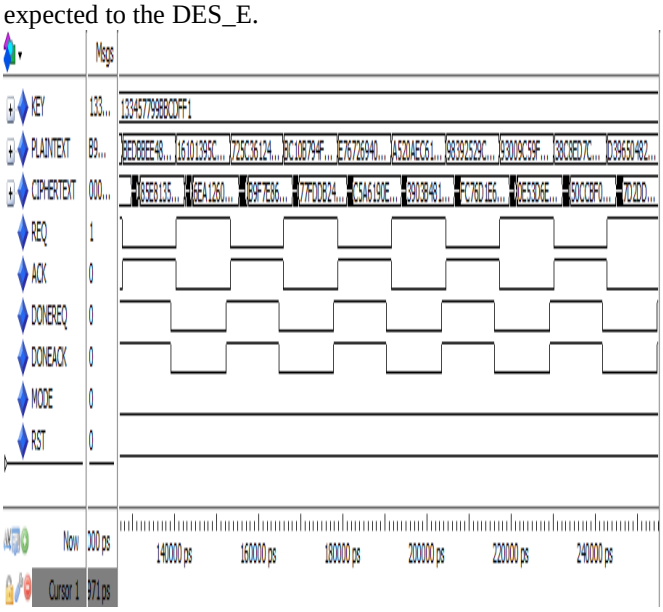

Fig. 11. Simulation: asynchronous pipeline of DES algorithm.

B. Results

Table I presents the results of the implementation of the DES algorithm in the versions of synchronous pipeline, multi-point GALS of [25] and proposed asynchronous pipeline. Comparing the three styles, the proposed pipeline architecture had an average penalty of 166.9% and 7.0%, respectively, in dynamic power and area (LUTs - Look-Up-Table + FFs) when compared to the other two styles. The proposed architecture had an average reduction of 66.3% in latency time and an average increase in throughput (MOPS - 10^6 operations per second) of 14.9% when compared with two other styles.

TABLE I. RESULTS: DESIGN STYLES OF DES_E ALGORITHM

Styles of Design		Time of Latency	Throughput MOPS	Dynamic Power	Macrocells	
					Number LUTS	Number Flip-Flops
DES	Synchronous Pipeline f_{MAX}=125 MHz	268ns	100.0	137.97mw	**4604**	1866
	Multi-point GALS of [25]	189ns	83.3	**46.41mw**	4926	2095
	Asynchronous Pipeline Proposed	**75.8ns**	**105.3**	246.06mw	6192	**1024**

VII. CONCLUSION

This work proposed a Data Encryption Standard FPGA-based implementation based on the device ALTERA Cyclone III family, in EP3C16F484C6, using a 8-stages asynchronous pipeline design style and sub-key multiplexing system for the encryption/decryption modes. At each clock cycle a new key and a new plaintext can be processed, and at each new cycle of clock after the pipeline fulfillment a new valid ciphertext can be obtained in the output. Furthermore, the system can be switched between the operation's mode of encryption and decryption by means of a single signal, which is used to select between two sets of sub-keys to be applied at each round, one for encryption and other for decryption. The implementation uses 6192 LUTs and 1024 D flip-flops, with a latency time of 75.8ns and a throughput of 105.3 MOPS. This encryption / decryption rate turns this architecture suitable to applications like smart cards and satellite communication, while the flexible change of key can be used in a key-search machine for the DES Algorithm. This encryption / decryption rate turns this architecture suitable to applications like smart cards, satellite communication and aerial drones, while the flexible change of key can be used in a key-search machine for the DES Algorithm.

REFERENCES

[1] S. Adee, "The Hunter for the Kill Switch". IEEE Spectrum, vol. 45-5, pp. 34-39, Jan 2008.

[2] E. Lussari, et al., "Software-Defined Radio design based on GALS architecture for FPGAs," IEEE 29th Symposium on Integrated Circuits and Systems Design (SBCCI). Pp.1-6, 2016.

[3] M. O. Ozmen, A. a. Yavuz, "Dronecrypt - An Efficient Cryptographic Framework for Small Aerial Drones," MILCOM 2018 - IEEE Military Communications Conference (MILCOM), pp.1-6, 2018.

[4] P. Kocher, "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and others Systems". In: 16th International Cryptology Conference on Advances in Cryptology (CRYPTO'96), pp. 104-113, Aug 1996.

[5] P. Kocher, et al., "Differential Power Analysis". In: 19th International Cryptology Conference on Advances in Cryptology (CRYPTO'99), pp. 388-397, Aug 1999.

[6] R. L. Rivest , A. Shamir , L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems" Communications of the ACM, Vol.:21, No. : 2, pp 120-126, Feb. 1978.

[7] D. J. Wheeler, R. Needham, TEA, a Tiny Encryption Algorithm, in the proceedings of FSE 1994, Lecture Notes in Computer Science, vol 1008, pp 363-366, Leuven, Belgium, December 1994, Springer-Verlag.

[8] Advanced Encryption Standard (AES). Federal Information Processing Standards Publication 197, National Institute of Standards and Technology, 2001. Available from http://csrc.nist.gov/publications/fips/ fips197/fips-197.pdf.

[9] Digital Encryption Standard (DES). Federal Information Processing Standards Publication 46-2, National Institute of Standards and Technology, December 1993. Available from http://www.itl.nist.gov/ fipspubs/fip46-2.htm.

[10] National Bureau of Standard (U.S.), "Data encryption Standard (DES)", Federal Information Processing Standards Publication 46, National Technical Information Service, Springfield, VA, April 1977.

[11] National Bureau of Standard (U.S.),"DES modes of Operation", Federal Information Processing Standards Publication 81, National Technical Information Service, Springfield, VA, December 1980.

[12] D. R. Stinson, "Cryptography. Theory and Practice". 2nd Edition, Chapman & Hall/CRC, Boca Raton, Florida, 2002.

[13] B. Schneier, "Applied Cryptography Second Edition: protocols, algorithms, and source code in C", Wiley & Sons, New York,USA, 2nd edition, 1996.

[14] K. Dichou, V. Tourtchine, F. Rahmoune, "Finding the best FPGA implementation of the DES algorithm to secure smart cards", Electrical Engineering (ICEE) 2015 4th International Conference on, pp. 1-4, 2015.

[15] K. Wong, M. Wark, M & E. Dawson. "Single-chip FPGA implementation of the Data Encryption Standard (DES) algorithm". 2. 827 - 832 vol.2. 10.1109/GLOCOM.1998.776849, 1998.

[16] J.P. Kaps, C. Paar. "Fast DES Implementations for FPGAs and Its Application to a Universal Key-Search Machine". In: Tavares S., Meijer H. (eds) Selected Areas in Cryptography. SAC 1998. Lecture Notes in Computer Science, vol 1556. Springer, Berlin, Heidelberg, 1999.

[17] C. Patterson, "A Dynamic Implementation of the Serpent Block Cipher", Proc. Workshop Cryptographic Hardware and Embedded SystemsCHES 2000, pp. 142-155, Aug. 2000.

[18] M. McLoone, J.V. McCanny, "A high performance FPGA implementation of DES", Signal Processing Systems 2000. SiPS 2000. 2000 IEEE Workshop on, pp. 374-383, 2000.

[19] K. M. A. Abd, H. F. A. Hamed, E. A. M. Hasaneen, "FPGA Implementation of the Pipelined Data Encryption Standard (DES) Based on Variable Time Data Permutation", The Online Journal on Electronics and Electrical Engineering, vol. 2, no. 3, pp. 298-302, 2011.

[20] S. Oukili and S. Bri, "FPGA implementation of Data Encryption Standard using time variable permutations," 2015 27th International Conference on Microelectronics (ICM), Casablanca, 2015.

[21] D. Liang, C. Hongyi, "An efficient and scalable VLSI implementation of DES," ASICON 2001, 4th International Conference on ASIC Proceedings (Cat. No.01TH8549), pp.341-343,2001.

[22] S. O'Melia, A. J. Elbirt, 'Enhancing the Performance of Symmetric-Key Cryptography via Instruction Set Extensions," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol.18, issue:11, pp.1505-1518,2010.

[23] Q. Zhang, et al. "Optimization design of a low power asynchronous DES for security applications based on Balsa and synchronous tools," International Conference on Electronics, Communications and Computers (CONIELECOMP), pp.124-129, 2015.

[24] R. Soares, N. Calazans, F. Moraes, P. Maurine, L. Torres (2011). "A Robust Architectural Approach for Cryptographic Algorithms Using GALS Pipelines," IEEE Design & Test of Computers. 28. 62-71. 10.1109/MDT.2011.69, 2011.

[25] T. Curtinha, et al. "FPGA Implementation of Low-Latency Robust Asynchronous Interfaces for GALS Systems," IEEE XXV International Conference on Electronics, Electrical Engineering and Computing (INTERCON), pp.1-4, 2018.

[26] D. A. Silva, O. Verducci, D. L. Oliveira, "Implementation of DES Algorithm in New Non-Synchronous Architecture Aiming DPA Robustness," IFIP/IEEE 27th International Conference on Very Large Scale Integration (VLSI-SoC), pp.1-2,2019.

[27] H. Wu, et al. 'A method to transform synchronous pipeline circuits to bundled-data asynchronous circuits using commercial EDA tools," IEEE International Conference on Electron Devices and Solid-State Circuits (EDSSC), pp.1-2, 2019.

[28] P. C. Kocher, "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems". In: Koblitz N. (eds) Advances in Cryptology — CRYPTO '96. CRYPTO 1996. Lecture Notes in Computer Science, vol. 1109. Springer, Berlin, Heidelberg, 1996.

[29] R. Lellis, R. Soares, A. A. Souza. "An energy-based attack flow for temporal misalignment countermeasures on cryptosystems." 2017 IEEE International Symposium on Circuits and Systems (ISCAS) (2017): 1-4.

[30] D. R. Stinson, "Cryptography. Theory and Practice". 2nd Edition, Chapman & Hall/CRC, Boca Raton, Florida, 2002.

[31] K. Wong, M. Wark, M & E. Dawson. "Single-chip FPGA implementation of the Data Encryption Standard (DES) algorithm". 2. 827 - 832 vol.2. 10.1109/GLOCOM.1998.776849, 1998.

[32] T. –A. Chu, "Synthesis of Self-Timed VLSI Circuits from Graph-Theory Specifications," PhD. Thesis, June 1987, Dep. Of EECS, MIT.

[33] P. Beerel, R. Ozdag and M. Ferretti, "A Designer's Guide to Asynchronous VLSI". Cambridge University Press, p. 337, 2010.

[34] Altera-corporation,2020-www.altera.com.