An Asynchronous Implementation of Cryptographic Algorithm using High-Level Automatic Synthesis

Kledermon Garcia[1,2], Duarte L. Oliveira[1], Lester A. Faria[1], Higor A. Delsolto[1], Leonardo Romano[3]
[1]Electronic Engineer Division – Technological Institute of Aeronautics – ITA – IEEA
[2]Aeronautical Systems Division – Institute of Aeronautics and Space – IAE - DCTA
[3]Electrical Engineer Department – University Center of FEI

***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, which are subject to attacks, often using the clock signal to reveal the secret data. To deal with this major problem, the asynchronous paradigm presents interesting features, due to the lack of the clock signal, being an option for the project of digital systems. In this paper, we propose a bundled-data architecture to implement an asynchronous cryptosystem. The cryptographic algorithm was chosen based on its simplicity and is called TEA (*Tiny Encryption Algorithm*). For its implementation, it was considered FPGAs (Field Programmable Gate Array) devices as target platforms. Compared to synchronous designs, the asynchronous ones, besides being more robust, presented a reduction in the latency time of up to 15% and in power dissipation of up to 11.9%.**

***Keywords*—logic asynchronous, XBM specification, data-path single-rail**

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 the use of 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].

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 [2,3]. 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. These attacks seek to establish a relationship between the analyzed physical features and the processed data.

Kledermon Garcia, kledermonkg@iae.cta.br; Duarte L. Oliveira, duarte@ita.br, Tel. +55-12-3947-6813; Lester A. Faria, lester@ita.br; Higor A. Delsoto, higordel@hotmail.com; Leonardo Romano, leoroma@uol.com.br.

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) [4], TEA (Tiny Encryption Algorithm) [5], AES (Advanced Encryption Standard) [6] and DES (Digital Encryption Standard) [7]. SoC systems focused on security are often embedded, thus following the constraints of the embedded design, such as reduced power dissipation, high level of integration, critical performance, etc. The SoC system design is based on the reuse of technical pre-designed and pre-validated components, called IP (Intellectual Property Core) cores [8], in order to reduce design time and cost. Different SoCs can be found in areas such as remote sensing [9], processing of secure data [10], etc.

SoC systems are traditionally designed in synchronous paradigm, i.e. they use a global clock to synchronize their operations. They are quite popular due to their simplicity of design and availability of commercial CAD tools for automatic synthesis. In DSM-MOS (Deep-sub-Micron MOS) technology, a clock signal requires major attention due to its noise generation, electromagnetic interference and power consumption [11]. Besides these factors, the distribution of the clock signal along the chip is a task with increasing complexity due to clock skew problems, which decrease the system performance. The overhead caused by the clock signal can reach up to 130% in a VLSI (Very Large Scale Integration) implementation [12] and is even worse when FPGAs (Field Programmable Gate Array) are employed. Beside all these problems, the analysis techniques of encrypted systems, seeking to reveal the data, use the clock signal as the main analysis parameter.

The asynchronous design shows to be an alternative and an interesting paradigm, once it eliminates some problems of the synchronous approach. Asynchronous circuits operate "by events" and the synchronization is achieved by local handshake signals rather than by a global clock. Therefore, the concerns associated with global clock are eliminated. However, asynchronous circuits are difficult to design. Besides the lack of CAD tools for an automatic synthesis, they should also be free of hazard and of critical race [13]. Different styles have been proposed for behavioral synthesis of asynchronous systems [14-19]. These styles can be categorized into two classes: a) direct translation syntax [16]; b) optimized synthesis [18,19].

Methods for direct translation syntax, despite of the recent efforts of using re-synthesis technique for better optimization, is already limited [17]. On the other hand, the optimized synthesis follows the traditional steps of behavioral synthesis (synchronous paradigm), such as scheduling, registers and functional units assignment, etc. [20].

Different proposals have been made for the implementation of cryptographic systems, aiming at a greater reliability facing to attacks. We can cite the implementations in the GALS (Globally-Asynhronous Locally-Synchronous) paradigm, proposed by Spadavecchia [21] and by Soares [22], aiming at DES and AES algorithms, which show a great performance. The GALS architecture allows operating with multiple clocks, which hampers the SCA. In this paper we propose an asynchronous architecture, based on a bundled-data implementation, in the decomposition style. It was obtained by optimized automatic behavioral synthesis, previously proposed by Garcia [19] and that is focused on FPGA devices, harming SCA due to its asynchronous nature. Comparing it with synchronous designs, there was a reduction in the latency time and in power dissipation of up to 15% and 11.9%, respectively.

II. Automatic Synthesis of Asynchronous Systems: Overview

The optimized behavioral synthesis can be directed to the asynchronous pipeline style or to the "decomposition" style, also known as division or sharing resources. It is very familiar to designers, allowing high optimization and addressing “intensive control” applications. Many proposals for these two styles have been made focusing on Bundled-data Implementation [15,18,19]. In bundled-data, the transferring of N data bits can be represented by N+2 signals, called "bundle". Each bit of data is represented by a single signal and the “+2” is a pair of local handshake signals: *request* (req) and *acknowledge* (ack). Data transfer starts with the *req* signal, and finishes with the *ack* signal, where the communication, based on the handshake protocol, may be in four or two phase way (Fig. 1a, b).

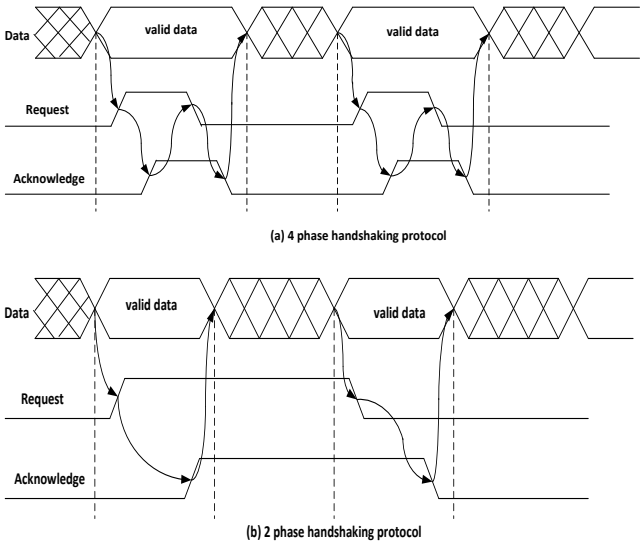

Fig. 1. Handshake protocol: a) 4-phase; b) 2-phase.

An important architecture for asynchronous pipeline is the so called MOUSETRAP, and was developed by Singh et al. [15] (Fig. 2). This architecture operates in two phases handshake protocol, presents low latency time and high throughput. Another advantage is that its control is composed only by a XNOR gate and transparent D latch.

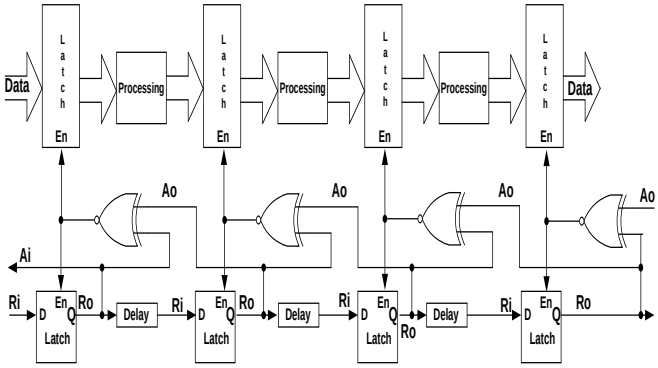

Fig. 2. MOUSETRAP linear pipeline [15].

Considering the decomposition style, there are two variants:

a) Projects that use dual-rail data-path, also known as asynchronous data-path. The STG specification (Signal Transition Graph - Petri-net), proposed by Chu [23], is the most appropriate one to describe its controller, which will interact with the asynchronous data-path without the insertion of any delay element, allowing a high concurrency. The asynchronous data-path uses dual-rail components, which have high implementation cost; and

b) Projects that use single-rail data-path, also known as synchronous data-path. The implementation is bundled-data, where a delay element is inserted, setting the cycle time of the state transition. The “Extended Burst-Mode” specification (XBM), proposed by Yun [24], is the most appropriate one to describe the controller (asynchronous finite state machine – AFSM), which interacts with the synchronous data-path (Fig. 1). We can mention, at least, three reasons why it is the most appropriate one:

i. the status variables are conditional ones, working by level and having a non-monotonic behavior;

ii. the interaction between the AFSM and the data-path is performed by a timing discretization. The cycle times of different state transitions are defined with the insertion of delay elements, which synchronize the interaction between AFSM and the data-path. This process fits in the burst-mode operation (fundamental mode) [24]; and

iii. the XBM specification is more familiar and more compact than the STG specification to the great majority of designers.

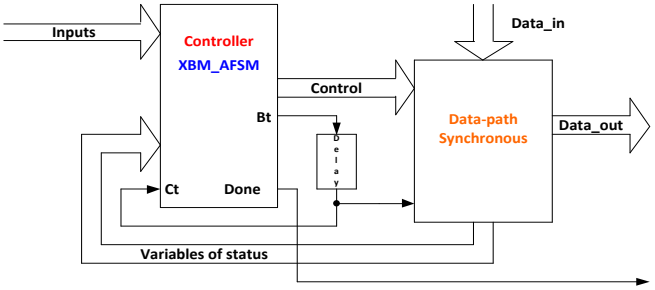

Fig. 3. General architecture: Bundled-Data asynchronous systems in the decomposition style.

III. CRYPTOGRAPHIC ALGORITHM: TEA

The TEA algorithm (Tyni Encryption Algorithm) was developed by Wheeler and Needhan, in 1994, and further expanded in [25]. It is based on the concept of symmetric encryption, due to the use of symmetric keys, i.e., using the same key to encrypt (TEA_E) and decrypt (TEA_D). The basic operation of the algorithm consists of additions and logical exclusive-OR, to generate the nonlinearity and logical shifts to the left and right, providing the mixture of data and the key. These operations are repeated several times, being suggested at least six iterations by the author. To ensure that the encryption is different after each iteration, it is used a constant *delta*, based on the golden ratio, as shown in (1). The value of *delta* ensures that the sub-keys will be distinct and its numerical value has no relevance to the quality of encryption. Figure 1 presents a diagram representing a cycle of the TEA algorithm.

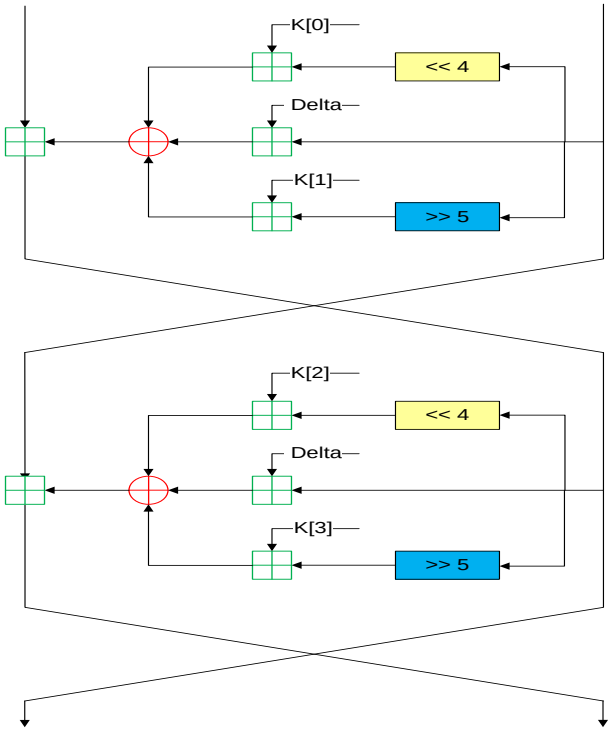

Fig. 4. Cycle of TEA algorithm of [5].

$$Delta = \left(\sqrt{5} - 1\right) \times 2^{31} \quad (1)$$

IV. ASYNCHRONOUS IMPLEMENTATION OF TEA CRYPTOGRAPHIC ALGORITHM

The asynchronous design of encryption algorithms (TEA_E and TEA_D) follows the decomposition style and have been implemented in the target architecture of Fig. 3, using FPGA devices.

A. Asynchronous implementation on FPGA

Commercial FPGAs devices have become a very popular way to prototype and implement digital systems due to its low cost and short time project [26]. They were developed to support synchronous designs, thus the asynchronous implementation is not natural [27,28]. Two main difficulties arise in asynchronous designs based on LUTs (Look-Up-Table) in FPGAs involving AFSM_XBM: *a)* Process of mapping hazard-free Boolean functions in logic blocks (macrocells). The commercial tools used for decomposition and mapping Boolean functions in LUTs are not prepared to meet the requirements of logical hazard. This may cause a circuit malfunction, if manual intervention to fix the problem is not performed. The mapping function must satisfy the decomposition requirements proposed in Sigel et al. [29]; *b)* Internal routing process among macrocells can introduce significant delays. These delays can result in essential hazard and lead to a circuit malfunction [27]. The circuit delay model defines how to solve the problem of essential hazard: insertion of delay elements in the feedback lines or employ macrocells that satisfy the isochronic fork condition [13,30].

B. Asynchronous optimized design using decomposition style

The TEA_E and TEA_D algorithms were synthesized with an asynchronous behavioral synthesis tool proposed by Garcia [19]. The tool accepts an algorithm described in LASYN language (language synthesis - input file for the tool). Analyzing the synthesis of TEA_E algorithm, in the first step the tool synthesizes the optimized single-rail data-path using only synchronous paradigm components. Figure 5 shows the DFG (Data Flow Graph) [20] of the TEA_E algorithm, in which there are ten steps with a single machine cycle. The data-path required 5 ALUs (arithmetic logic unit), of which 4 ALUs have different operations, and 16 registers (Fig. 7). In the second step, it synthesizes the AFSM_XBM with Sicarelo tool [31], presenting 15 states, 16 states transitions, 3 input signals and 61 output signals.

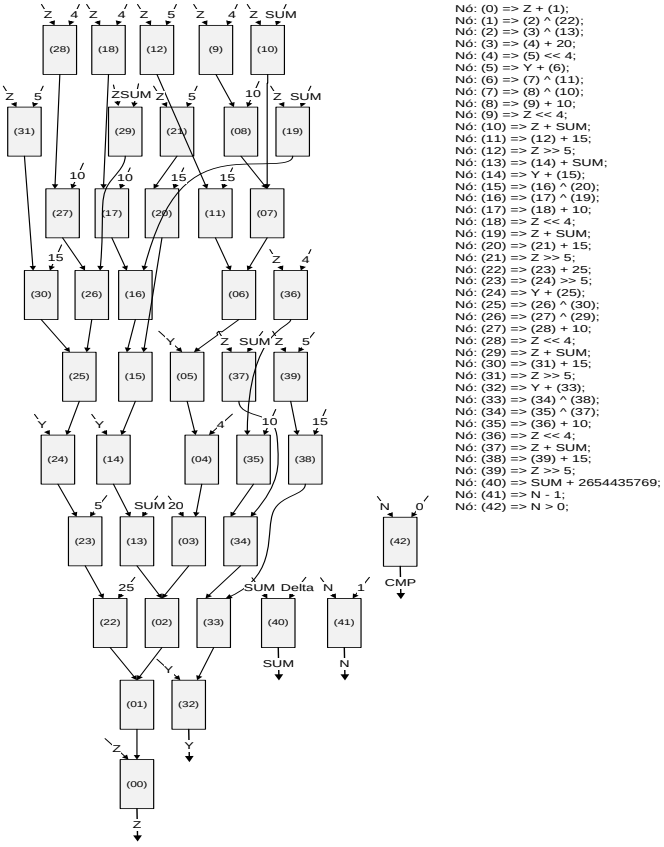

Fig. 5. DFG of TEA_E algorithm.

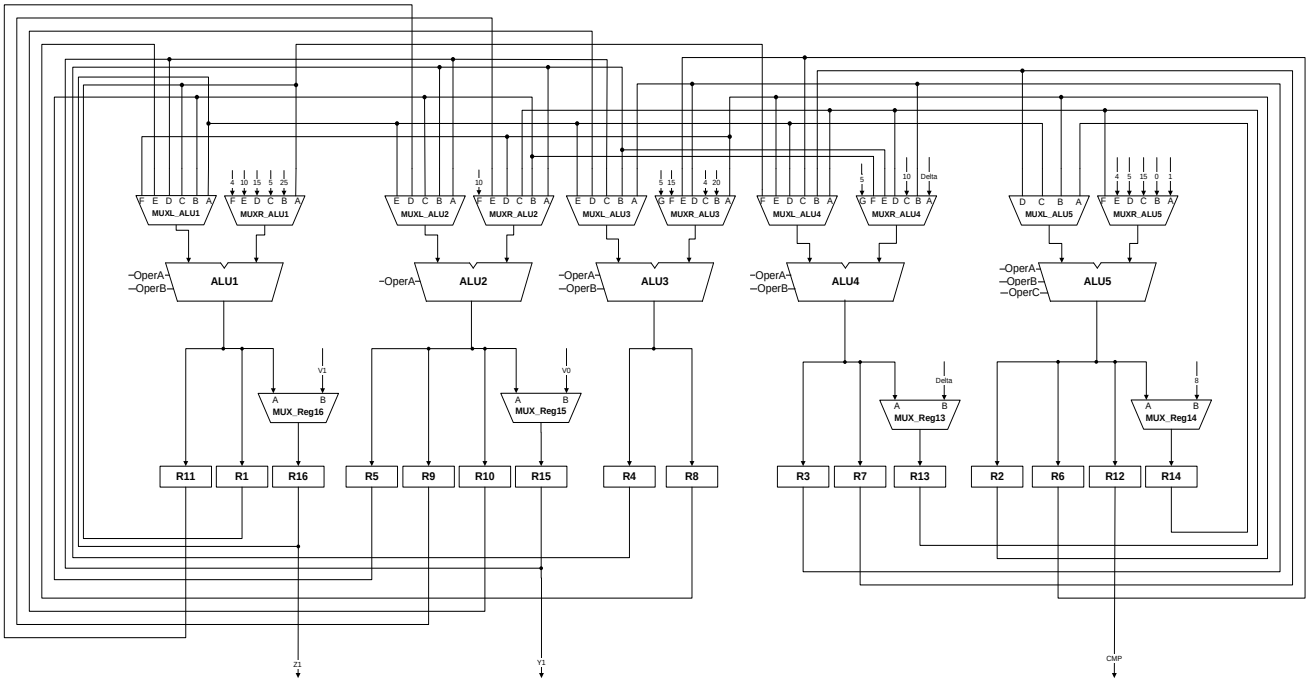

Fig. 6. Data-path single-rail for asynchronous TEA_E.

V. SIMULATIONS & RESULTS

The simulations and designs in the proposed synchronous and asynchronous styles were made for the TEA_E and TEA_D algorithms, and were performed in Quartus II software, version 9.1 [32], considering Altera STRATIX II (EP2S15F484C3) as target device.

A. Simulations of synchronous and asynchronous designs

Figure 7 shows the simulation of the synchronous version of TEA_E algorithm. The waveforms are exactly as expected to the TEA_E, where V0 and V1 are the inputs of both data, and Y1 and Z1 are the encrypted outputs. Figure 8 shows a simulation of the asynchronous version of TEA_E algorithm. The waveforms are exactly as expected to TEA_E, arriving to the same results of the synchronous version.

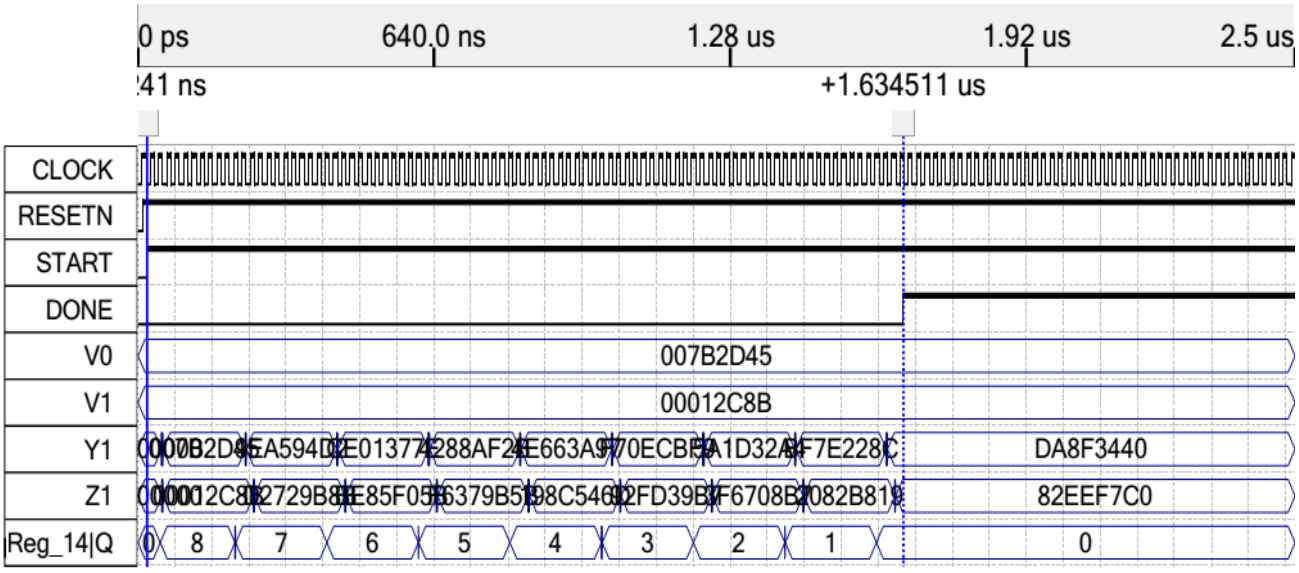

Fig. 7. Simulation: synchronous TEA_E.

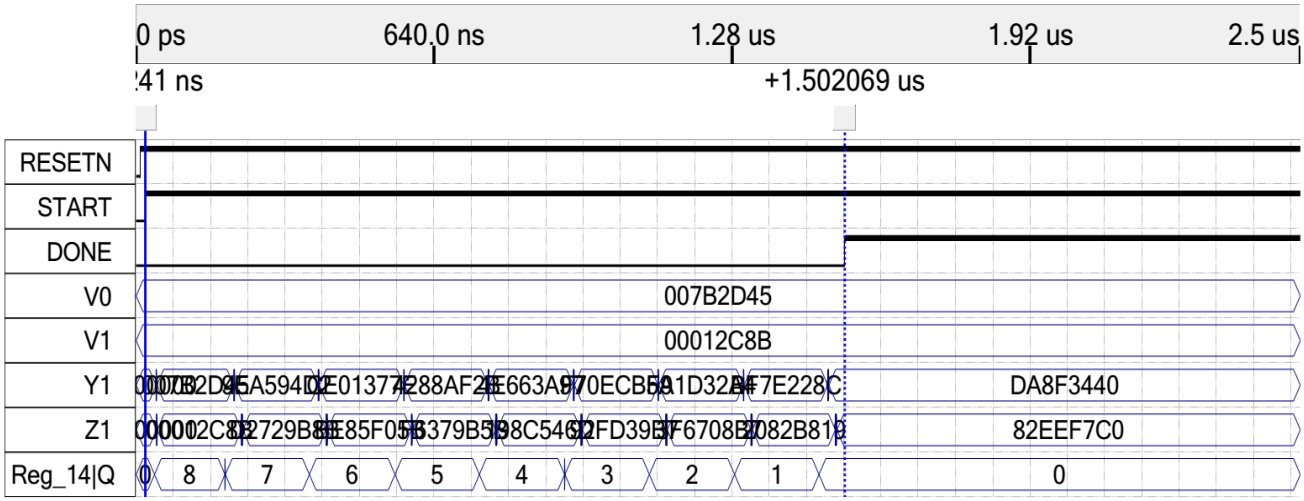

Fig. 8. Simulation: asynchronous TEA_E.

B. Results of synchronous and asynchronous designs

Table I shows the results of synchronous and asynchronous versions of TEA_E algorithm. Compared with the synchronous version, the asynchronous proposal achieved an 8% reduction in processing time and a reduction in power loss of 11.9%. There was a penalty in area (LUTs + FFs) of only 1.2%.

Table II shows the results of synchronous and asynchronous versions of TEA_D algorithm. Compared with the synchronous version, the asynchronous proposal achieved a reduction of 15% in processing time and a reduction in the dissipated power of 12%. There was a penalty in the area (LUTs + FFs) of 4%.

TABLE I RESULTS: ENCRYPTION ALGORITHM TEA_E

		Time of Latency	Power Dissipation	Macrocells	
				Number LUTS	Number Flip-Flops
TEA_E	Synchronous f_{MAX}=55 MHz	1634.5ns	519.15mw	2090	**471**
	Proposal Asynchronous	**1502.01ns**	**462.12mw**	**2074**	520

TABLE II RESULTS: DECRYPTION ALGORITHM TEA_D

		Time of Latency	Power Dissipation	Macrocells	
				Number LUTS	Number Flip-Flops
TEA_D	Synchronous f_{MAX}=50 MHz	1.818,78ns	541,49mw	**2005**	**471**
	Proposal Asynchronous	**1.546,01ns**	**477,00mw**	2054	523

VI. CONCLUSION

Digital systems focusing on the protection of data are increasingly demanded, being based on the encryption concept. The encryption algorithms are subject to attacks, which are based on the analysis of physical quantities, often using the clock signal to do that. In this paper we proposed two asynchronous implementations of TEA_E and TEA_D algorithms, not using the clock signal, what makes them more robust the SCA attacks. These algorithms have been implemented in the optimized FSM_XBM architectures of "local clock" + "conventional data-path", which can be mapped onto any PLD device, without requiring any kind of macro-cells mapping. For future works we will propose TEA_E and TEA_D implementations in other architectures focused on FSM_XBM.

REFERENCES

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

[2] 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.

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

[4] 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.

[5] 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.

[6] 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.

[7] 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.

[8] R. Bergamaschi, et al. "Automating the Design of SoC using Cores". *IEEE Design & Test of Computers*, vol. 18-5, Sep 2001, pp. 32-45.

[9] X. Han, et al. "Fault-Tolerante Relay Node Placement in Heterogeneous Wireless Sensor Networks". *IEEE Transactions on Mobile Computing*, vol. 9-5, pp. 643-656, May 2010.

[10] B. Morris, et al. "How to Encipher Messages on a Small Domain Deterministic Encryption and the Thorp Shuffle". In: 29th Annual International Cryptographic Conference (CRYPTO'09), pp. 286-302, Aug. 2009.

[11] D. Goldhaber-Gordon, et al., "Overview of Nanoelectronic Devices," *Proc. of the IEEE*, vol. 85, No. 4, pp.521-540, April 1997.

[12] J. Cortadella, et al., "Coping with the variability of combinational logic delays," *ICCD*, pages 505–508, 2004.

[13] C. J., Myers, "*Asynchronous Circuit Design*", Wiley & Sons, Inc., 2004, 2ª edition.

[14] I. E. Sutherland, "Micropipelines", *Communication of the ACM*, vol. 32, No.6, pp.720-738, June, 1989.

[15] M. Singh and S. M. Nowick, "MOUSETRAP: High-Speed Transition-Signaling Asynchronous Pipelines", *IEEE Trans. on VLSI Systems*, vol.15, no. 6, pp.684-698, June, 2007.
[16] S. F. Nielsen, J. Sparsø and J. Madsen, "Behavioral Synthesis of Asynchronous Circuits using Syntax Directed Translation as Backend," IEEE Trans. on VLSI Systems, vol. 17, Nro. 2, pp.248-261, February 2009.
[17] T. Chelcea, and S. M. Nowick, "Resynthesis and Peephole Transformations for the Optimization of Large-scale Asynchronous Systems," Proc. ACM –DAC, pp.405-410, 2002.
[18] Minoru I. et al., "A Tool Set for the Design of Asynchronous Circuits with bundled-data Implementation," Proc. IEEE 29th Int. Conf. on Computer Design (ICCD), pp.78-83, 2011.
[19] K. Garcia, "Behavioral synthesis of Optimized Asynchronous Circuits," Master of Science, Instituto Tecnológico de Aeronáutica – ITA, Brazil, 2015, (in portuguese).
[20] G. De Micheli, "*Synthesis and Optimization of Digital Circuits*" McGraw Hill International Editions, 1994.
[21] L. Spadavecchia, "A Network-based Asynchronous Architecture for Cryptographic Devices," PhD thesis, University of Edinburgh, 2005.
[22] R. I. Soares, "Arquiteturas GALS Pipeline para Criptografia robusta a Ataques DPA e DMA," Tese de Doutorado, PUC-RS, 2010.
[23] T. –A. Chu, "Synthesis of Self-Timed VLSI Circuits from Graph-Theory Specifications," PhD. Thesis, June, 1987, Dep. Of EECS, MIT.
[24] K. Y. Yun e D. L. Dill, "Automatic Synthesis of Extended Burst-Mode Circuits: Part I (Specification and Hazard-.Free Implementation) and Part II (Automatic Synthesis)," IEEE Trans. on CAD of Integrated Circuit and Systems, Vol. 18:2, pp. 101-132, Feb. 1999.
[25] G. S. Mahdi, "A modification of TEA block cipher algorithm for data security (MTEA),"Engineering & Technology Journal, v. 29, n. 5, 2011. University of Technology, Iraq Academic Scientific Journals.
[26] J. J. Rodriguez, et. Al., "Features, Design Tools, and Applications Domains of FPGAs", *IEEE Trans. on Industrial Electronics*, vol. 54, No. 4, pp.1810-1823, August 2007.
[27] E. Brunvand, "Using FPGAs to Implement Self-Timed Systems", *Journal of VLSI Signal Processing*, Special issue on field programmable logic, vol.6(2), pp.173-190, August, 1993.
[28] M. Tranchero and L. M. Ryneri, "Implementation of Self-Timed Circuits onto FPGAs Using Commercial Tools", 11th Euromicro Conf. on Digital System Design Architectures, Methods and Tools, pp.373-380, 2008.
[29] P. Sigel, G. De Michele and D. Dill, "Decomposition methods for library binding of speed-independent," Proc. Int. Conf. Computer-Aided Design, pp.558-565, 1994.
[30] D. L. Oliveira, et al., "Burst-Mode Asynchronous Controllers on FPGA," *Int. Journal of Reconfigurable Computing*, vol. 2008, pp.1-10, 2008.
[31] T. Curtinhas, et al. "SICARELO: A tool for synthesis of Locally-clocked Extended Burst-Mode Asynchronous Controllers," IEEE 5rd Latin American Symposium on Circuit and Systems, pp.1-4, 2014.
[32] Altera Corporation, 2016, www.altera.com.