A Fully Synthesized Key Establishment Core based on Tree Parity Machines in 65nm CMOS

Héctor Gómez, Óscar Reyes and Elkim Roa
Design Group of Integrated Systems CIDIC - Universidad Industrial de Santander
Bucaramanga, Santander, Colombia
hector.gomez@correo.uis.edu.co,{efroa,omreyes}@uis.edu.co

***Abstract*—This paper presents a low-area ASIC implementation of a fully-synthesized symmetric key establishment architecture based on tree parity machines (TPMs) in 130nm and 65nm standard-cell CMOS technologies. The proposed circuit architecture has a serial datapath with re-keying characteristic enabled by a proposed pseudo-random binary sequence (PRBS) generator based on variable-length linear-feedback shift register (LFSR). A circuit technique is proposed that enhances datapath access to add re-keying feature. Fully-synthesized results for 130nm and 65nm show an area consumption of 0.016mm^2 and 4800μm^2 respectively. Relative area and power consumption are studied by comparing synthesized TPMs with an implementation of a CRC16 error detection code used within security applications. Comparison is made through a proposed figure of merit that include the generated key length in order to show scalability of the architecture with the available technologies.**

I. Introduction

Low cost systems with sensitive information impose energy and size limitations on the implementation of security primitives [1], [2]. Energy limitation forces to exploit availability of resources avoiding programmable devices that consume extra power or are not optimized for security functionality. Size limitation restricts memory and logic implying that software-based security solutions have to move to hardware resulting in size and power reductions.

Restricted systems prefer implementation of symmetric cryptography requiring the establishment of a shared key pair over a public unsecured channel. Key establishment based on elliptic curve and hyper elliptic curve public cryptography is very common in embedded systems [3]. However, public cryptography is a software-oriented solution making difficult the integration in resource limited scenarios.

Neural cryptography presents a hardware friendly alternative to implement key establishment in a completely symmetric way [4]. The usage of tree parity machine (TPM) neural networks in synchronization mode provides an algorithm to perform key establishment with low computational resources. Only a few implementations are reported with incomplete information of final resources usage such as in [3], [2], [5] and [6]. Volkmer [3], [2], proposes an architecture for FPGA but power consumption is not reported. On the other hand, Mühlbach [5], [6] proposes a synthesized-architecture on FPGA and 180nm CMOS technology but area is not reported. In addition, Volkmer and Mühlbach do not report layout results.

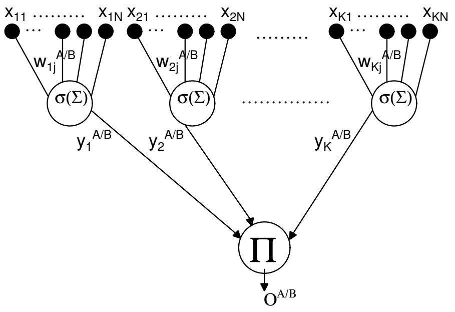

Fig. 1: Tree parity machine(TPM) neural network.

This paper proposes a more secure key establishment algorithm with two new circuit implementations techniques. First, full control over stored information in memory device is provided by a modified datapath. Second, additional obfuscation is added by a proposed variable-length PRBS to increase the complexity for attackers without significant impact on the resources required. Furthermore, a detailed implementation of a fully-synthesized TPM core to perform key establishment in 130nm and 65nm CMOS technologies. Analysis of resources consumption is made by comparing TPMs implementations with CRC16 code detection error showing a relative consumption to a common used circuit in communication systems. A figure of merit (FoM) is proposed to set a criteria for comparing different TPM implementations including the key length.

II. Neural Cryptography: TPM algorithm

Neural cryptography is an alternative to achieve key establishment in light-weight applications using a special property: two randomly initialized neural networks learning from each other can synchronize [7]. These particular feed-forward neural networks, having one layer of hidden units, are called parity machines (PM). Both networks have common inputs and exchange information about their outputs. In case of agreement, the two PMs are trained by a Hebbian rule based on their mutual outputs, resulting in many cases in a complete-synchronization state of synaptic weights. This behavior can work as a possible key exchange protocol for data transmission because it is difficult for an attacker to reveal the common parameters after synchronization [4].

978-1-5090-0493-5/16/$31.00 © 2016 IEEE

Multi-layer feedforward PMs (tree PM or TPM) show better performance in synchronization mode. TPM structure has three layers as shown in the Fig. 1. Input layer has N non-overlapping binary inputs that are random and common to the networks. One hidden layer with K units (input field of K $\cdot$ N) and bounded discrete weights $w_{kj}^{A/B} \in [-L, L]$. Finally, one binary output $O^{A/B}(t) \in \{-1, 1\}$ is calculated by a parity function of the signs of summations:

$$\begin{aligned} O^{A/B}(t) &= \prod_{k=1}^{K} y_k^{A/B}(t) \\ &= \prod_{k=1}^{K} \sigma \left(\sum_{j=1}^{N} w_{kj}^{A/B}(t) x_{kj}(t) \right) \end{aligned} \tag{1}$$

where A and B are the two parties or networks, N is the number of inputs and K the number of hidden units.

Regarding other applications, through the process of synchronization it is possible to obtain a common key transmitting information that is not directly related to the key itself. Therefore, a key exchange based on TPM can be used as a complement to another type of encryption to increase security [3]. Final key length depends on TPM parameters and is given by:

$$KEY_{length} = \log_2((2 \times L + 1)^{KN}). \tag{2}$$

III. Symmetric Key Establishment Architecture Based on TPMs

Hardware-based encryption engine in wireless applications make use of symmetric key algorithms -due to their light-weight calculations- to provide security with low resources. TPM algorithm produces an output based on a parity function where multiplications are avoided reducing the number of operations. Therefore, an architecture based on TPMs is an interesting alternative due to the associated light-weight calculations.

A Datapath based on [8] with several improvement and additions is shown in Fig. 2. Datapath uses serial data flow to compute the output. The architecture uses a pseudo-random binary sequence generator (PRBS) based on variable-length linear feedback shift register (LFSR) as shown Fig. 3. The variable characteristic is implemented using a mux to select what registers are used as feedback and the serial output of the PRBS is selected in the same way as the feedback signals (depending on the required length). Furthermore, PRBS produces initial synaptic weights where the LFSR output is enabled by a counter. From any part of the generated sequence the final output is selected and masked with logical gates. It is possible to select all of these options externally and randomly. In addition, the LFSR provides pseudo-random inputs with fixed options using the serial output feature in order to develop the synchronization process.

The objective of the architecture, once synchronization occurs, is the fast re-keying. Re-keying consists in randomly modifying a part of the synchronized-synaptic weights and running the synchronization process again, as consequence the synaptic weights can be aligned faster than in the initial process. Previous works ([2], [6]) are not clear with the strategy for re-keying and its implementation, thus, this work proposes the following. The proposed datapath has additional inputs to access the memory device and to provide full control over stored information in order to change partial weights and to re-start synchronization process. Moreover, the proposed PRBS allows to pseudo-randomly change these weights taking advantage of variable length and masked output, improving statistical properties of re-keying and obfuscating the process for any adversary.

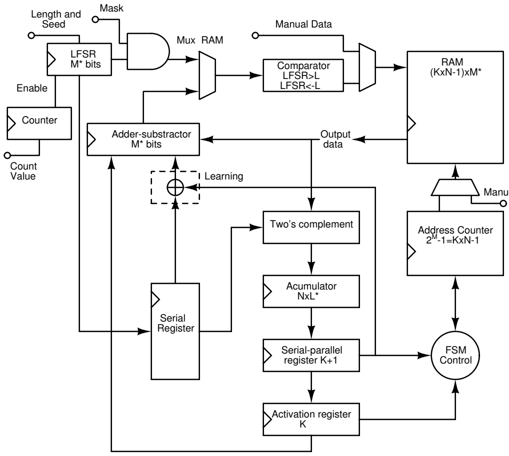

Fig. 2: TPM's serial datapath architecture.

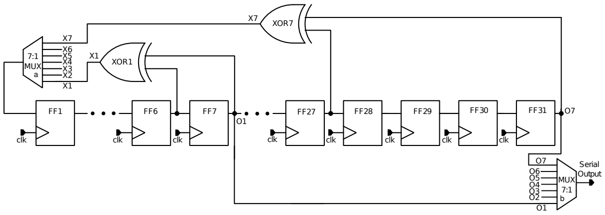

Fig. 3: Proposed PRBS with variable length and masked output.

This work uses the finite state machine (FSM) as is shown in Fig. 4 which consists of ten states. The objective of the FSM is to allow data serial calculation besides providing control over the proposed-re-keying feature. The FSM uses a pre-initialization to store the initial pseudo-random weights and after the synchronization process the FSM allows to modify the aligned weights in two ways: a completely arbitrary modification or using the proposed PRBS to give better statistical properties to the modification.

IV. Results

Two different TPMs have been fully-synthesized and verified in two different technologies. Synthesis process is performed by using the different corners available in both technologies 130nm and 65m, corresponding to the exposed in Table I. Synthesized TPMs have an input field of 42 (N=14

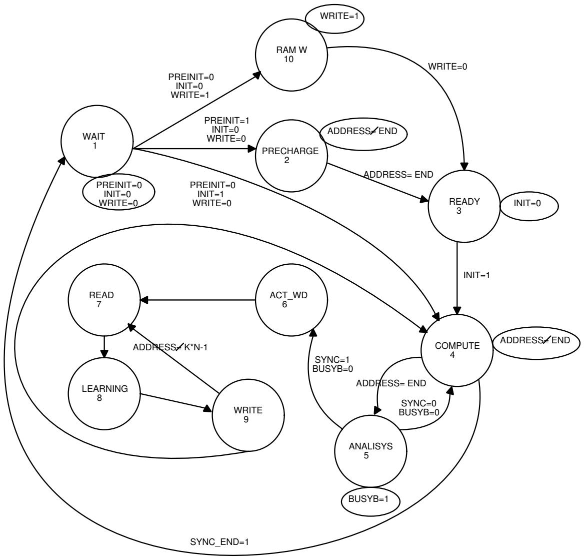

Fig. 4: TPM's finite state machine.

TABLE II: TPM used cells.

Process	Sequential	Inverters + buffers	logic
Worst case			
130nm	347	100	573
65nm	347	116	562
Typical case			
130nm	347	104	570
65nm	347	120	568
Best case			
130nm	347	107	562
65nm	347	113	566

TABLE III: TPM synthesis results.

Process	Area [μm^2]	Energy [μW/MHz]	Max. Freq. [MHz]
Worst case			
130nm	16600	19.4	350
65nm	4800	7.6	434
Typical case			
130nm	16600	22.4	500
65nm	4800	8.7	667
Best case			
130nm	16600	31.5	770
65nm	4780	10.2	800

and K=3) with synaptic weights of 4 given an approximate key length of $K_L = \log_2((2\times4+1)^{42}) \approx 133-\text{bits}$. Concordance of results is evident with the number of sequential cells which is 347 for both technologies as shown Table II. Moreover, logic cells and the usage of buffers and inverters are similar.

Timing constraints considering non-ideal clock network provide the maximum operation frequency for the three corner cases and Table III shows that this frequency correspond for 65nm and 130nm to 434MHz and 350MHz respectively in worst-case condition. More accurate measurement of power consumption is achieved by extracting activity of cells.

Synchronization process of the proposed TPM implementation takes about 200 training with 40000 clock cycles resulting in about 200 cycles per training. Cycles and number of training agree with the reported by [2] and [6] for keys around 128 bits considering an ideal communication channel. These number of cycles result in synchronization times of 92.1us and 114us for 65nm and 130nm respectively. In addition, the re-keying procedure allows it to reduce the synchronization steps about 100 due to the weights are partially aligned at the time to re-start the establishment.

A. *Chip layout*

Layout for both technologies is built by using the synthesis results of worst case library and taking into account clock tree specification for buffering. Figs. 5 and 6 show the final layout for both technologies with power rings. In addition, figures highlight the area and placement of datapath and FSM which help to verify the accomplishment of constraints. Final layouts area are 160μm$\times$160μm and 113μm$\times$113μm for 130nm and 65nm respectively where in both cases the datapath occupies most of the area.

TABLE I: Library corners.

Corner	Process	Voltage [V]	Temperature [°C]
Worst case	Slow-Slow	1.08	125
Typical case	Typical-Typical	1.2	25
Best case	Fast-Fast	1.32	-40

B. *A cyclic redundancy check (CRC) implementation.*

Communication systems use code-detection errors. Wireless systems such as smart cards use CRC due to the algorithm computational efficiency to detect errors in data transmission. For example, Lee [9] propose the use of a CRC16 in an RFID passive tag with cryptographic functions where the main limitation is that power supply comes from the tag antenna. Moreover, this kind of tag should be not only power efficient but also area efficient to reduce chip cost. The impact of cryptographic functions such as the TPM algorithm in area

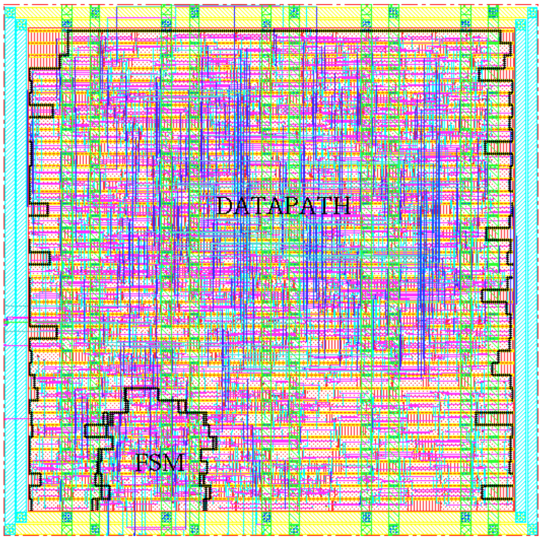

Fig. 5: TPM layout in 130nm. Final area consumption is 160μm$\times$160μm using a double power ring of 3μm width.

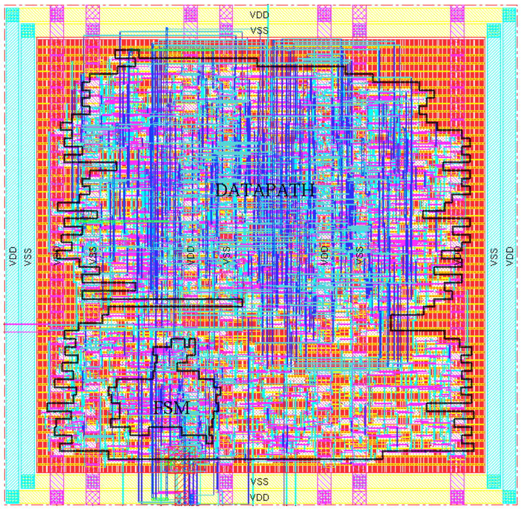

Fig. 6: TPM layout in 65nm. Final area consumption is 113μm×113μm using a double power ring of 3μm width.

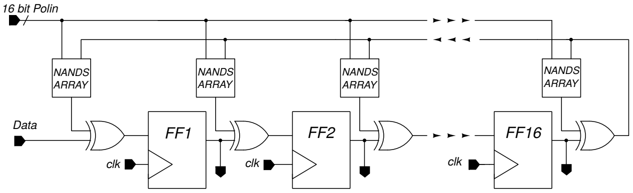

Fig. 7: CRC16 block diagram.

utilization can be estimated by doing a comparison with other components implemented within security applications, for example CRC16.

Fig. 7 shows the CRC16 block diagram which its polynomial is programmable with a fixed order. CRC cells results in Table IV show good hardware description with 16 sequential cells for both 130nm and 65nm technologies for worst case library. In addition, CRC synthesis shows area and power specification in Table V including operation frequencies.

Quantification of comparison between TPM and CRC16 implementation is made through the proposed FoM given by (3). The FoM includes area and energy consumption besides taking into account the key length of the establishment process. This FoM allows to compare different implementations based on the resources consumption and the effective key length, being better the implementation with the minimum FoM value. The resulting FoM in Table V shows the scalability of proposed circuital architecture. The FoM for related works of Volkmer [3], [2] and Mühlbach [5], [6] cannot be calculated due to the absence of used resources. Moreover, neither system integration nor layout are reported by Volkmer and Mühlbach.

TABLE IV: CRC used cells.

Process	Sequential	Inverters + buffers	logic
130nm	16	7	66
65nm	16	8	66

TABLE V: CRC synthesis results and comparison with TPMs using the proposed FoM.

Process	Area [μm^2]	Energy [uW/MHz]	Freq. [MHz]	FoM [bit^{-1}]
130nm	930	1.7	350	1.591
65nm	280	0.8	500	0.598

$$FoM = \left(\frac{Engine_{Energy}}{CRC16_{Energy}} \times \frac{Engine_{area}}{CRC16_{area}} \times \frac{1}{Key_{length}}\right) \tag{3}$$

V. Conclusions

This paper presents a fully-synthesized TPM core in 65nm and 130nm technology. The proposed TPM implementation enables the key establishment and re-keying using a serial datapath and a FSM to control the process of synchronization and re-keying. A post-processed seed using variable length PRBS adds obfuscation to the re-keying increasing the hardening of the system. Two proposed circuits techniques were implemented to reduce implementation cost. An improved re-keying feature is achieved by using the PRBS with variable length and a masked output that is also used to generate initial weights and in synchronization process to generate pseudo random inputs in order to reduce the implementation cost. This proposed implementation can be used in low-cost systems such as in radio-frequency identification (RFID) where symmetric cryptography is mandatory for security purposes, implying the need of a shared key. Common low-cost systems usually use well-known CRC algorithms to detect errors making CRC implementation appropriate to be used as a measurement reference for size and power. A FoM is proposed to study resources consumption relative to CRC16 implementation showing the scalability of the circuital architecture.

References

[1] H. Houssain, M. Badra, and T. F. Al-Somani, "Hardware implementations of elliptic curve cryptography in wireless sensor networks," in *Internet Technology and Secured Transactions (ICITST), 2011 International Conference for*, Dec 2011, pp. 1–6.

[2] M. Volkmer and S. Wallner, "Tree parity machine rekeying architectures," *IEEE Transactions on Computers*, vol. 54, no. 4, pp. 421–427, Apr. 2005.

[3] ——, "A Key Establishment IP-Core for Ubiquitous Computing," in *16th International Workshop on Database and Expert Systems Applications (DEXA'05)*. Copenhagen: IEEE, 2005, pp. 241–245.

[4] A. Ruttor, W. Kinzel, L. Shacham, and I. Kanter, "Neural cryptography with feedback," *Physical Review E*, vol. 69, no. 4, p. 46110, Apr. 2004.

[5] S. Muhlbach and S. Wallner, "Secure and Authenticated Communication in Chip-Level Microcomputer Bus Systems with Tree Parity Machines," in *2007 International Conference on Embedded Computer Systems: Architectures, Modeling and Simulation*. IEEE, Jul. 2007, pp. 201–208.

[6] S. Mühlbach and S. Wallner, "Secure communication in microcomputer bus systems for embedded devices," *Journal of Systems Architecture*, vol. 54, no. 11, pp. 1065–1076, 2008.

[7] I. Kanter and W. Kinzel, "Cryptography based on neural networks - analytical results," *Journal of Physics A: Mathematical and General*, vol. 35, no. 47, pp. 1–4, 2002.

[8] J. Blanco, "Implementación sobre FPGA de un algoritmo de intercambio de llave simétrica basado en redes neuronales," Project Degree Universidad Industrial de Santander, 2015.

[9] J.-w. Lee, D. H. T. Vo, Q.-h. Huynh, and S. H. Hong, "A Fully Integrated HF-Band Passive RFID Tag IC Using 0.18-um CMOS Technology for Low-Cost Security Applications," *IEEE Transactions on Industrial Electronics*, vol. 58, no. 6, pp. 2531–2540, 2011.