Hindawi Publishing Corporation
VLSI Design
Volume 2008, Article ID 160728, 7 pages
doi:10.1155/2008/160728

Research Article

A Programmable Hardware Cellular Automaton: Example of Data Flow Transformation

Samuel Charbouillot, Annie Pérez, and Daniele Fronte

Laboratoire Matériaux et Microélectronique de Provence (L2MP-POLYTECH), UMR CNRS 6137, IMT-Technopôle de Château Gombert, 13451 Marseille Cedex 20, France

Correspondence should be addressed to Annie Pérez, perez@polytech.univ-mrs.fr

Received 13 April 2007; Accepted 9 December 2007

Recommended by Jean-Baptiste Begueret

We present an IP-core called PHCA which stands for programmable hardware cellular automaton. PHCA is a hardware implementation of a general purpose cellular automaton (CA) entirely programmable. The heart of this structure is a PE array with reconfigurable side links allowing the implementation of a 2D CA or a 1D CA. As an illustration of a PHCA program, we present the implementation of a symmetric cryptography algorithm called ISEA for Ising spin encryption algorithm. Indeed ISEA is based on a 2D Ising spin lattice presenting random series of disordered spin configurations. The main idea of ISEA is to use this disorder to encrypt data. Efficiency of ISEA and PHCA implementation results are given.

1. INTRODUCTION

Cellular automata (CA) were originally introduced by von Neumann for studying self-reproduction in biological systems [1]. Then they have been used for language recognition and modelling of physical systems [2]. The mathematical properties of cellular automata were also studied. Nowadays, CA automata are also used for high-quality random numbers generation [3] and implementations of reconfigurable hardware CA are proposed [4].

This paper proposes an intellectual property (IP) core for a programmable hardware cellular automaton (PHCA). PHCA is a powerful tool for the design and test of 1D or 2D cellular automata rule applications. An acyclic one-dimensional cellular automaton and a cyclic two-dimensional cellular automaton can be implemented on the PHCA. The architecture of the PHCA is a fine grained fully parallel structure inspired by a classic single instruction multiple data (SIMD) structure made of 1-bit processing elements (PEs) [5].

An example of PHCA program concerns a cryptography application. The cryptography field is still increasing nowadays. Electronic transactions become very important and require security since most of them are concerned with either payments or confidential data. Public key and secret key cryptographic algorithms provide a solution to this security problem. These algorithms are able to ensure data authenticity, integrity, and confidentiality [6]. Secret key algorithms are more suitable for hardware implementation.

In the context of the secret key algorithms, we propose a symmetric algorithm based on cellular automata rules. This algorithm is called Ising spin encryption algorithm (ISEA) because it uses a system of Ising spins. In this paper, we focus on a 2D Ising spin lattice [7, 8]. The time evolution of the spin configuration in this lattice is managed by local rules leading to disordered configurations in accordance with certain conditions. The configuration space is explored by a random walk imposed by a microcanonical Monte Carlo method [9]. ISEA uses the disordered spin configurations to encrypt data by combining the spin lattice and an array of data to be encrypted. This encryption process is rather fast. Moreover, the permanent exchanges between neighbor sites introduce a constant noise useful against the attacks based on power analysis.

The PHCA may be programmed according to 1D or 2D cellular automata (CA) rules. This work focuses on the PHCA with the 2D configuration and is programmed according to ISEA rules. Each site of the spin lattice system is

updated by a PE. All the PEs apply the same rule concurrently. An example of resulting encrypted data array is given below. A first version of a PHCA provided with a 32×32 PE array has been implemented on an Xilinx FPGA xc3s5000. The throughput of the encrypted data stream is 16 Mbps.

This paper is divided into six sections. Section 2 shows how the PHCA architecture maps a CA. Section 3 introduces the microcanonical Monte Carlo methods and describes the local rules of the algorithm ISEA. Section 4 shows in detail the encryption process. In Section 5, we present and discuss the ISEA en-/decryption results and the PHCA implementation performances. Finally, Section 6 gives our conclusion.

2. PHCA ARCHITECTURE

The aim is to realize a programmable hardware tool suitable for CA rules implementation. The architecture of this tool is inspired by a classic SIMD structure [5].

2.1. Mapping a cellular automaton

A cellular automaton consists of several identical cells governed by simple rules. The cellular automaton is globally synchronized; that is, at each time step each cell updates its state according to some set of local rules.

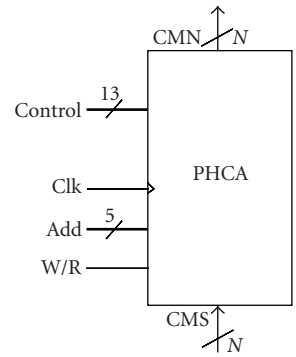
More precisely, the next state of each cell depends on the present state of the neighbor cells [10]. The cell itself may be included in its own neighborhood. A cellular automaton can be of any dimension and can be either cyclic or acyclic. Moreover, CAs are suitable for hardware implementation since they are simple, regular, locally interconnected, and modular.

This work focuses essentially on 2D CA with a north, east, west, and south (NEWS) array of cells that are synchronous, governed by local rules, uniform (i.e., all the cells obey the same rule), and with a von Neumann neighborhood. In this case, the next state $x_{i,j}(t+1)$ of the cell $(i; j)$ depends on its own present state and on the present state of its four nearest neighbors:

$$x_{i,j}(t+1) = f\left[x_{i,j}(t),\, x_{i-1,j}(t), x_{i+1,j}(t), x_{i,j-1}(t), x_{i,j+1}(t)\right]. \tag{1}$$

In order to design an IP-core mapping this definition, we chose to describe a multiprocessor fine-grained structure operating in fully parallel mode. For the instruction stream organization, we chose an SIMD scheme in order to avoid synchronization as well as connection problems.

The heart of this SIMD structure is an array of processing elements (PEs) controlled by the same instruction. The memory is distributed. At each clock cycle, all the PEs execute concurrently the same instruction on the data stored in their internal memory elements. We wanted to map one cell of the CA to one PE. We chose a one-bit architecture for each PE in order to integrate more PEs (more cells) in the array than in the case of more coarse-grained structures. Of course the consequence is that the computation performances slow down when multibit operands must be treated.



Pin name	Function
Control	Control bus
Clk	Clock
Add	RAM address bus
W/R	RAM write(1) read(0)
CMS	Data input bus
CMN	Data output bus

Figure 1: PHCA logic symbol and associated pin functions.

2.2. PHCA symbol and interconnections

The PHCA logic symbol for an $M \times N$ PE array is given in Figure 1. The external data enter through the N-bit south-data bus CMS and exit through the N-bit north-data bus CMN. The thirteen control lines bring the same instruction word to each PE. As we shall see below, each PE has a private 32×1 bit RAM controlled by the W/R input and addressed by the 5-bit Add input bus. All the registers of the PHCA are synchronized by the same clock Clk.

An example of a 4×4 PE array is shown in Figure 2. This regular processor square grid has fixed communication links between the nearest neighbors. Moreover, when all the switches of the west array side are in position 1, the PE array is wrapped around in a toric mode to implement a cyclic two-dimensional cellular automaton. Otherwise, when all the switches are in position 2, the PHCA becomes a chain of PEs to implement an acyclic one-dimensional cellular automaton. This last configuration is not explored in the present work.

2.3. PHCA processing element

The PHCA contains $M \times N$ single-bit processor elements. The structure of a PE is detailed in Figure 3. A PE is equipped with a 32×1-bit RAM, five multiplexers, one single-bit arithmetic and logic unit (ALU), four 1-bit registers (NS, EW, C, CM), and input/output ports on all four sides. The ALU is a full adder/subtractor. The result of an addition is given on the ALU outputs CY and SM, and the result of a subtraction on the ALU outputs BW and SM. These ALU outputs CY, SM, and BW correspond also to logic operations in accordance with certain conditions. The registers and RAM accept data from up to eight possible sources through the five multiplexers. The concatenation of these multiplexer's control bits gives the 13-bit instruction word. The instruction set of PHCA is given in Table 1. Up to five commands can be executed simultaneously during each instruction cycle.

N/S and E/W links connect a processor cell to its four neighbors. CMS/CMN links provide the PE array with a second vertical link system which is particularly useful because it does not communicate with the ALU. So these CMS/CMN links allow a south−north shift of the data stream through the whole array concurrently with other PE operations.

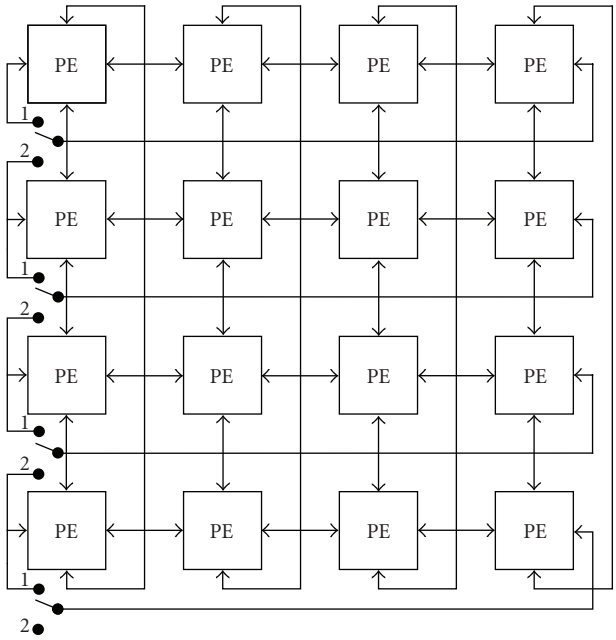


Figure 2: Fixed communication links between the nearest neighbors. Configurable links on the west side of the array.

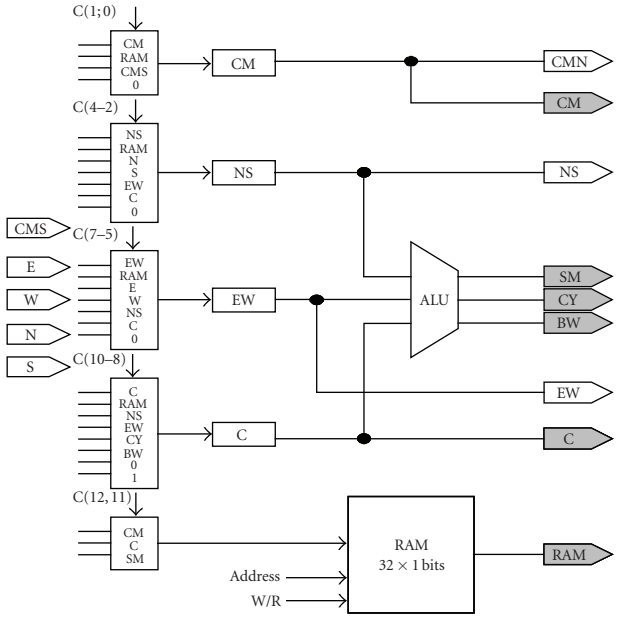


Figure 3: PE architecture.

The dark-grey outputs are reinjected as multiplexers inputs into the PE itself.

3. MICROCANONICAL MONTE CARLO METHOD

3.1. Main idea

Many processes in the nature include the randomness in themselves. This randomness can be used in order to generate long unpredictable key sequences needed by stream cipher schemes. Mathematical models which describe such physical phenomena are probability models.

Since a 2D Ising spin lattice presents a random series of disordered spin configurations, the main idea in the ISEA algorithm is to use this series of configurations to encrypt data. The associated probability model is implemented on a CA with determinist reversible rules.

Numerical simulations are powerful tools to simulate phase transitions on statistical systems. Monte Carlo and molecular dynamics represent two complementary schemes for such simulations. A microcanonical Monte Carlo (MMC) [9] method represents a simulation algorithm interpolating between the Monte Carlo and molecular dynamics techniques. The MMC method consists of taking a random walk on a surface of constant energy. This random walk will generate successive configurations of the statistical system.

In order to ensure a fast and secure encryption of sensitive data through the PHCA, we propose to use these configurations. The PHCA has to perform the three following actions:

(1) storing the successive rows of data to be encrypted coming from the south-input bus CMS and shifting these data through the PE array up to the north-output bus CMN;
(2) ensuring a permanent random walk by executing the microcanonical Monte Carlo local rules;
(3) combining the data flow and the lattice statistical system configurations in order to encrypt the data.

3.2. Microcanonical Monte Carlo method

The statistical system to simulate is the 2D Ising model. Let us consider a square lattice of $M \times N$ sites with one spin S at each site. The spins may be up or down. With the MMC method, each site i is also provided with a reservoir containing an energy E_{ri}.

Two kinds of energies are involved in this model. The first one is a magnetic interaction energy; for a link (i, j), between two neighbor sites i and j, the magnetic energy is expressed by

$$m_{ij} = S_i \text{ xor } S_j. \tag{2}$$

So $m_{ij} = 0$ if the two considered spins point towards the same direction, otherwise $m_{ij} = 1$. The second kind of energy is called "reservoir" energy; it is the sum of all the private site reservoir energies E_{ri}.

At each time step, all the spins try to flip. Nevertheless, the flip has a cost in terms of magnetic energy. Indeed, if the spin S_i of site i flips, the magnetic energy varies by

$$\Delta M_i = -2\left[\Sigma_j (S_i \text{ xor } S_j) - 2\right], \tag{3}$$

where j refers to the four neighbors of the site i. An illustration is given in Figure 4. The local rule is that if ΔM_i is smaller than or equal to E_{ri}, the spin S_i flips. Otherwise, S_i does not change. In other words, if the site has enough reservoir energy to pay the flip, then the spin can flip effectively.

Table 1: Instruction set.

Description	Code C12 · · · C0	Description	Code C12 · · · C0
NOP	XXXXXXXXXXX00	EW← NS	XXXXX100XXXXX
CM← RAM	XXXXXXXXXXX01	EW← C	XXXXX101XXXXX
CM← CMS	XXXXXXXXXXX10	EW← 0	XXXXX110XXXXX
CM← 0	XXXXXXXXXXX11	NOP	XX000XXXXXXXX
NOP	XXXXXXXX000XX	C← RAM	XX001XXXXXXXX
NS← RAM	XXXXXXXX001XX	C← NS	XX010XXXXXXXX
NS← N	XXXXXXXX010XX	C← EW	XX011XXXXXXXX
NS← S	XXXXXXXX011XX	C← CY	XX100XXXXXXXX
NS← EW	XXXXXXXX100XX	C← BW	XX101XXXXXXXX
NS← C	XXXXXXXX101XX	C← 0	XX110XXXXXXXX
NS← 0	XXXXXXXX110XX	C← 1	XX111XXXXXXXX
NOP	XXXXX000XXXXX	RAM← RAM	00XXXXXXXXXXX
EW← RAM	XXXXX001XXXXX	RAM← CM	01XXXXXXXXXXX
EW← E	XXXXX010XXXXX	RAM← C	10XXXXXXXXXXX
EW← W	XXXXX011XXXXX	RAM← SM	11XXXXXXXXXXX

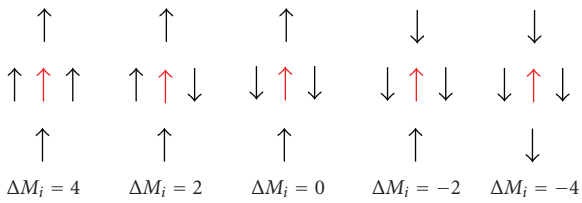


Figure 4: Magnetic energy costs ΔM_i for the central site i spin flip.

4. ENCRYPTION PROCESS WITH ISEA

The three actions enumerated in Section 3.1 are quite suitable for cellular automata. Each PE of the PHCA updates one site. A spin-up is coded 0; a spin-down is coded 1. The reservoir energy is 4-bit coded. So two arrays of 1-bit values coexist simultaneously in the PHCA: the array of spins is updated at each time-step and the array of data shifts to the north. In order to encrypt the data, each PE xors the bit of data and the bit of spin.

During the initialization phase, the programmer has to choose the initial spin configuration and to distribute the reservoir energy. Then he has to choose the number of iterations of the MMC rules to compute before xoring the spin bit and the data bit. These choices constitute the key S_k of the encryption process. This cryptography algorithm is symmetric and the key is secret. Let us detail how to store the initial values in the PE array and how to manage iterations of the MMC method on the spin array.

(i) During the loading phase, the spin and the reservoir energy values are presented to the southern side of the PE array through the CMS data bus (see Figure 1). Then these data are shifted to the north. When all the PEs receive the first bit to store through their CMS input (see Figure 3), they store it in their CM register and then transfer it from CM to the RAM. This process is iterated, in bit-serial mode, till all the initial values are stored in the array.

(ii) During the computation phase, according to the MMC method, the operations to be performed are rather simple: xor, shift, addition, subtraction.

5. RESULTS AND DISCUSSION

The efficiency of the ISEA algorithm and the results of our first FPGA implementation of PHCA are presented thereafter.

5.1. Application to image en-/decryption

An application example of our hardware CA programmed with the ISEA algorithm is the color image encryption/decryption system shown in Figure 5. The clear original 640 × 853 picture is given in Figure 6(a). Each pixel is coded with 3 bytes (red, green, and blue) so each line of this image can be divided into 120 128-bit words to fit with the PE array horizontal size.

In order to ensure a secure data exchange, both the sender and the receiver need a PHCA with, for instance, 128 × 128 PEs. The operations required to encrypt and decrypt are detailed thereafter.

(1) The sender imposes the initial spin values S and distributes the total reservoir energy R. Then he programs the PHCA in order to perform U initial spin lattice configuration updates. In the example leading to Figure 6(b) results, the initial configuration of the Ising lattice was all the spins pointing towards down. For the distribution of the reservoir energy R, an energy of 2 was distributed to each cell except for 3 cells (called "hot cells") which received an energy of 4. Hot cell coordinates constitute the information R'. Moreover, 2000 initial spin lattice configuration updates were carried on. The concatenation of S, R', and U constitutes

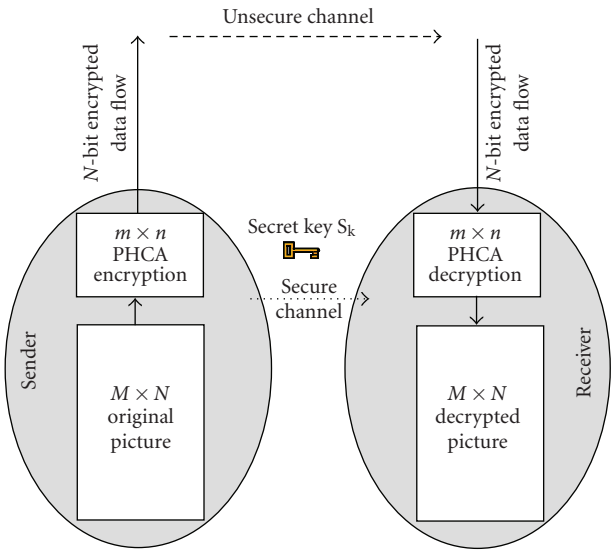


Figure 5: Complete PHCA-based encryption/decryption system.

(a) (b)

Figure 6: (a) Original picture. (b) Encrypted picture.

the secret key S_k which must be transmitted to the receiver through a secure channel.

(2) The sender introduces the clear image through the south side of its PHCA, one word at a time. These data shift to the north and after each shift step, they are xored with the spin lattice configuration. The resulting encrypted image is shown in Figure 6(b). One can notice that the initial picture is completely scrambled at this step.

(3) The receiver gets the secret key S_k through a secure channel. Then he initializes its PHCA with S and R and programs it to perform U spin lattice configuration updates.

(4) The receiver introduces the encrypted message into the south side of its PHCA. These operations allow to exactly recover the initial data picture at the north side of the receiver PHCA.

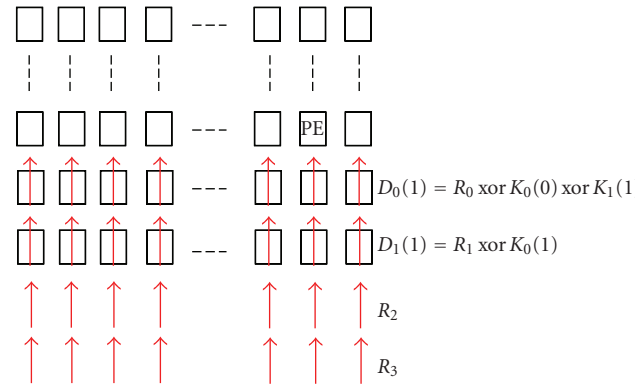


Figure 7: Random key sequence generation at time $t = 1$.

5.2. Test of randomness

The pixels in Figure 6(b) seem to be randomly distributed. In order to test the quality of the two-dimensional CA random number generator (RNG) produced by the ISEA algorithm, the Diehard tests [11] were used. The input file for the Diehard test program is a binary file resulting from the concatenation of the random keys C_i generated by the Ising spin configuration.

How can we generate the long unpredictable key sequences C_i necessary for the cipher? Let K_i be the concatenation of all the spin values of a whole lattice row. At time $t = 0$, at the beginning of the encryption process illustrated in Figure 7, the first row R_0 of clear data is introduced through the south of the PE array and xored with $K_0(0)$. Then at time $t = 1$, the result $D_0(0)$ is shifted to the north and xored with $K_1(1)$, and so on.

At time $t = t_m$, the first encrypted data row $D_0(t_m)$ available at the north of the PE array is given as follows:

$$D_0(t_m) = R_0 \text{ xor } C(t_m), \tag{4}$$

where $C(t_m) = K_0(0)$ xor $K_1(1)$ xor,..., xor $K_m(t_m)$ is the first encryption key of the random sequence.

The battery of the 17 Diehard tests was applied on a sequence of 70 M keys, $C(t_m), C(t_m+1), \ldots, C(t_m+a)$. Figure 8 gives the proportion of passed Diehard tests versus the total reservoir energy R. On one side, for low R, the spins are "frozen" because the sites have no sufficient E_r to flip their spin. On the other side, for high R, all the spins flip simultaneously. These results show that R must be chosen between 1000 and 3000 to obtain high-quality randomness.

Other energy-band values are found depending on the S and U parameter values, on the way of distributing the initial reservoir energy, and on the lattice size. A deeper investigation on the ISEA algorithm efficiency and a comparison with other RNGs are actually in progress.

One can notice other advantages of PHCA and ISEA. First, concerning PHCA, the permanent exchanges between neighbor sites introduce a constant noise useful against the attacks based on power analysis. Then, concerning the ISEA algorithm, the fact that the MMC method conserves the total

Table 2: Encryption cores resource and performance.

Core	Std Helion	PHCA	ISM
Application	AES	ISEA 32 × 32 sites	ISEA 32 × 32 sites
Technology	Spartan 3–5	Spartan xc3s5000	Spartan xc3s5000
Logic resource	251 slices 3 block rams	32589 slices	14148 slices
Max clock frequency	151 MHz	161 MHz	132 MHz
Max data rate	402 Mbps	16 Mbps	2110 Mbps
Programmable	No (dedicated to AES)	Yes (with 1D or 2D AC rules)	No (dedicated to ISEA)

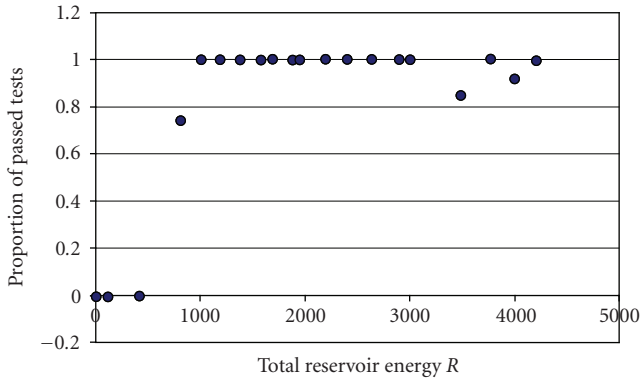


Figure 8: Diehard test results.

energy of the spin system can be used as a test to reveal some hardware anomaly.

5.3. PHCA implementation performances

Since the PHCA is programmable and has configurable interconnect switches, it is suitable for 1D or 2D CA rules. So it can constitute a powerful tool to elaborate and test cellular automata rules. It can also be used as a multialgorithm CA.

We implemented an Xilinx FPGA xc3s5000, a first version of the PHCA containing a 32 × 32 PE array. PHCA implementation results are reported in Table 2. The clock frequency of the PHCA is 161 MHz. The 309 PE instructions are necessary to update a spin array configuration. The throughput of the en-/decrypted data stream is 16.7 Mbps. The 1-bit architecture of the ALU and of the registers constitutes a throughput limitation. For instance, 25 clock cycles are necessary to perform an addition of two bytes.

In order to perform a faster en-/decryption process, we designed a machine called Ising spin machine (ISM). It is dedicated to the implementation of the ISEA algorithm with 32 × 32 sites. ISM performs one update every 2 clock-cycles. Targeting also a Spartan-3 device, the throughput is 2 Gbps (see Table 2). ISM goes 125 times faster than PHCA and uses twice less resources; in return it is not a multialgorithm CA.

Table 2 presents also the implementation result of the core Helion [12], a commercial implementation of the well-known secret-key AES algorithm [13]. Helion data rate performance is 5 times slower than ISM and 25 times faster than PHCA. However, Helion is only dedicated to the AES algorithm.

6. CONCLUSION

The IP-core PHCA proposed in this work has a fine grained SIMD architecture very suitable to implement cellular automata-based algorithms. The heart of the structure is a PE array with reconfigurable-side links allowing to get a cyclic 2D CA or an acyclic 1D CA.

An application of the 2D CA configuration of PHCA to data flow encryption/decryption using the proposed ISEA algorithm has been presented here and leads to two kinds of conclusions concerning the hardware and the algorithm, respectively.

Concerning the hardware, the implementation of ISEA on PHCA leads to a data rate of 16 Mbps which is 125 times lower than the performance obtained from a core that we designed to be dedicated to ISEA. Nevertheless, PHCA has the important advantage to be programmable. So it can be used as an experimentation platform to test the algorithms efficiency and their implementation on a 2D cell array architecture. If the test is successful, in a second step, macros dedicated to the chosen algorithms can be designed to improve the performances and get smaller area.

Concerning the ISEA algorithm, we saw that ISEA allows to code a data stream using a random walk on a surface of constant energy generated by the MMC method. The high quality of the random number generated by ISEA has been tested by the battery of the 17 Diehard tests.

The random numbers generated by ISEA are used as the long and unpredictable keys needed by the data stream encryption/decryption as presented in this work. Moreover, the elaboration of an experimentation platform for stream ciphers comparison is actually in progress. It uses two Virtex-II FPGA boards (one for encryption and one for decryption). Postimplementation Xilinx ISE simulation results already show that the encryption method using ISEA runs faster than the ciphers using the algorithms presented in [3, 4, 14].

Otherwise, the security of the whole encryption/decryption system compared to secret key security standards is also under investigation.

REFERENCES

[1] P. Sarkar, "A brief history of cellular automata," *ACM Computing Surveys*, vol. 32, no. 1, pp. 80–107, 2000.

[2] F. Bagnoli and A. Francescato, "A cellular automata machine," in *Cellular Automata and Modeling of Complex Physical*

Systems, P. Manneville, N. Boccara, G. Y. Vichniac, and R. Bidaux, Eds., p. 312, Springer, Berlin, Germany, 1990.

[3] M. Tomassini, M. Sipper, and M. Perrenoud, "On the generation of high-quality random numbers by two-dimensional cellular automata," *IEEE Transactions on Computers*, vol. 49, no. 10, pp. 1146–1151, 2000.

[4] R.-J. Chen, Y.-T. Lai, and J.-L. Lai, "Architecture design and VLSI hardware implementation of image encryption/decryption system using re-configurable 2-D Von Neumann cellular automata," in *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS '06)*, pp. 153–156, Island of Kos, Greece, May 2006.

[5] NCR GAPP Application Notes, *NCR Corporation*, Dayton, USA, 1985.

[6] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, Boca Raton, Fla, USA, 1997.

[7] C. F. Baillie, *Lattice Spin Models and New Algorithms: A Review of Monte Carlo Computer Simulations*, World Scientific, River Edge, NJ, USA, 1990.

[8] M. Creutz, "Microcanonical Monte Carlo simulation," *Physical Review Letters*, vol. 50, no. 19, pp. 1411–1414, 1983.

[9] M. Creutz, "Deterministic Ising dynamics," *Annals of Physics*, vol. 167, no. 1, pp. 62–72, 1986.

[10] T. Toffoli and N. Margulus, "Programmable matter: concepts and realization," *Physica D*, vol. 47, no. 1-2, pp. 263–272, 1991.

[11] G. Marsaglia, "Diehard," 1998, http://www.stat.fsu.edu/pub/diehard/.

[12] http://www.heliontech.com/aes.htm.

[13] J. Daemen and V. Rijmen, "AES Proposal: Rijndael," September 1999, http://www.esat.kuleuven.ac.be/.

[14] M. D. Galanis, P. Kitsos, G. Kostopoulos, N. Sklavos, O. Koufopavlou, and C. E. Goutis, "Comparison of the hardware architectures and FPGA implementations of stream ciphers," in *Proceedings of the 11th IEEE International Conference on Electronics, Circuits and Systems (ICECS '04)*, pp. 571–574, Tel-Aviv, Israel, December 2004.