13

Cryptography in Quantum Cellular Automata

Mohammad Amin Amiri[1], Sattar Mirzakuchaki[2] and Mojdeh Mahdavi[3]
[1,2]*Iran University of Science and Technology*
[3]*Islamic Azad University, Shahr-e-Qods Branch*
Islamic Republic of Iran

1. Introduction

During past several decades, the microelectronics industry has improved the integration, the power consumption, and the speed of integrated circuits by means of reducing the feature size of transistors. But it seems that even by decreasing the transistor sizes, some problems such as power consumption cannot be ignored. QCA which was first introduced by Lent et al. (Lent et al., 1993) represents an emerging technology at the nano technology level.
Utilizing the QCA technology for implementing logic circuits is one of the approaches which in addition to increasing the clock frequency of these circuits to tera-hertz frequencies and decreasing the size of logic circuits to nanoscale, decreases the power consumption of these circuits (Lent et al., 1993; Tougaw & Lent, 1996). QCA cells have quantum dots in which the position of electrons will determine the binary levels of 0 and 1. This is the most noticeable feature of QCA against conventional logic in which logical states are stored by means of voltage levels.
Serpent block cipher was a finalist candidate of Advanced Encryption Standard (AES). This cryptographic algorithm has 32 rounds with an 128-bit block size and a 256-bit key size. This cryptographic algorithm consists of an initial permutation IP, 32 rounds, and a final permutation FP. Each round involves a key mixing operation, a pass through S-boxes, and a linear transformation. In the last round, the linear transformation is replaced by an additional key mixing operation(Anderson et al., 1998).
As an application of QCA technology, we have implemented the Serpent block cipher. Simulation results of this implementation are obtained from QCADesigner v2.0.3 software. QCADesigner is developed by the ATIPS lab at the University of Calgary in Canada. QCADesigner v2.0.3 has different simulation engines. Throughout this paper, the coherence vector simulation engine is used due to its accurate and detailed evaluation of QCA.
The remainder of this chapter is organized as follows. In Section 2, a brief review of QCA is presented. In Section 3, Serpent block cipher and its important modules are discussed. The implementation of the Serpent block cipher by means of the implementation of its modules is presented in Section 4. Section 5 concludes this chapter.

2. Quantum Cellular Automata

Each cell in Quantum Cellular Automata is composed of four quantum dots, as schematically shown in Fig. 1. The quantum dots are schematically shown as open circles. Each cell contains two electrons which are schematically shown as solid dots. The electrons are allowed to

jump between the various quantum dots in a cell by the mechanism of quantum mechanical tunneling but they are not permitted to tunnel between two individual cells. The barrier height between each two individual cells is supposed to be high enough to completely block intercellular tunneling. If the electrons are left alone, they will meet the arrangement

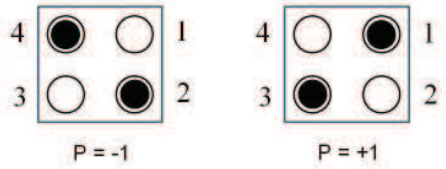

Fig. 1. QCA cell and its ground states

corresponding to the physical ground state of the cell. It is clear that the Coulombic force between electrons will make them occupy different dots. By these concepts, the ground states of a cell will be two basic arrangements with electrons at opposite corners, as shown in Fig. 1. The positions of the electrons are also illustrated in this figure.

Coulombic interaction between electrons in different cells will control their Coupling. Fig. 2

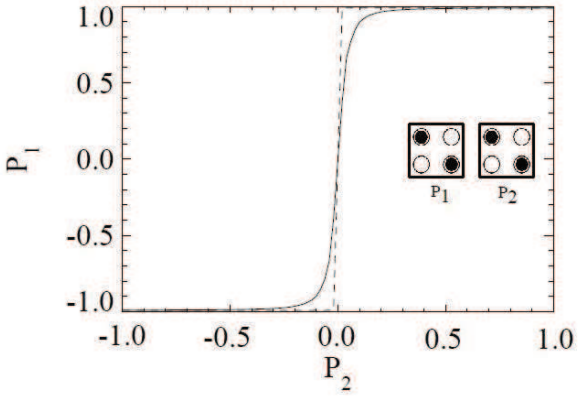

Fig. 2. Coupling of QCA cells

illustrates how one cell is affected by the state of its neighbor cell (Tougaw et al., 1993). This figure shows the two cells where the polarization of cell 1 is affected by the polarization of cell 2. The polarization of cell 2 (P2) is assumed to be fixed at a given value and this polarization will influence the cell 1, thus determining its polarization. The non-linear nature of the cell-cell coupling is a result that can be drawn here. Cell 1 is almost completely polarized in presence of cell 2 which might be partially and not completely polarized (Tougaw & Lent, 1996)(Tougaw et al., 1993). Utilizing the physical interactions between cells, basic Boolean logic functions can be realized. The elementary logic gates in QCA are the Majority gate and the Inverter gate which are illustrated in Fig. 4(a) and Fig. 3, respectively. The Majority gate can be realized by only 5 QCA cells (Tougaw & Lent, 1994). The logic AND function can be implemented by a Majority gate by setting one of its inputs permanently to 0 and the logic OR function can be implemented by a Majority gate by setting one of its inputs permanently to 1. Synchronization of the information flow throughout the QCA circuits and the direction of information flow in these circuits are provided by QCA clocking mechanism. The required power for the

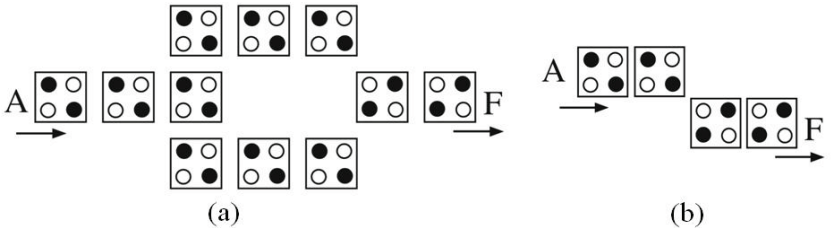

Fig. 3. (a) Redundant inverter gate and (b) Inverter gate

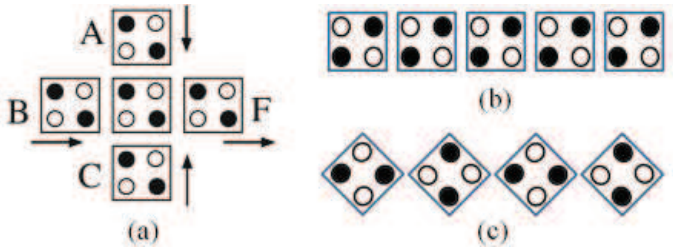

Fig. 4. (a) Majority logic gate, (b) Binary wire, and (c) Inverter chain

operation of QCA circuit is provided by QCA clocking mechanism too. More precisely, the QCA clocking mechanism is used to control the tunneling barrier height in QCA cells. The electrons are trapped in their positions, when the clock is low and they cannot tunnel to other dots, therefore latching the cell (Hold phase). This condition is caused by the intracellular barriers which are held at their maximum height. The cell goes to the null polarization state (Relax phase) when the clock signal is high. This condition is caused by the intracellular barriers which are held at their minimum height. Between these two conditions, the cells are either switching or releasing.

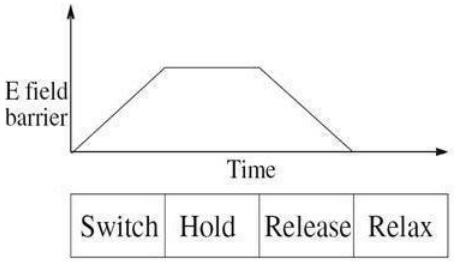

Fig. 5. Barrier hight in four phases of clock

Fig. 5 illustrates the barrier height in four phases of the QCA clock. Each cell in an individual clocking zone is connected to one of the four available phases of the QCA clock which is demonstrated in Fig. 6. Each QCA cell is synchronously latched and unlatched with the changing of the clock signal and therefore the information is distributed throughout the circuits (Hennessy & Lent, 2001)(Amiri et al., 2008)(Kim et al., 2007)(Vankamamidi et al., 2008).

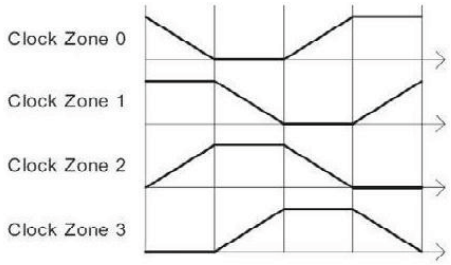

Fig. 6. QCA clock zones

3. Serpent block cipher

Serpent is a 32-round substitution permutation network (SPN) operating on four 32-bit words, thus having a block size of 128 bits (Anderson et al., 1998; 2006). Serpent encrypts a 128-bit plaintext to a 128-bit ciphertext in 32 rounds with 33 subkeys. The user key length is assumed to be variable but in the proposal, it is fixed to be 128, 192 or 256 bits. It should be mentioned that the short keys with less than 256 bits are mapped to 256 bits keys by appending one '1' bit to the MSB end followed by as many '0' bits as required to produce 256 bits. The cipher consists of an initial permutation IP, 32 rounds, and a final permutation FP. Each round involves a key mixing operation, a pass through S-boxes, and a linear transformation. In the last round, the linear transformation is replaced by an additional key mixing operation(Anderson et al., 1998).

3.1 Key mixing

At each round, a 128-bit subkey K_i is exclusively ORed with the current intermediate data B_i.

3.2 The S-boxes

Serpent has 8 individual 4×4 S-Boxes which repeat every 8 round. Every 128-bit input of the S-Box will be divided to 32 blocks of 4-bits and every block will be applied to a 4×4 S-Box. The outputs of these 32 S-Boxes will be concatenated again together to perform a 128-bit block.

3.3 Linear transform

The 128-bit output of S-Box will be divided to four 32-bits and these words are linearly mixed by some shift, rotate and XOR operations. A complete round of Serpent is described as:

$X_0, X_1, X_2, X_3 := S_i(B_i \oplus K_i)$
$X_0 := X_0 <<< 13$
$X_2 := X_2 <<< 3$
$X_1 := X_1 \oplus X_0 \oplus X_2$
$X_3 := X_3 \oplus X_2 \oplus (X_0 << 3)$
$X_1 := X_1 <<< 1$
$X_3 := X_3 <<< 7$
$X_0 := X_0 \oplus X_1 \oplus X_3$
$X_2 := X_2 \oplus X_3 \oplus (X_1 << 7)$
$X_0 := X_0 <<< 5$
$X_2 := X_2 <<< 22$
$B_{i+1} := X_0, X_1, X_2, X_3$

Where <<< means Rotation and << means Shift. In the last round, this linear transform is replaced by an additional key mixing:
$B_{32} := S_7(B_{31} \oplus K_{31}) \oplus K_{32}$

4. QCA implementation

In this work, each QCA cell is assumed to have the width and length of 18 nm like previous works such as (Cho & Swartzlander, 2007)(Cho & Swartzlander, 2009). The neighbor cells have a center to center distance of 20 nm. Implementation of initial and final permutation which are just bit reordering functions (Anderson et al., 1998), is accomplished by routing of inputs to desired outputs. Key mixing or XOR function is implemented by 3 majority gates. Fig. 7 illustrates the implementation of the key mixing function. This module has two inputs and one output. One of the inputs of this module comes from the round input and the other is the corresponding bit of the round key. This implementation has a latency of two clock periods, a complexity of 68 cells and an area of about 79200 nm^2.

The linear transform function has been first simplified and then it has been implemented.

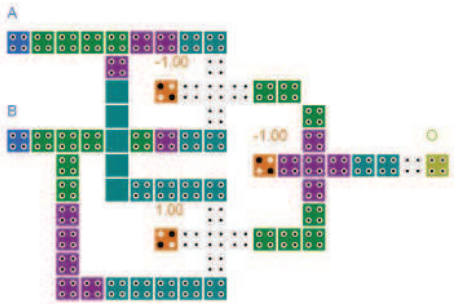

Fig. 7. Key mixing function or XOR

The simplified function was only constructed by XOR gates. As an example, the simplified 33rd and 34th bits of linear transform stage are as follows:

$$LTO(32) = LTI(14) \oplus LTI(33) \oplus LTI(68) \quad (1)$$

$$LTO(33) = LTI(15) \oplus LTI(34) \oplus LTI(69) \quad (2)$$

Fig. 8 illustrates the implementation of the 33rd output of linear transform. As illustrated in this figure, 2 XORs are applied to 3 inputs to perform the output.

This implementation has a latency of three clock periods, a complexity of 143 cells and an area of about 179200 nm^2.

4.1 Design and implementation of S-boxes

Serpent block cipher has 8 individual S-Boxes named S0 to S7. Here, the S0 substitution function is selected among Serpent's S-Boxes to be implemented. Other S-Boxes can be designed and implemented in such a manner. The S0 substitution function is illustrated in Table 1. Input and output values of this S-Box are shown in binary format and they have the length of 4 bits each.

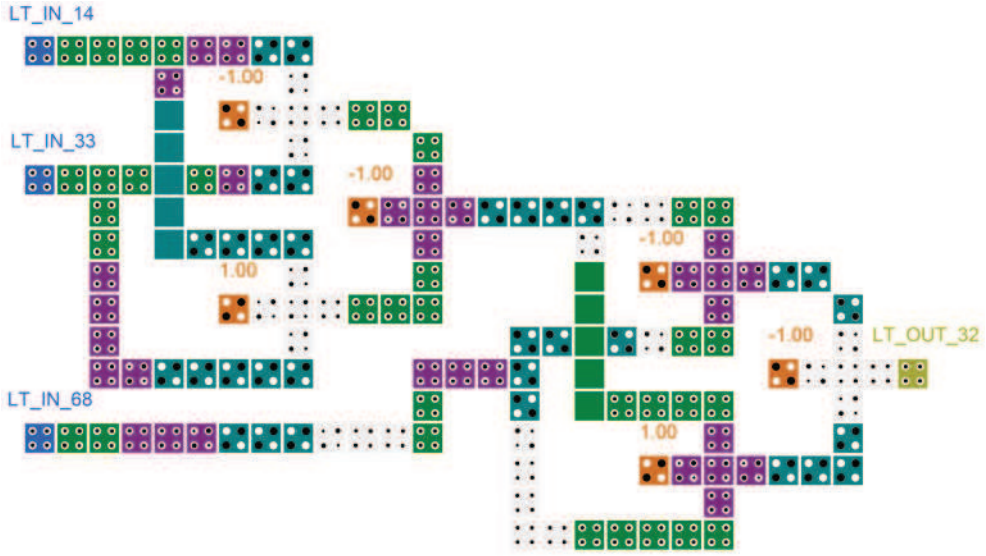

Fig. 8. Linear transform for 33rd output of linear transform

S-Box Input				S-Box Output			
S3	S2	S1	S0	O3	O2	O1	O0
0	0	0	0	0	0	1	1
0	0	0	1	1	0	0	0
0	0	1	0	1	1	1	1
0	0	1	1	0	0	0	1
0	1	0	0	1	0	1	0
0	1	0	1	0	1	1	0
0	1	1	0	0	1	0	1
0	1	1	1	1	0	1	1
1	0	0	0	1	1	1	0
1	0	0	1	1	1	0	1
1	0	1	0	0	1	0	0
1	0	1	1	0	0	1	0
1	1	0	0	0	1	1	1
1	1	0	1	0	0	0	0
1	1	1	0	1	0	0	1
1	1	1	1	1	1	0	0

Table 1. The S0 S-Box of Serpent Block Cipher

4.1.1 LUT-based design

In this method, a Look-Up-Table or memory is used to implement the S-Box. All the output bits of the S-Box *i.e.* O3, O2, O1, and O0 are implemented separately. The O0 design and implementation is discussed here. All the other output bits are implemented in such a manner. Just the memory contents are the differences between the implementation of these

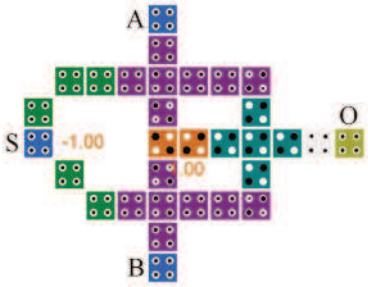

Fig. 9. QCA layout of 2 to 1 multiplexer

output bits. It means that for implementation of each output bit, the corresponding values should be stored in the memory. A 16 to 1 multiplexer is used to design a LUT-Based S-Box. The 16 to 1 multiplexer is composed of fifteen 2 to 1 multiplexers. A 2 to 1 multiplexer is illustrated in Fig. 9.
This novel multiplexer has a complexity of 31 QCA cells. When the "S" input of this multiplexer has the value of '0', the value of the "A" input will appear on the "O" output and when the "S" input has the value of '1', the value of the "B" input will appear on the "O" output.
The input bits of the S-Box are connected to the "S" input of the 16 to 1 multiplexer. The output values of the S-Box are fixed on the Data inputs of the 16 to 1 multiplexer. Using this structure, when the input value is applied to the "S" input of the multiplexer, the corresponding values will appear on the output port of the multiplexer after 10 clock periods of latency. This latency is the result of clocking which is used for data propagation throughout the circuit. It should be mentioned that after 10 clock periods of latency, the output value of the multiplexer will be valid in each clock period.

LUT-Based QCA implementation of the O0 bit of the S0 S-Box is illustrated in Fig. 10. Each output bit of the S0 S-Box is exhaustively simulated. Simulation results of the O0 bit is illustrated in Fig. 12. The results of LUT-Based QCA implementation of the S0 S-Box is discussed in Table 2. It can be seen that the Delay, Complexity and Area of all the output implementations are the same.

4.1.2 Logic-based design

In this method, the logic function of each output is used to implement it. All the output bits of the S0 S-Box *i.e.* O3, O2, O1, and O0 are implemented separately. Design and Implementation of O0 output bit is discussed here. All the other output bits are implemented in such a manner. The following logic functions are extracted from Table 1. Considering any output bit, the implementation of each term of logic function is composed of one, two, or three majority gates which are used as logic AND functions.

$$O3 = \bar{S_2}\bar{S_1}S_0 + \bar{S_3}\bar{S_2}S_1\bar{S_0} + \bar{S_3}S_2\bar{S_1}\bar{S_0} + S_2S_1S_0 + S_3S_2S_1 + S_3\bar{S_2}\bar{S_1} \quad (3)$$

$$O2 = \bar{S_3}S_1\bar{S_0} + \bar{S_3}S_2\bar{S_1}S_0 + S_3S_2S_1S_0 + S_3\bar{S_1}\bar{S_0} + S_3\bar{S_2}\bar{S_1} + S_3\bar{S_2}\bar{S_0} \quad (4)$$

$$O1 = \bar{S_1}\bar{S_0} + \bar{S_3}S_2S_0 + \bar{S_3}\bar{S_2}\bar{S_0} + S_3\bar{S_2}S_1S_0 \quad (5)$$

$$O0 = \bar{S_3}\bar{S_2}\bar{S_0} + S_3S_2\bar{S_0} + \bar{S_3}S_1 + S_3\bar{S_2}\bar{S_1}S_0 \quad (6)$$

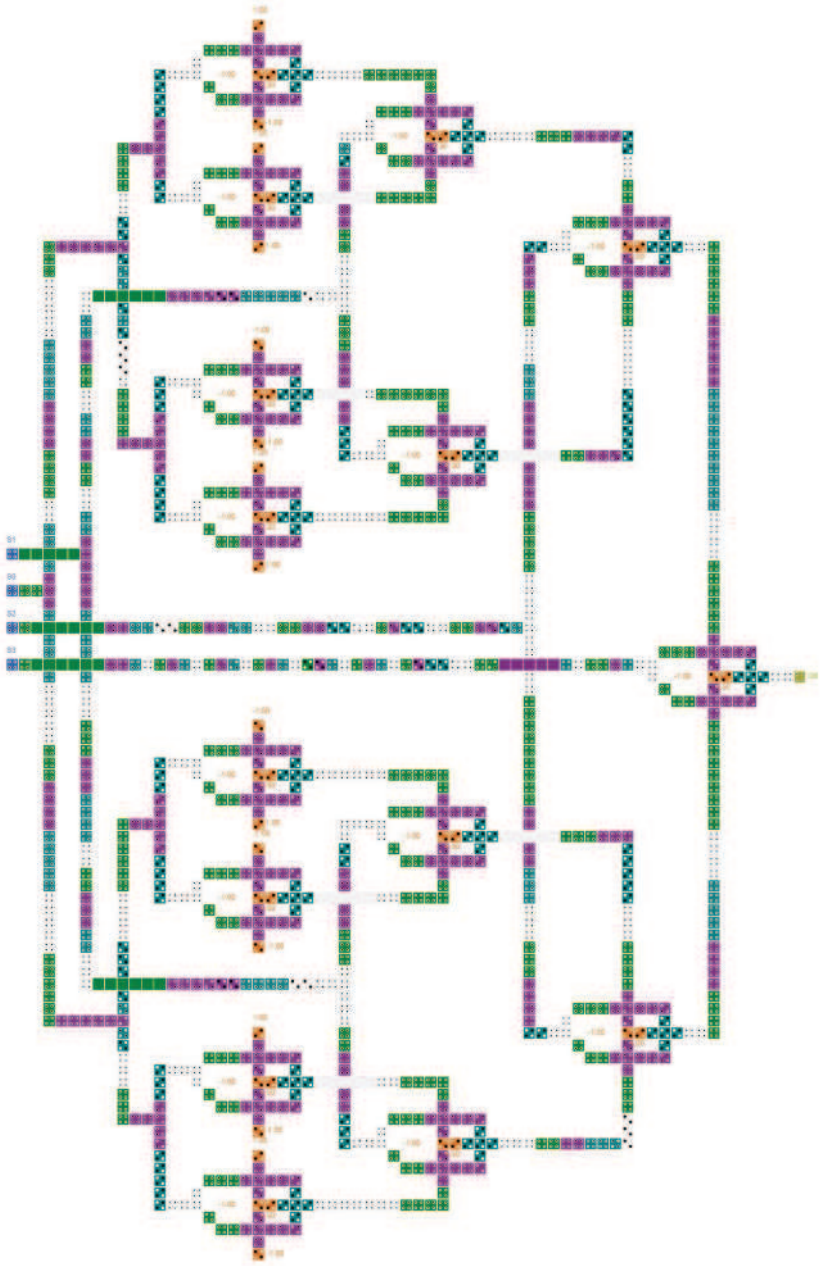

Fig. 10. LUT-Based QCA Implementation of O0 bit of S0 S-Box

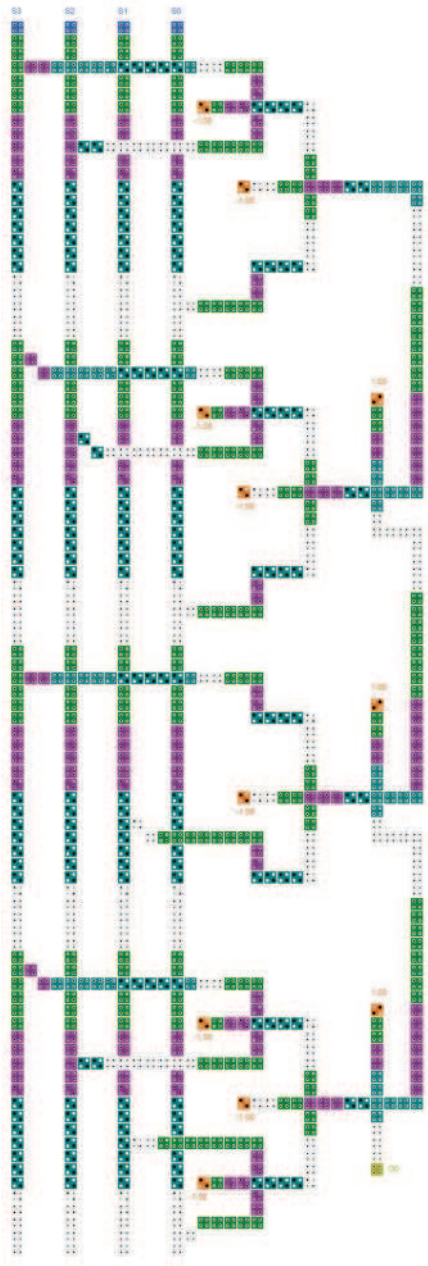

Fig. 11. Logic-Based QCA Implementation of O0 bit of S0 S-Box

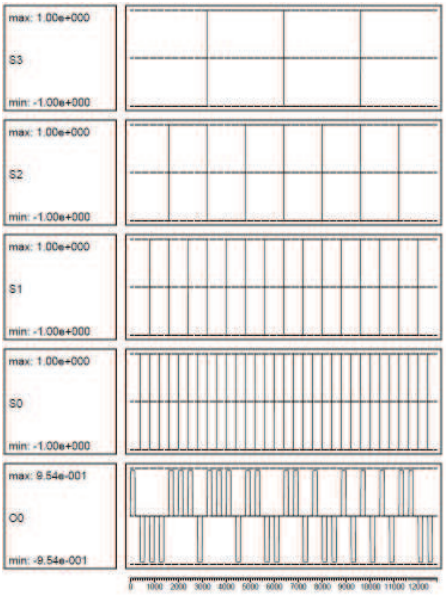

Fig. 12. Simulation Result of LUT-Based S-Box

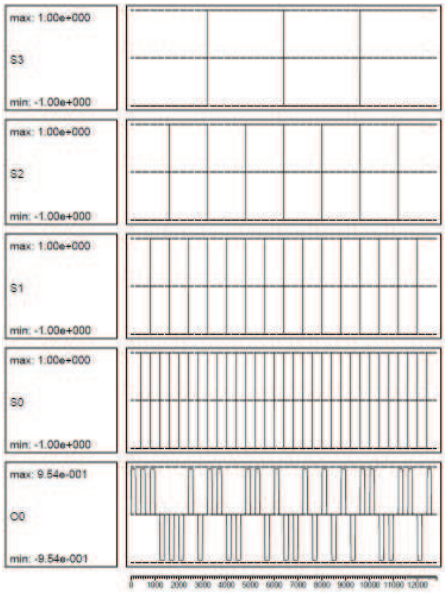

Fig. 13. Simulation Result of Logic-Based S-Box

	O3	O2	O1	O0	S0 S-Box
Complexity(Cells)	1200	1200	1200	1200	4800
Area(μm^2)	2.626	2.626	2.626	2.626	10.504
Delay(Clocks)	10	10	10	10	10

Table 2. Implementation Results of the LUT-Based S0 S-Box

	O3	O2	O1	O0	S0 S-Box
Complexity(Cells)	1204	1205	758	798	3965
Area(μm^2)	1.7236	1.7236	1.1532	1.1532	5.7536
Delay(Clocks)	8	8	6	6	8

Table 3. Implementation Results of the Logic-Based S0 S-Box

One majority gate is used for the terms which contain only two inputs, two majority gates are used for the terms which contain three inputs, and three majority gates are used for the terms which contain four inputs. The output of AND functions are also logically ORed to result the desired output.
Logic-Based QCA implementation of the O0 bit of the S0 S-Box is illustrated in Fig. 11. Each output bit of the S0 S-Box is exhaustively simulated. Simulation results of the output bits are illustrated in Fig. 13. The results of Logic-Based QCA implementation of the S0 S-Box is discussed in Table 3.
The O0 and O1 output values are valid after 6 clock periods of latency and the O2 and O3 output values are valid after 8 clock periods of latency. The maximum Delay among four output bits is considered to be the Delay of Logic-Based S-Box.

5. Conclusion

The implementation of the Serpent block cipher in Quantum Cellular Automata is investigated here. The main modules of this cryptographic algorithm are implemented in this technology and the implementation results are discussed. The two methods of S-Box design, *i.e.* LUT-Based and Logic-Based methods are inspected. The Serpent's S-Boxes are designed and simulated by these two methods. The implementation results show that the Logic-Based method has better advantages than LUT-Based method. Its Delay, Complexity and Area are less than the other method. A novel multiplexer is also introduced. This multiplexer is composed of only 31 QCA cells.

6. References

Lent, C. S.; Tougaw, P. D.; Porod, W. & Bernstein, G. H. (1993). Quantum Cellular Automata. *Nanotechnology*, Vol. 4, No. 1, (1993) page numbers 49-57.

Tougaw, P. D. & Lent, C. S. (1996). Dynamic Behavior of Quantum Cellular Automata. *Journal of Applied Physics*, Vol. 80, No. 8, (Oct. 1996) page numbers 4722-4735.

Anderson, R.; Biham, E. & Knudsen, L. (1998). Serpent: A proposal for the Advanced Encryption Standard, emphNIST AES Proposal, 1998.

Tougaw, P. D.; Lent, C. S. & Porod, W. (1993). Bistable Saturation in Coupled Quantum-dot Cells. emphJournal of Applied Physics, Vol. 74, No. 5, (Sep. 1993) page numbers 3558-3565.

Tougaw, P. D. & Lent, C. S. (1994). Logical Devices Implemented Using Quantum Cellular Automata. emphJournal of Applied Physics, Vol. 75, No. 3, (1994) page numbers 1818-1825.

Hennessy, K. & Lent, C. S. (2001). Clocking of Molecular Quantum-dot Cellular Automata. emphJournal of Vacuum Science and Technology B, Vol. 19, No. 5, (Sep. 2001) page numbers 1752-1755.

Lent, C. S. & Isaksen, B. (2003). Clocked Molecular Quantum-dot Cellular Automata emphIEEE Transaction on Electron Devices, Vol. 50, No. 9, (Sep. 2003).

Amiri, M. A.; Mahdavi, M. & Mirzakuchaki, S. (2008). QCA Implementation of a Mux-Based FPGA CLB, emphProceedings of International Conference On Nanoscience and Nanotechnology, pp. 141-144, Australia, Feb. 2008, Melbourne.

Kim, K.; Wu, K. & Karri, R. (2007). The Robust QCA Adder Designs Using Composable QCA Building Blocks. emphIEEE Transaction on Computer Aided Design of Integrated Circuits and Systems, Vol. 26, No. 1, (2007) page numbers 176-183.

Vankamamidi, V.; Ottavi, M. & Lombardi, F. (2008). Two-Dimensional Schemes for Clocking/Timing of QCA Circuits. emphIEEE Transaction on Computer Aided Design of Integrated Circuits and Systems, Vol. 27, No. 1, (2008) page numbers 34-44.

Anderson, R.; Biham, E. & Knudsen, L. (2006). Smart Card. Research and Applications, Berlin/Heidelberg, 2006, Springer.

Cho, H. & Swartzlander, E. E. (2007). Adder Designs and Analyses for Quantum-Dot Cellular Automata, emphIEEE Transaction on Nanotechnology, Vol. 6, No. 3, (2007) page numbers 374-383.

Cho, H. & Swartzlander, E. E. (2009). Adder and Multiplier Design in Quantum-Dot Cellular Automata. emphIEEE Transaction on Computer, Vol. 58, No. 6, (2009) page numbers 721-727.

Cellular Automata - Innovative Modelling for Science and Engineering

Edited by Dr. Alejandro Salcido

ISBN 978-953-307-172-5
Hard cover, 426 pages
Publisher InTech
Published online 11, April, 2011
Published in print edition April, 2011

Modelling and simulation are disciplines of major importance for science and engineering. There is no science without models, and simulation has nowadays become a very useful tool, sometimes unavoidable, for development of both science and engineering. The main attractive feature of cellular automata is that, in spite of their conceptual simplicity which allows an easiness of implementation for computer simulation, as a detailed and complete mathematical analysis in principle, they are able to exhibit a wide variety of amazingly complex behaviour. This feature of cellular automata has attracted the researchers' attention from a wide variety of divergent fields of the exact disciplines of science and engineering, but also of the social sciences, and sometimes beyond. The collective complex behaviour of numerous systems, which emerge from the interaction of a multitude of simple individuals, is being conveniently modelled and simulated with cellular automata for very different purposes. In this book, a number of innovative applications of cellular automata models in the fields of Quantum Computing, Materials Science, Cryptography and Coding, and Robotics and Image Processing are presented.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Mohammad Amin Amiri, Sattar Mirzakuchaki and Mojdeh Mahdavi (2011). Cryptography in Quantum Cellular Automata, Cellular Automata - Innovative Modelling for Science and Engineering, Dr. Alejandro Salcido (Ed.), ISBN: 978-953-307-172-5, InTech, Available from: http://www.intechopen.com/books/cellular-automata-innovative-modelling-for-science-and-engineering/cryptography-in-quantum-cellular-automata

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821