High Level Modeling of AES in QCA Technology

Mojdeh Mahdavi[1], Mohammad Amin Amiri[2]
1- Department of Electronics, Shahr-e-Qods Branch, Islamic Azad University, Tehran, Iran.
Email: m.mahdavi@qodsiau.ac.ir (Corresponding author)
2- Department of Electrical and Computer Engineering, Malek Ashtar University of Technology, Tehran, Iran.
Email: maamiri@mut.ac.ir

Received: July 2018 Revised: September 2018 Accepted: October 2018

ABSTRACT:
Lent has created QCA nanoscale devices by merging the cellular automata and quantum electronics. These devices are capable of achieving very high switching speeds and very low electrical power consumption. AES block cipher is now used worldwide. This algorithm is based on the Rijndael cipher which was submitted as a proposal to NIST during the AES selection process. The implementation of this cryptographic algorithm in QCA technology is presented in this paper. On the other hand, the QCADesigner software which is used to simulate QCA circuits is sensitive to the QCA cell count, inputs and outputs. It seems that by increasing the QCA cell count, inputs and outputs, the simulation time will increase and sometimes the simulation will be impossible. A higher level modeling of QCA circuits by VHDL hardware description language and simulation of these models by ModelSim software is presented in this paper to solve the mentioned problem. It is shown that the QCA implementation of the AES algorithm with key, input and output length of 128 bits is easily modeled and simulated in ModelSim software. The implementation results of various implementation methods are also compared in this paper for AES algorithm. It is illustrated that the QCA implementation of this algorithm is the most efficient implementation among existing methods.

KEYWORDS: Quantum Cellular Automata, Advanced Encryption Standard, Implementation, Modeling.

1. INTRODUCTION

Reducing the feature size of transistors has led to improvement of the integration and the speed of digital integrated circuits but some problems such as power consumption cannot be disregarded. Quantum Cellular Automata was first introduced by Lent et al. by means of merging the cellular automata and quantum electronics [1].

QCA represents an emerging technology at the nanotechnology level. Utilizing the QCA technology to implement logic integrated circuits is one of the approaches which in addition to reducing the power consumption and the size of logic circuits, increases the clock frequency of these circuits. QCA cells are composed of quantum dots, in which the position of electrons in the mentioned dots will determine the binary level of that cell.

The Advanced Encryption Standard (AES) specifies a FIPS-approved cryptographic algorithm that can be used to protect electronic data. The AES algorithm is a symmetric block cipher that can encrypt and decrypt information. The AES algorithm is capable of using cryptographic keys of 128, 192, and 256 bits to encrypt and decrypt data in blocks of 128 bits.

QCADesigner is developed by the ATIPS lab at the University of Calgary in Canada [2]. This software is used to simulate QCA circuits with coherence vector and bistable approximation engines. One of the main problems with this software and others is that they use the Landauer clocking and are unable to deal with multi types of clocking signals. M. Ottavi et al. used Verilog HDL to describe QCA devices, but he couldn't expand his library to include the dynamic behavior of QCA devices [3]. Tiago et al. introduced the QCA-LG that was a tool for the automatic layout generation of QCA combinational circuits [4]. This tool generates the QCA circuit of an HDL input.

Some previous works are introduced to implement cryptographic algorithms in QCA technology [5-7]. Their main focus is to implement the main blocks of these algorithms at QCA level and simulate them using QCADesigner. But in this paper, the whole algorithm is modeled, implemented and simulated by ModelSim software. An extended review of QCA technology is presented in some mentioned references [8-14].

The remainder of this paper is as follows. Section 2, investigates the modeling of QCA basic devices. In Section 3, the Advanced Encryption Standard, VHDL implementation and QCA implementation of this algorithm are presented. Finally, Section 4 will conclude this paper.

2. MODELING OF BASIC ELEMENTS

Higher abstraction level modeling of basic elements of QCA circuits is presented in this section. Using hardware description languages, these basic modules can be described.

VHDL is a hardware description language which is used in electronic design automation to describe various systems such as field-programmable gate arrays and integrated circuits. It can describe the behavior and structure of electronic systems, but is especially suited as a language to describe the digital electronic hardware designs. The clocking scheme modeling is also illustrated in this section.

2.1. Four Phases of Clock

Clock signals in QCA circuits are applied to four clock zones. Each zone's clock has a 90 degree phase difference with its neighbor zone's clock. The CLK0 corresponds to zone 0, the CLK1 corresponds to zone 1, the CLK2 corresponds to zone 2, and finally the CLK3 corresponds to zone 3.

2.2. Clock Zone Modeling

The ZONE_LATCH module is used for modeling of clock zones. This module has three inputs and one output. The INPUT and OUTPUT ports are used for data input and data output, the CLK port is used for clock input and the CLR port is used for module reset. Investigating the function of this module, if the CLK signal has the logic value of one, the INPUT will appear on the OUTPUT and otherwise, the OUTPUT will remain unchanged. When the CLR signal has the logic value of zero, the logic value of zero will appear on the OUTPUT port.

2.3. Binary Wire Modeling

The WIRE module models the binary wire with one clock delay. This module which is composed of four ZONE_LATCH modules has six inputs and one output. The INPUT and OUTPUT ports are used for data input and data output, the CLK0, CLK1, CLK2, and CLK3 ports are used for clock inputs and the CLR port is used for module reset.

Single clock delay generation between the INPUT signal and the OUTPUT signal is the function of this module. When the CLR signal has the logic value of zero, the logic value of zero will appear on the OUTPUT port of the module and its sub modules.

2.4. Inverter Gate Modeling

The inverter gate is modeled by the INVERTER module. This module has one input and one output. The INPUT and OUTPUT ports are used for data input and data output. The function of this module is to invert the logic value of INPUT signal to be appeared on the OUTPUT port.

2.5. Majority Gate Modeling

The MAJORITY module is used for modeling of majority gate. This module has three inputs and one output. The INPUT1, INPUT2, and INPUT3 ports are used for data input and the OUTPUT port is used for data output. Majority signal generation between the input signals is the function of this module. As mentioned before, the OR gate and the AND gate are implemented using this module.

3. AES IMPLEMENTATION

The implementation of the AES block cipher is discussed in this section. First, the AES algorithm is presented. Next, the VHDL implementation and the QCA Implementation will be investigated.

3.1. AES Block Cipher

The design principle of AES is a Substitution-permutation network which has a fixed block size of 128 bit and key size of 128, 192 and 256 bits with 10, 12 and 14 rounds of encryption respectively [15]. The last round of algorithm is different while all other rounds are same. Encryption and decryption operates on 16 byte of data as an array of 4 by 4 bytes named state matrix. The encryption rounds consist of four main steps which are, shift row, substitution of bytes, mix column and add round key. The AES steps are shown in Fig. 1.

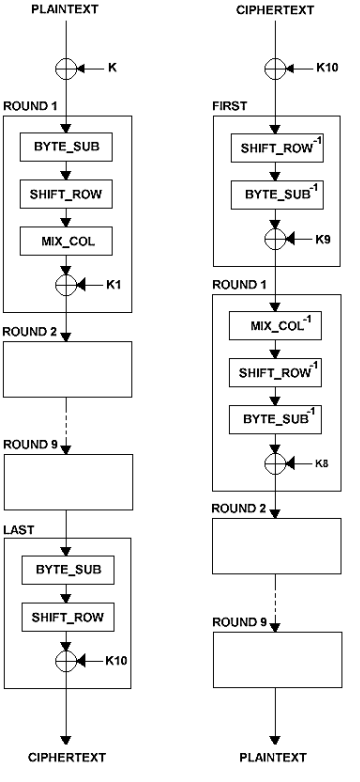

Fig. 1. AES Encryption and Decryption.

Shift row operation shifts byte of first, second and third rows of state matrix to left by an offset of 1, 2, and 3 respectively. Each byte of state matrix is replaced by corresponding SBox entry in substitution byte step. Mix column operation multiplies fixed matrix with state matrix in GF (2^8) while add round key operation adds state matrix with key in GF (2^8).

3.2. VHDL Implementation

The four previously mentioned modules, Substitution of bytes, Shift row, Mix Column and Round Key Addition are described by VHDL language. The round module and the algorithm module are also described and the unrolled architecture is used to implement the algorithm. Using a testbench in ModelSim software, the standard test vectors are used to simulate the algorithm module, as illustrated in fig. 2. The corresponding test vector values of INPUT, KEY, and the OUTPUT are presented below.
INPUT = "00010203040506070809 0a0b0c0d0e0f"
KEY = "00112233445566778899aabbccddeeff"
OUTPUT = "69c4e0d86a7b0430d8cdb78070b4c55a"

all byte substitution functions in a same way and also in a same appearance. Of course the Logic-based implementation method is more efficient but here the simplicity was preferred. Using the LUT-based method, each 256-to-1 multiplexer is composed of seventeen 16-to-1 multiplexers and each 16-to-1 multiplexer is composed of seventeen 2-to-1 multiplexers. Fig. 3 illustrates the schematic model of a 2-to-1 multiplexer. As illustrated in this figure, a 2-to-1 multiplexer has 8 inputs and one output. The main inputs are A_IN, B_IN and SEL ports and the output is OUTPUT port. Other inputs, the CLK0, CLK1, CLK2, and CLK3 ports are used for clock inputs and the CLR port is used for module reset. This module is composed of 3 MAJORITY modules, one INVERTER module and 7 ZONE_LATCH modules. The delay of this module is one clock period.

As mentioned above, each 16-to-1 multiplexer is composed of seventeen 2-to-1 multiplexers. A 16-to-1 multiplexer has 8 inputs and one output. The main inputs are the 16-bit DATA and the 4-bit SEL ports and the output is the OUTPUT port. Other inputs, the

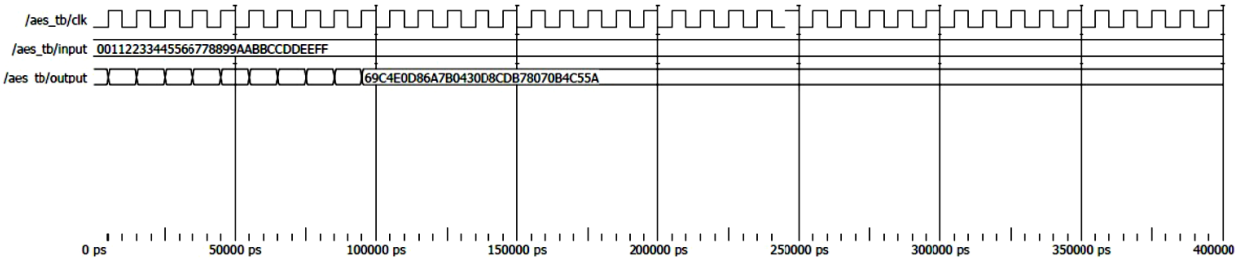

Fig. 2. Simulation result of AES.

3.3. QCA Implementation

The QCA implementation of the AES block cipher is presented in this section. According to the intrinsic latency and pipelined structure of QCA circuits and also simpler control circuits, the unrolled architecture is selected for QCA implementation of AES algorithm. The loop architecture will not be used here because of the same reasons. It means that when the data flow is controlled by the clock signal and each sub module has more than one clock latency, a partially unrolled architecture may be used instead of loop architecture to enhance the data throughput. The partially unrolled architecture is not used because of its complex control circuits to be implemented in QCA.

3.3.1 Byte Substitution Implementation

As explained in [6, 7], there are two methods for implementing the byte substitution functions. These two methods are Logic-based and LUT-based implementation methods. In this paper the LUT-based method is used to implement the mentioned function. This method has simpler structure and can implement

CLK0, CLK1, CLK2, and CLK3 ports are used for clock inputs and the CLR port is used for module reset. This module is composed of fifteen 2-to-1 multiplexer modules and 52 WIRE modules. The delay of this module is ten clock period. Eight 256-to-1 multiplexers are used to implement an 8 by 8 SBOX. Fig. 4 illustrates the schematic model of an 8 by 8 SBOX. The input value of each multiplexer is fixed on the SBOX values. When an input value appears on the SEL input of each multiplexer, the output value will appear after 16 clock periods.

3.3.2 Mix Column Implementation

The multiply-by-2 and the multiply-by-3 modules are designed and implemented. The schematic model of the multiply-by-3 module is illustrated in fig. 5. This module has a delay of four clock periods and is composed of a multiply-by-2 module, a wire with delay of 2 clock periods, and an 8 bit XOR2 module.

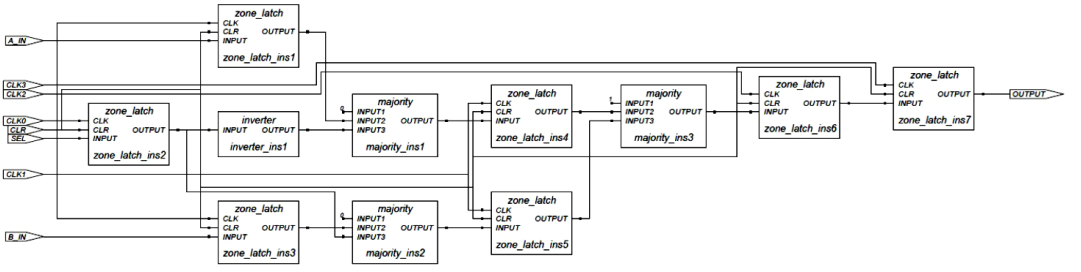

Fig. 3. Schematic Model of a 2-to-1 multiplexer.

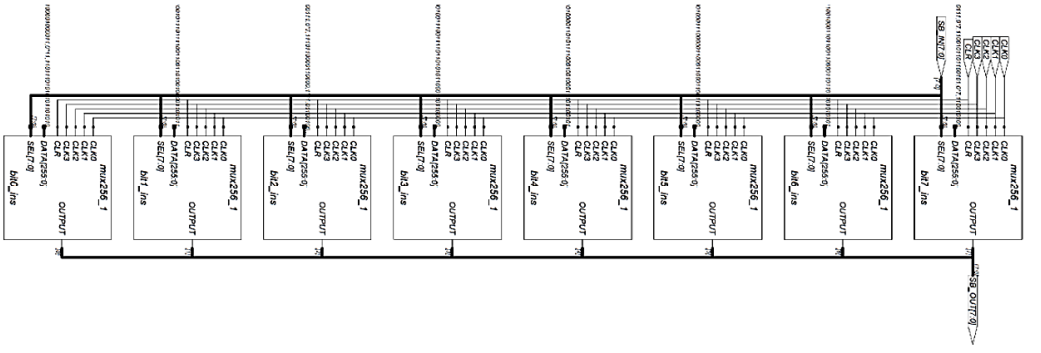

Fig. 4. Schematic Model of an 8 by 8 SBOX.

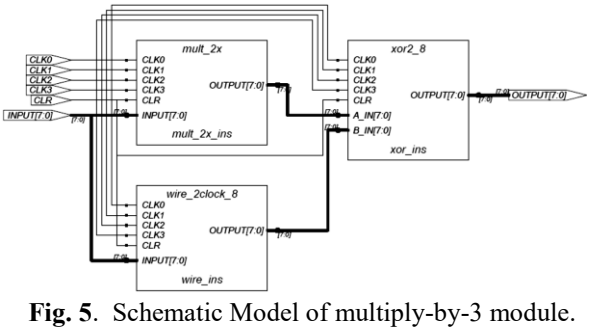

Fig. 5. Schematic Model of multiply-by-3 module.

The XOR2 module is illustrated in fig. 6. As illustrated in this figure, an XOR2 module has 7 inputs and one output. The main inputs are A_IN and B_IN ports and the output is OUTPUT port. Other inputs, the CLK0, CLK1, CLK2, and CLK3 ports are used for clock inputs and the CLR port is used for module reset. This module is composed of 3 MAJORITY modules, one INVERTER module and 16 ZONE_LATCH modules. The delay of this module is two clock periods. The mat-mult module is composed of a multiply-by-2 module, a multiply-by-3 module and an 8 bit XOR2 module and has a delay of eight clock periods. The mix-column module is composed of four mat-mult modules and has a delay of eight clock periods. Fig. 7 illustrates the schematic model of the mix-column module.

3.3.3 Round Implementation

The round module is composed of a sbox module, a shift-row module, a mix-column module and a xor module. The schematic model of this module is illustrated in fig. 8. This module has a delay of twenty six clock periods.

The final-round module is almost like the round module but the mix-column module is absent in the final-round module. This module has a delay of eighteen clock periods.

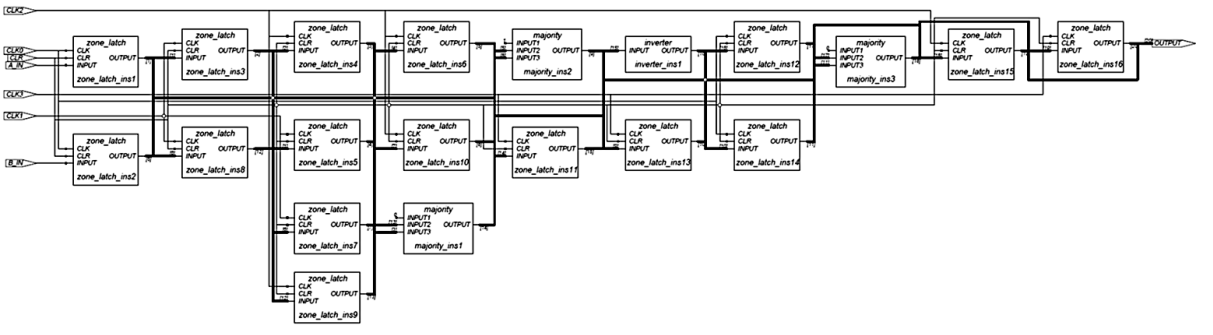

Fig. 6. Schematic Model of XOR2 module.

3.3.4 Algorithm Implementation

The algorithm module is composed of nine round modules, a final-round module and a 128 bit xor module. The algorithm module has a delay of 254 clock periods. Using a testbench in ModelSim software, the standard test vectors are used to simulate the algorithm module. Here, just the simulation result of the first round is illustrated in fig. 9. Having the below INPUT and SUBKEY for the first round, the OUTPUT of this round will obtain the mentioned value. Also, the twenty six clock periods delay of the round module can be seen in this figure.

INPUT = "000102030405060708090a0b0c0d0e0f"
SUBKEY = "d6aa74fdd2af72fadaa678f1d6ab76fe"
OUTPUT = "89d810e8855ace682d1843d8cb128fe4"

3.4. Comparison of Implementations

For the comparison of the implementations, parameters such as frequency, consumed area, initial delay, throughput and throughput/area are investigated. For the VHDL implementation, the Spartan3 family FPGA of the Xilinx Corporation is evaluated and for these FPGAs, each cell's area is about 2100 square micrometers [11].

According to the QCADesigner software, each QCA cell's area is about 20*20 square nanometers. The working frequency of QCA circuits is assumed to be 100 GHz as previously used in mentioned references [16, 17]. The functional verification of these two proposed modules is accomplished by standard test vectors and by means of the ModelSim software.

The simulation results, verify that the modules are well implemented and work correctly. The initial delay of the VHDL implementation is 10 clock periods but for QCA implementation, this delay is 254 clock periods. According to the higher frequency of QCA circuits, the initial delay of QCA implementation is much lower.

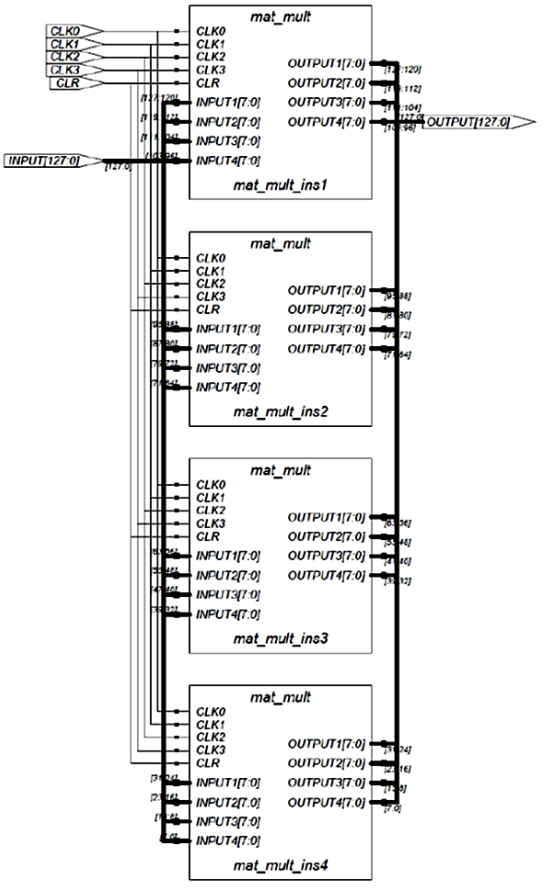

Fig. 7. Schematic Model of mix-column module.

The throughput (Thr) of the implementations is 128 bit per clock period and again according to higher

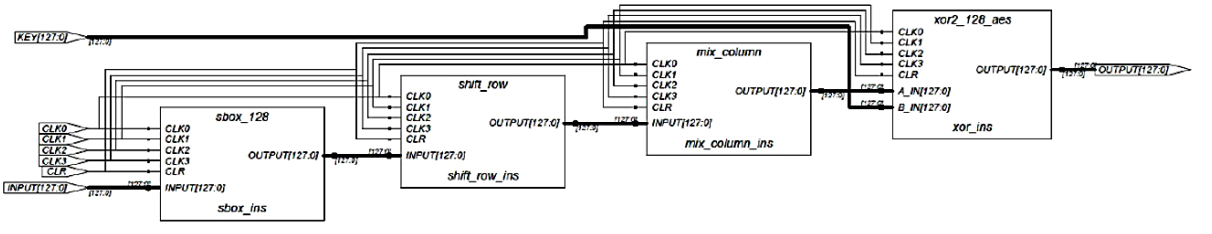

Fig. 8. Schematic Model of round module.

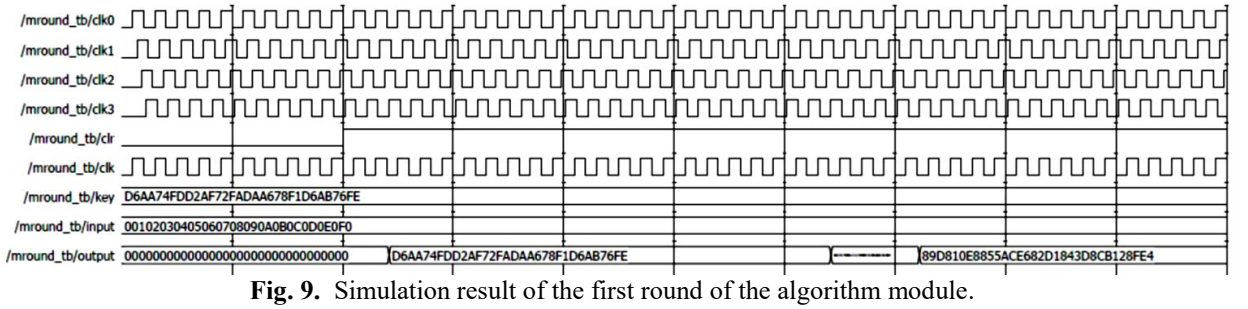

Fig. 9. Simulation result of the first round of the algorithm module.

frequency of QCA circuits, the throughput of QCA implementation is much higher.

The comparison results of these two methods of implementation and two ASIC implementations [18, 19] are illustrated in table 1. It is illustrated that the QCA implementation of AES algorithm is much more efficient than others types of implementations. The throughput/area (Thr/Area) of this method is almost one million times more than other methods.

Table 1. Comparison results of the implementation methods.

Imp. Method	Area	Freq.	Init. Delay	Thr	Thr/Area
Proposed HDL Imp.	20.3238 [mm^2]	92 [MHz]	110 [ns]	11.776 [Gbps]	579 [bps/μm^2]
Proposed QCA Imp.	0.08 [mm^2]	100 [GHz]	2.54 [ns]	12800 [Gbps]	160 [Mbps/μm^2]
Ref [18]	1.3 [mm^2]	110 [MHz]	12.5 [ns]	1.03 [Gbps]	792 [bps/μm^2]
Ref [19]	0.0211 [mm^2]	12 [MHz]	n. a.	4.31 [Mbps]	204 [bps/μm^2]

4. SUMMARY

A method for modeling of QCA circuits by VHDL hardware description language and simulation of these models by ModelSim software is presented in this paper. The simulation and verification of QCA circuits with high density of cells, inputs and outputs is possible by this method.

The proposed VHDL implementation and the QCA implementation of the AES algorithm and two ASIC implementations of this algorithm are also presented in this paper. The implementation results are investigated and compared. Results show that the QCA implementation of AES block cipher is several times more efficient than other implementation of this cryptographic algorithm.

REFERENCES

[1] C.S. Lent, P.D. Tougaw, W. Porod and G.H. Bernstein; **"Quantum Cellular Automata,"** *Nanotechnology*, Vol. 4, pp.49–57, 1993.

[2] K. Walus, V. Dimitrov, G.A. Jullien and W.C. Miller; **"QCADesigner: A CAD Tool for an Emerging Nano-Technology,"** *Micronet Workshop*, Available online, 2003.

[3] Marco Ottavi, Luca Schiano and Fabrizio Lombardi; **"HDLQ: A HDL Environment for QCA Design,"** *ACM J. on Emerging Tech. in Computing Systems*, Vol. 2, pp. 243–261, 2006.

[4] T. Teodosio and L. Sousa; **"QCA-LG: A tool for the Automatic Layout Generation of QCA Combinational Circuits,"** *Norchip Conference*, 2007.

[5] M.A. Amiri, M. Mahdavi and S. Mirzakuchaki; **"QCA Implementation of A5/1 Stream Cipher,"** *2nd Int. Conf. on Advances in Circuits, Elec. and Micro-elect.*, 2009.

[6] M.A. Amiri, S. Mirzakuchaki and M. Mahdavi; **"Logic-Based QCA Implementation of a 4×4 S-Box,"** *Informacije MIDEM*, Vol. 40, pp. 197-203, 2010.

[7] M.A. Amiri, S. Mirzakuchaki and M. Mahdavi; **"Cryptography in Quantum Cellular Automata,"** *Cellular Automata - Innovative Modelling for Science and Engineering, Dr. Alejandro Salcido (Ed.)*, InTech, 2011.

[8] Karim, Fazal, and Konrad Walus; **"Efficient simulation of correlated dynamics in quantum-dot cellular automata,"** *IEEE Trans. on Nanotechnology*, Vol. 13, pp. 294-307, 2014.

[9] Yang, Xiaokuo, Li Cai, S. Wang, Zhuo Wang and C. Feng; **"Reliability and Performance Evaluation of QCA Devices with Rotation Cell Defect,"** *IEEE Transactions on Nanotechnology*, Vol. 11, pp. 1009-1018, 2012.

[10] Das, Kunal and Debashis De; **"A Study On Diverse Nanostructure for Implementing Logic Gate Design for QCA,"** *International Journal of Nanoscience*, Vol. 10, pp. 263-269, 2011.

[11] V. Vankamamidi, M. Ottavi and F. Lombardi; **"Two-Dimensional Schemes for Clocking/Timing of QCA Circuits,"** *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst*., Vol. 27, pp. 34-44, 2008.

[12] K. Hennessy and C. S. Lent; **"Clocking of Molecular Quantum-dot Cellular Automata,"** *J. Vac. Sci. Technol.*, Vol. 19, pp. 1752-1755, 2001.

[13] D. Tougaw and M. Khatun; **"A Scalable Signal Distribution Network for Quantum-Dot Cellular Automata,"** *IEEE Transactions on Nanotechnology*, Vol. 12, pp. 215-224, 2013.

[14] J. S. Chandra, K. Suresh and B. Ghosh; **"Clocking Scheme Implementation for Multi-Layered Quantum Dot Cellular Automata Design,"** *Journal of Low Power Electronics*, Vol. 10, pp. 272-278, 2014.

[15] Advanced Encryption Standard (AES), http://www.csrc.nist.gov/publications/fips/fips197/fips-197.pdf

[16] Minjun Yan; **"Electric Field Detection by Electrostatic Force Microscopy for Clocking Quantum dot Cellular Automata Molecules,"** *PhD Thesis, University of Notre Dame, Notre Dame*, Indiana, 2006.

[17] Mo Liu; **"Robustness and Power Dissipation in Quantum-dot Cellular Automata,"** *PhD Thesis, University of Notre Dame, Notre Dame*, Indiana, 2006.

[18] Huang Yin, He Debiao, Kang Yong and Fei Xiande, **"High-speed ASIC implementation of AES supporting 128/192/256 bits,"** *International Conference on Test and Measurement*, 2009.

[19] Tim Good and Mohammad Benaissa, **"692-nW Advanced Encryption Standard (AES) on a 0.13-μm CMOS,"** *IEEE Transactions on VLSI Systems*, Vol. 18, No. 12, pp. 1753-1757, 2010.