A5/1 Algorithm Based On Quantum-Dot Cellular Automata (QCA) Technology

Rashmi Chawla[1], Priyanka Gaur[2], Sunita Rani[3], Naresh Tanwar[4]

[1]Asst. Professor, YMCAUST,
[2]Student, M.Tech-VLSI, YMCAUST,
[3]Student, M.Tech-VLSI, YMCAUST,
[4]Asst. Professor, ACEM,

VLSI Dept., YMCA University of Science and Technology, Faridabad, INDIA

***Abstract*–The motivation for this paper has been basically the need of security for mobile communication which not only effects military but also personal usage. Quantum Cellular Automata (QCA) is an emerging new nano-technology that not only serves as an alternative solution to CMOS which decreases lots of physical limits and circuitry boundations [18]. QCA is a transistor-less technology and here information is passed based on the charge on electron and by mutual electrostatic repulsion between them. QCA has very high device density, faster switching speed clocking and extremely low power consumption. This paper describes the basic way to generate cipher text in QCA that can be helpful in QCA based secure nano-communication. The implementation and testing of results is performed using QCA Designer-2.0.3[4].Utilizing the QCA technology, we have implemented the A5/1 stream cipher which was the original encryption algorithm for GSM. The implementation of this cryptographic algorithm is accomplished by means of the implementation of its main modules. The key properties of the QCA implementation of A5/1 stream cipher such as latency, area and complexity are discussed in this paper [3].**

***Keywords* – QCA-technology implementation, stream cipher, A5/1, GSM network, cryptography**

I. INTRODUCTION

In recent time the main aim of designers was towards the scaling down of the key parameters for designing an integrated circuit, i.e. increasing the density of the circuit (VLSI, CMOS). So the need to move from transistor circuits to transistor-less circuits raised which was fulfilled by QCA. In QCA information are passed by electron charges through coulombic-force present within quantum dots in place of electrical energy (pulse) as in CMOS circuit.

In cryptography cipher text plays a major role to represent the actual data into an encrypted (non-readable) form to ensure security from unauthorized access to the actual data [23].

Before starting with the A5/1 algorithm, it is beneficial to have a general view to the GSM system in which this algorithm is used. GSM network tells us the basic functioning of the A5/1 algorithm and also why we need to encode the speech and at which platform.

A. Description of GSM

GSM (Global System for Mobile Communications) is a digital cellular technology used for transmitting voice and data services. GSM allows users to roam seamlessly from one network to another, while also providing personal mobility. In addition, both speech and signalling channels are digitalized, which essentially labelled GSM as the second-generation (2G) mobile system.

GSM communication takes place through a specified procedure. Firstly, the authenticity of the SIM(Subscriber Identity Module) is verified either by VLR (Visitor location Register) or by HLR (Home Location Register), and then various algorithms (A3, A8, A5) are implemented to get the process of communication completed[9].

B. Key generation and encryption in GSM

Authentication is one of the most important security functions in GSM. It involves several functional components: the SIM card, MSC (Mobile Switching Centre), VLR (Visitor Location Register) and the AuC(Authentication Centre). The MSC verify the identity of the subscriber through a challenge-response process. Upon request from the MSC, the VLR returns an authentication triplet consisting of a 128-bit random number (RAND), a 32-bit signed response (SRES) and the session key K. When a MS (Mobile station) requests service, the MSC challenges it by sending the RAND and a 3-bit

Cipher Key Sequence Number (CKSN) in a *MM Authentication Request* message. The MS must answer this challenge correctly before being granted access to the network. The RAND is forwarded from MS to the SIM for processing. Fig.1[12] shows the information each entity contains in order to authenticate the SIM. The SIM takes the RAND value and the 128-bit Individual Subscriber Authentication Key Ki and produces a 32-bit signed response (SRES). This is the MS's response to the challenge which is sent back to the network in an *RR Authentication Response*. If the SRES value is identical to the one given in the authentication triplet, the authentication is successful.

C. Confidentiality

In addition to the parameters needed for the authentication of subscribers, the SIM card also contains parameters needed to provide confidentiality. Fig.2 [9] shows the information stream which is encrypted over the air-interface. An algorithm called A8 is used to generate a session key K_c.

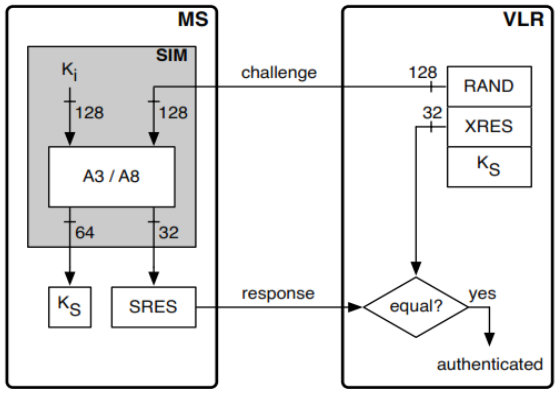

Fig. 1 Challenge-Response Authentication in GSM

By using RAND and the key K_i, the SIM runs the A8 algorithm to produce a 64-bit K_c, which is then used by a third algorithm called A5. The A5 is used to produce a key stream of 228 bits from the K_s (or K_c) and the current F_n of the timeslot in which the next segment of the message is sent. The key stream is decomposed into two halves. While the first half encrypts the downlink frame, the second half is used to encrypt the uplink frame. For each transmitted frame, a new 228-bit key stream is calculated by A5 to encrypt (and decrypt) the frame. The session key K_s generated above is taken as K_c in the Fig.2.

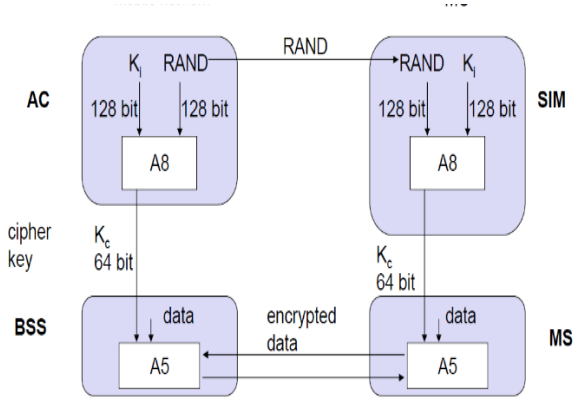

Fig.2 Key Generation and Encryption in GSM

II. INTRODUCTION TO A5/1

A5/1 was developed in 1987. This algorithm is built from three short linear feedback shift registers (LFSR) of lengths 19, 22, and 23 bits, which are named R1, R2 and R3, respectively as shown in figure 3. The rightmost bit in each register is named as bit zero [10]. The taps of R1 are at bit positions 13, 16, 17, 18. The taps of R2 are at bit positions 20, 21. The taps of R3 are at bit positions 7, 10, 21, 22. When a register is clocked, its taps are XORed together, and the result is stored in the rightmost bit of the left shifted register. The three registers are specially clocked using the following majority rule: Each register has a single clocking tap (bit 8 for R1, bit 10 for R2, and bit 10 for R3). In each clock cycle, the majority function of the clocking taps is calculated and only those registers whose clocking taps are equal to the majority bit, are actually clocked.

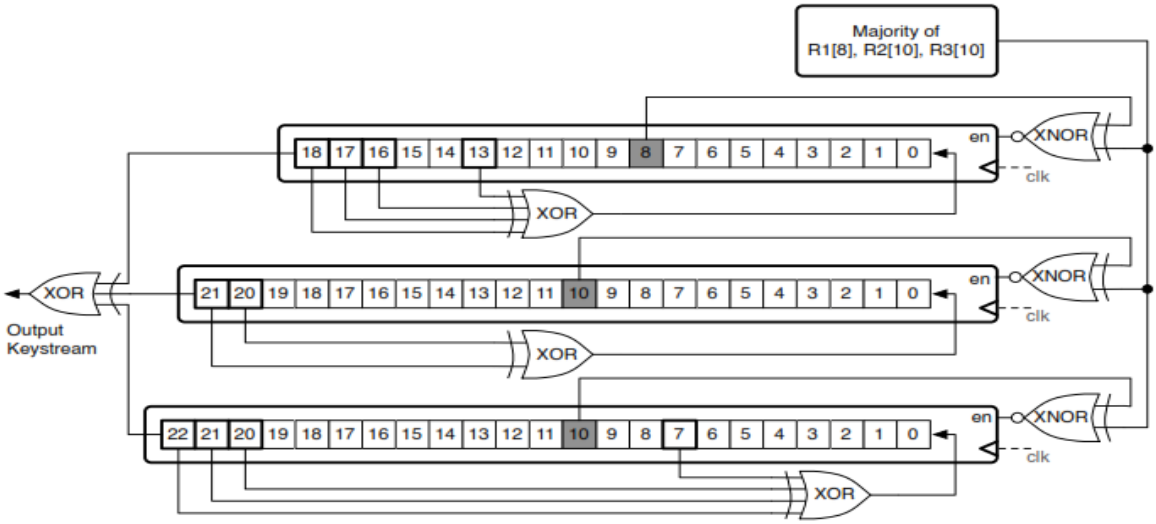

Fig.3 Design of A5/1

In an LFSR, all bits are shifted one place towards the end of the register every time the register is clocked. This leaves the least significant bit (LSB) empty, meaning that this bit needs to get its new value in a different way. This is solved by letting each register have a few tapped bits. As seen in Fig.3, the tapped bits of R1 are bit 13, 16, 17 and 18, of R2 bit 20 and 21, and of R3 bit 7, 20, 21 and 22. Whenever a register is clocked the value of these taps are read out before the shifting has been performed. Their values are then XORed together and the result used as input to bit 0. Whether a register is clocked or not is decided by the majority function. Three clocking bits exist, denoted by C1, C2 and C3 respectively. C1 is bit 8 of R1, C2 is bit 10 of R2 and C3 is bit 10 of R3. At each clock cycle the value of these bits are analyzed, and the two or three registers whose bit agrees with the majority are clocked [12].

The A5/1 algorithm takes as input a 64-bit key K_c and a 22 bit frame number F_n and produces 228 bits of key stream. The output is the result of bit 18, 21and 22 XORed together. Messages between the MS and the BTS (Base Transceiver Station) are sent in bursts containing 114-bits of payload, and since the same frame number can be used two times in a row, A5/1 needs to produce 228 bits of key-stream. The first 114 bits of the key-stream are used to encrypt downlink traffic (BTS to MS), while the last 114 bits are reserved for encryption of uplink traffic (MS to BTS).

A. A5/1 Algorithm

From initialization to 228-bits of key-stream, the A5/1 algorithm goes as follows [17]:

1. All three registers are set to 0. Then, for 64 cycles, the key is mixed into the registers in parallel using the following algorithm:

For i = 0 to 63 do

$R1[0] = R1[0] \otimes K[i]$

$R2[0] = R2[0] \otimes K[i]$

$R3[0] = R3[0] \otimes K[i]$

Clock all three registers according to the regular clocking scheme.

end for

Where Ri [0] denotes the LSB of register Ri, K [0] the LSB in the key and K [63] the most significant bit. It's important to notice that the majority clocking mechanism is not used at this stage.

2. 22 additional cycles are clocked, still overlooking the majority function. During this period the frame number is XORed into the lsb of the registers in the same way as with the key, that is:

   For i = 0 to 63 do

   $R1[0] = R1[0] \otimes F[i]$

   $R2[0] = R2[0] \otimes F[i]$

   $R3[0] = R3[0] \otimes F[i]$

   Clock all three registers according to the regular clocking scheme.

   end for

3. 100 additional clocks are performed with the regular majority clocking mechanism

activated, but the output is discarded. The content of the registers at the end of this state is what we refer to as the *initial state* of A5/1.

4. 228 clocks are performed to produce 228 bits of key stream. This is key stream is then XORed with the plaintext in order to encrypt the data.

B. *Primitive Polynomials of LFSRs:*

The general form of an LFSR of degree m is shown in Fig.4. It shows m flip-flops and m possible feedback locations, all combined by the XOR operation. Whether a feedback path is active or not, is defined by the *feedback co-efficient* $p0, p1,..., p_{m-1}$:

- If pi= 1 (closed switch), the feedback is active.
- If pi= 0 (open switch), the corresponding flip-flop output is not used for the feedback.

With this notation, we obtain an elegant mathematical description for the feedback path. If we *multiply* the output of flip-flop i by its co-efficient p_i, the result is either the output value if p_i= 1,

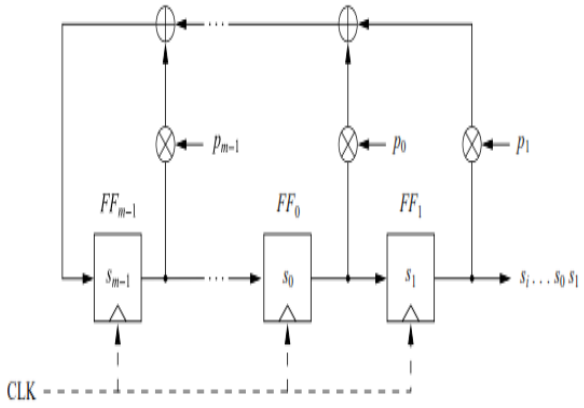

Fig.4 General LFSR with Feedback Coefficient P_i and Initial State S_i

which corresponds to a closed switch, or the value zero if p_i=0, which corresponds to an open switch. The values of the feedback co-efficients are crucial for the output sequence produced by the LFSR. Let's assume the LFSR is initially loaded with the values $s_0,\ldots\ldots s_{m-1}$. The next output bit of the LFSR s_m, which is also the input to the leftmost flip-flop, can be computed by the XOR-sum of the products of flip-flop outputs and corresponding feedback co-efficient:

$$s_m = s_{m-1}p_{m-1}+\cdots+s_1p_1+s_0p_0 \bmod 2$$

An LFSR with a feedback co-efficient vector (p m-1,...,p1, p0) is represented by the polynomial

$$P(x)=x^m+ p_{m-1}\, x^{m-1}+\ldots+p_1x+p_0$$

Register	Register Length	Clocking Bit	Primitive Polynomials	Tapping Bits
R_1	19 bits	8	$1+x+x^2+x^5+x^{19}$	18, 17, 16, 13
R_2	22 bits	10	$1+x+x^{22}$	21, 20
R_3	23 bits	10	$1+x+x^2+x^{15}+x^{23}$	22, 21, 20, 7

Fig.5 Feedback Polynomials of LFSRs in A5/1 Construction

III. IMPLEMENTATION USING QCA

The microelectronics industry has improved the integration, the power consumption, and the speed of integrated circuits during past several decades 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. The use of QCA technology for realizing logic circuits is one of the approaches which in addition to decrease the size of logic circuits and to increase the clock frequency of these circuits, reduces the power consumption of these circuits. QCA which was first introduced by Lent et al. represents an emerging technology at the nanotechnology level. QCA cells have quantum-dots, in which the position of electrons will determine the binary levels of 0 and 1[3].

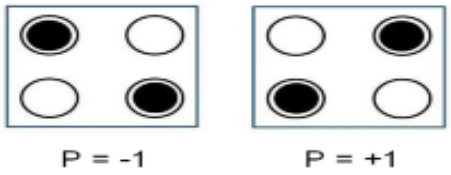

Fig.6 Basic QCA Cells

A. *Memory Cell Implementation*

In this section, the implementation of A5/1 stream cipher is investigated by means of implementation of its main blocks. One of its main blocks is the memory cell. We have implemented a Loop-Based memory cell which has a write enable input, an input and an output. QCA implementation of this unit is illustrated in Fig.7. It can be seen that if the ENABLE input has

a high logic value, the stored memory datum will remain unchanged and if the ENABLE input has a low logic value for a period of clock, the complementary INPUT value will be stored in the memory cell. The OUTPUT will display the stored value in the memory cell in all clock periods [3].

B. Majority Function Implementation

Another main block of the A5/1 stream cipher is the Majority function. The Majority function produces the enable signals for shifting the A5/1 registers. This is accomplished by using the clocking bits of three predefined registers. If the clocking bit of a register is equal to majority of three clocking bits, that register should be shifted in the next operation. QCA implementation of this function for R2 register is illustrated in Fig.9. The other Majority functions forR1 and R3 registers are almost the same as this one [3].

C. R1 Register Implementation

Fig.11 shows the QCA implementation of the R1 register. This register has 19 memory cells. There are also three XOR gates and a multiplexer in R2 register. The multiplexer is used to select the R1-INPUT when the R1-LOAD is activated. Otherwise, the feedback value is selected. The 8th bit of this register is the clocking bit and the 18th bit is the output bit. The feedback of this register is computed by 13th, 16th, 17th and 18th bits which are XORed. When the R1-LOAD input has the logic value of one, the R1-INPUT value will be stored in the 0th bit of R1 register. The R1-ENABLE input is used to shift the data throughout the R1 register. When this input has the value of one for a clock period, all bits of R1 register, except 0th bit, will store the previous bit's value after five clock periods of latency. The value of 0th bit will be changed after seven extra clock periods because its feedback value should be computed by three XOR gates and the XOR gates have a total latency of seven clock periods. It means that a shift operation will be accomplished in twelve clock periods. In short, we can say that R1 register will give us (1+x+x2+x5+x19) as the output [3]. The implementation of the R1 register has a complexity of 196 cells and an area of about $4.95*10^{-5}nm^2$.

D. R2 Register Implementation

Fig.12 shows the QCA implementation of the R2 register. This register has 22 memory cells. There is also an XOR gate and a multiplexer in R2 register. The multiplexer is used to select the R2-INPUT when the R2-LOAD is activated. Otherwise, the feedback value is selected. The 10th bit of this register is the clocking bit and the 21st bit is the output bit. The feedback of this register is computed by 21st and 20th bits which are XORed. When the R2-LOAD input has the logic value of one, the R2-INPUT value will be stored in the 0th bit of R2 register. The R2-ENABLE input is used to shift the data throughout the R2 register. When this input has the value of one for a clock period, all bits of R2 register, except 0th bit, will store the previous bit's value after five clock periods of latency. The value of 0th bit will be changed after four extra clock periods because its feedback value should be computed by an XOR gate and the XOR gate has the latency of four clock periods. It means that a shift operation will be accomplished in nine clock periods. Here, the output will be (1+x+x22)[3]. The implementation of the R2 register has a complexity of 100 cells and an area of about $2.52*10^{-5}nm^2$.

E. R3 Register Implementation

Fig.13 shows the QCA implementation of the R3 register. This register has 23 memory cells. There are also three XOR gates and a multiplexer in R3 register. The multiplexer is used to select the R3-INPUT when the R3-LOAD is activated. Otherwise, the feedback value is selected. The 10th bit of this register is the clocking bit and the 22nd bit is the output bit. The feedback of this register is computed by 7th, 20th, 21st and 22nd bits which are XORed. When the R3-LOAD input has the logic value of one, the R3-INPUT value will be stored in the 0th bit of R3 register. TheR3-ENABLE input is used to shift the data throughout the R3 register. When this input has the value of one for a clock period, all bits of R3 register, except 0th bit, will store the previous bit's value after five clock periods of latency. The value of 0th bit will be changed after six extra clock periods because its feedback value should be computed by three XOR gates and the XOR gates have a total latency of six clock periods. It means that a shift operation will be accomplished in eleven clock periods. The primitive polynomial output for R3 register is (1+x+x2+x15+x23) [3].The implementation of the R3 register has a complexity of 196 cells and an area of about $4.95*10^{-5}nm^2$.

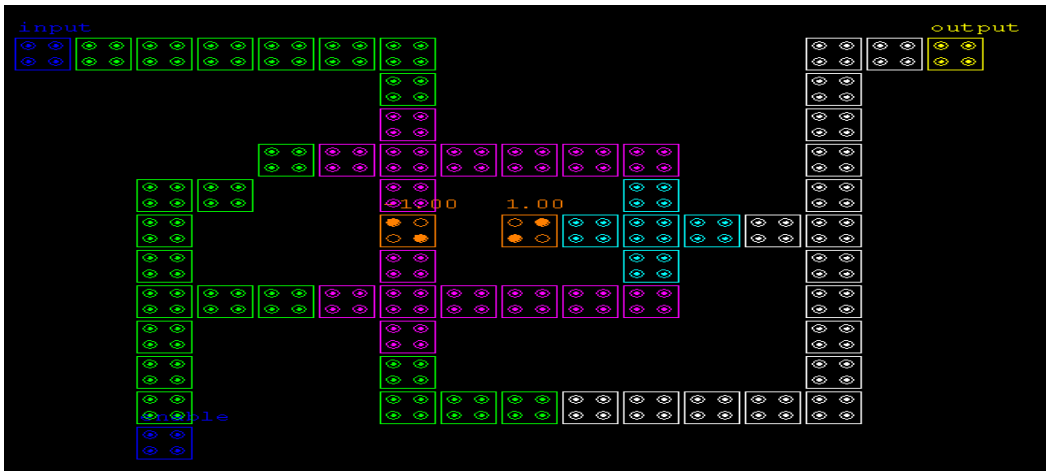

Fig.7 Memory Cell

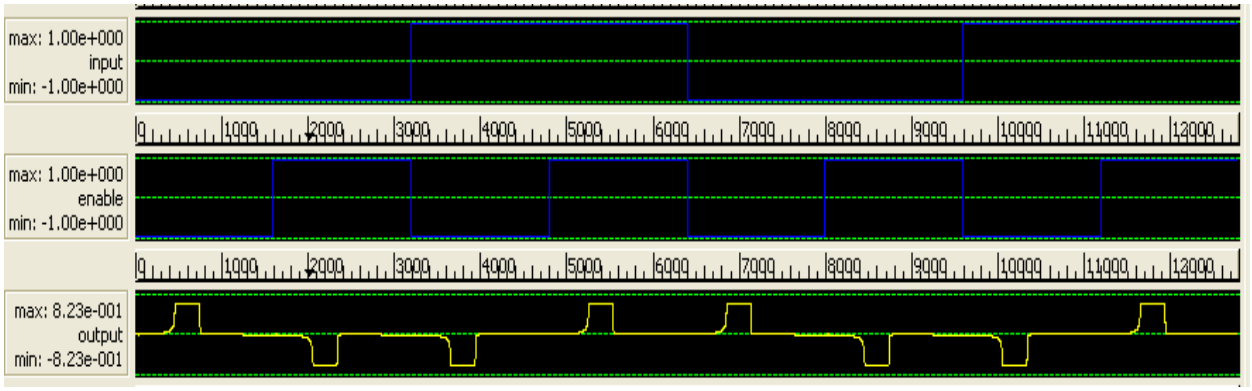

Fig.8 Output of Memory Cell

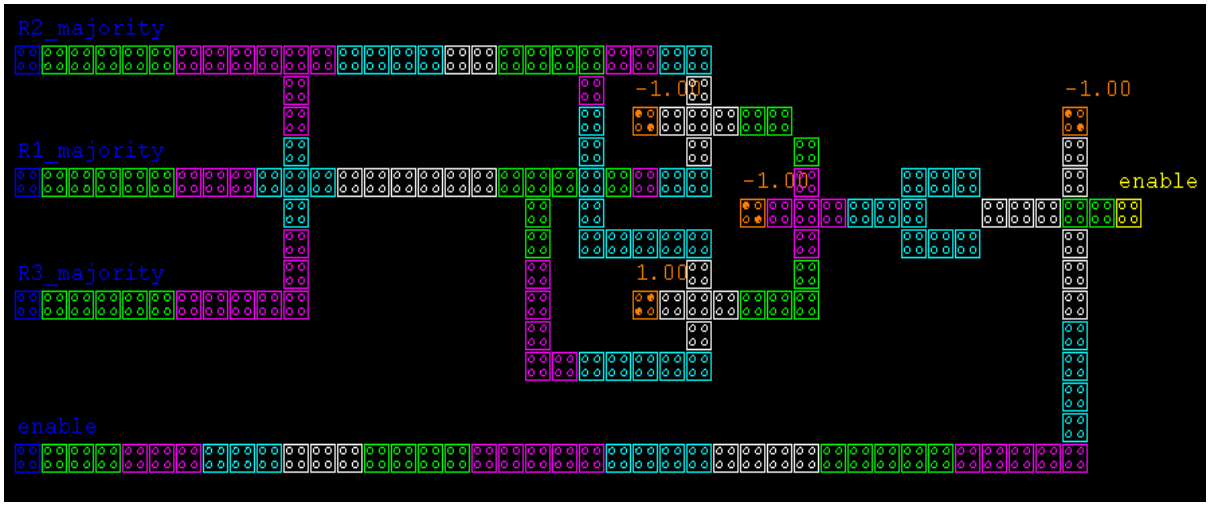

Fig.9 Majority Cell

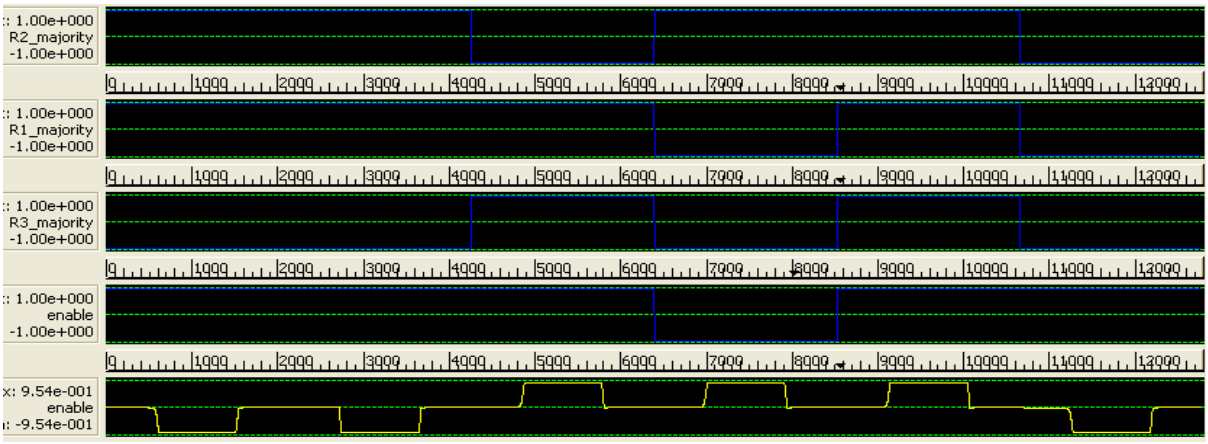

Fig.10 Output of Majority Cell

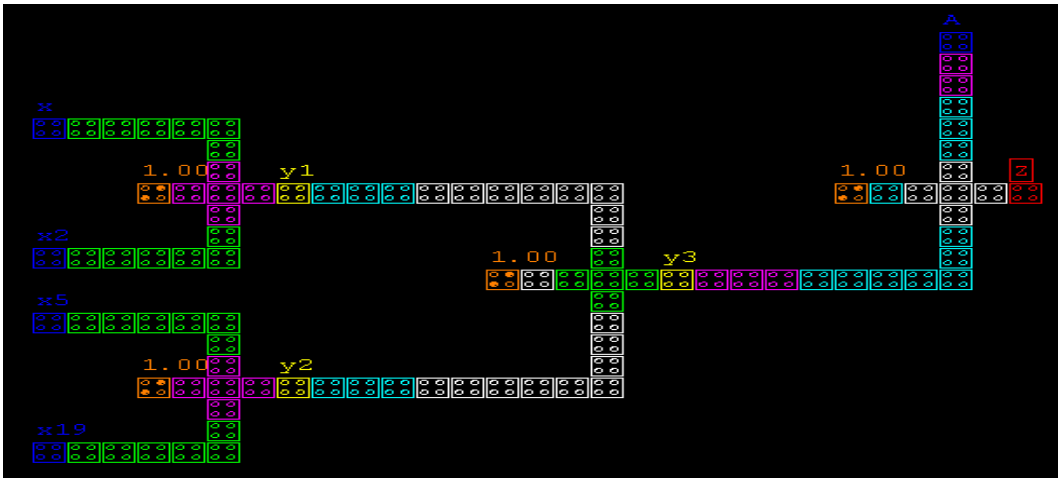

Fig.11 R1 Register

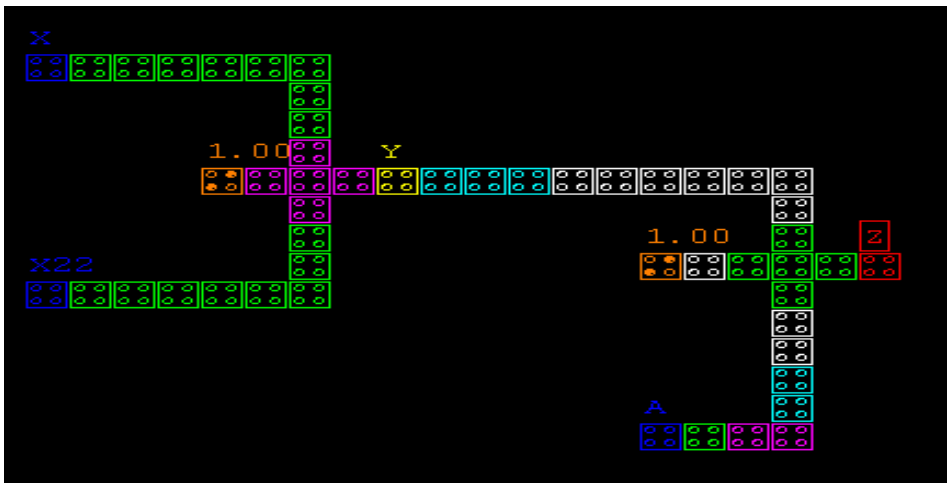

Fig.12 R2 Register

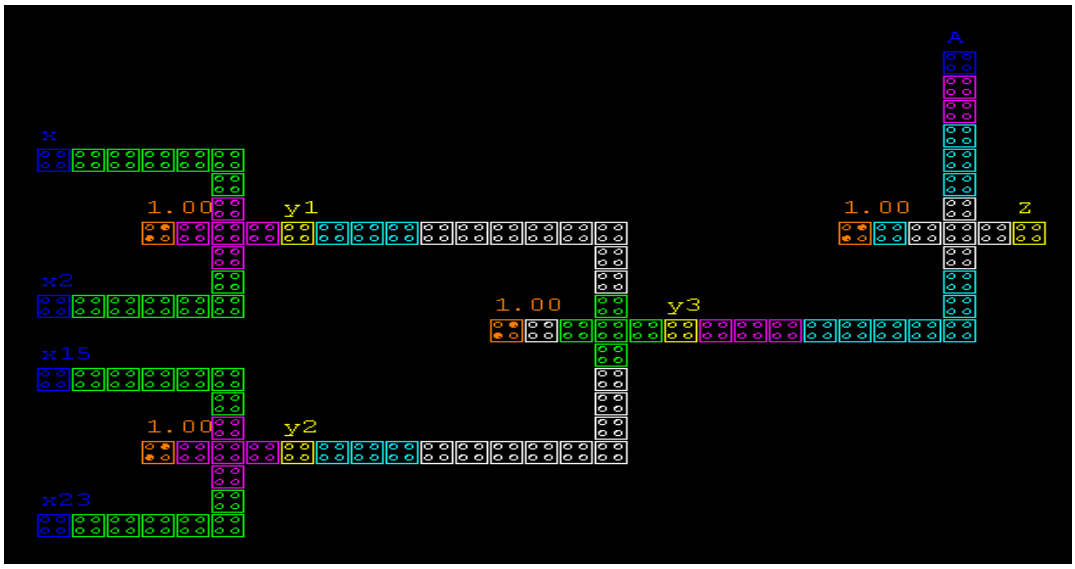

Fig.13 R3 Register

In the above simulations, we have just shown the basic algorithm of A5/1 cryptography in GSM phones. But what the actual process of cryptography is shown in the next simulation, taking an example of few bits, whereas in actual process 228 bits are taken.

Eg. If Xi denotes any plain text communication being received, then those bits are being XORed with the key bits, say Si and Yi is the ciphered text [23]. So, here is an example showing the above explanation:-

Sender

Plain text(x0, . . . ,x6) = 1000001

⊕

Key(s0, . . . , s6) = 0101100

Cipher text(y0, . . .,y6) = 1101101

Receiver

Cipher text(y0, . . .,y6) = 1101101

⊕

Key(s0, . . . , s6) = 0101100

Plain text(x0, . . . ,x6) = 1000001

Fig.14An Example Showing Encryption and Decryption

In the terms of QCA implementation, the cipher text and the plain text generation can be done as follows:
Yi = Xi XOR Si = Xi.Si'+Xi'.Si
Xi = Yi XOR Si = Yi.Si'+ Yi'.Si

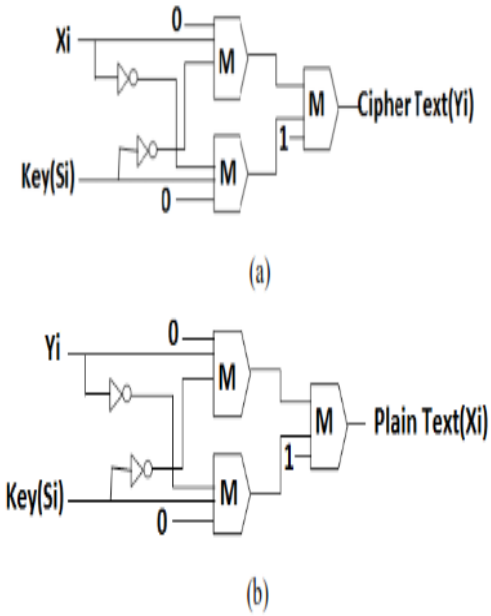

Fig.15 QCA Schematic of (A) Cipher Text Generation (B) Plain Text Generation

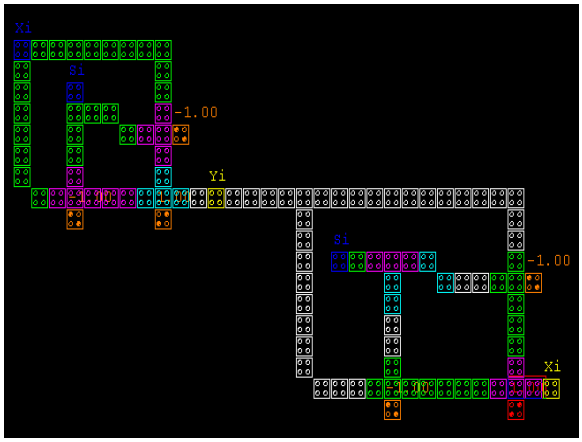

Fig.16 QCA Encoder and Decoder

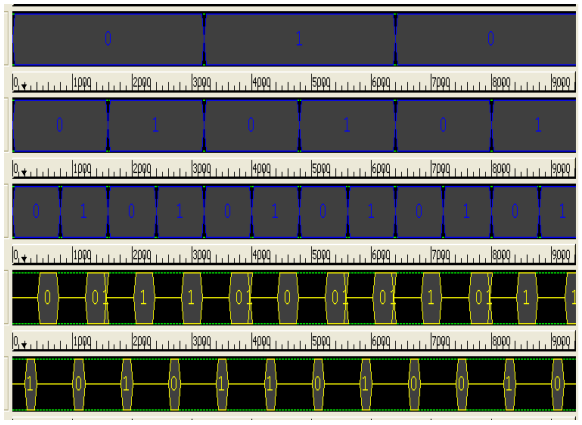

Fig.17 Output of QCA Encoder and Decoder

IV. RESULTS AND DISCUSSION

The aim of this project was the usage of nano-technology in much needed privacy terms of speech encoding in mobile communication and this has been achieved in this project by using the Quantum dots principle in A5/1 algorithm.

We have used QCADesigner[4,8] software to implement our project. A5/1 algorithm has been running with the use of microscopic technology and with our project, we have given a try to implement the same in nano-technologic terms. Now, this QCA implementation has given us advantage in terms of area, latency and speed. As now, the area covered by the same circuit is in nano-meters rather than micrometers; area covered by the circuit is decreased. And as the area decreases, the latency parameter is also reduced as now the QCA wires dissipate minimal energy and also, the loss of signal is recovered by the clock signals in QCA circuit, speed is also increased.

A. *Calculation of Area Covered*

The circuit is implemented and simulated using the Bi-stable simulation engine QCA Designer-2.0.3. The distance between the cells and quantum-dots are fixed and with the help of these, I can calculate the area of the circuit simulated.

The area covered by R1, R2, and R3 registers is ($4.95*10^{-5}$, $2.52*10^{-5}$ and $4.95*10^{-5}$) nm^2. The area of encoder and decoder circuit is to be calculated yet. So, the area covered by the circuit can be calculated by counting the number of cells in the circuit horizontally and vertically. The cells are 42 each for encoder and decoder, so area is $5.02*10^{-5}$ nm^2.

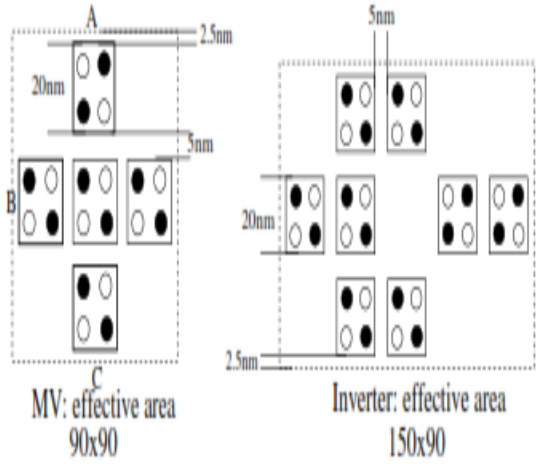

Fig.18 Effective Area of MV and Inverter

V. CONCLUSION

From the above implementation, we can conclude that the QCA circuits occupy less area and are much faster in speed (so, latency is also reduced) than the CMOS technology.

Due to the invention of side-channel attacks, cryptographic device can be victimized in terms of power and electromagnetic analysis attacks. So, the proposed circuit can be sound reference to build a secure QCA cryptographic module than traditional system as QCA has ultra low power consumption and very high clocking. The encryption and decryption is performed on limited-bits message using limited-bits key but it can be done on any length of message bits

as well key bits using proposed circuit. Here, circuit required only 3 MVs, 2 inverters, 3 clock zones, 42 cells and $5.02*10^{-5}$ nm² areas for both encoder and decoder together. Future implementation of cryptographic algorithms for QCA-based secure nano-communication system can be achieved using the proposed circuit [23].

REFERENCES

[1] Lent, C., et l., "Quantum Cellular Automata," *Nanotechnology*, Vol. 4, 1993, pp. 49–57.

[2] Quantum-dot Cellular Automata, *by Weiqiang Liu, Máire O'Neill, and Earl E. Swartzlander, Jr.*

[3] QCA Home Page," website, 2013, http://www.nd.edu/~qcahome/.

[4] Walus, K., et al., "QCADesigner: A Rapid Design and Simulation Tool for Quantum-Dot Cellular Automata," *IEEE Transactions on Nanotechnology*, Vol. 3, 2004, pp. 26–31.

[5] Lombardi, F., and J. Huang, *Design and Test of Digital Circuits by Quantum-Dot Cellular Automata*, Norwood, MA: Artech House, Inc., 2007.

[6] Frost, S., et al., "Memory in Motion: A Study of Storage Structures in QCA," in *Proceedings of 1st Workshop on Non-Silicon Computing*, Vol. 2, 2002, pp. 30–37.

[7] Orlov, A., et al., "Experimental Demonstration of A Binary Wire for Quantum-Dot Cellular Automata," *Applied Physics Letters*, Vol. 74, No. 19, 1999, pp. 2875–2877.

[8] Orlov, A., et al., "Experimental Demonstration of Clocked Single-Electron Switching in Quantum-Dot Cellular Automata," *Applied Physics Letters*, Vol. 77, No. 2, 2000, pp. 295–297.

[9] Decoding GSM, by Magnus Glendrange, Kristian Hove, Espen Hvideberg, Master of Science in Communication Technology, June-2010

[10] A5/1 Security Project, http://reflextor.com/trac/a51, May 2010. Last accessed June 11, 2010.

[11] S. Babbage, A. Space/Time Trade off in Exhaustive Search Attacks on Stream Ciphers. In *European Convention on Security and Detection*, number 408 in IEEE Conference Publication, May 1995.

[12] Hardware-Based Cryptanalysis of the GSM A5/1 Encryption Algorithm, by Timo-Gendrullis, may-2008.

[13] Implementation of Cryptographic Algorithms for GSM Cellular Standard, by Ganpat University Journal Of Engineering & Technology, VOL.-1, ISSUE-1, Jan-June-2011.

[14] Raj Pandya, "*Mobile and Personal Communication Systems and Services*, 2001 IEEE PRESS, New York

[15] V. K. Garg, "*Wireless and Personal Communication System",* 1997 Prentice Hall of India Private Ltd., New Delhi.

[16] Complexity-analysis on attacks of A5/1 by *Tartu2006.*

[17] A real-world attack breaking A5/1 within hours, by Timo Gendrullis, Martin Novotn´y, and Andy Rupp

[18] Quantum Dot-Cellular Automata Based Cipher Text Design for Nano-Communication, by Jadav Chandra Das, and Debashis De, *2012 International Conference on Radar, Communication and Computing (ICRCC), SKP Engineering College, Tiruvannamalai, TN., India. 21 – 22 December, 2012. pp. 224-229.*

[19] International Technology Roadmap for Semiconductors,--(ITRS) 2005 [Online]. Available: hnp://www.itrs.net.

[20] R. Zhang, K. Walus, W. Wang, and G. A. Jullien, "A method of majority logic reduction for quantum cellular automata," IEEE Trans. Nanotechnol., vol. 3, no. 4, pp. 443-450, Dec. 2004.

[21] W. Porod, ―Quantum-dot devices and quantum-dot-cellular automata, Int. J. Bifurcation Chaos, vol. 7, no10, pp. 2199–2218, 1997.

[22] E.N.Ganesh1, Lal Kishore2 and M.J.S. Rangachar, ―*Implementation of Quantum cellular automata combinational and sequential circuits using Majority logic reduction method*, International Journal of Nanotechnology and Applications, Vol. 2, pp. 89106, Nov. 2008.

[23] C. S. Lent and B. Isaksen, ―*Clocked molecular quantum dot cellular automata,‖* IEEE Trans. Electron Devices, vol. 50, no. 9, pp. 18901896, Sep.2003.1