International Journal of Engineering & Technology, 7 (1.5) (2018) 237-244

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

FPGA implementation of tunable arbitrary sequencer for key generation mechanism

M. Siva Kumar[*], B. Murali Krishna, N. Sai Tejeswi, Sanath Kumar Tulasi,
N. Srinivasulu, K. Hari Kishore

Department of ECE, Koneru Lakshmaiah Educational Foundation, Vaddeswaram,
Guntur, Andhra Pradesh, India 522502
**Corresponding author E-mail: n.saitejeswi1@gmail.com*

Abstract

In the present scenario information security has become a predominant issue. Cryptography is the process used for the purpose of information security. In cryptography message is encrypted with key produces cipher and decrypts the original message from cipher uses variety mechanisms and permutations. This paper presents a key generation mechanism suitable in cryptography applications which plays a vital role in data security. Random key generation techniques are multiplexed and configured in FPGA. In run time based on priority of section inputs randomly one method selectively produces a key which inputs to cryptosystem. Jitter process generates random numbers based on clock frequency triggered to oscillators, which produces pseudo random keys, but it consumes more resources when compared with other methods, but randomness in generated key is exponential. Pre stored random numbers in Block Memory are generated using IP core generator. The main advantage of the proposed model is to produce random keys which will be secure, predictable and attains high security. Due to its configurable nature, FPGA's are suitable for wide variety of applications which can configure in runtime to implement custom designs and needs. Random number generation techniques are designed using Verilog HDL, simulated on Xilinx ISE simulator and implemented on Spartan FPGA.

Keywords:*LFSR, Cryptography, Verilog, Xilinx, FPGA.*

1. Introduction

Cryptography is a technique used whenever there is need of securing the information. Cryptography is mainly used in communication system where information must be transmitted from sender to receiver without getting interrupted by third party. Encryption is a process of transmission of known information into a different structure (cipher text) based on algorithm which cannot be decrypted without knowing decoding technique. The cipher is a structure which contains the encryption and decryption process. The working of cipher is controlled by the algorithm. The information security is maintained by making the key known only to transmitter and receiver and without which information cannot be decrypted.

Two types of key algorithms are available in cryptography one is symmetric key algorithm and the other is asymmetric key algorithm. The major difference between symmetric key and asymmetric key is symmetric key algorithms uses same key for both encryption and decryption whereas asymmetric algorithm uses different keys for encryption and decryption.

Random numbers are used for the generation of different keys used for encryption and decryption. These Random numbers are used in various fields such as ATM'S, OTP generation and for various fields where security is the main issue. Due to flexibility and easy design change FPGA's are configured for implementation of the Random numbers which are required for the generation of security keys in communication and cryptography applications [6].

A. Description of Proposed Model:

This paper presents the various methods for the random key generation mechanisms are used, such as ,

i) Jitter
ii) BRAM
iii) LFSR
iv) Galois LFSR
v) Fibonacci LFSR
vi) Combined LFSR with XOR.
vii) Low Power LFSR.
Viii) Run Time Polynomial Change.

randomly gets selected. The implementation is shown below in fig 1.

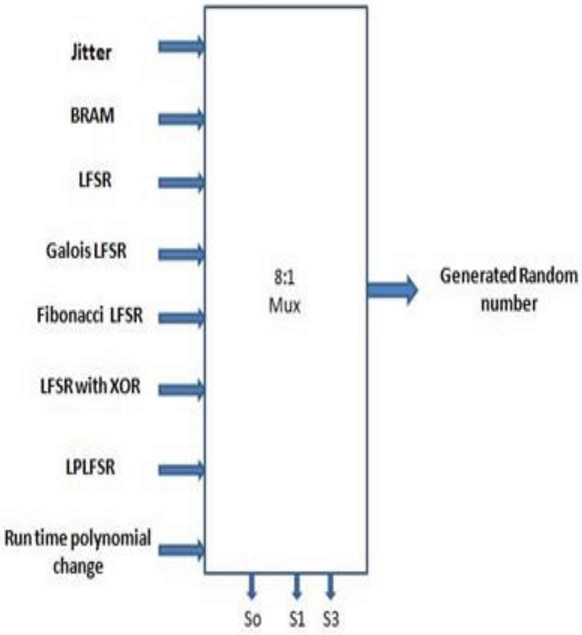


Fig1: Multiplexer Implementation for generating Random Numbers using different methods

Generation of random number out of eight methods by making the selection lines, to be configured by usage of 3 bit LFSR to obtain any one of the method.

B. Adoption of methods

i) Selection of 000 adopts Jitter method for the generation of random numbers.
ii)) Selection of 001 adopts BRAM method for the generation of random numbers.
iii) Selection of 010 adopts LFSR method for the generation of random numbers.
iv) Selection of 011 adopts Fibonacci LFSR method for the generation of random numbers
v) Selection of 100 adopts Galois LFSR method for the generation of random numbers
vi) Selection of 101 adopts combinational LFSR/XOR method for the generation of random numbers
vii) Selection of 110 adopts Low Power LFSR for the generation of random numbers .
viii)Selection of 111 adopts Run Time Polynomial Change method for the generation of random numbers .

2. Existing methods

A. Generation of Random Numbers using Ring Oscillators

Noise Generators are one of the source which produces random data which is unpredictable. Ring Oscillator is the main principle used in noise generator, Odd numbers of inverters which are interconnected forms a ring oscillator. Inverter based Ring Oscillator is as shown in fig 2.

Jitter on Ring Oscillators is one of the concept used for acquiring Random Numbers. Jitter is the timing variation of signal edges from their ideal values. This variation gives us a scope for the generation of random numbers. If the Selection lines are of 000 Ring Oscillator method [1] gets selected.

Ring Oscillators consists of definite frequency based on arrangement and placement of components and temperature .Thus, every ring oscillator has its own definite frequency. If we enable two ring oscillators there would be slight frequency difference which gives rise jitter. Placement of invertors for ring oscillator is as shown fig 2(1).

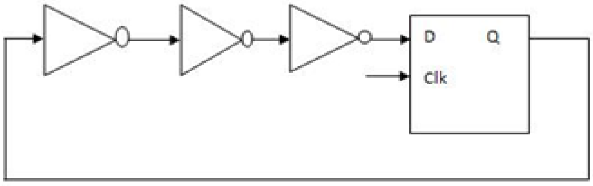


Fig. 2(1): Ring Oscillator

The architecture for the generation of random numbers using jitter [3] is very simple and is as shown in fig2(2).

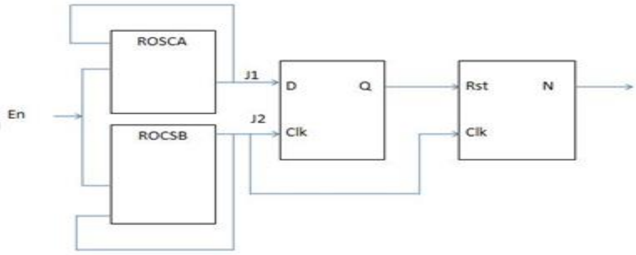


Fig. 2(2): Generation of random numbers using Jitter

Jitter is the existence of difference infrequencies from reference frequency of original signal. Ring oscillators whose output is having a frequency difference which is nothing but beat frequency jitter is produced between the two signals.[2]
1) When the clock is triggered to the ring oscillators it produces an output by using Xor operation between the output signals of Ring oscillators. We are considering Ring Oscillator A is little faster than that of Oscillator B thus there exists a frequency little frequency difference[3] (jitter).
2) The output of ring oscillator A is sampled by output of ring oscillator B by making the output of
ring oscillator A as input to the D Flipflop and output of ring oscillator B is given as clock signal to D Flipflop.
3) The output of D Flipflop is 1 when the outputs of ring oscillators are not equal and for similar outputs the output will be 0.
4) The output of D Flipflop is fed as input to the counter and the output of ring oscillator B is given as Clock to the counter.
5) When two output signals of ring oscillators are different D Flipflop output is 1 and there by counter gets Reset and whenever the inputs are not equal D Flipflop output is 0 the counter gets incremented.
Whenever the clock is triggered the oscillators started generating the signal with some frequency difference and if we observe in the simulation waveform in fig 2(3) at this frequencies D flip flop output has a transition to 1 and making the counter reset and for the other frequencies counter counts the no of clock cycles coming from the output of oscillator B.

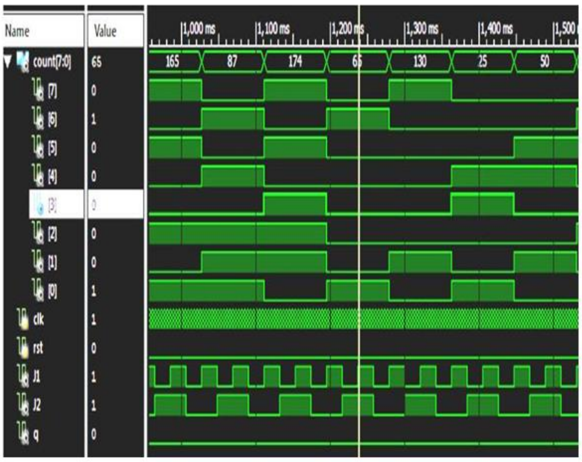


Fig. 2(3): Simulations obtained using the frequency difference between output of two Ring Oscillators.

Where J1 and J2 are outputs of the Ring Oscillators and Q is the output of D Flipflop and Reset is input of the Counter and Clock is the initial enable signal.

By the observation in the fig2(2) , whenever J1 and J2 are not equal the D Flipflop is 1 and counter is getting reset and whenever the output of D Flipflop is 0 the counter increments. Random numbers generated through this process is pseudo random number and next state can’t be estimated as the output is due to frequency difference between the outputs of ring oscillators. The D Flipflop output is 1 at random instants as frequency difference.

B. Generation of random numbers using BRAM (Block RAM)

Random values are stored in Block RAM. Selection of 010 at multiplexer adopts BRAM method [1] for the generation of
random numbers. BRAM produces the random values that are stored in the memory shown in fig2(4).

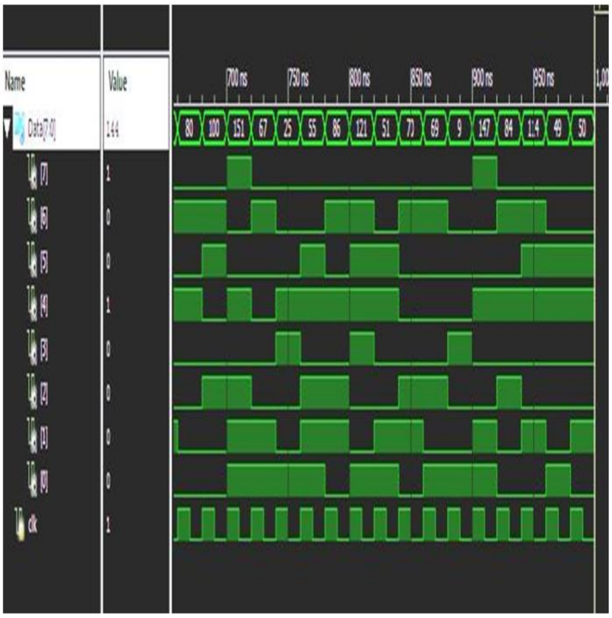


Fig. 2(4): Simulations obtained using Block Ram

C. Generation of Random Numbers usingLinear Feedback Shift Register (LFSR)

Linear Feedback Shift Register is a combination of series of registers which shifts the given data and there by produces random numbers .The random number is produced by linear function of two inputs and the linear function that is performed on the bits to produce random numbers is XOR gate [4] .The initial value given to the LFSR is called seed. The bit positions which are responsible for the generating next state are called taps .As the operator which is performed for the random number generation is known so the set of random number generated by particular taps can be determined initially. The tap positions can be represented using a polynomial .The primitive polynomial is one which represents the taps that gives the maximum random numbers for n bits.Based on the taps that are selected and definite random numbers are generated [4] as shown in fig 2(5).

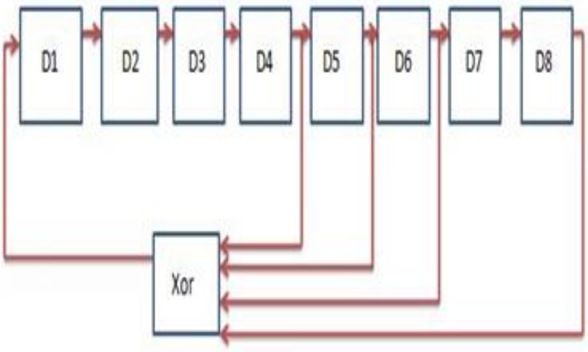


Fig. 2(5): Generation of Random Numbers using 8bit LFSR

Whenever the selection lines are 010 LFSR method for the generation of Random numbers gets
Selected. Initially when clock is given and some value is stored in the Flip flops then in the next clock cycle the outputs of fourth, fifth, sixth and eighth flip flops are XOR’d and the output of the XOR gate is given to first flip flop [5] and the process repeats and this can be seen in the simulation waveform fig2(6).

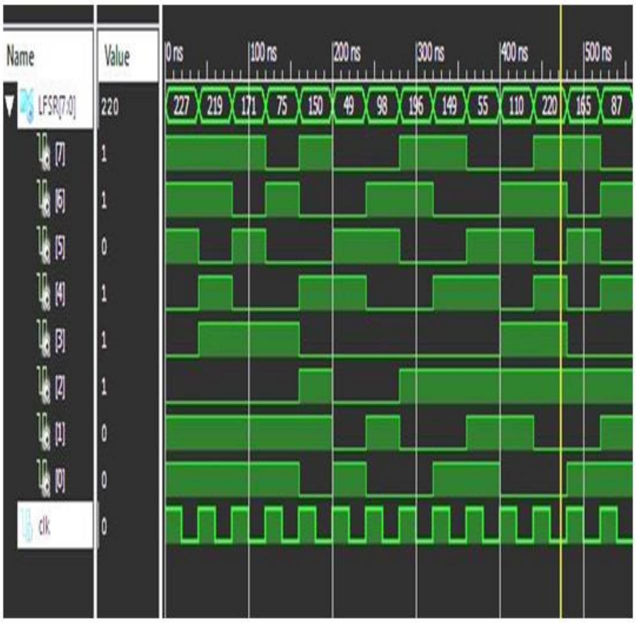


Fig. 2(6): Simulations obtained for 8bit LFSR

D. Generation of Random Numbers using Galois LFSR

Galois LFSR is named after French Mathematician EvaristeGalois .In Galois LFSR. The position of bits which changes the output is called taps thus When the clock is enabled the bits that are not taps are shifted one bit right unchanged and the bits position which acts as taps are XOR’d with the output bit before storing to next position. The new output bit acts as next input bit .The conclusion is whenever the output bit is 0all bits in registers shifts and whenever the output bit is 1 the bits in the taps gets flipped and entire LFSR gets shifted to one bit right position[6].
The Taps are based on the feedback polynomial. The feedback polynomial for 8bit LFSR is ,

$$F(x) = 8 + 6 + 5 + 4 + 1 \tag{1}$$

The representation of Galois LFSR based on feedback polynomial is as shown in fig 2(7).

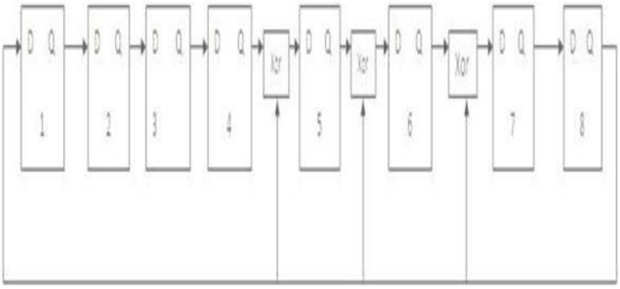


Fig. 2(7): Galois LFSR

When the selection lines are 011 Galois method gets selected for the key generation. Initially

some value is given into flip flops and when clock is enabled the output of 4th flip flop and output of 8th flip flop gets XOR’d and the output of XOR gate is given as input to the 5th flip flop and similarly the output of 5th flip flop and output of 8th flip flop gets XOR’d and result is given as input to 6th flip flop and finally the output of 6th flip flop and 8th flip flop are XOR’d and the result is given as input to 7th flip flop and the process continues and there by produces random numbers which can be seen in the simulations waveform in fig 2(8).

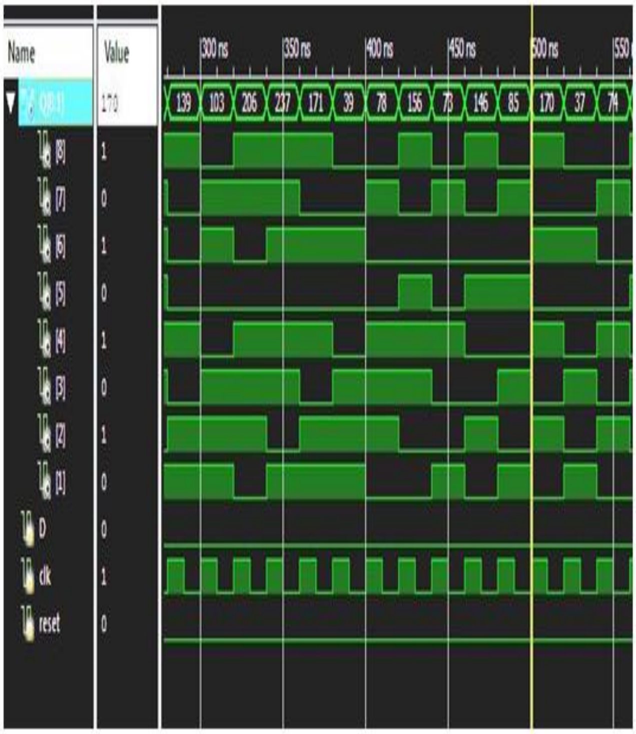


Fig. 2(8): Simulations obtained for 8bit Galois LFSR

E. Generation of random numbers using Fibonacci LFSR

Fibonacci LFSR is one of the LFSR used for the generation of random numbers. The bit position which influences the next state are called taps and the right most bit in the LFSR is called output bit.The taps are adjusted based on feedback polynomial .The feedback polynomial for 8 bit polynomial is equation1.The taps of the LFSR are XOR’d sequentially with the output bit and feedback to the leftmost bit .The Fibonacci LFSR is as shown in fig 2(9).

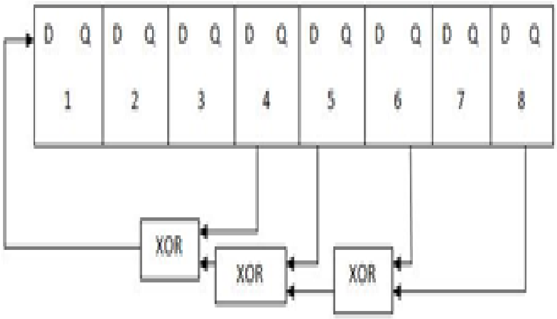


Fig. 2(9): Fibonacci LFSR

If the selection lines are 100 Fibonacci LFSR gets selected for the generation of encryption keys .Initially some value is stored in flip flop ,here we are taking 8bit LFSR so taps are selected based on feedback polynomial .Thus when clock is enabled the output of 6th and 8th flip flop are XOR’d and the outputis given as input to another XOR gate whose inputs are output of first XOR gate and output of 5th flip flop and the output of 2nd XOR gate is given as input to the 3rdXor gate whose inputs are output of 2nd XOR gate and output of 4th flip flop . Lastly the output of 3rd XOR gate is given as input to 1st flip flop [6] and the process continues and random numbers are generated which can be seen in simulation results in fig .2(10)

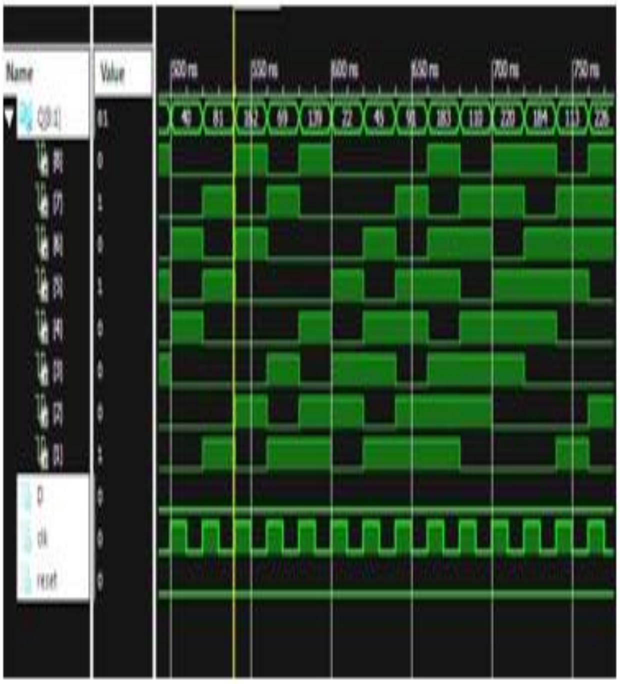


Fig. 2(10): Simulation results for Fibonacci LFSR

F. Generation of random numbers using combined LFSR/XOR.

Combined LFSR/XOR is random number generator which is combination of LFSR and a XOR. It requires at least two seed to produce random number which are used as key for encryption purpose. Combined LFSR/XOR requires (N-1) bit LFSR for N bit random number generation. The architecture for Combined LFSR/XOR is as shown in fig 2(11). The output of LFSR is XOR’d to produce N bit .By the usage Combined LFSR/XOR the architecture is reduced as (N-1) bit LFSR is used. Overall Structure of Combined LFSR/XOR is as shown in fig 2(12).

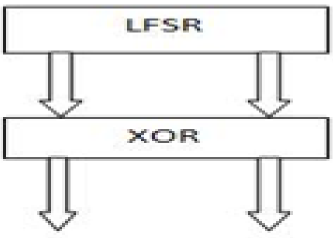


Fig. 2(11): Overview Architecture for Combined LFSR/XOR

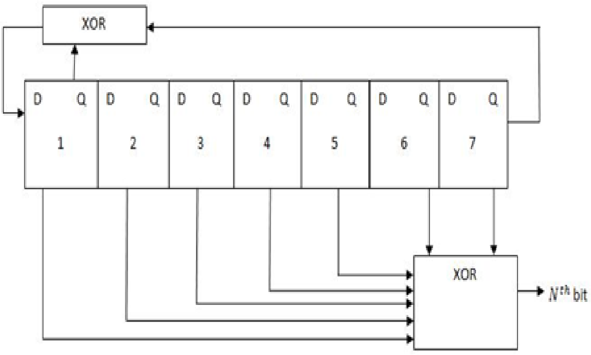


Fig. 2(12): Overall architecture of Combined LFSR/XOR

The overall architecture is as shown if the selection lines are 101 Combined LFSR/XOR method gets selected to produce random numbers. When the clock is enabled and some value is stored in flip flops and all the seven flip flop outputs are XOR'd and the output of XOR is taken as 8thbit. for the generation of random number, the output of 7th flip flop and the output of 1st flip flop are XOR'd and the output is given as input to 1st flip flop and the simulations results are shown in fig 2(13).

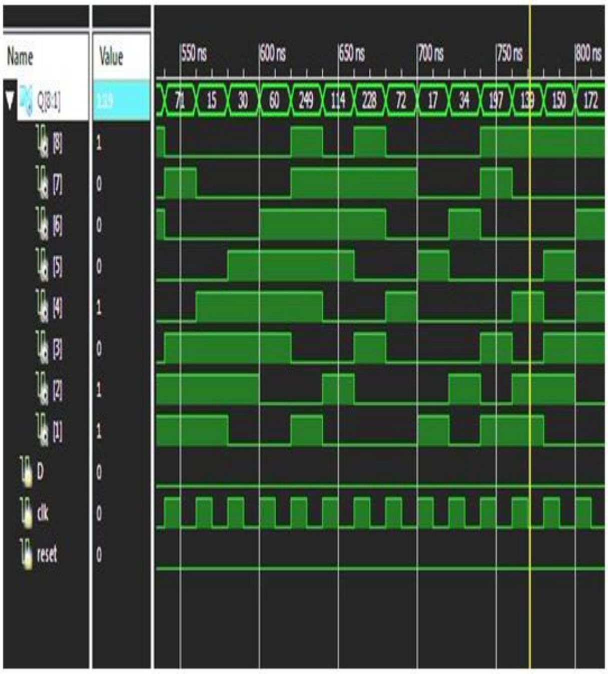


Fig. 2(13): Simulation results for Combined LFSR/ XOR

G. Generation of Random Numbers using Low power LFSR

In the present scenario where power reduction is the predominant issue. Different methods are proposed for the purpose of random number generation but generation using low power should also be taken into consideration. Clock gating is introduced to reduce the power consumption. For the purpose of clock gating we are using one NAND gate and one XOR gate. Average power consumed by the LFSR is reduced by using clock gating. The schematic for the clock gating is as shown in fig2(14).

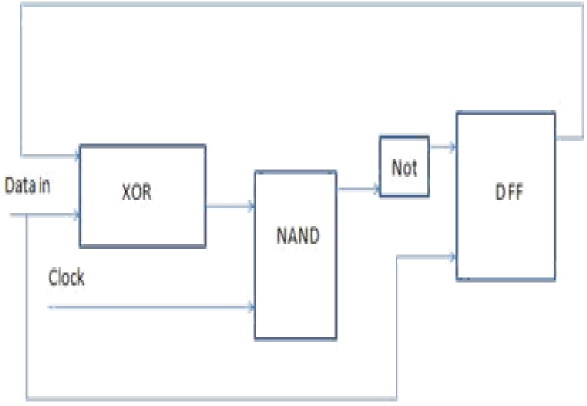


Fig. 2(14): Clock Gating

If the selection lines are 110 Low power LFSR [7] method for the generation of random numbers in which power consumption is reduced by using clock gating process. Concept after applying to the LFSR is as shown in fig.2(15).

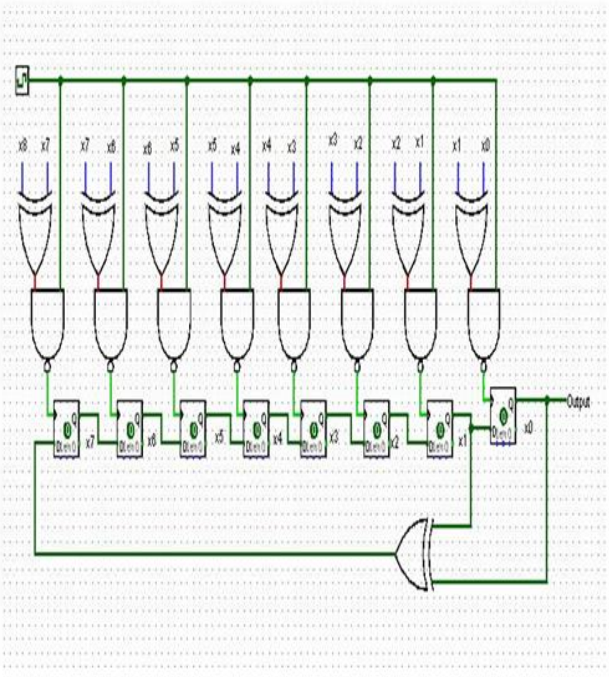


Fig. 2(15): Random numbers generated using Low power LFSR

When the selection lines are 110 low power LFSR method gets selected for random number generation. when the clock is enabled by usage of and gate and or gate clock gating is performed and thus power consumption has reduced and the output of 7th and 8th flip flops are XOR'd and the output is given as input to 1st flip flop [8]. Power consumption is reduced and random numbers are also generated which can be observed in the simulations results shown in fig.2(16).

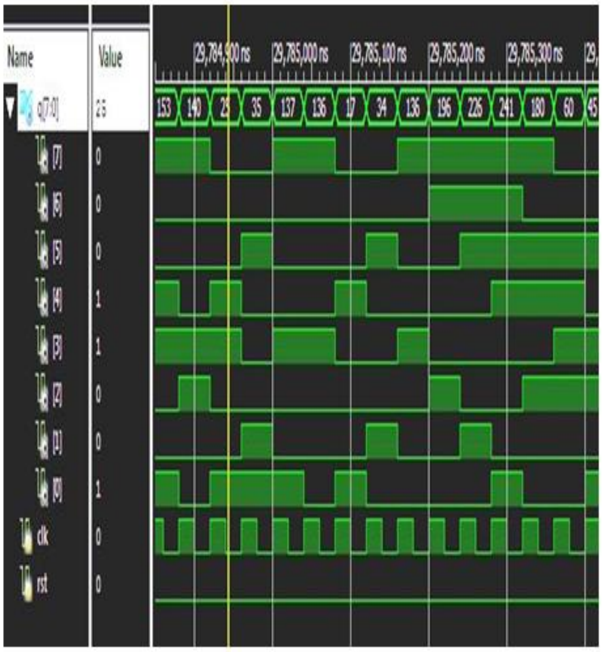


Fig. 2(16): Simulations obtained using Low power LFSR.

H. Generation of random numbers by run time polynomial change method

Run time polynomial uses a simple multiplexer, by the usage of selection lines taps for the LFSR gets selected as shown in fig 2(21.) The flip flop which are responsible for the change in output are known as taps i.e. if the selection line is 00 , 1 & 2 are made as taps for the 4bit LFSR .The architecture is as shown in fig 2(17)

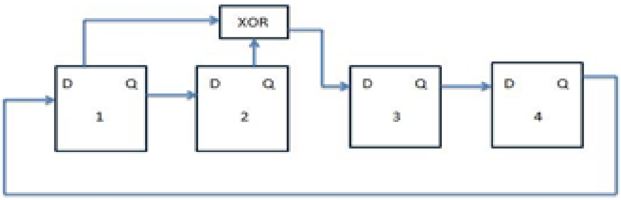


Fig. 2(17): Architecture of LFSR with taps 1&2

If the selection line is01, 2 & 3 are made as taps for the 4bit LFSR. The architecture is as shown in fig 2(18)

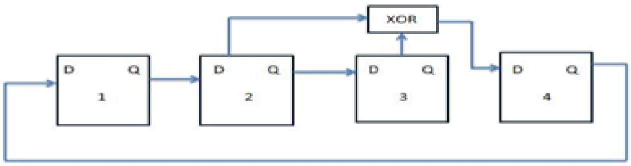


Fig 2. (18): Architecture of LFSR with taps 2&3

If the selection line is 10 , 3 & 4 are made as taps for the 4bit LFSR .The architecture is as shown in fig 2(19).

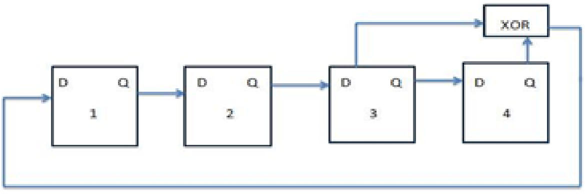


Fig. 2(19): Architecture of LFSR with taps 3&4

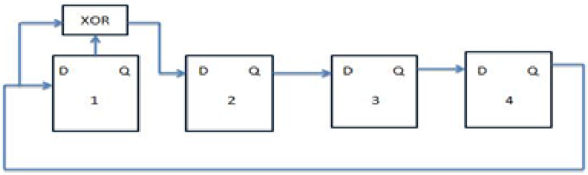


Fig. 2(20): Architecture of LFSR with taps 4&1

If the selection line is 11 , 4 & 1 are made as taps for the 4bit LFSR and as shown in fig 2(20) .The over allarchitecture is as shown in fig 2(21).

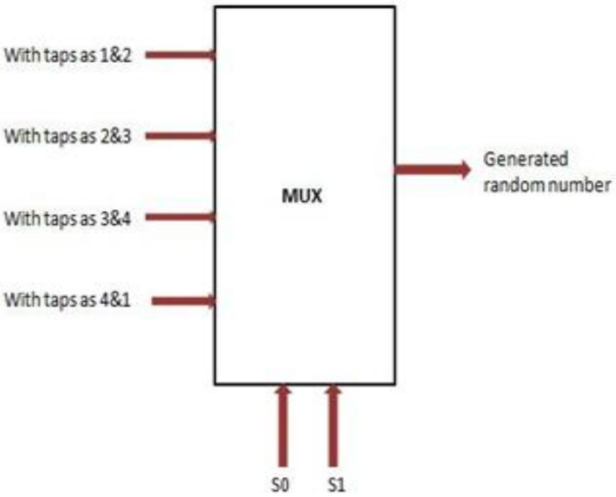


Fig. 2(21): Random number generation using Run time polynomial.

When selection lines are 111 run time polynomial change method gets selected for key generation. Due to the random selection lines different LFSR are evolved based on taps. Thus different numbers are produced with different LFSR that are produced due to different LFSR's simulations are as shown in fig 2(22).

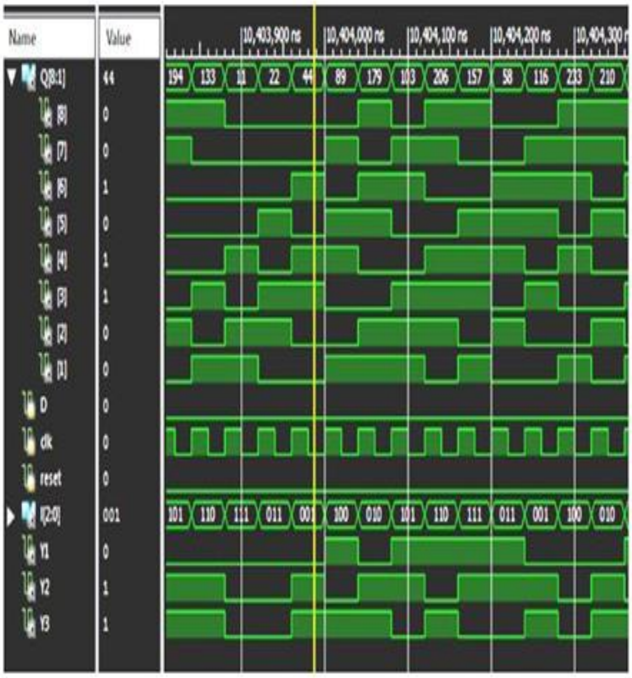


Fig. 2(22): Simulation for Random number generation using Run time polynomial

3. Random Key Generation through arbitrary selection inputs

By the usage of a 3 bit LFSR, the output of LFSR which is a random number from 0-7 is given as input to selection lines. Different methods get selected to produce random numbers. Thus key which is required for Encryption is obtained. The Architecture for the proposed model is shown in fig 3(1).

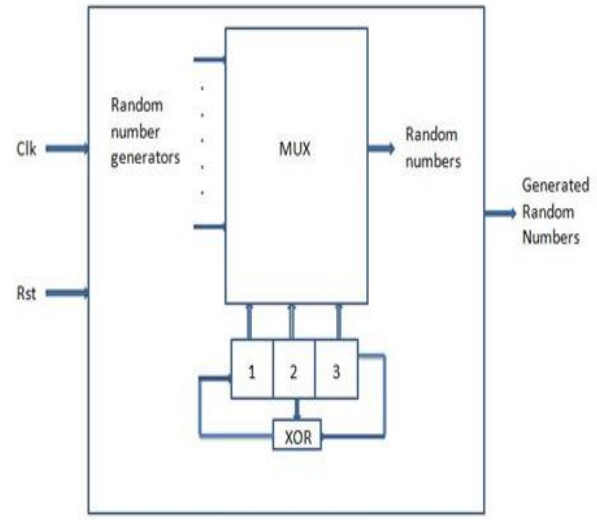


Fig. 3(1): Architecture for the proposed model.

The overall key generation process is done and simulations are shown in fig 3(2).

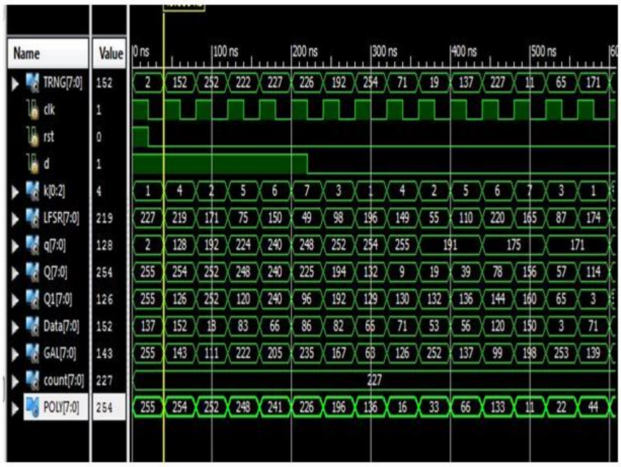

Name
Value
0 ns
100 ns
200 ns
300 ns
400 ns
500 ns
TRNG[7:0] 152 | 2 152 252 222 227 226 192 254 71 19 137 227 11 65 171
clk 1
rst 0
d 1
k[0:2] 4 | 1 4 2 5 6 7 3 1 4 2 5 6 7 3 1
LFSR[7:0] 219 | 227 219 171 75 150 49 98 196 149 55 110 220 165 87 174
q[7:0] 128 | 2 128 192 224 240 248 252 254 255 191 175 171
Q[7:0] 254 | 255 254 252 248 240 225 194 132 9 19 39 78 156 57 114
Q1[7:0] 126 | 255 126 252 120 240 96 192 129 130 132 136 144 160 65 3
Data[7:0] 152 | 137 152 18 83 66 86 82 66 71 53 56 120 150 3 71
GAL[7:0] 143 | 255 143 111 222 205 235 167 63 126 252 137 99 198 253 139
count[7:0] 227 | 227
POLY[7:0] 254 | 255 254 252 248 241 226 196 136 16 33 66 133 11 22 44

Fig. 3(2): Key Generation Mechanism

A. Implementation of Proposed System

All the above eight methods i.e. Random Key generation techniques are implemented in FPGA. Run time polynomial method is selectively produce random keys using any one of technique and performs permutations with message to produce cipher.

Fig. 3(3): Implementation of Low Power LFSR

Fig. 3(4): Implementation of LFSR

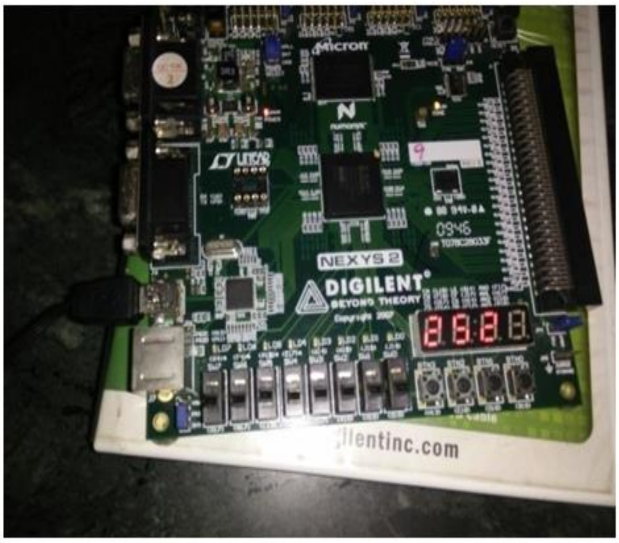
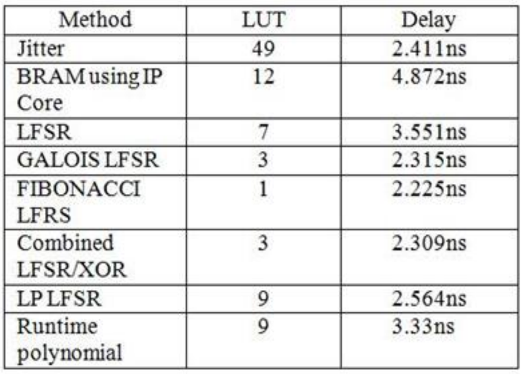


Fig 3(5): Implementation of BRAM

Table 3.1: Device Utilization Summary for Random Key Generation Mechanism



Method	LUT	Delay
Jitter	49	2.411ns
BRAM using IP Core	12	4.872ns
LFSR	7	3.551ns
GALOIS LFSR	3	2.315ns
FIBONACCI LFRS	1	2.225ns
Combined LFSR/XOR	3	2.309ns
LP LFSR	9	2.564ns
Runtime polynomial	9	3.33ns

4. Conclusion

Pseudo random keys are generated using different methods. Several methods are configured randomly in run time based on priority of section inputs which produces random keys. We observed that Jitter process generates random numbers based on clock frequency triggered to oscillators, run time polynomial method produces random keys by selectively changing tap positions in LFSR both methods will be more secure, predictable and attains high security, but it Jitter consumes more Look up Tables (LUT) when compared to other methods. Pre stored random numbers in memory are generated using BRAM with IP core generator and this method generates keys at a span of delay. By the usage of LFSR methods like Galois LFSR and Fibonacci LFSR, Combined LFSR/XOR and by the usage of Low power LFSR power and resources can be saved. By using clock gating architecture power consumption can be reduced through Low power LFSR. Fibonacci LFSR uses less resources and delay in comparison with other methods. Device utilization summary for above mentioned random generation methods are placed in the table3. 1. Look up Tables and delays for different methods have analyzed using Xilinx Synthesis Technology (XST). The advantage of proposed model is to produce high randomness in generated key as it is used for cryptography applications.

References

[1] Anju P. Johnson Member, IEEE, RajatSubhra Chakraborty Senior Member, IEEE and DebdeepMukhopadyayMember, IEEE "An Improved DCM-based Tunable True Random Number Generator for Xilinx FPGA" 1549-7747, 2016 IEEE, DOI 10.1109/ TCSII. 2016.2566262, IEEE Transactions on Circuits and Systems II: Express Briefs.

[2] Andrei Marghescu1), Paul Svasta2), Emil Simion3)" Politehnica " University of Bucharest, Romania1) 2) CETTI, Romania " Optimizing Ring Oscillator-based True Random Number Generators Concept on FPGA" 978-1-5090-1389-0/16 2016 IEEE.

[3] Qianying Tang, Bongjin Kim, Yingjie Lao, Keshab K. Parhi, and Chris H. Kim University of Minnesota, Minneapolis, MN 55455 USA," True Random Number Generator Circuits Based on Single- and Multi-Phase Beat Frequency Detection", 978-1-4799-3286-3/14/ 2014 IEEE.

[4] Neha Agrawal, Neelesh Gupta, Neetu Sharma," Linear Feed Back Shift Register Base Multiple Long Period Random Binary Sequences Generator" International Journal of Computer Applications (0975-8887) Volume 138 – No.3, March 2016.

[5] Amit Kumar Panda, Praveena Raj put, Bhawna Shukla Dept. of ECE, IT Guru GhasidasVishwavidyalaya " FPGA Implementation of 8, 16 and 32 Bit LFSR with Maximum Length Feedback Polynomial using VHDL" 978-0-7695-4692-6/12 2012 IEEE.

[6] Shivshankar Mishra1 , Ram Racksha Tripathi2 and Devendra Kr. Tripathi" Implementation of Configurable Linear Feedback Shift Register in VHDL" 978-1-5090-2118-5/16 2016 IEEE.

[7] M.S Shyam, M.Tech (VLSI&ES), Nagadastagiri Reddy Suddamalla" Low Power Linear Feedback Shift Register For Random Pattern Generation in BIST" international Journal of Advanced Scientific Technologies in Engineering and Management Sciences (IJASTEMS-ISSN:2454-366X)

[8] Review On Power Optimized TPG Using LP-LFSR For Low Power BIST" 978-1-4673-9214-3/16/ 2016 IEEE

[9] GU Xiao-chen, ZHANG Min-xuan," Uniform Random Number Generator using Leap-Ahead LFSR Architecture", 978-0-7695-3906-5/09 2009 IEEE.

[10] SitiHazwani, Sheroz Khan, Mohammad Umar Siddiqi, Khalid A.S. Al-Khateeb, Mohamed HadiHabaebi, ZeeshanShahid," Randomness Analysis of Pseudo Random Noise Generator Using 24-bits LFSR" 2166-0662/14 ,2014 IEEE.

[11] Chengani Vinod Chandra1 S.Ramasamy2," TEST PATTERN GENERATION FOR BENCHMARK CIRCUITS using LFSR", IEEE - 31661 4th ICCCNT 2013.

[12] E. Aleksejev, A.Jutman, R.Ubar," LFSR Polynomial and Seed Selection Using Genetic Algorithm", 1-4244-0415-0/06 IEEE..

[13] C. P. Souza1, F. M. Assis2, R. C. S. Freire3," Mixed Test Pattern Generation Using a Single Parallel LFSR", 0-7803-9360-0/06 2006 IEEE.

[14] HENK HOLLMANN," Design of Test Sequences for VLSI Self-Testing Using LFSR", OOlS-9448/9O/O300-0386$O1 .OO 0 1990 IEEE. Balwinder Singh, Arun Khosla, SukhleenBindra" Power Optimization of Linear Feedback Shift Register (LFSR) for Low Power BIST" 2009 IEEE International Advance Computing Conference (IACC 2009) Patiala, India.

[15] Dr. Seetaiah Kilaru, Hari Kishore K, Sravani T, Anvesh Chowdary L, Balaji T "Review and Analysis of Promising Technologies with Respect to fifth Generation Networks", 2014 First International Conference on Networks & Soft Computing, ISSN:978-1-4799-3486-7/14,pp.270-273,August2014.

[16] Meka Bharadwaj, Hari Kishore "Enhanced Launch-Off-Capture Testing Using BIST Designs" Journal of Engineering and Applied Sciences, ISSN No: 1816-949X, Vol No.12, Issue No.3, page: 636-643, April 2017.

[17] N Bala Dastagiri, Kakarla Hari Kishore "Reduction of Kickback Noise in Latched Comparators for Cardiac IMDs" Indian Journal of Science and Technology, ISSN No: 0974-6846, Vol No.9, Issue No.43, Page: 1-6, November 2016.

[18] A Murali, K Hari Kishore, D Venkat Reddy "Integrating FPGAs with Trigger Circuitry Core System Insertions for Observability in Debugging Process" Journal of Engineering and Applied Sciences, ISSN No: 1816-949X, Vol No.11, Issue No.12, page: 2643-2650, December 2016.

[19] Mahesh Mudavath, K Hari Kishore "Design of CMOS RF Front-End of Low Noise Amplifier for LTE System Applications Integrating FPGAs" Asian Journal of Information Technology, ISSN No: 1682-3915, Vol No.15, Issue No.20, page: 4040-4047, December 2016.

[20] P Bala Gopal, K Hari Kishore, B.Praveen Kittu "An FPGA Implementation of On Chip UART Testing with BIST Techniques", International Journal of Applied Engineering Research, ISSN 0973-4562, Volume 10, Number 14 , pp. 34047-34051, August 2015

[21] S Nazeer Hussain, K Hari Kishore "Computational Optimization of Placement and Routing using Genetic Algorithm" Indian Journal of Science and Technology, ISSN No: 0974-6846, Vol No.9, Issue No.47, page: 1-4, December 2016.

[22] N Bala Gopal, K Hari Kishore "Analysis of Low Power Low Kickback Noise in Dynamic Comparators in Pacemakers" Indian Journal of Science and Technology, ISSN No: 0974-6846, Vol No.9, Issue No.44, page: 1-4, November 2016.

[23] T. Padmapriya and V. Saminadan, "Priority based fair resource allocation and Admission Control Technique for Multi-user Multi-class downlink Traffic in LTE-Advanced Networks", International Journal of Advanced Research, vol.5, no.1, pp.1633-1641, January 2017.

[24] S.V.Manikanthan and K.Baskaran "Low Cost VLSI Design Implementation of Sorting Network for ACSFD in Wireless Sensor Network", CiiT International Journal of Programmable Device Circuits and Systems, Print: ISSN 0974–973X & Online: ISSN 0974–9624, Issue: November 2011, PDCS112011008.

[25] Rajesh, M., and J. M. Gnanasekar. &quot;Congestion control in heterogeneous wireless ad hoc network using FRCC.&quot; Australian Journal of Basic and Applied Sciences 9.7 (2015): 698-702.

[26] S.V.Manikanthan and V.Rama"Optimal Performance Of Key Predistribution Protocol In Wireless Sensor Networks" International Innovative Research Journal of Engineering and Technology ,ISSN NO: 2456-1983,Vol-2,Issue –Special –March 2017.