# Architectural Design of Random Number Generators and Their Hardware Implementations

**Sarmad Fakhrulddin Ismael**
University of Mosul/computer Engineering Department
sarmad.fakhraldeen@gmail.com

**Dr. Basil Shukr Mahmood**
University of Ninevah
basil_mahmood@yahoo.com

## Abstract

The architectural design of the random number generators for uniform distribution, normal distribution, exponential distribution and Rayleigh distribution using Box-Muller and inverse transformation method has been hardware implemented on FPGA. Any of the random number generators can generate one sample every clock cycle. The generators have been implemented on Xilinx Spartan 3E XC3S500E FPGA. The designed generators work properly up to maximum frequency of 418.41MHz .The outcome results of the generators have been tested by the chi-square test at a 5% level of significance which provided the correct required distributions.

**Keyword: Box-mulle, Chi-square, Inverse transformation, FPGA.**

## تصميم معمارية لتوليد الارقام العشوائية وتنفيذها مادياً

**سرمد فخر الدين إسماعيل**
قسم هندسة الحاسوب/ جامعة الموصل

**د. باسل شكر محمود**
كلية هندسة ألألكترونيات

## الملخص

المعمارية المصممة لتوليد الارقام العشوائية بتوزيع منتظم وتوزيع طبيعي وتوزيع اسي وتوزيع (رايلي ) باستخدام طريقة الـ (Box-muller) وطريقة التحويل العكسي تم بناءها ماديا باستخدام ال FPGA. اي واحد من مولدات الارقام العشوائية ممكن ان تولد رقم واحد في كل دورة. المولدات تم بناءها على Xilinx Spartan 3E XC3S500E FPGA. المولدات المصممة مناسبة للعمل بتردد مقداره 418.41MHz النتائج التي تم الحصول عليها من المولدات تم اختبارها بواسطة فحص مربع كاي بمستوى اهمية مقدارها 5% والتي حققت التوزيع المطلوب.

**Received: 9 – 5 - 2013** **Accepted: 10 – 10 - 2013**

## 1. Introduction

Random number generators are used in a large number of computationally intensive modeling and simulation applications such as traffic light simulation [1],communication system [2],cryptography system [3] ,the most commonly random generators used are uniform distribution, normal distribution ,exponential distribution and Rayleigh distribution. All types of random number generators can be derived from the uniform random generators. The most famous methods for mapping between the generators are Box-muller method and inverse transformation method.

There have been many researches on their hardware implementation to generate random number, A VLSI implementation of universal random number generation with uniform distribution, exponential distribution, Rayleigh distribution and Rayleigh distribution is proposed by CUI, LI and et.al. [4]. A hardware Gaussian noise generator using the box-muller method and the error analysis of its elementary function is proposed by Dong-U, John and et.al. [5], A hardware designs for generating random numbers from arbitrary distributions using the inversion method are proposed by Ray, Dong-U and et.al. [6]. A ZIGGURAT based method to generate random number with Gaussian distribution is proposed by Guanglie, Philip and et.al. [7], FPGA implementation and performance analysis of 8, 16, and 32 bit LFSR pseudo random number generator system to generate a uniform distribution is proposed by Amit , Praveena and et.al. [8]. A random number is generated by various methods such as LFSR and linear congruental generator and blum blum shub to generate random number with uniform distribution is proposed by Jay, Sudhanshu and et.al. [9].

The paper is organized as follows. In section 2 the background and theory of the architectural design is presented. Section 3 shows the hardware Implementation details. The Experimental results are given in section 4. Section 5 concludes this paper.

## 2. Background and Theory

There are many methods to generate random number with arbitrary distribution, most of these method based on converting the random number with uniform distribution to another type of distribution. Review of the methods used in this research is shown below:

### 2.1 Box-muller method:

This method generates a pair of random variables with normal distribution from two uniformly distributions over (0,1) with zero mean and standard deviation =1 (N(0,1))[10]

$$X1 = \sqrt{-2\ln U1} \quad cos 2\pi U2 \qquad (1)$$

$$X2 = \sqrt{-2\ln U1} \quad sin 2\pi U2 \qquad (2)$$

U1and U2 are random variables with uniform distributions.
X1 and X2 are random variables with a normal distributions.
Thus the proposed architecture is based on this method to generate normal distribution.

## 2.2 The inverse transformation method (ITM):

The inversion method for generating non-uniform random variables is based on the simple observation that a random variable X with F as cumulative distribution function (CDF) can be generated by:

$$X = F^{-1}(U) \tag{3}$$

Where U denotes a uniform U (0, 1) random variable and the X is a random variable with desired distribution [11].

The proposed architectures for exponential distribution and Rayleigh distribution are based on this method.

The required steps to compute the desired random variable using inverse transformation method are:

1- Compute the CDF of the desired random variable *X*.
2- Set *F*(*X*) = *U*.
3- Solve the equation *F*(*X*) = *U* for *X* in terms of *U*.
4- Generate uniform random numbers *U*1, *U*2, *U*3, ... and compute The desired random variable by $X\,i = F^{-1}(U\,i)$

By applying the above steps for exponential distribution:
- Since the PDF (probability density function) for exponential distribution:

$$f(x) = \lambda e^{-\lambda x} \tag{4}$$

Then the CDF will be:

$$F(x) = 1 - e^{-\lambda x} \tag{5}$$

$$F(x) = U \tag{6}$$

$$x = -\frac{\ln(1-U)}{\lambda} \tag{7}$$

For more simplicity, because the U is an interval between (0,1) then (1-U) is also an interval between (0,1) and eq. (7) can be approximated as follows:

$$x = -\frac{\ln(U)}{\lambda} \tag{8}$$

Repeating the above steps for Rayleigh distribution results:
The PDF for Rayleigh distribution:

$$f(x) = \frac{x}{\sigma^2} e^{\frac{-x^2}{2\sigma^2}} \tag{9}$$

Then its CDF will be:

$$F(x) = 1 - e^{\frac{-x^2}{2\sigma^2}} \tag{10}$$

$$F(x) = U \tag{11}$$

$$x = \sigma\sqrt{-2\ln U} \tag{12}$$

## 3. Hardware Implementation Details

### 3.1 Linear Feedback Shift Register (LFSR):

It is a shift register whose input bit is a linear function of its previous state. The most commonly used linear function is XOR. Thus, an LFSR is most often a shift register whose input bit is driven by the exclusive-or (XOR) of some bits of the overall shift register value deterministic. The initial value of the LFSR is called the seed. The stream of the values produced by the register is completely determined by its current (or previous) state. Likewise, because the register has a finite number of possible states, it must eventually enter a repeating cycle. However, an LFSR with a well-chosen feedback function can produce a sequence of bits which appears random and which has a very long cycle. [8][9][12].

The maximum cycle length of the uniform random variable generated from LFSR is equal $2^n$-1 where n: is the length of shift register in LFSR. In this paper we use (n=12bit) then the number of random variable = 4k =4096 numbers.

To choose feedback polynomials that generate maximum period we use a table that is presented by Xilinx in [13]. The feedback polynomial for 12-bit is $x^{12} + x^6 + x^4 + x + 1$ . Figure 1 shows the architecture design for 12-bit LFSR

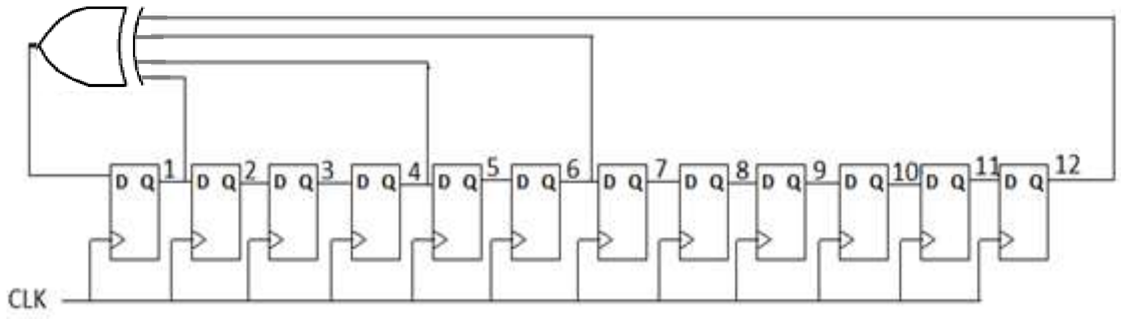


Figure (1): Architecture design for 12-bit LFSR

### 3.2 The proposed architecture:

Figure 2 shows the design of the proposed architecture. Since it has been used the two 12-bit LFSR to generate two uniform random variables, then look-up table method (LUT) is used to evaluate the two terms in eq. (2) for Box-muller method $y1 = \sqrt{-2\ln U1}$ and $y2 = \sin 2\pi U2$ ,since the number of states in LFSR is 4096 state ,then the number of y1 and y2 is also 4096 states and the architecture uses 4 block RAMs in Spartan3E (each block ram is 1k*16bit ) for each term (y1 and y2) then y1 and y2 have use to generate many distribution as shown in figure 2:

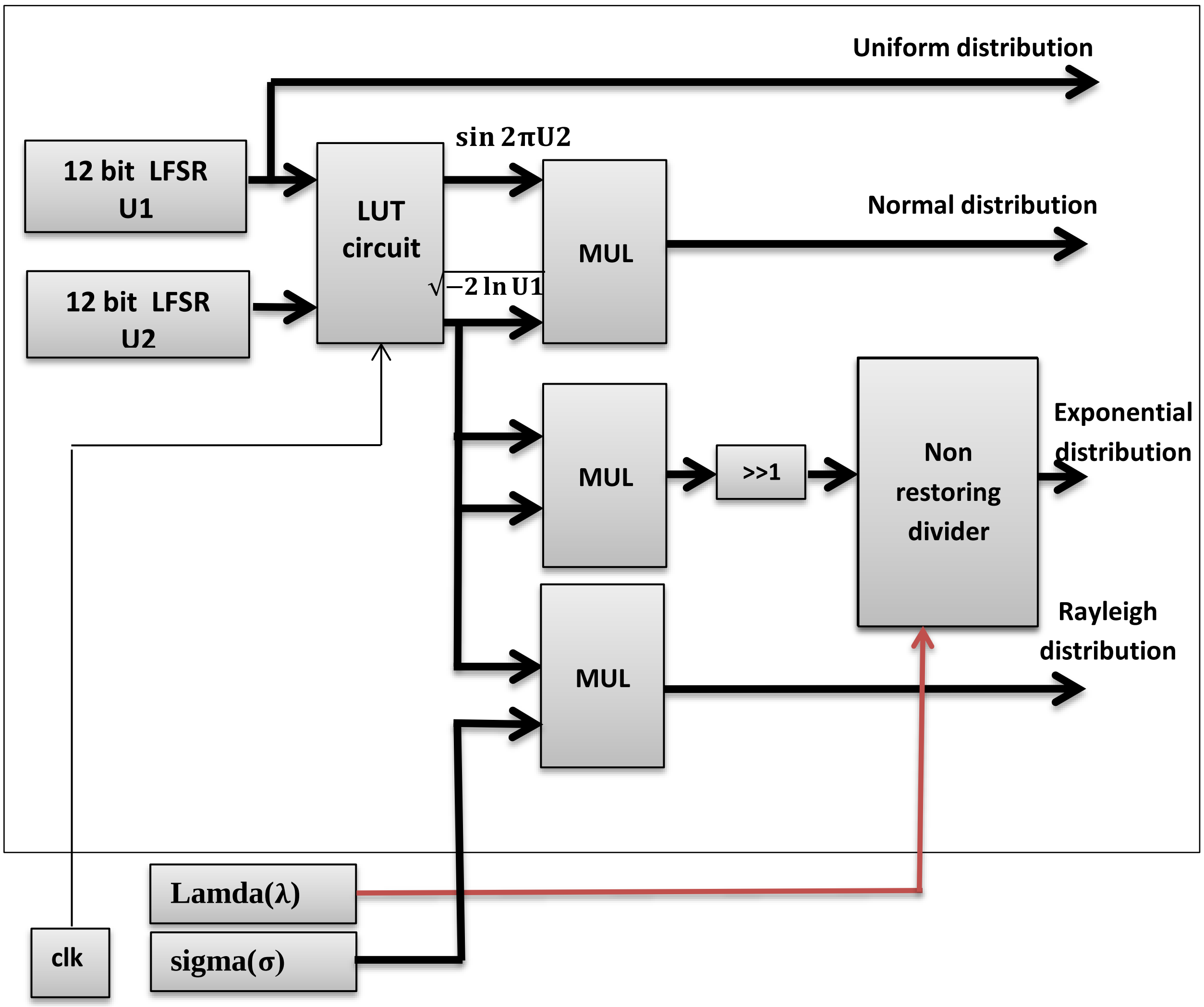


Figure (2): The proposed architecture of the random number generators

The LUT circuit is shown in figure 3, the circuit uses 8 block RAMs (4 block RAMs for each term y1 and y2) each block RAM is 1kx16bit .The 30-bit non restoring divider [14] shown in figure(2) is used to divide the value –lnU1 by lamda ($\lambda$) as input in order to generate the exponential distribution. The non-restoring divider technique is used to increase the throughput [14]. The Fixed point package [15] is used for representing variables in the VHDL code. Table (1) shows the length and the format of each variable used in the design.

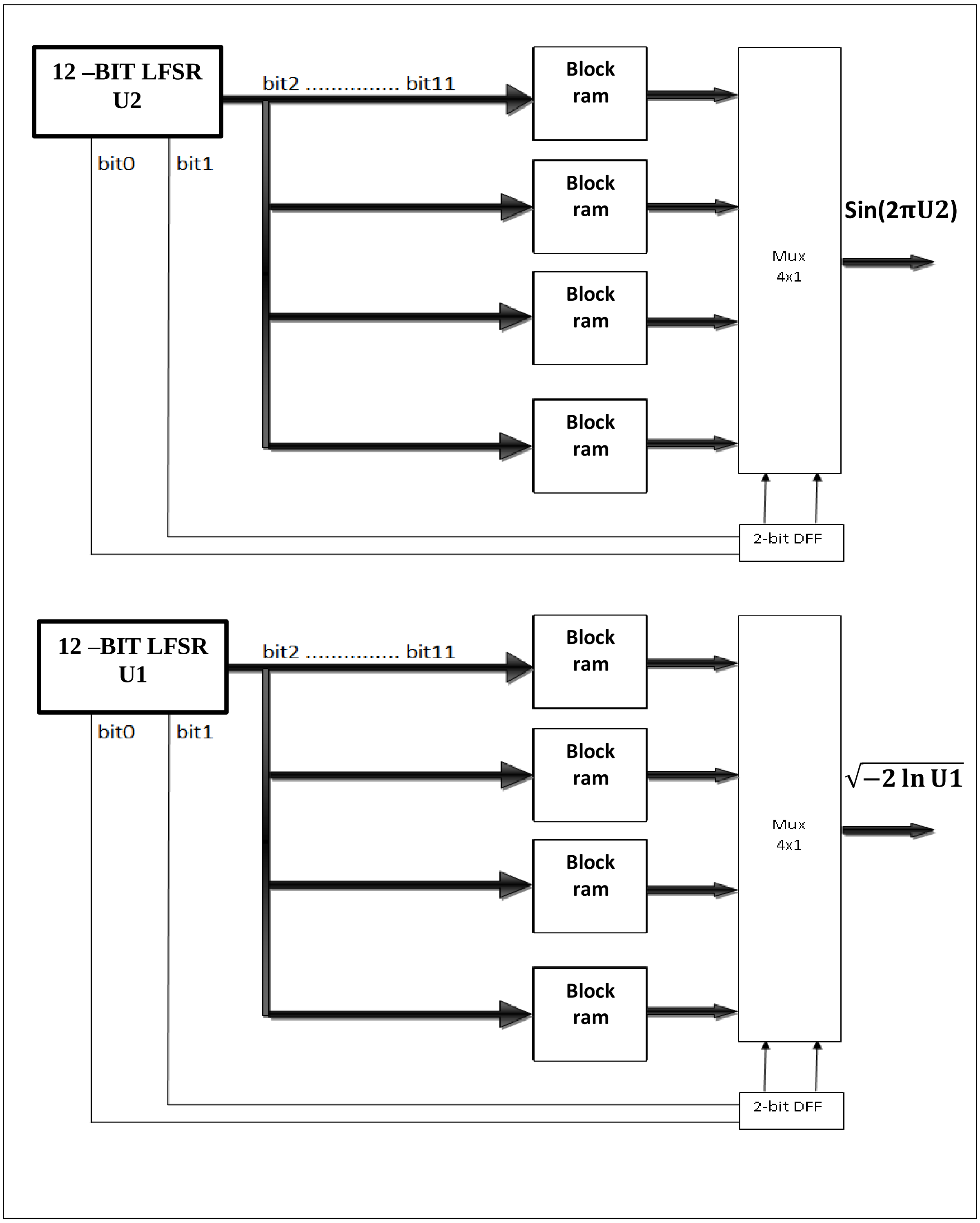


Figure (3): LUT circuit design for the evaluation the elementary functions

**Table (1): Number of Bits Representation for each Element**

| variable | No. of bit for real part with sign bit | No. of bit for fraction part | Total No. of bit |
|---|---|---|---|
| u1,u2 | 0 | 12 | 12 |
| $\sqrt{-2\ln U1}$ | 4 | 12 | 16 |
| sin2$\pi$u2 | 2 | 14 | 16 |
| lamda | 4 | 8 | 12 |
| sigma | 5 | 8 | 13 |
| uniform distribution | 0 | 12 | 12 |
| normal distribution | 6 | 26 | 32 |
| Exponential distribution | 13 | 17 | 30 |
| rayleigh distribution | 9 | 20 | 29 |

## 4. Experimental results

### 4.1 Simulation Results:

The complete architecture design was implemented on Xilinx Spartan 3E XC3S500E FPGA using ISE14.1 and the results of simulation for lamda=2 and sigma=2 are shown in figure 4. It is clear from the simulation results that all the numbers for all distributions are generated in every clock cycle.

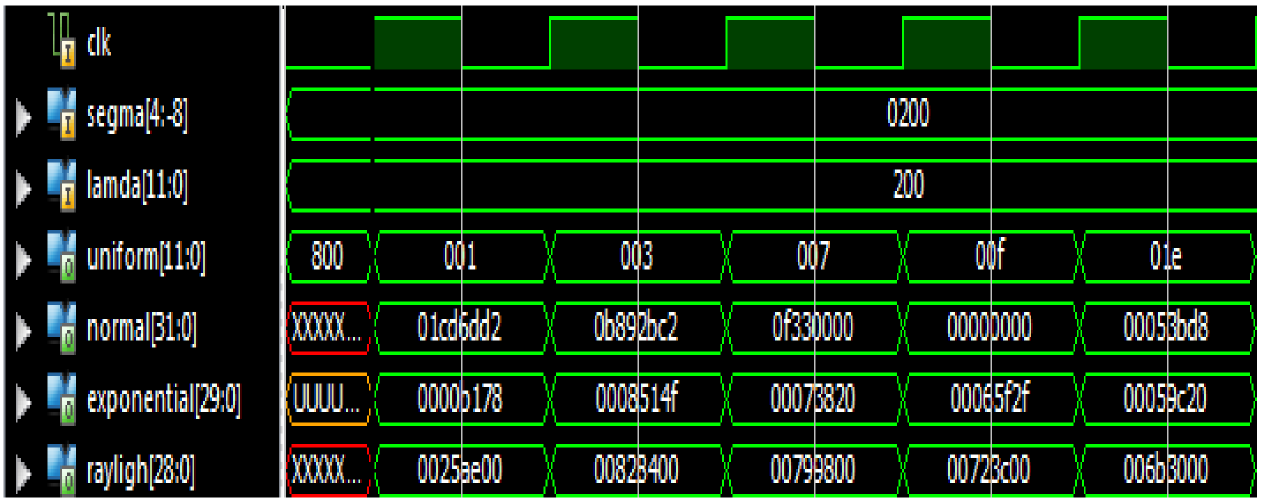


Figure (4): Simulation Results for the Proposed Architecture

The distributions for the generated numbers of the four distributions were plotted as shown in figure 5 using Matlab program.

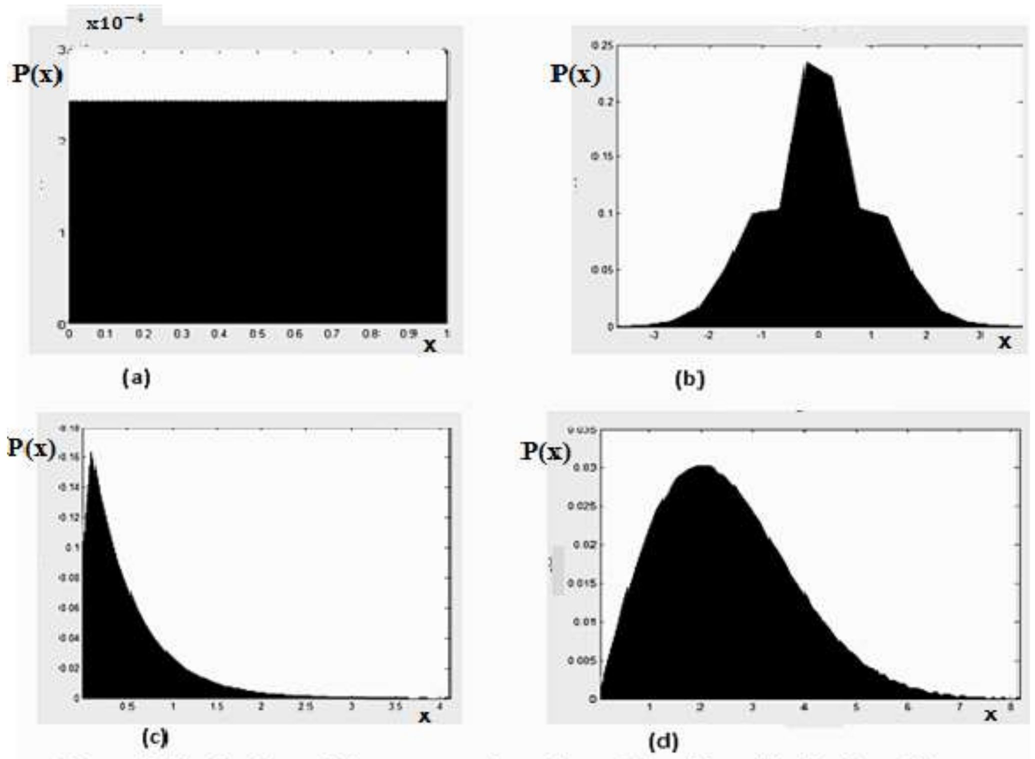
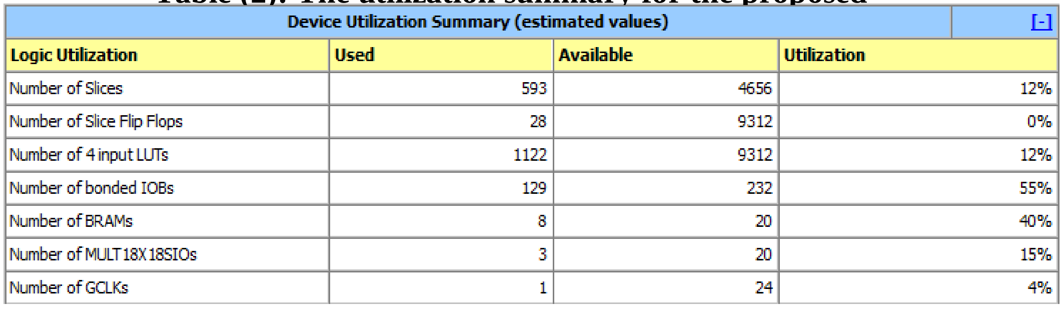


Figure 5 distributions of the generated numbers: (a) uniform distribution , (b) normal distribution N(0,1)(c) Exponential distribution with λ=2 (d) Rayleigh distribution with σ=2

## 4.2 Speed and resources utilization:

After synthesizing the design by ISE14.1, it is found that the maximum operating frequency becomes 418.41MHz. The resources utilization summary from the FPGA chip is shown in table 2.

Table (2): The utilization summary for the proposed



| Device Utilization Summary (estimated values) | | | [-] |
|---|---|---|---|
| **Logic Utilization** | **Used** | **Available** | **Utilization** |
| Number of Slices | 593 | 4656 | 12% |
| Number of Slice Flip Flops | 28 | 9312 | 0% |
| Number of 4 input LUTs | 1122 | 9312 | 12% |
| Number of bonded IOBs | 129 | 232 | 55% |
| Number of BRAMs | 8 | 20 | 40% |
| Number of MULT18X18SIOs | 3 | 20 | 15% |
| Number of GCLKs | 1 | 24 | 4% |

To make a fair comparison, Table 3 shows a comparison between our design and three previous related works .These works were implemented on other FPGA target kit so we resynthesized our design on another FPGA target kit as shown in Table 3.

**Table (3): comparisons of different RNG implemented on different Xilinx FPGA target kit**

| Design | [5] | [6] | This work | This work |
|---|---|---|---|---|
| Method / type distributions | Box-Muller/ Gaussian | Inversion method/ Gaussian, exponential and log-normal | Box-Muller and inversion method/ uniform, normal, exponential and Rayleigh | Box-Muller and inversion method/ uniform, normal, exponential and Rayleigh |
| Target kit | Xilinx Virtex-4 XC4VLX 100-12 FPGA | Xilinx Virtex-4 XC4VLX 100-12 FPGA | Xilinx Spartan 3E XC3S500E FPGA | Xilinx Virtex-4 XC4VLX 100-12 FPGA |
| slices | 1528 | 487 | 593 | 593 |
| Block rams | 3 | 2 | 8 | 8 |
| MULT18x18 | 12 | 2 | 3 | 3 |
| clk speed [MHz] | 233 | 371 | 418.41 | 549.753 |
| Sample/clk | 2 | 1 | 4 | 4 |

## 4.3 Statistical test:

The chi-square Goodness of fit (GoF) test is used for testing the distribution characteristic and used to establish whether an assumed distribution is correct [4][16]. Table 4 show the test results for the generated numbers of the four distributions. For each test, 1000 sample were considered.

**Tables (4): Chi-square test results for uniform, normal distribution, and exponential distribution and Rayleigh distribution**

| Distribution type | Degree of freedom | Statistical quantity | Test result |
|---|---|---|---|
| Uniform | 5 | 2.958 | Accept distribution |
| Normal | 5 | 4.965 | Accept distribution |
| Exponential | 6 | 3.904 | Accept distribution |
| Rayleigh | 6 | 4.632 | Accept distribution |

All tested distributions passed the chi-square test at a 5% level of significant.

## 5- Conclusions

The proposed architecture generates all the four distinct distributions with better performance by using a small area and high frequency of Spartan 3E chip and few clock cycles. The used area of Spartan 3E is 13% and the maximum operating frequency becomes 418.41MHz. This enhancement in performance is due to the usage of LUT for constructing the elementary functions and also applying the modified non-restoring division algorithm. The proposed architecture is used as good in real time application. To check the Goodness of fit (GoF) the resulted numbers pass the chi-square test at 5% level of significance.

## 6- References

[1] Zhen Shen, Kai Wang And Fenghua Zhu,"Agent-Based Traffic Simulation And Traffic Signal Timing Optimization With GPU", 14th International IEEE Conference On Intelligent Transportation Systems, October 5-7, 2011, pp. 145-150.

[2] Dong-U Lee, Wayne Luk, John Villasenor And Peter Y.K. Cheung," A Hardware Gaussian Noise Generator For Channel Code Evaluation",FCCM '03 Proceedings Of The 11th Annual IEEE Symposium On Field-Programmable Custom Computing, 2003, pp. 69-78.
[3] Franciszek Seredynski, Pascal Bouvry and Albert Y. Zomaya", Cellular Automata Computations And Secret Key Cryptography", parallel computing, Volume 30, Issues 5-6, May–June 2004, pp. 753–766.
[4] CUI Wei, LI Chengshu And SUN Xin "FPGA Implementation Of Universal Random Number Generator", Proceedings. ICSP'04. 7th International Conference on , IEEE, Vol. 1, 2004, pp. 495–498.
[5] Dong-U Lee, John D. Villasenor, Wayne Luk And Philip H.W. Leong," A Hardware Gaussian Noise Generator Using The Box-Muller Method And Its Error Analysis" , IEEE transactions on computers, Vol. 55, No. 6, June 2006, pp. 659-671.
[6] Ray C. C. Cheung, Dong-U Lee, Wayne Luk And John D. Villasenor," Hardware Generation Of Arbitrary Random Number Distributions From Uniform Distributions Via The Inversion Method" , IEEE transactions on very large scale integration (VLSI) systems, Vol. 15, No. 8, August 2007, pp. 952-962.
[7] Guanglie Zhang, Philip H.W. Leong, Dong-U Leey, John D. Villasenor Ray C.C. Cheung and Wayne Luk," Ziggurat-Based Hardware Gaussian Random Number Generator " International Conference On Field Programmable Logic And Applications, 2005, pp. 275-280.
[8] Amit Kumar Panda, Praveena Rajput And Bhawna Shukla, "Design Of Multi Bit LFSR PNRG And Performance Comparison On FPGA Using VHDL", International Journal of Advances in Engineering & Technology (IJAET), Vol. 3, Issue 1, , 2012, pp. 566-571.
[9] Jay Kumar1, Sudhanshu Shukla, Dhiraj Prakash, Pratyush Mishra And Sudhir Kumar, "Random Number Generator Using Various Techniques Through VHDL", International Journal Of Computer Applications In Engineering Sciences, Vol. I, Issue II, June 2011.
[10] G. Box and M. Muller, "A Note on the Generation of Random Normal Deviates", Annals Math. Statistics, Vol. 29, 1958, pp. 610-611.
[11] W. Ho¨rmann and J. Leydold, "Continuous Random Variate Generation by Fast Numerical Inversion," ACM Trans. Modeling and Computer Simulation, Vol. 13, No. 4, 2003, pp. 347-362.
[12] Linear Feedback Shift Register ,Wikipedia website. [Online]. Available: http://en.wikipedia.org/wiki/Linear_feedback_shift_register 2013.
[13] Efficient Shift Registers, LFSR Counters, and Long Pseudo-Random Sequence Generators, http://www.xilinx.com/support/documentation/application_notes/xapp052.pdf
[14] Bojan Jovanovic and Milun Jevtic,"FPGA implementation of throughput increasing techniques of the binary dividers", international scientific conference,19-20 November 2010, pp. 397-401.
[15] David Bishop, "Fixed point package user's guide", http://www.vhdl.org/vhdl-200x/vhdl-200x-ft/packages/files.html,2006.
[16] Jorge Luis Romeu,"The Chi-square: a large-sample Goodness of Fit test", Vol. 10, No. 4, 2003-2004.

The work was carried out at the college of Engineering. University of Mosul