ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com
International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 3, Special Issue 19, April 2016

RANDOM NUMBER GENERATION USING WELL METHOD

[1]A.N.Duraivel, [2]A.Komathi, [3]R.Arthiereena, [4]V.Aruna

[1]Associate professor, Dept of ECE, kings engineering college,
[2,3,4] UG Students, Dept of ECE, kings engineering college,

[1]duraivel.n@gmail.com, [2]komathianandhan@gmail.com, [3]arthiereena@gmail.com, [4]varuna3095@gmail.com

Abstract : **High quality random numbers are of critical importance to many scientific applications. Being the key component of various scientific applications, designing PRNGs that can rapidly provide independent parallel streams of high quality random numbers is also becoming increasingly important in modern systems. This paper presents hardware architecture for efficient implementation of the well equidistributed long-period linear (WELL) algorithm. WELL algorithm is implemented in BRAM-register hybrid architecture which provides 6R/2W operations in a single cycle. Our design achieves a throughput of one sample-per- cycle. The proposed architecture is also implemented on targeting different devices for the comparison of other types of pseudorandom number generators. In addition, we design a software/hardware framework that is capable of dividing the WELL stream into an arbitrary number of independent parallel sub streams. With support from software, this framework can obtain speedup roughly proportional to the number of parallel cores. This design can achieve long period random numbers.**

KEYWORDS: **field-programmable gate array (FPGA), parallel random number generator (PRNG), well equidistributed long-period linear (WELL) algorithm.**

I. INTRODUCTION

High quality random numbers play vital role in many scientific applications, Particularly for Monte Carlo simulations. As application sizes scale, one emerging trend is to develop parallelized version of the applications to exploit the available parallel hardware resources, such as in field-programmable gate arrays (FPGAs), to achieve high speed in performance. One prevalent F2-linear PRNG is the Mersenne Twister (MT) [1], which has very long period and good equidistribution. However, MT is also proved to have certain drawbacks. For example, one serious issue is that it is sensitive to poor initialization and can take a long time to recover from a zero-excess initial state .The well eequidistributed long-period linear (WELL) algorithm is proposed [2] to fix this problem. Compared with MT, WELL has better equidistribution while retaining an equal period length. The fast jump ahead technique [4] provides an Thus allowing multiple PRNGs to generate independent sub streams in parallel and providing strong theoretical support for Parallelizing F2-linear PRNGs with long-period. A large body of research is done on F2-linear PRNGs, most of which focus on algorithms and relevant software implementations. Only a few hardware implementations can be found in the literature. For those hardware implementations, most of them employ the MT method, including nonparallel [5] and parallel [7], [8] hardware implementations. With its advantages over MT, WELL also receives great attention from the software community. Being the key component of various scientific applications, designing PRNGs that can rapidly provide independent parallel streams of high quality random numbers is also becoming increasingly important in modern systems. PRNG is an algorithm for generating a sequence of numbers whose properties approximate the properties of sequences of random numbers. Most PRNGs focus on algorithms and software implementations. Only a few hardware implementations can be found. In previous works, a BRAM-and-register-hybrid architecture for WELL19937 with a throughput of one sample per cycle was used. Later a method with a more resource-efficient structure that reduces the usage of BRAMs from four to two was proposed and it retained the same throughput. The total resource used is also as much as 50% compared with the original structure. This design also provides a software reduced /hardware framework to parallelize its output stream based on the new structure and a resource-efficient hardware architecture for WELL with a throughput of one sample per cycle.

II. WELL USING 4 BRAM

Earlier we have WELL method using 4 BRAM. In the existing system 4 BRAM s are used . thus more resources are used for this BRAM s when we dumped into the FPGA kit. And moreover we can't obtain six read and two write operations in a single cycle . so the resource overhead can't be reduced. This paper is a further work of the conference paper. So we propose a more resource-efficient structure that reduces the usage of BRAMs from four to two, while retaining the same throughput. The total resource used is also reduced as much as 50% compared with the original structure. We

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com
International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 3, Special Issue 19, April 2016

III. WELL USING 2BRAM

We design a software/hardware framework to parallelize its output stream based on the new structure. More specifically, we make the following contributions.1) A resource-efficient hardware architecture for WELL with a throughput of one sample per cycle. (improved TWOBRAMs-based WELL architecture) 2) A dedicated 6R/2W RAM structure for WELL, which is capable of providing six Reads and two Writes concurrently in a single cycle, with little resource overhead.3) A software/hardware framework to generate parallel random numbers. Also the implementation of the WELL structure is discussed more detailed and the comparison is enhanced by introducing more types of FPGA-based PRNGs.

A. BLOCK DIAGRAM(HARDWARE ARCHITECTURE)

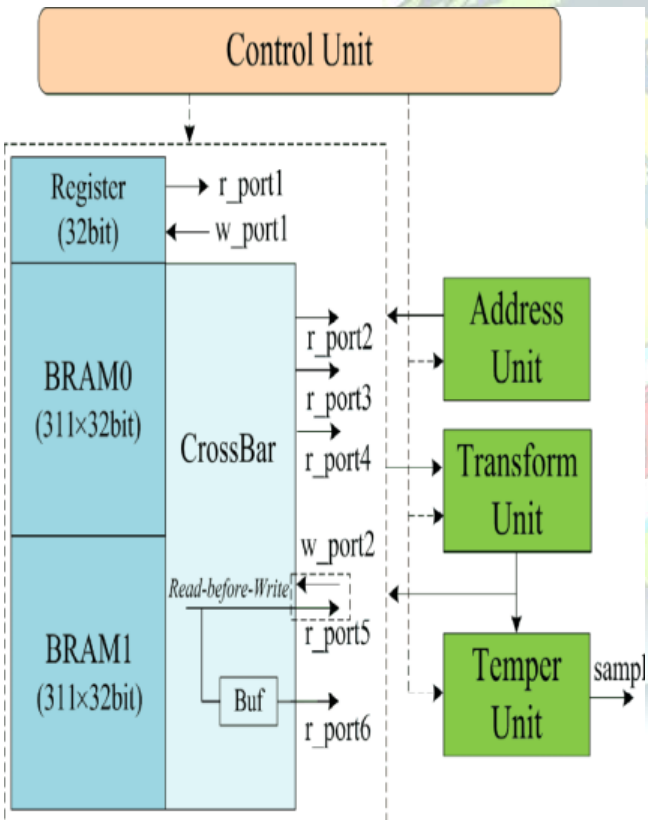

Fig. 1 Block diagram

Hardware architecture for WELL19937 consists of five blocks: the Control Unit, the Address Unit, the Transform Unit, the Temper Unit, and a 6R/2W RAM. The core component is the RAM, which stores the 624 32-bit state vectors and is capable of concurrently supporting six Reads and two Writes. The Address Unit generates appropriate R/W addresses for the RAM. The Transform Unit and the Temper Unit perform the Transform and Temper operations of the WELL algorithm, and can be fully pipelined. The Control Unit produces the control signals to coordinate the system. The state vector S [0] is read and updated in each cycle. We therefore can use a single register to store S [0] and provide the necessary 1R/1W operations. Christo Ananth et al. [6] proposed a system which contributes the complex parallelism mechanism to protect the information by using Advanced Encryption Standard (AES) Technique. AES is an encryption algorithm which uses 128 bit as a data and generates a secured data. In Encryption, when cipher key is inserted, the plain text is converted into cipher text by using complex parallelism. Similarly, in decryption, the cipher text is converted into original one by removing a cipher key. The complex parallelism technique involves the process of Substitution Byte, Shift Row, Mix Column and Add Round Key. The above four techniques are used to involve the process of shuffling the message. The complex parallelism is highly secured and the information is not broken by any other intruder. The embedded output registers of both BRAMs are enabled to improve the clock speed. Thus the high quality random numbers are generated using this frame work.

B. MODULES

1)BRAM: Based on the transformation process of WELL algorithm in each generation process, six blocks from the state vector are fetched while two blocks are updated. Therefore, to achieve the expected throughput, the RAM should read six operands and store two results concurrently in a single cycle. Such a RAM can be directly implemented using 624 32-bit registers, but this is not area-efficient and is impractical when building parallel PRNGs. It is also not straightforward to provide eight ports by simply assembling four BRAMs together, as we need to guarantee that the read and write operations are distributed across different BRAMs evenly. Instead, we propose a BRAM-and-register hybrid structure to build the required 6R/2W multiport RAM, which is the key component to achieve one sample per cycle throughput. The BRAMs of the FPGA allow a Read-before-Write operation, i.e., when writing a new data into an address, the data previously stored at this address can be fetched concurrently in the same clock cycle.

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com
International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 3, Special Issue 19, April 2016

2)TRANSFORM UNIT: It perform row wise shifting operations .it can be designed with the help of shifting units, XOR operations ,multiplexer and a concatenation unit.

3)TEMPER UNIT: This work is to shift the output of transform unit in column wise with the help of shifting units, XOR operations, and operations

4)ADDRESS UNIT:It is used to access the locations of BRAM using concern addresses . here we have used 10 bit address. So address decoder is used to perform this work. An address decoder is a binary decoder circuit that has two or more bits of an address bus as inputs and that has one or more device selection lines as outputs.When the address for a particular device appears on the address bus, the address decoder asserts the selection line for that device. When a single address decoder serves multiple devices, an address decoder with n address input bits can serve up to 2n separate devices. This address decoder has ten address inputs and 1024 device selector outputs.

4)CONTROL UNIT : it is considered as the brain of overall framewok.since it provides control signals to all the units.it can be designed with conditional and loop statements.it provides control signals for multiplexers in crossbar implementation for allowing correct read and write operations.it provides control signals for multiplexers in transform unit to select the number. It provides control signals for address decoder for accesing the locations of BRAM.

C. HARDWARE

1)Cyclone II FPGAs: It offer up to 1.1 Mbits of embedded memory through the popular M4K memory blocks, which can be configured to support a wide range of operation modes including RAM, ROM, first-in first-out (FIFO) buffers, and single-port and dual-port modes.Cyclone II FPGAs offer up to 150 18 x 18 multipliers that are ideal for low-cost digital signal processing (DSP) applications. These multipliers are capable of implementing common DSP functions such as finite impulse response (FIR) filters, fast Fourier transforms (FFTs), correlators, encoders/decoders, and numerically controlled oscillators (NCOs). Cyclone II FPGAs provide support for external memory interfaces, which allow you to integrate external SDR, DDR, and DDR2 SDRAM devices, and QDRII SRAM devices at data rates up to 668 Mbps.This is used for showing the output in the hardware.

D. SOFTWARES

1)Model Sim: It is intended for users of UNIX, Linux, and Microsoft Windows.All versions of Model Sim can be supported on all platforms. Model Sim is a verification and simulation tool for VHDL, Verilog , SystemVerilog, and mixed language designs In Model Sim, all designs are compiled into a library. Model sim is used for compile and simulate codes of the modules

2) Quartus: It is a registered trademark of Altera Corporation in the US and other countries.This can be used to compile and synthesise the VHDL,Verilog files.With the help of this software we can obtain timing report,RTL diagrams,power summary .

IV. SOFTWARE FRAMEWORK

WELL algorithm is used to generate seed vectors. It has better equidistribution, by counting the fraction of nonzero coefficients in the characteristic polynomial of the recurrence, and by an empirical comparison of the diffusion capacity of the old and new generators. It is very useful and safe for the vast majority of simulation applications, especially when the generator's speed is important. WELL algorithm performs more bit transformations and is better equidistributed than (for example) the Mersenne twister, while having the same period length and approximately the same speed. WELL algorithm has better equidistribution and "bit-mixing" properties for equivalent period length and speed. This can reduce the impact of persistent dependencies among successive output values, which can be observed in certain parts of the period of gigantic generators such as the Mersenne twister.

V. SOFTWARE AND HARDWARE FRAMEWORK

As application sizes scale, one emerging trend is to develop parallelized version of the applications to exploit the available parallel hardware resources, such as in field-programmable gate arrays (FPGAs), to achieve high speed in performance . The fast jump ahead technique [4] provides an efficient method to determine the starting point of a new substream from an existing substream, thus allowing multiple PRNGs to generate independent substreams in parallel and providing strong theoretical support for parallelizing F2-linear PRNGs with long-period The jump ahead unit in Software is responsible for: 1) generating the initial vector states for each PRNG according to user configurations, i.e., the total random numbers, the number of simulation cores and the initial seed; 2) offloading initial state vectors to the hardware; and 3) dealing with the simulation results. The core component of the hardware is a PRNG Array consisting of a number of parallel WELL PRNGs. It can be constructed by simply replicating the single generator. After receiving proper state vectors, an array of size n can produce n random numbers in parallel in every clock cycle, which are denoted as $y0$, $yv1, . . . , {}^{y}vn\text{-}1$.

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com
International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 3, Special Issue 19, April 2016

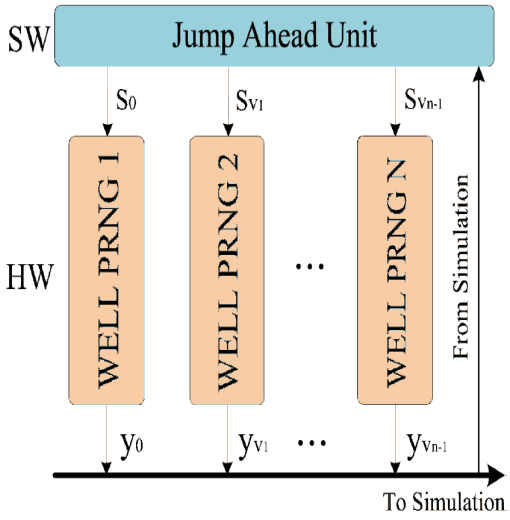

Fig. 2 Jump ahead unit

VI. IMPLEMENTATION

Hardware architecture is designed using VHDL language .In this all the blocks are designed as modules so they are concatenated to work as a entire hardware architecture .registers are implemented with the help of shift registers. BRAM is a Block RAM, which is a two-port memory (PORT A and PORT B) containing several kilobits of RAM. These blocks are denoted as FPGA blocks Each FPGA block consists of small block called Look up table (LUT). It is used for performing logical functions which we can then reconfigure it as few bits of RAM. BRAM is implemented using the array of ram cells.ram cells are created using d flip-flop with appropriate logic operations such as AND,NOT,OR etc…the address unit is designed by the help of address decoder .crossbar is implemented with eight multiplexers. control unit consists of set of rules implemented by conditional and loop statements .transform and temper units are implemented by XOR and shift operations .thus the whole hardware architecture is designed .the state vectors are produced with the help of WELL algorithm. This is the software framework we have used. Thus both hardware and software framework is created to produce high quality random numbers.

VII. EVALUATION

	Period (log2	Crush Faile	Slices	FFs	LUTs	BRAMs	Frequency .(MHz)	Throughput (Gb/s)
Proposed work	19937 2	2	112	259	317	2	423	13.5
Existing Work	19937 2	2	168	535	571	4	424	13.6

VIII. RESULTS

A. WELL GENERATOR OUTPUT

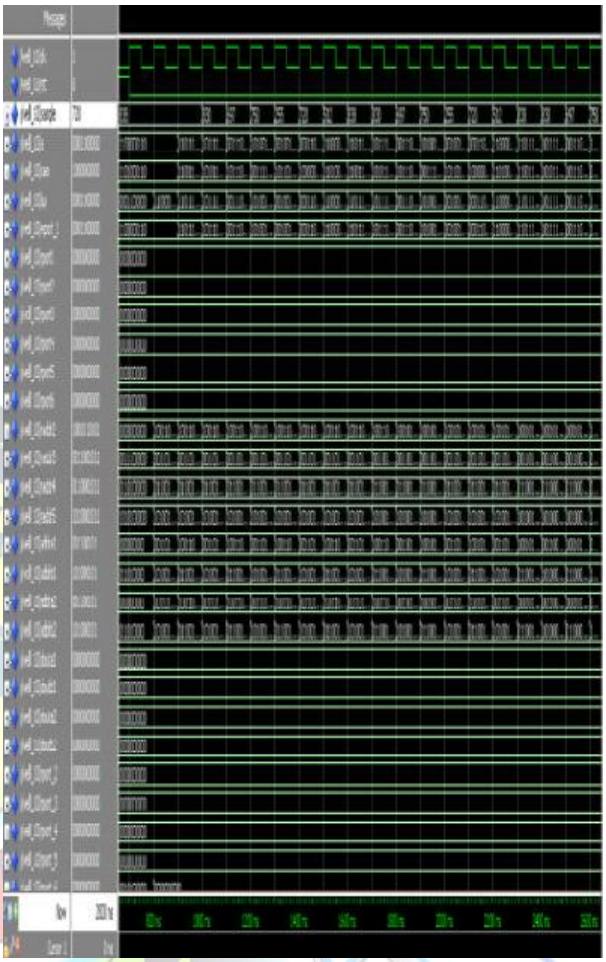

Figure. 3 Simulation result of WELL random number generator using BRAM

In BRAM, when WE is enabling, 32-bit input is given and write operation is performed. When WE become low, the read operation is performed and therefore the output will be displayed at Doubt. The output Z4 (32-bit) value is obtained by performing various shift and XOR operations on the read and write ports according to the architecture given in transform unit. The output of the temper unit is sample, which is obtained by shifting and XOR operations of Z4 (output of transform unit) and other Hexadecimal values. Register unit is used to store the given values. i.e., the value of 32-bit w_port1 is provided as the input and the same value is obtained as the output in 32 bit r_port1.By combining the output values of transform unit, temper unit, register unit.The sample is declared as a final 32bit output value of a single WELL generator. Therefore the random number is generated as sample for the WELL algorithm. Now, the 32-bit outputs (sample) of 4 WELL PRNGs are

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com
International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 3, Special Issue 19, April 2016

arranged in parallel manner to produce the final Pseudo-random Number.

IX. CONCLUSION

Through our study, we demonstrated that our proposed one-sample-per-cycle hardware architecture for the WELL algorithm achieved high performance, low area cost, and high quality output at the same time. The SW/HW framework we develop could parallelize the WELL sequence into arbitrary number of independent parallel sub streams. The random patterns are generated by using WELL generator and its performance was analyzed. The speed is enhanced by parallelizing the WELL PRNGs into number of sub streams.

REFERENCES

[1] Panneton, P. L'Ecuyer, and M. Matsumoto, "Improved long-period generators based on linear recurrences modulo 2," *ACM Trans. Math. Softw.,* vol. 32, no. 1, pp. 1–16, Mar. 2006.

[2] P. L'Ecuyer and F. Panneton, "Fast random number generators based on linear recurrences modulo 2: Overview and comparison," in *Proc. 37th Conf. Winter Simul.,* 2005, pp. 110–119.

[3] H. Haramoto, M. Matsumoto, T. Nishimura, and P. L'Ecuyer, "Efficient jump ahead for F2-linear random number generators," *Inf. J. Comput.,* vol. 20, no. 3, pp. 385–390, 2008.

[4] V. Sriram and D. Kearney, "An area time efficient field programmable Mersenne Twister uniform random number generator," in *Proc. Int. Conf.Eng. Reconfigur. Syst. Algorithms*, 2006, pp. 244–246..

[5] Shrutisagar and A. Abbes, "High performance FPGA implementation of the mersenne twister," in *Proc. 4th IEEE Int. Symp. Electron. Design, Test Appl.*, Jan. 2008, pp. 482–485.

[6] Christo Ananth, H.Anusuya Baby, "High Efficient Complex Parallelism for Cryptography", IOSR Journal of Computer Engineering (IOSR-JCE), Volume 16, Issue 2, Ver. III (Mar-Apr. 2014), PP 01-07

[7] L. Dalal and D. Stefan, "A hardware framework for the fast generation of multiple long-period random number streams,"in *Proc. 16th ACM Int. Symp. FPGAs*, Feb. 2008, pp. 245–254.

[8] Y. Li, P. Chow, J. Jiang, and M. Zhang, Software/hardware framework for generating parallel long-period random numbers using the WELL method," in *Proc. 21st Int. Conf. Field Program.*

[9] Thomas and W. Luk, "High quality uniform random number generation through LUT optimised linear recurrences," in *Proc. IEEE Int. Conf. Field-Program. Technol.*, Dec. 2005, pp. 61–68.

AUTHOR DESCRIPTION

Mr.A.N.Duraivel, is currently an Associate Professor with the Department of Electronics and communication Engineering at Kings Engineering College

A.Komathi is an undergraduate student who is pursuing his B.E degree in Electronics and communication engineering at Kings Engineering College

R.Arthie Reena is an undergraduate student who is pursuing his B.E degree in Electronics and communication engineering at Kings Engineering College

V.Aruna is an undergraduate student who is pursuing his B.E degree in Electronics and communication engineering at Kings Engineering College