ISSN (Online) 2278-1021

IJARCCE

ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 11, November 2017

Increasing randomness using DCM-based tunable True Random Number Generator

J.Shouba Tharani[1], Dr.V.Seethalakshmi[2]

PG Scholar, Dept of VLSI Design, KPR Institute of Engineering and Technology, Coimbatore, India[1]

Associate Professor, Dept of ECE, KPR Institute of Engineering and Technology, Coimbatore, India[2]

Abstract: There is a need for enhancing the randomness in modern cryptographic systems. True Random Number Generators (TRNGs) play a very important role in modern cryptographic systems. True Random Number generator (TRNG) circuit utilizes tunability feature for determining the degree of randomness. Random number generators will be required to protect the medical, financial and personal data of entities connected to the networks. A digital true random number generator that can be synthesized using standard digital tools will enable designers to address these privacy concerns more efficiently. To provide advanced clocking capabilities, Digital Clock Managers (DCMs) is used. The DCM modules allow greater designer control over the clock waveforms, and their usage eliminates the need for initial calibration.The well equidistributed long-period linear (WELL) is used to fix poor initialization and take less time to recover from zero-excess initial state.This systems creates randomness with high authentication.

Keywords: Tunable Random Number Generator (TRNG), Digital Clock Manager (DCM), Well equidistributed long-period linear (WELL).

I. INTRODUCTION

Random number generators are used in security for generating secrets such as session keys and large primes for key exchange and exponentiation. Random number generators are also used for simulating random events and for professional gaming. The security applications are of primary importance as the number and complexity of networks continues to grow. Random number generators will be required to protect the medical, financial and personal data of entities connected to these networks. A digital true random number generator that can be synthesized using standard digital tools will enable designers to address these privacy concerns more efficiently. True random number generators (TRNGs) have become an indispensable component in many cryptographic systems, including PIN/password generation, authentication protocols, key generation, random padding and nonce generation. TRNG utilize a nondeterministic random process, usually in the form of electrical noise, as a basic source of randomness. Along with the noise source, a noise harvesting mechanism to extract the noise and a post processing stage to provide a uniform statistical distribution are other important components of the TRNG. For a perfect true random number generator, the probability of the next generated number being any specific value should be equal to the probability of the next generated number being any other specific value. Since a certainty is always a probability of 1 and since some specific value will certainly be generated, the probability of any particular value being generated to 1 (certainty) divided by the number of possible values in the range. As a simple illustration, Consider a six sided die. The probability of any one of the six sides facing up after rolling the die is 1 (certainty) divided by 6 (the number of possible values). For a digital random number composed of N bits where N is positive definite, the range of values has 2N possible values. So the probability of any particular value being generated next by a true N-bit digital random number generator is given in equation 1:

$$P=1/2^N \qquad (1)$$

At the inception of this research, true random number generators always required some analog components be included in ICs (Integrated Circuits). True random number generators were always based on an analog property like junction or thermal noise that was often whitened and scaled to produce a uniformly distributed random number generator. Whenever a SOC (System On Chip) required random number generation, an analog IC designer was required to complete the design. A digital random number generator that can be designed using standard digital design tools would significantly reduce the cost and complexity of including a true random number generator in a design. When a true random number generator is implemented in an FPGA (Field Programmable Gate Array), either several additional analog components such as resistors and operational amplifiers must be added to the design, or the designer must measure and match performance of individual logic blocks to achieve acceptable performance. Again, a digital random number generator that can be designed and implemented using standard digital design tools would alleviate the need for extra components and/or the tedious hand-matching of logic blocks. This dissertation documents the design and implementation of a true random number generator using standard digital design methodology.

ISSN (Online) 2278-1021

IJARCCE

ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 11, November 2017

A digital clock manager is an electronic component available on some FPGA (notably ones produced by Xilinx). A DCM is useful for manipulating clock signals inside the FPGA, and to avoid clock skew which would introduce errors in the circuit. Digital Clock Managers (DCMs) provide advanced clocking capabilities to Spartan™-3 FPGA applications. DCMs optionally multiply or divide the incoming clock frequency to synthesize a new clock frequency. DCMs also eliminate clock skew, thereby improving system performance. Similarly, a DCM optionally phase shifts the clock output to delay the incoming clock by a fraction of the clock period. The DCMs integrate directly with the FPGA's global low-skew clock distribution network. The Digital Clock Manager (DCM) primitive in Xilinx FPGA parts is used to implement delay locked loop, digital frequency synthesizer, digital phase shifter, or a digital spread spectrum. The digital clock manager module is a wrapper around the DCM primitive which allows it to be used in the EDK tool suite. The DCM module is used in Multipling or dividing an incoming clock, Making sure the clock has a steady duty cycle, Adding a phase shift with the additional use of a Delay-locked loop, Eliminatig clock skew within an FPGA design.

II. RELATED WORK

A.Single Phase BFD-TRNG Model

The structure and working of the (single phase) BFDTRNG [6] can be summarised as follows, in conjunction with Fig.1:

1) The circuit consists of two quasi-identical ring oscillators (it is termed as ROSCA and ROSCB), with similar construction and placement. Due to inherent physical randomness originating from process variation effects associated with deep sub-micron CMOS manufacturing, one of the oscillators (say, ROSCA) oscillates slightly faster than the other oscillator (ROSCB). In addition, the author proposed to employ trimming capacitors to further tune the oscillator output frequencies.
2) The output of one of the ROs is used to sample the output of the other, using a D flip-flop (DFF). Without loss of generality, assume the output of ROSCA is fed to the D-input of the DFF, while the output of ROSCB is connected to the clock input of the DFF.
3) At certain time intervals (determined by the frequency difference of the two ROCs), the faster oscillator signal passes, catches up, and overtakes the slower signal in phase. Due to random jitter, these capturing events happen at random intervals, called "Beat Frequency Intervals". As a result, the DFF outputs a logic-1 at different random instances.
4) A counter controlled by the DFF increments during the beat frequency intervals, and gets reset due to the logic-1 output of the DFF. Due to the random jitter, the free running counter output ramps up to different peak values in each of the count-up intervals before getting reset.
5) The output of the counter is sampled by a sampling clock before it reaches its maximum value.
6) The sampled response is then serialized to obtain the random bitstream.

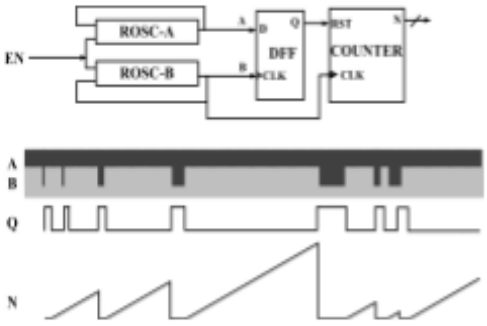

Fig.1: Architecture of single phase BFD–TRNG.

Shortcoming of the BFD-TRNG One shortcoming of the previous BFD-TRNG circuit is that its statistical[5] randomness is dependent on the design quality of the ring oscillators. Any design bias in the ring oscillators might adversely affect the statistical randomness of the bitstream generated by the TRNG. Designs with same number of inverters but different placements resulted in varying counter maximas. Additionally the same ring-oscillator based BFDTRNG implemented on different FPGAs of the same family shows distinct counter maxima. Unfortunately, since the ring oscillators[1] are free-running, it is difficult to control them to eliminate any design bias. The problem is exacerbated in FPGAs where it is often difficult to control design bias because of the lack of fine-grained designer control on routing in the FPGA design fabric. A relatively simple way of tuning clock generator hardware primitives on

ISSN (Online) 2278-1021

IJARCCE

ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 11, November 2017

Xilinx FPGAs, particularly the Phase Locked Loop (PLL) or the Digital Clock Manager (DCM) as used in this work, is by enabling dynamic reconfiguration via the Dynamic Reconfiguration Ports (DRPs). Once enabled, the clock generators can be tuned to generate clock signals of different frequencies by modifying values at the DRPs on–the–fly, without needing to bring the device off– line. The proposed tunable BFD-TRNG[5] suitable for FPGA platforms described next.

III. TUNABLE BFD–TRNG FOR FPGA BASED APPLICATIONS

Fig 2 shows the overall architecture of the Digital Clock Manage based tunable BFD–TRNG. In place of two ring oscillators, two DCM modules generate the oscillation waveforms. The DCM primitives are parameterized to generate slightly different frequencies, by adjusting two design parameters M (Multiplication Factor) and D (Division Factor). In the proposed design, the source of randomness is the jitter presented in the DCM circuitry. The DCM modules allow greater designer control over the clock waveforms, and their usage eliminates the need for initial calibration. Tunability is established by setting the DCM parameters on–the–fly using DPR capabilities using DRP ports. This capability provides the design greater flexibility than the ring oscillator based BFD-TRNG. The difference in the frequencies of the two generated clock signals is captured using a DFF.

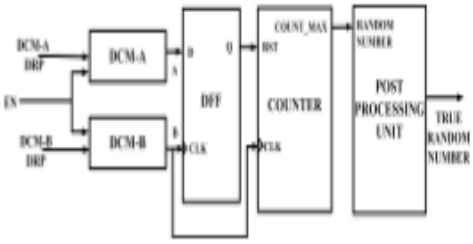

Fig.2: Overall architecture of Digital Clock Manage based tunable BFD–TRNG.

The DFF sets when the faster oscillator completes one cycle more than the slower one (at the beat frequency interval). A counter is driven by one of the generated clock signals, and is reset when the DFF is set. Effectively, the counter increases the throughput of the generated random numbers. The last three LSBs of the maximum count values reached by the count were found to show good randomness properties.The target clock frequency is determined by the set of parameter values actually selected. The random values reached by the counter, as well as the jitter are related to the chosen parameters M and D. This makes it possible to tune the proposed TRNG using the predetermined stored M and D values. As unrestricted DPR has been shown to be a potential threat to the circuit, the safe operational value combinations of the D and M parameters for each DCM are predetermined during the design time, and stored on an on-chip Block RAM (BRAM) memory block in the FPGA. There are actually two different options for the clock generators – one can use the Phase Locked Loop (PLL) hard macros available on Xilinx FPGAs, or the DCMs. We next describe analytical and experimental results which compelled us to choose DCM in favor of the PLL modules for clock waveform generation.

A.Well Equidistributed Long-Period Linear Method

High quality random numbers are of critical importance to many scientific applications, particularly for Monte Carlo simulations. Given the advantages of high performance and reproducibility, pseudorandom number generators (PRNGs) based on linear recurrences over F2 are widely adopted in such simulations. One prevalent F2-linear PRNG is the Mersenne Twister (MT), 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 equidistributed long-period linear (WELL) algorithm is proposed to fix this problem. Compared with MT, WELL has better equidistribution [8] while retaining an equal period length 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. 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. The fast jump ahead technique 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.

With its advantages over MT, WELL also receives great attention from the software community. However, few hardware implementations can be found. The Ukalta Engineering Corporation gave a brief introduction to its product that employs the WELL algorithm. However, it only achieves a throughput of one sample every two cycles and no

ISSN (Online) 2278-1021
IJARCCE
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 11, November 2017

structural details are revealed. 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 also design a software/hardware framework to parallelize its output stream based on the new structure[8]. More specifically, we make the following contributions.

1) A resource-efficient hardware architecture for WELL with a throughput of one sample per cycle.
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.

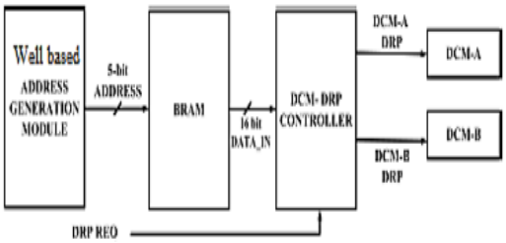

Fig. 3: Well equidistributed long-period linear (WELL) TRNG

Fig 3 shows the circuit with well based TRNG. 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. 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 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. 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. Using this feature, the w_port2 and the r_port5 can be provided by a single port of a BRAM thus saving one R/W operation. Since the Read addresses of the r_port5 and r_port6 are always adjacent, the data from r_port5 can be buffered and reused by r_port6 in the next clock cycle. This saves one more Read operation. By utilizing these two optimizations, the remaining six R/W operations are decreased to only four operations. This can be provided by two dual-ported 311 × 32-bit BRAMs. Assume the access delay of the BRAM is one clock cycle, the pseudocode for generating the access addresses for the Read ports from r_port2 to r_port5 (addr[2..5]) and the Write port w_port2 (addr[5]). The crossbar implements the mapping rules [8], to forward the R/W ports to appropriate BRAMs, where the access address is generated by the Address Unit based on the pseudocode.

The statistical testing is two-fold: 1) the sequential testing to check for correlations within a stream and 2) the parallel testing to check for correlations between different substreams. For the sequential testing, we just have verified that the outputs of the hardware and the standard software version are exactly the same when starting with the same seed. Since the statistical properties of the WELL algorithm was already well proven to be good, hence the same should be true of the hardware WELL generator. The parallel testing is performed by interleaving different substreams into a single stream. For example, if the parallel degree is n and the stream i is given by $xi,0, xi,1, xi,2, \ldots$, then the new stream is in the form of $x0,0, x1,0, \ldots, xn-1,0, x0,1, x1,1, \ldots, xn-1,1, \ldots$. We apply the new stream to the standard statistical test suites, Diehard and Crush from TestU01. Testing results show that the parallel generators pass all tests, which verify the independence of different parallel streams generated by our framework

IV. CONCLUSION

An improved fully digital tunable TRNG for FPGA based applications, based on the principle of Beat Frequency Detection and clock jitter, and with in-built error correction capabilities is presented. The TRNG utilizes this tunability feature for determining the degree of randomness, thus providing a high degree of flexibility for various applications. The proposed design successfully passes all NIST statistical tests.

ISSN (Online) 2278-1021

IJARCCE

ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 11, November 2017

REFERENCES

[1] DongshengLiu, Zilong Liu, Lun Li, and Xuecheng Zou(2016) A Low-Cost Low-Power Ring Oscillator-Based Truly Random Number Generator for Encryption on Smart Card.

[2] Johnson A.P. , R. S. Chakraborty and D. Mukhopadhyay(2015)"APUFEnabled Secure ArchitectureforFPGA BasedIoTApplications,"in EEE Transactions on Multi-Scale Computing Systems.

[3] Johnson A.P. , R. S. Chakraborty and D. Mukhopadhyay(2015) "A Novel Attack on a FPGA based True Random Number Generator", 10th Workshop on Embedded Systems Security.

[4] Johnson A.P. , S. Saha, R. S. Chakraborty, D. Mukhopadyay and Sezer Goren(2014)"Fault Attack on AES via Hardware Trojan Insertion by Dynamic Partial Reconfiguration of FPGA over Ethernet", 9th Workshop on Embedded Systems Security.

[5] Rukhin A., J. Soto, J. Nechvatal, M. Smid and E. Barker(2001) "A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications", DTIC Document.

[6] Tang N., B. Kim, Y. Lao, K. K. Parhi and C. H. Kim(2014)"True Random Number Generator circuits based on single- and multi-phase beat frequency detection," Proceedings of the IEEE 2014 Custom Integrated Circuits Conference.

[7] Von Neumann J. ,(1951)"Various Techniques used in Connection with Random Digits.", National Bureau of Standards Applied Mathematics Series.

[8] Yuan Li, Paul Chow, Senior Member, IEEE, Jiang Jiang, Minxuan Zhang, and Shaojun Wei(2014) Software/Hardware Parallel Long-Period Random Number Generation Frame'work Based on the WELL Method.