A BBS Random Number Generator for Low Power Applications

Alireza Hassanzadeh
ECE Department,
The Shahid Beheshti University,
Tehran, Iran

Vahid Mahboubi
ECE Department,
The Shahid Beheshti University
Tehran, Iran

ABSTRACT

In this paper a low power random number generator has been designed and implemented using BBS algorithm. The BBS random number generator is known for high statistical quality and power consumption. Dynamic power dissipation has been reduced significantly comparing to regular implementation. Low power techniques such as power gating and pipelining have been employed to reduce power consumption of the module. Experimental measurements for a 32-bit BBS random number generator shows at least 30% reduction in dynamic power consumption. The proposed low power BBS random number generator has been implemented on Xilinx Spartan-6 FPGA evaluation board.

Keywords

RNG, BBS, Low power, VHDL, FPGA.

1. INTRODUCTION

A random number generator is a hardware or software that can generate a series of numbers with unknown and unpredictable pattern. Random number generators (RNG) can be used in simulation, big data sampling, genetic algorithm, numerical analysis, decision making, complex statistical events, cryptography and many other applications. Random number generators are divided into two categories: true random (TRNG) and pseudo random generators (PRNG) [1]. Cryptographically Secured Pseudo Random Number Generators (CSPRNG) are widely used in cryptography, key generation and random streams. For a successful coding approach the statistical quality of the RNG is critical. Applications such as low power secure networks, RFIDs, One Time Password (OTP) and many military applications require low power operation of the RNG. In CSPRNG applications low power consumption is critical, because of limited power available for such applications.

CSPRNGs can be used in unique key generators. Because of high security requirement of these applications high distribution algorithms are used for it. Unfortunately, these generators are low speed and consume large amount of power. Obviously, for cryptography and portable applications low power consumption is important. In this report, because of high statistical distribution of Blum Blum Shub (BBS) algorithm, it has been used for implementation in low power application. Low power digital design techniques have been employed for low power implementation of the algorithm.

This paper is arranged as follows: Random number generators and BBS generator have been introduced in section 2 and 3. Low power BBS algorithm has been covered in section 4. Simulation and experimental results are in section 5. Conclusion remarks are at the end.

2. RANDOM NUMBER GENERATION

Random numbers can be generated using various methods. Generation methods can be different in principal and have different time of generation, generation pattern and repeatability [2]. Every application requires specific property of the RNG. For example for statistical applications speed of generation is very important. In cryptography, high distribution or statistical quality is critical. The generation method has important impact on power consumption, production cost and generation time. Some of the methods used in TRNGs are: numerical methods, quantum and thermal effects, atmospheric and electromagnetic disturbances, Hash functions, radioactive degeneration, lava lamp, two free running oscillators, Perlin noise that have advantages and disadvantages. In practice, physical generators based on natural phenomena are only used for seed generation for cryptography applications. For cryptographic applications mathematical and algorithmic methods are more common.

3. BBS RANDOM NUMBER GENERATOR

The BBS method was introduced by Lenore Blum and Manuel Blum and Michael Shub in 1986 [3]. BBS Algorithm is well known for its application in CSPRNG implementation. The method uses equation (1) to generate the number:

$$X_{n+1} = X_n^2 \bmod m \qquad (1)$$

Where m is:

$$m = p * q \qquad (2)$$

where p and q are large prime numbers. In every iteration of the algorithm, the output value is stored in X_{n+1}. The output is Xn+1 or a few least significant bits of X_{n+1}. The two prime numbers p and q should have residue of 3 when divided by 4, which guarantees that every residue has square root and is a second order residue. The initial value of S (seed number) is chosen to have no factor of m. Therefore, p and q are not any factor of S.

The algorithm should be repeated to generate as many as bits required for the random number. Obviously, this method is not considered a fast method of random number generation and is not suitable for applications that require speed. However, complexity and high precision of the algorithm makes the code hard to break and is very attractive method for security applications [3]. To evaluate the method following values have been considered:

p=11, q=19, S=3 and therefore m=p*q=209. The seed number is $s^2 = 9$. The following iteration can be done to obtain the X_n values:

$$\mathrm{X}_0 = S^2 \bmod m = 9 \qquad (3)$$

$$X_1 = X_0^2 \bmod m = 81 \qquad (4)$$

$$X_2 = X_1^2 \bmod m = 82 \qquad (5)$$

X_0 is produced based on the seed number and other X_{n+1}'s are produced based on the previous X_n number. For statistical quality pseudo random number generation the seed number selection is very important. The seed number can be generated using true random number generator (TRNG). The even or odd parity or LSB of the number or any other combination of the random number is generated and used. Table 1 shows a sequence of random number generation using BBS algorithm and the above initial values.

Table 1- Random number sequence using BBS CSPRNG

	value	Even Parity	Odd Parity	LSB
X_0	9	0	1	1
X_1	81	1	0	0
X_2	82	1	0	1
X_3	36	0	1	0
X_4	42	1	0	0
X_5	92	0	1	0

Many blocks are not used all the time when the BBS CSPRNG is operating. This is the key feature that can be used to for low power BBS random number generation.

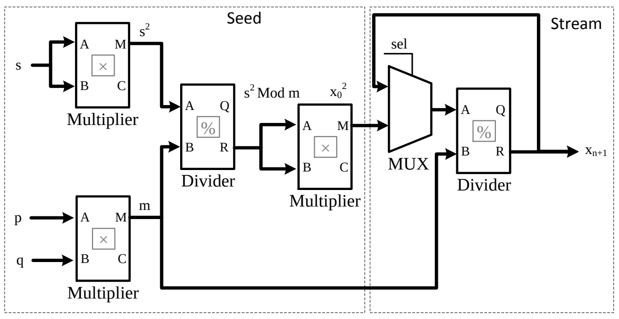


Fig. 1: Regular BBS CSRNG Building Blocks

As shown in figure 1, most of the time the system is generating random sequence and the seed generation block is not used, unless the seed number needs to be updated. As the seed number input is always changing due to its TRNG generation nature, application of this number to the multiplier and divider section can produce large amount of dynamic power dissipation.

4. LOW POWER BBS RANDOM NUMBER GENERATOR

The main building blocks of a BBS CSPRNG consists of units for generating equation (1), prime number multiplication (2), initial iteration equation (3) and bit generation such as the sequence shown in Table 1. The main modules of a BBS CSPRNG which are multiplier and divider blocks are shown in figure 2.

Each module potentially can consume large dynamic power. If partitioning and power gating is used, idle modules can be turned off to save power. Using FSM power saving techniques, power consumption can be reduced effectively. The BBS CSPRNG consists of multiplication modules, divider module for residue calculation, loading registers, data path and output module. Figure 2 shows the block diagram of the BBS CSPRNG including pipelining and glitch reduction registers. This block diagram has been optimized using FSM techniques for power consumption.

In the first step, outputs S and m are generated using s, p and q inputs. S and m are generated in clock pulses φ_1 and φ_2 and stored on φ_3 in the register. This value is used as a seed number or the initial value for iteration. Major power reduction techniques used for low power operation is to clock gate the seed generating section after initial three phases of system operation and while the seed is unchanged. By disabling the clock of this section a large part of the system will be placed in low power mode. In the next clock phases, where initial value X_0 has already been obtained and transferred to the output stage, all stages except the rear end stages will not receive the clock for reducing power dissipation. In this step, X_1-X_n are stored in the specified registers and are used for next phase of the iteration. In the block diagram of figure 2 the select input (*sel*) has been also inactive to reduce more power consumption. Since φ_2 is inactive, the *m* value will remain constant too. Therefore, input A of the multiplexer, divider and output register are the only values that will change. These units will consume less power since most other units are clock gated.

In order to reduce power dissipation because of glitches, input registers have been used. Since multiplication and division blocks are relatively complex and have multilevel logic, glitches can cause significant power dissipation. Using registers can suppress this kind of dissipation in the module. If input registers are not in place, any change in s, p and q can easily propagate in the next two multiplier and divider sections and cause large power dissipation [4].

The FSM designed in this paper for BBS CSPRNG is based on clock gating and glitch reduction techniques that have been simulated for power approximation.

It is worth mentioning that BBS algorithm is considered a high power consumption technique among CSPRNG methods. The proposed method is a high quality statistical method and if speed is not critical can be used in low power applications.

5. SIMULATIONS AND EXPERIMENTAL RESULTS

The proposed low power BBS CSPRNG has been simulated and implemented using Xilinx ISE and FPGA board shown in figure 3. Figure 4 shows simulation results for 8, 16 and 32 bit RNG using the low power algorithm for BBS implementation. As the number of bits increases the power dissipation increases too and the effects of low power design techniques become more relevant. The conventional algorithm power dissipation increases faster than the low power method due to nonlinear relation of power dissipation and number of bits. This clearly shows the effectiveness of the techniques used. The speed of generation can be increased by increasing clock frequency and can be comparable to regular method. The statistical distribution quality of both regular and low power algorithm have been simulated for 8-bit data and 512 samples as shown in figure 5. This shows that there is no major difference in the statistical quality of both methods.

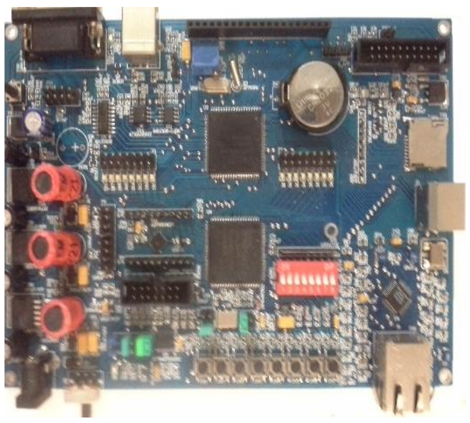


Fig. 3: Xilinx Spartan-6 FPGA evaluation board used for simulation of the BBS algorithm

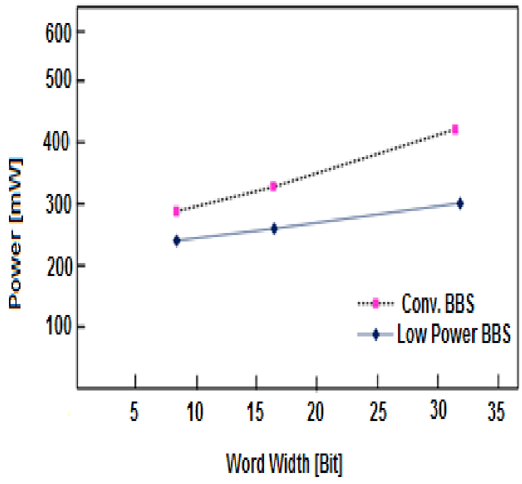
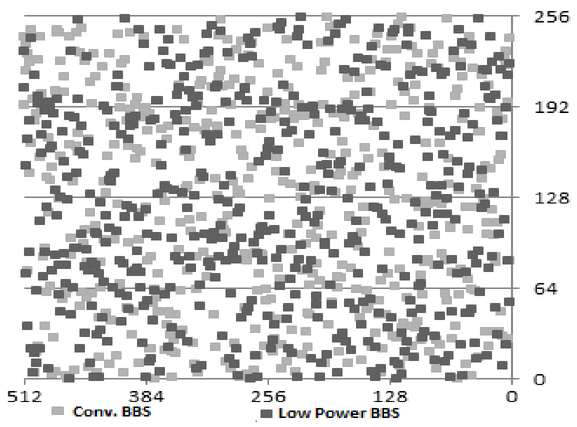


Fig. 4: Power dissipation for conventional and low power BBS implementation



6. CONCLUSION

A low power pseudo random number generator based on BBS algorithm has been simulated and implemented using Xilinx ISE and FPGA evaluation board. Low power techniques such as clock gating and pipelining has been used to reduce dynamic power dissipation of the algorithm. The proposed method proves usability of the BBS method for low power applications without sacrificing security. FPGA implementation proves the concept. The method shows at least 30% power reduction comparing to regular BBS algorithm implementation. The BBS method will no longer be considered a high power dissipation method and can be used in high security, low power random number generation application. Further investigation of the block diagram of the BBS CSPRNG shows that some other methods such as power gating can be used to reduce both dynamic and static power dissipation. Other technology dependent methods such as MTCMOS and subthreshold implementation can also be used to further reduce power consumption by sacrificing the speed of generation. However, these methods can not be used in FPGA based implementations and should be used in ASIC and SoC applications.

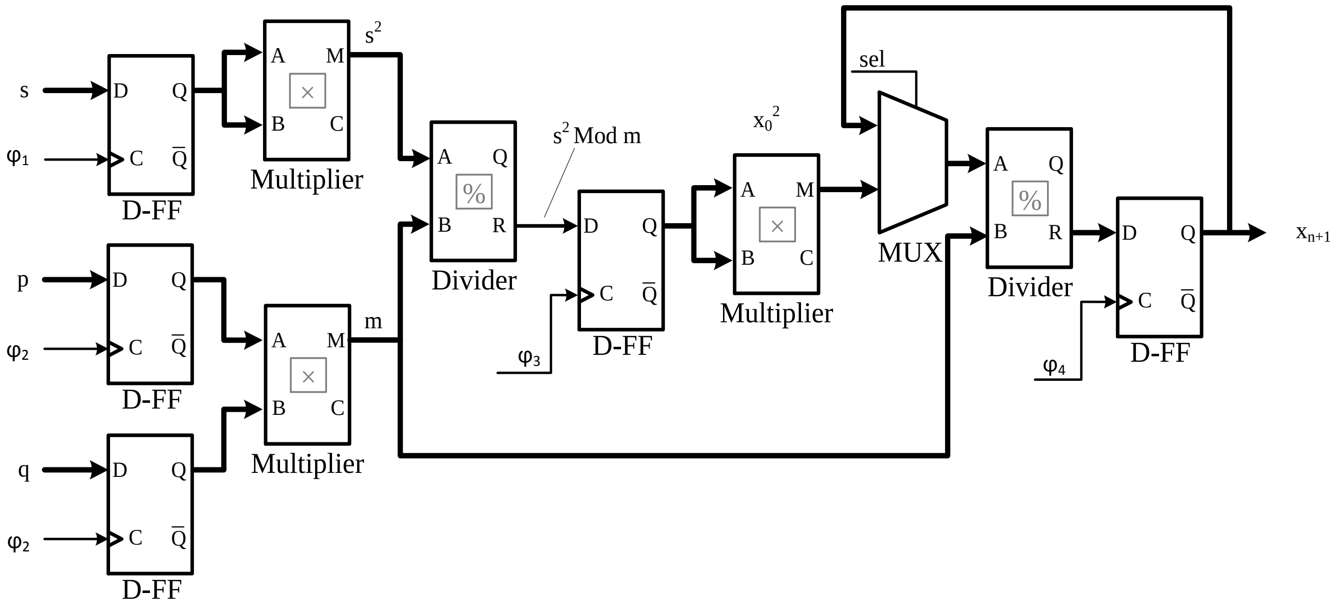


Fig. 2: BBS CSPRNG Building Blocks

7. ACKNOWLEDGMENTS
The authors would like to thank the APT Company for providing the hardware for simulation and verification of the proposed low power BBS algorithm.

8. REFERENCES

[1] A. Tomoaki Sato B. Kazuhira Kikuchi C. Masa-aki Fukase, "A PRNG Circuit on PLD with Feature of Low-Power, High-Speed, and Various Generation of Random Number Sequence", Hirosaki, Aomori 036-8561 JAPAN, IEEE 2006.

[2] A. K.H. Tsoi B. K.H. Leung C. P.H.W. Leong,“Compact FPGA-based True and Pseudo Random Number Generator", Proceedings of the 11th Annual IEEE Symposium on Field-Programmable Custom Computing Machines IEEE 2003.

[3] A.Khushboo Sewak, B.Praveena Rajput, C. Amit Kumar Panda, ”FPGA Implementation of 16 bit BBS and LFSR PN Sequence Generator: A Comparative Study" 2012 IEEE Students’ Conference on Electrical, Electronics and Computer Science FPGA.

[4] A.Tomoaki Sato B.Kazuhira Kikuchit C. Masa-aki Fukaset, "Chip Design of a Wave-Pipelined PRNG",2006 IEEE

[5] Xilinx, Inc. Xilinx Libraries Guide, 2011.