

AN AREA TIME EFFICIENT FIELD PROGRAMMABLE MERSENNE TWISTER UNIFORM RANDOM NUMBER GENERATOR

Vinay Sriram
University of South Australia
vinay.sriram@unisa.edu.au

David Kearney
University of South Australia
david.kearney@unisa.edu.au

ABSTRACT

Reconfigurable computing offers an attractive solution to accelerating infrared scene simulations. In infrared scene simulations, the modeling of a number of atmospheric and optical phenomena like scintillation, refraction, blurring due to lens optics and photon noise may be implemented in parallel. All of these require simultaneous and continual generation of random numbers. Furthermore, random number generation is only a small component of all of these algorithms. Current software random number generators are too slow whilst current hardware random number generators are plagued by issues such correlations and are not area efficient. We describe a reconfigurable computing based uniform random number generator based on the mersenne twister algorithm that is area time efficient and that does not suffer from correlations.

1. INTRODUCTION

There is extensive literature in the field of software random number generators (see the survey paper [9,10]). There are a number of published papers describing the hardware acceleration of random number generators. [1,2]. One of the requirements of infrared simulation is that after supplying one single seed a large pool of uniform random numbers are then be immediately generated. This also facilitates the generation of Gaussian random numbers by adding a processing step that converts a pair of these uniform numbers into a single Gaussian one. There do not appear to be any hardware accelerated uniform random number generators that can do this. The survey of the literature indicates that there is scope for providing the first hardware accelerated uniform generator that has a high period and this may facilitate the development of smaller and faster hardware Gaussian generators based on an hardware optimized combination of this uniform generator and one of the existing hardware Gaussian generators. On this basis, we propose the development of an area time efficient uniform random number generator based on the mersenne twister algorithm.

2. MERSENNE TWISTER THEORY

The mersenne twister, or the MT19937, algorithm is an improved variant of the twisted generalized feedback shift register (TGFSR). Both the TGFSR and the MT19937 were developed by Mastsumoto and Kurita [4].

The MT19937 algorithm is a good solution for hardware optimization as the output it generates has been proven to be free of long-term correlations [4]. It has a massive period of $2^{19937}-1$. Furthermore, it uses less computationally intensive mathematical functions. This would therefore enable the development of an area efficient hardware implementation.

3. EQUATIONS

The MT19937 algorithm generates sequences of uniformly distributed pseudo random integers 32 or 54 bit numbers between $[0, 2^w-1)$. The MT19937 algorithm is based on the following linear recurrence formula, where x and a denote word vectors, and A is w by w matrix. The proof of the algorithm is provided in [4].

$$x_{k+n} := x_{k+m} \otimes (x_k^u \parallel x_{k+1}^l) A, \text{ where } k=(0,1, \dots)$$

4. MT19937 HARDWARE ARCHITECTURE OVERVIEW

We have developed three staged hardware architecture for the MT19937 algorithm illustrated in figure 1 below. The three stages are seed generator, seed value modulator and output generation.

The seed generator accepts a 32 bit seed, stores it in an on chip dual ported block RAM and uses this seed to generate another seed. This process is repeated until a total of 624 seeds are generated.

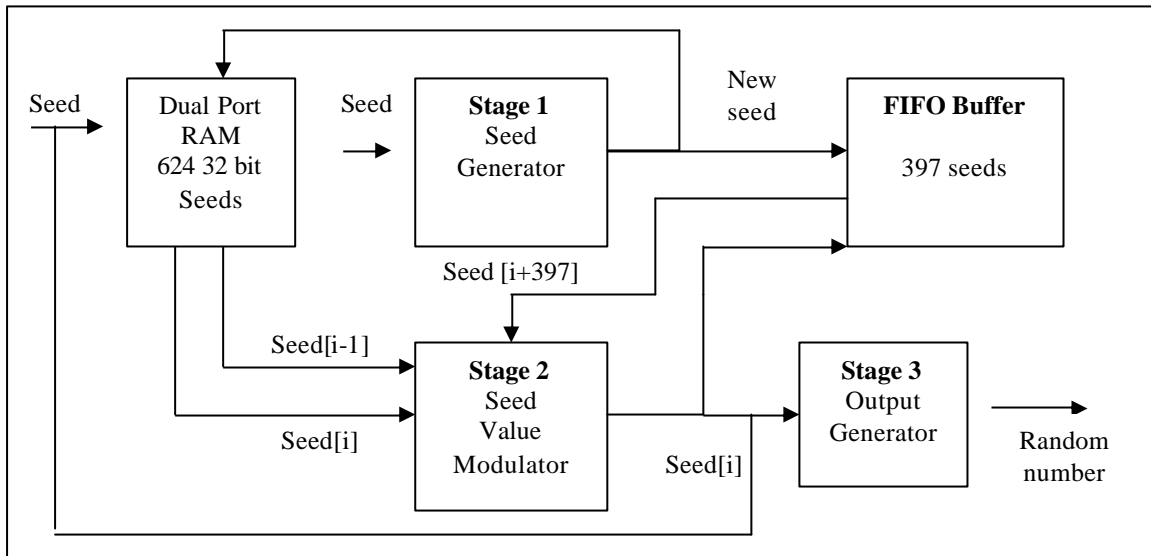


Figure 1: MT19937 Hardware Architecture Overview

On the generation of the 397th seed, the first, second seeds from block ram and the 397th seed are passed onto the stage 2. This stage involves the modification of the seed value based on the recurrence formula 1. The newly generated seed is stored back in block RAM and is output via the output generator in stage 3. Then onwards, each subsequent seed generated, along with the next two seeds from block ram are sent to the seed value modulator stage. The algorithm allows for both integer and uniform distribution of random numbers.

5. RESULTS

We have tested the implementation using the diehard tests of for random number generators. It has successfully passed the tests. The random number generated, as in the software implementation of MT19937, adheres to the uniform distribution and has a period of $2^{19937}-1$. The long period is particularly important as in scene simulations it prevents the problem of repetition of random numbers during the generation of each scene.

Our hardware implementation is significantly faster than the software implementation provided by Matsumoto [4]. The hardware optimized implementation requires only 26 milliseconds to generate 1 million random numbers as compared to 93 milliseconds in software.

We have considered hardware implementations of the Ziggurat random number generator, proposed by Zhang and Leong, and the Wallace random number generator, proposed by Lee and Leong [1]. Both these model the Gaussian probability density function.

On comparison against both software and hardware solutions, hardware optimized MT19937 generates random numbers faster and on comparison with other existing hardware implementations it is area efficient as well.

Table 1: Comparison of some software and hardware random number generators

Random number generator	H/W or S/W	Slices	Time to generate (10^6 numbers)	Gaussian /Uniform
MT19973	H/W	420	26 msec	Uniform
MT19937 [8]	S/W	NA	93 msec	Uniform
TT800 [8]	S/W	NA	60 msec	Uniform
Ziggurat[2]	H/W	770	62 msec	Gaussian
Wallace[1]	H/W	891	64 msec	Gaussian

6. CONCLUSION

The results of this paper are very significant. We were able to develop a hardware uniform random number generator that is area time efficient and has a very long period. All of measurement metrics are very important for infrared scene simulations.

7. ACKNOWLEDGEMENTS

The authors would like to thank Mr. Ross Frick for providing valuable advice which was influential in the development of this design.

8. REFERENCES

- [1] D. Lee, W. Luk, J.D. Villasenor, G. Zhang and P.H.W. Leong, "[A hardware Gaussian noise generator using the Wallace method](#)", *IEEE Transactions on VLSI Systems*, volume 13, number 8, pages 911-920, Aug 2005.
- [2] G. Zhang, P.H.W. Leong, D. Lee, J.D. Villasenor, R.C.C. Cheung and W. Luk, "[Ziggurat-based hardware Gaussian random number generator](#)", In *Proc. IEEE International Conference on Field Programmable Logic and its Applications (FPL)*, Tampere, Finland, Sep 2005.
- [3] Kiran, B. and Prasanna, V. K., "Reconfigurable Computing Systems", in *Proceedings of the IEEE*, 2002, pp:1201-1217.
- [4] Matsumoto, M. and Nishimura, T. (1998) Mersenne Twister: A 623-dimensionally equidistributed uniform pseudo-random number generator, *ACM Transactions on Modeling and Computer Simulation*, **8**, 3-30.
- [5] Matsumoto, M and Kurita, Y. Twisted GFSR generators. *ACM Transactions on Modeling and Computer Simulation*, **2**(3):179-194, 1992.
- [6] P. L'Ecuyer, "Random Number Generation", Chapter 4 of the *Handbook on Simulation*, Jerry Banks Ed., Wiley, 1998, 93--137.
- [7] Press, W.H., B.P. Flannery, S.A. Teukolsky, W.T. Vetterling, 1986; *Numerical Recipes, The Art of Scientific Computing*, Cambridge University Press, Cambridge, 818 pages, ISBN 0-512-30811-9
- [8] *Nihat Karaoglu* Mersenne Twister - A Study on random number generators [Internet]. Date Unknown. Last updated 2005. [cited 3 January, 2006]. Available from: <http://student.vub.ac.be/~nkaraogl/mt/mt.html>
- [9] R. P. Brent, Random number generation and simulation on vector and parallel computers (invited paper), *Proc. Fourth International EuroPar Conference* (Southampton, UK, 1-4 Sept 1998), D. Pritchard and J. Reeve (editors), *Lecture Notes in Computer Science*, Vol. 1470, Springer-Verlag, Berlin, 1998, 1-20
- [10] R. Brent. Random Number Generation and Simulation on Vector and Parallel Computers. In *Proceedings of EuroPar '98, Lecture Notes in Computer Science*, 1470, pages 1--20, September 1998.