Multiple long-period random number binary sequences Generator using LFSR

Neha Agrawal
MTech Scholar Deptt.of ECE, Truba Inst. Of Engg. & Info Tech. Bhopal.

Neelesh Gupta
Deptt.of ECE ,Truba Inst. of Engg. & Info. Tech. Bhopal.

Neetu Sharma
Deptt.of ECE ,Truba Inst. Of Engg. & Info. Tech. Bhopal.

Abstract:
This work presents the VHDL implementation for SRAM base linear feedback shift register long-period pseudorandom number generator. Random number generator is use forstrengthening and securing data of electronics communication. LFSR is the most efficient method for one bit random number generator. When many bits are required, LFSR can be extended by design with extra circuitry. The increase in length of random number sequence consumes more area. The SRAM base random number generator is area efficient.

Introduction:
For the testing hardware in digital communication transmission and reception of a signal uses Pseudo random binary sequences (PRBSs). The most RNG are software base which are random in sequence but follow a predefined sequence. Thus, these numbers are not truly random and can easily be predicted by having knowledge of the generating algorithm. Random number is generated using shift register with series connected flipflop with some flipflop output is fedback to the input offirst flipflop through xor logic. The length of sequence depends on which output of flipflop is feedback to first flipflop input, this hardware structure is linear feedback shift register (LFSR). An LFSR is a shift register that, when clocked, advances the signal through the register from one bit to the next most-significant bit. It propagates the binary data stores in series connected flipflops. It generates all possible states except all os state and will generate a maximum length pseudo random sequence. For an n stage LFSR, there are 2n-1 states, and the M sequence is 2n-1 bits long. Hence, the M-sequence is periodic, and after the 2n-1 distinct values, it repeats itself in the next samples.
It propagates the binary data stores in series connected flipflops. It generates all possible states except all os state and will generate a maximum length pseudo random sequence. For generation of required sequence the LFSR is initialized with start sequence. The feedbak through xor logic causes the value in the shift register to cycle through a set of unique values which is repeated after specified length of sequence.

Linear Feedback Shift Register LFSR :
The n number of flipflop base Linear Feedback Shift Registers generates the random sequence of 2n-1 states. At every edge trigging of clock input, the binary data of the registers are shifted right by one position. There is feedback from flipflops output to the first flipflop through XNOR or XOR gate. A value of all "1"s is illegal in the case of a XNOR feedback. A count of all "0"s is illegal for an XOR feedback. The design of eight bit linear shift shift register requires 8 flipflops, 1 xor logic gate, and 255states. The sixteen bit linear shift shift register requires 16 flipflops, 1 xor logic gate, and 65535 states. The 32 bit linear

shift shift register requires 16 flipflops, 1 xor logic gate, and 4294967295 states.

The following example shows the implementation of 8-bit LFSR having bit 8, bit 4, bit 3, and bit 2 as the parity bits. So the tap sequence will look like; $x^8+x^4+x^3+x^2+1$ This shift register is initialized with a key. Every clock cycle, all of the bits in the register are shifted such that the least significant bit is output. The new most significant bit is computed from the XOR of certain bits in the register. This shift register is rising edge triggered with an active-high enable and active-high reset.

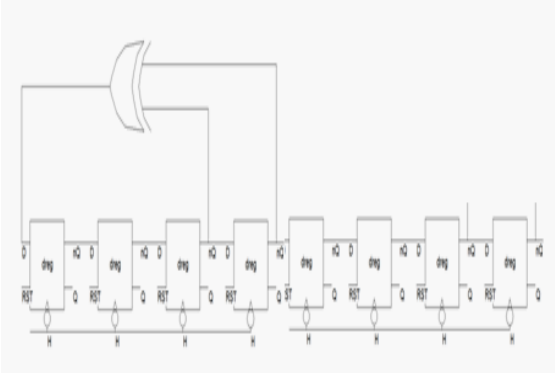

Fig Schematic circuit for 8 bit LFSR.

Implementation Methodology:

We will Increase number of bits in generator by generating the sequence using a Linear feedback shiftregister and making measurements on VHDL simulator. Using Finite state machine to genrate wide range of code with small amount of of logic using VHDL. We calculates the latency , throughput and computational time of LFSR and LUTs. The linear feedback shift register is made up of two parts: a shift register and a feedback function. The shift register is initialized with n bits (called the key), and each time a keystream bit is required, all of the bits in the register are shifted 1 bit to the right. So the least significant bit is the output bit. The new left-most bit is computed as the XOR of certain bits in the register. This arrangement can potentially produce a 2n-1 bit-long pseudo-random sequence (referred to as the period) before repeating.

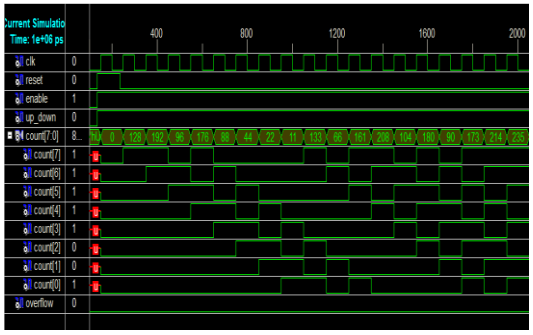

Fig Eight bit LFSR with updown logic

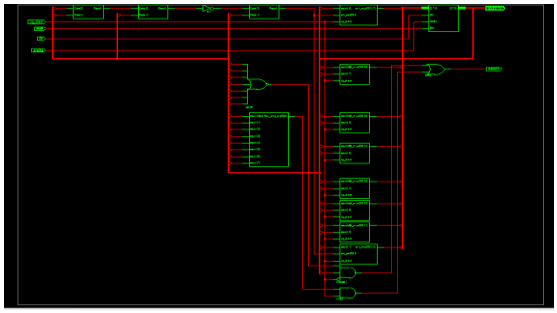

Fig RTL view of eight bit LFSR with updown logic

The fig above shows the timing simulation of 8-bit LFSR with up down logic. The tap sequence will look like; $x^8+x^4+x^3+x^2+1$ This shift register is initialized with a key. Every clock cycle, all of the bits in the register are shifted such that the least significant bit is output. The new most significant bit is computed from the XOR of certain bits in the register. This shift register is rising edge triggered with an active-high enable and active-high reset.

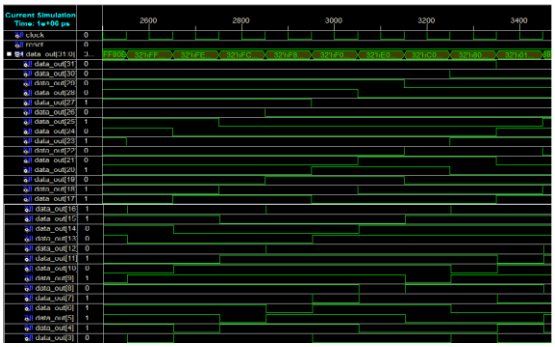

Fig Timing simulation of 32 bi random number generator

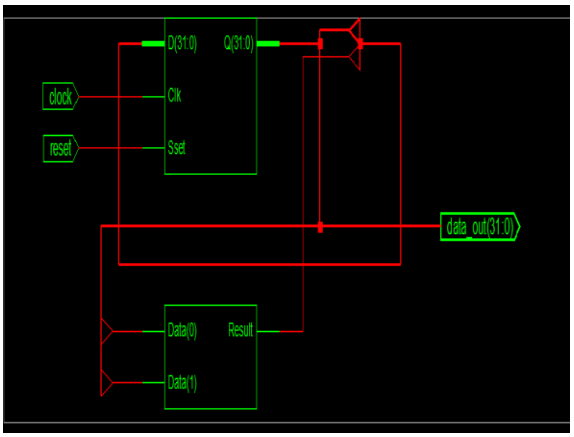

Fig RTL view of 32 bi random number generator

Implementation:

The proposed design is implemented using the VHDL hardware description language. The implementation supports a range of parameters to facilitate the experimental evaluation of design choices.

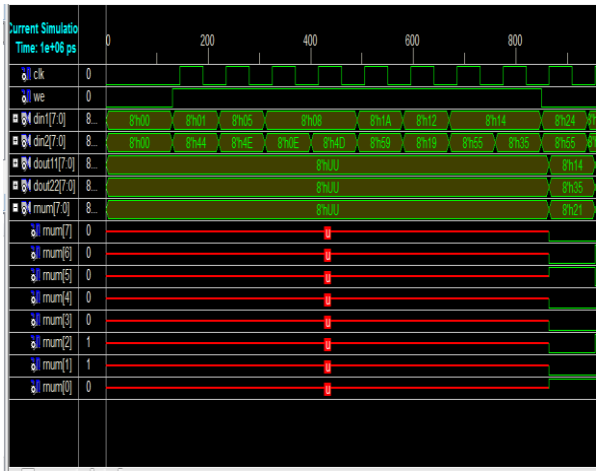

Fig Timing simulation of propose SRAM base RNG writing operation.

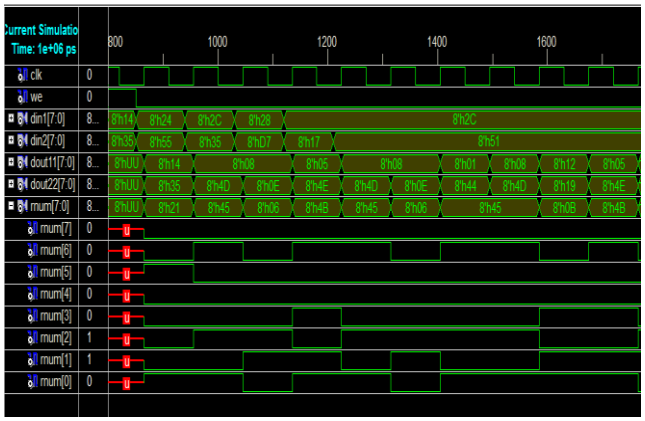

Fig Timing simulation of propose SRAM base RNG random number generate operation.

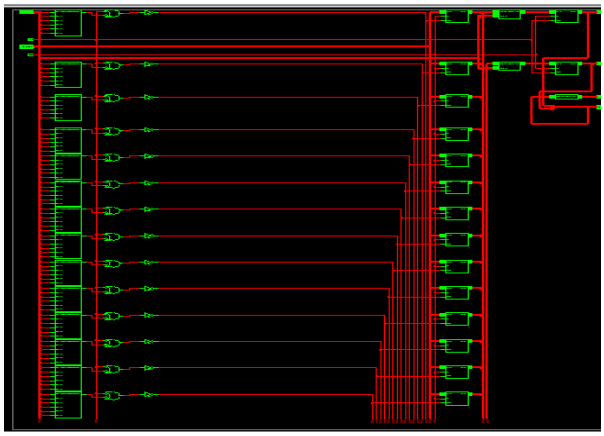

Fig RTL view of propose SRAM base RNG random number generate operation.

Synthesis report:

# Registers	: 130
8-bit register	: 130
# Multiplexers	: 2
8-bit 64-to-1 multiplexer	: 2
# Xors	: 1
8-bit xor2	: 1
# Registers	: 1040
Flip-Flops	: 1040
# Multiplexers	: 2
8-bit 64-to-1 multiplexer	: 2
# Xors	: 1
8-bit xor2	: 1

The design is implemented using VHDL, a standard hardware description language. The implementation serves two main purposes. First, it verifies the correctness of the design.

Second, it allows for verifying the analysis experimentally. The VHDL implementation is tested and verified both through simulation and on hardware. Xilinx is used for logic simulation. A simulation waveform is used to verify the correctness of the implementation, by checking the produced results.

Table 1 Device Utilization.

Design Module	Slices	FFs	LUTs	RAMs	Freq MHz	Throughput Gb\s
LFSR 8 bit	5	8	-	-	10	0.081
LFSR 10 bit	5	10	-	-	10	0.1
LFSR 16 bit	4	17	-	-	10	0.16
LFSR 32 bit	5	32	-	-	10	0.32
SRAM base RNG	84	130	130	2	11.11	0.088

Table 1 shows the comparison of a number of FPGA-based PRNGs, in terms of quality metric, resource usage, and performance, although consuming more logic resource (this is because of the algorithm complexity), the our random number generator achieves a bit higher throughput. In addition, it is worth to note that the resource usage is less.

Conclusion:

In an LFSR, the bits contained in selected positions in the shift register are combined in some sort of function and the result is fed back into the register's input bit. By definition, the selected bit values are collected before the register is clocked and the result of the feedback function is inserted into the shift register during the shift, filling the position that is emptied as a result of the shift.Feedback around an LFSR's shift register comes from a selection of points (taps) in the register chain and constitutes XORing these taps to provide tap(s) back into the register. Register bits that do not need an input tap, operate as a standard shift register. It is this feedback that causes the register to loop through repetitive sequences of pseudo-random value. The choice of taps determines how many values there are in a given sequence before the sequence repeats. The implemented LFSR uses a one-to-many structure, rather than a many-to-one structure, since this structure always has the shortest clock-to-clock delay path.

References:

[1] Yuan Li, Paul Chow, Jiang Jiang, Minxuan Zhang, and Shaojun Wei "Software/Hardware Parallel Long-Period Random Number Generation Framework Based on the WELL Method" Ieee Transactions On Very Large Scale Integration (Vlsi) Systems year 2013.

[2] David B. Thomas, and Wayne Luk "The LUT-SR Family of Uniform Random Number Generators for FPGA Architectures " IEEE Transactions On Very Large Scale Integration (Vlsi) Systems, Vol. 21, No. 4, April 2013.

[3] Fabio Pareschi, Gianluca Setti, and Riccardo Rovatti " Implementation and Testing of High-Speed CMOS True Random Number Generators Based on Chaotic Systems" IEEE Transactions On Circuits And Systems—I: Regular Papers, Vol. 57, No. 12, Pp No. 3124 December 2010.

[4] Walter Aloisi and Rosario Mita "Gated-Clock Design of Linear-Feedback Shift Registers" Ieee Transactions on Circuits and Systems—II: Express Briefs, Vol. 55, No. 6, pp no 546 year june 2008.

[4] Sharmitha.E.K, Sharmitha.E.K, Nisha Angeline. M, Palanisamy.C "High Throughput LFSR Design for BCH Encoder using Sample Period Reduction Technique for MLC NAND based Flash Memories " International Journal of Computer Applications (0975 – 8887) Volume 66– No.10, March 2013.

[5] François Arnault, Thierry Berger, Marine Minier, and Benjamin Pousse " Revisiting LFSRs for Cryptographic Applications " IEEE Transactions On Information Theory, Vol. 57, No. 12, December 2011 pp no.8095.
[6] Srinivasan Krishnaswamy and Harish K. Pillai "On the Number of Linear Feedback Shift Registers With a Special Structure" IEEE Transactions On Information Theory, Vol. 58, No. 3, March 2012 pp no. 1783.
[7] Wei Wei, Guodong Xie, Anhong Dang, and Hong Guo "High-Speed and Bias-Free Optical Random Number Generator" IEEE Photonics Technology Letters, Vol. 24, No. 6, March 15, 2012 pp no. 437.
[8] Jui-Chieh Lin, Sao-Jie Chen, and Yu Hen Hu "Cycle-Efficient LFSR Implementation on Word-Based Microarchitecture " IEEE Transactions On Computers, Vol. 62, No. 4, April 2013 pp no .832.

Author`s Profile

Neha agrawal1 has received the B.E. degree in electronics and communication engineering from patel college of science technology Bhopal (R.G.P.V. University),India, in 2010.she is pursuing her M.Tech. in Digital Communication from Truba Institute of Engineering and Information Technology Bhopal India. her research interest includes computer network , digital communication and VLSI.

Neelesh Gupta2 is working as A.P. electronics and communication engineering in Truba Institute of Engineering and Information Technology Bhopal India. He is also H.O.D. in dept. of electronics and communication engineering. He guided me through this entire thesis wherein he gave his insights on varying aspects of this topic.

Neetu Sharma3 is working as A.P. electronics and communication engineering in Truba Institute of Engineering and Information Technology Bhopal India. she also guided me in collecting various knowledge and data about thesis topic.