International Journal of Engineering and Techniques - Volume 2 Issue 3, May – June 2016

RESEARCH ARTICLE OPEN ACCESS

Design and Implementation of 10-Bit Pseudo Random Sequence Generator for 50 MHz

G.Hemanth Kumar[1], Dr.M.Saravanan[2] , Charan Kumar.K[3]
[1,2,3](Dept. of Eectronics and Instrumentation Engineering , Sree Vidyanikethan Engineering College, Tirupati)

Abstract:

The FPGA based implementation of Pseudo Random Sequence generator an algorithm is used for generating a sequence of random numbers i.e. the pseudo-random number generator (PRNG). Pseudorandom numbers are important in securing the Internet traffic in places where low memory utilization in the area of cryptography and low level of security is required. The pseudo-random number generator (PRNG) is based on linear feedback shift register (LFSR). pseudorandom bit sequences are provide by Linear feedback shift registers for a variety of purposes and widely used as synchronization codes, masking or scrambling codes, and signal sets in CDMA (code division multiple access) communications, white noise signals in Communication systems key stream generators in stream cipher cryptosystems, random number generators in many cryptographic primitive algorithms, and for testing vectors in hardware design.

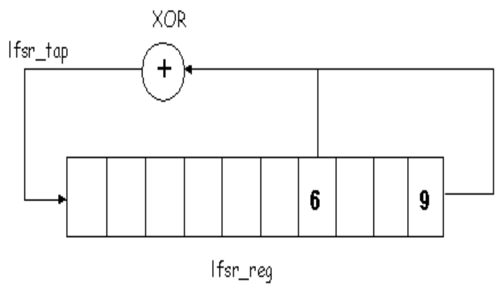
Keywords— **PRNG, LFSR, FPGA, ML-LFSR, CDMA, True" Random Number Generators (TRNG)**

I. INTRODUCTION

For various cryptographic applications, random numbers are necessary. The generated sequence is uniformly distributed or non-uniformly distributed. Random numbers are needed for a wide range of application in Science and Engineering which require statistical random input. The random number generator that produces a sequence of number i.e. appears random. The two types of generator have used 1] True random number generator [TRNG]; 2] Pseudorandom number generator [PRNG]. TRNG generator used for generating random data, but have existed. PRNG which uses a computational method based on certain algorithms produces random sequences that repeats are known as PRNG. This PRNG is achieved by LFSR whose input is a linear function of the previous state and the linear congruence algorithm. Using a linear congruence algorithm requires time consuming operation and its hardware implementation is very complicated. But using LFSR which is made up of shift register permits very fast generation of random sequences.

In this paper we propose design of ML-LFSR 10-bit. The principles outlined for 10-bit design are applicable to any N-bit ML-LFSR. In order to design the 10-bit ML-LFSR, the inputs to the lfsr_tap are defined by the characteristic polynomial that considers

Fig. 1 Maximum Length LFSR



The data generated by ML-LFSR is almost random. The data generated by LFSR can be can be taken in serial bit stream or as a parallel-out form. Usually the serial bit stream is taken from the MSB of the LFSR. Random number sequence is normally generated at lower frequencies (say e.g.: 4MHz). Based on the requirements for various purposes, the appropriate FPGA is used. The scope is to generate the random sequence of wide range of frequencies (i.e. from 4 MHz to 50 MHz).

II. METHODOLOGY

A. Linear Feedback Shift Register Sequences

Feedback shift register sequences have been widely used as white noise signals in communication systems synchronization codes and formasking or scrambling codes, signal sets in CDMA communications, key stream generators in stream cipher cryptosystems, random number generators in many cryptographic primitive algorithms, and for testing vectors in hardware design.

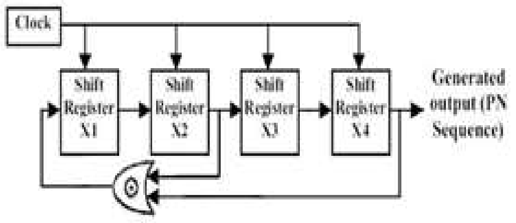


Fig. 2 Linear feedback shift register (LFSR)

If the feedback functionf $(X_0, X_1, \ldots, X_{n-1})$ is a linear function, i.e., if it can be expressed as

$$f(X_0, X_1, \ldots, X_{n-1}) = C_0X_0 + C_1X_1 + \cdots + C_{n-1} X_{n-1},\ C_i \in F$$

The maximal period of an LFSR sequence with n stage is 2n − 1, which is called a maximal length sequence, shortened as m-sequence, or pseudo noise sequence in communications. Usually, associate a linear feedback Boolean function with a polynomial

$$f(x) = X_n + C_{n-1}X^{n-1} + \cdots + C_1X + C_0$$

B. Non-Linear Generators

Linear feedback shift register (LFSR) sequences are widely used as basic functional blocks in key stream generators in stream chipper models due to their fast implementation in hardware as well as in software in some cases. In LFSR based stream ciphers, there are

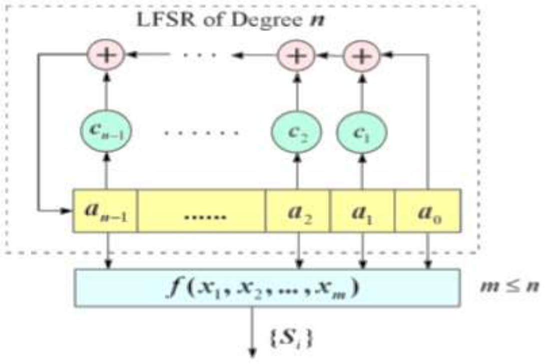


Fig.3. Filtering Sequence Generator

mainly two types of operations which operate on the LFSRs: Outputs of one LFSR or multiple LFSRs are transformed by nonlinear functions with or without memory states, including those by using mixed finite field operations and integer Modular operations and change the clock of the LFSRs to an irregular clock or by deleting some output bits of the LFSRs (clock control or shrinking generators).

III. IMPLEMENTATION OF LFSR BASED PRNG

For generation of random numbers the Middle square algorithm is used for the implementation of LFSR based PRNG. This algorithm follows take any number square it, remove the middle digits of the resulting number as the "random number", then use that number as the seed for the next iteration. For a generation of n-digit numbers, the period can be no longer than 8n. If the middle 4 digits are all zeroes, the generator outputs zeroes forever. If the first half of a number in the sequence is zeroes, the subsequent numbers will be decreasing to zero. For simulation, the random numbers generator must have the following desire properties:

It should produce numbers that are random. It should be fast. It should not require much computer storage. It should have a long period before cycling. The object is to select a generation method that

produces all of the random numbers that are needed before cycling occurs.
It should not regenerate; that is not repeating the same number over and over.
It should generate stream of random numbers that can be reproduced. That is for debugging and testing the response of the simulation program for the same stream of random numbers.
STEP1- Declare all the input and output ports The inputs are clock, reset. The outputs are data_out, bit_out
STEP2- Declare the architecture part of the entity
STEP3- Declare lfsr_reg as signal
STEP4- Initialize a clock sequence using an if- else statement
STEP5- Initialize the default sequence when reset =1
STEP6-When reset=0 perform xor sequence of the lfsr
STEP7- If reset =1 at any stage repeat step 5 and step 6.
STEP8- Transfer the value of last bit as output

IV. 10-BIT LFSR ITS DESIGN AND SIMULATION

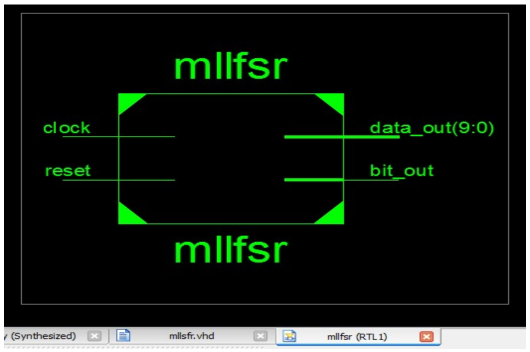


Fig4: MLLSFR RTL SCHEMATIC

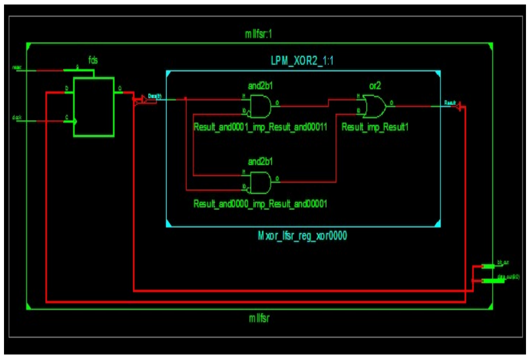


Fig5: RTL INTERNAL

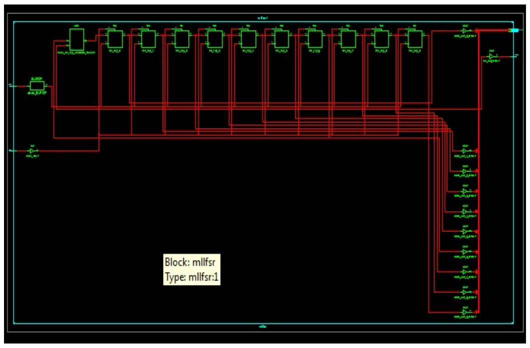
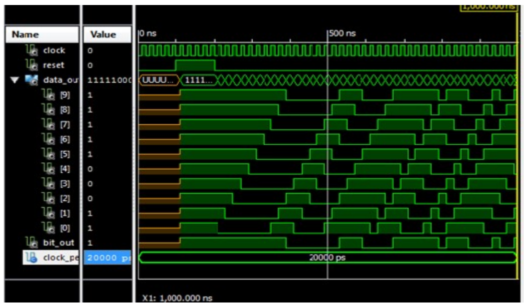
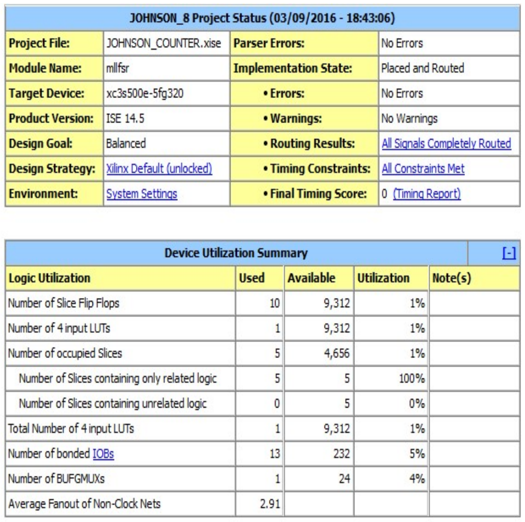


Fig6: MLLSFR TECHNOLOGY SCHEMATIC





JOHNSON_8 Project Status (03/09/2016 - 18:43:06)			
Project File:	JOHNSON_COUNTER.xise	**Parser Errors:**	No Errors
Module Name:	mllfsr	**Implementation State:**	Placed and Routed
Target Device:	xc3s500e-5fg320	**• Errors:**	No Errors
Product Version:	ISE 14.5	**• Warnings:**	No Warnings
Design Goal:	Balanced	**• Routing Results:**	All Signals Completely Routed
Design Strategy:	Xilinx Default (unlocked)	**• Timing Constraints:**	All Constraints Met
Environment:	System Settings	**• Final Timing Score:**	0 (Timing Report)

Device Utilization Summary				[-]
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Flip Flops	10	9,312	1%	
Number of 4 input LUTs	1	9,312	1%	
Number of occupied Slices	5	4,656	1%	
Number of Slices containing only related logic	5	5	100%	
Number of Slices containing unrelated logic	0	5	0%	
Total Number of 4 input LUTs	1	9,312	1%	
Number of bonded IOBs	13	232	5%	
Number of BUFGMUXs	1	24	4%	
Average Fanout of Non-Clock Nets	2.91			

Fig7: Simulate results for MLLFSR

V. CONCLUSIONS

A 10 bit code of maximal length is realized by shifting the input through the D-flip flops and feed backing the outputs of some registers known as taps again into the first register after passing them through the XOR gate. A dead lock condition arises in the case when the initial input into the first register of the XOR gate are all 0's. Under this condition the output of all the register of the PRSG remains as 0 at all instants of time. Therefore it is necessary that the initial to the PRSG to be equal to 1, the output of the XOR gate.

FPGA are the only one that could generate higher frequencies of the range of 50MHZ. The code is generated and simulated using Xilinx ISE7.1i. Different types of random generators are briefly discussed and the suitable type is chosen for more randomness sequence

REFERENCES

1. Panda Amit K, Rajput P, Shukla B, Design of Multi Bit LFSR PNRG and Performance comparison on FPGA using VHDL",International Journal of Advances in Engineering & Technology (IJAET), Mar 2012, Vol. 3, Issue 1, pp. 566-571.

2. Sewak K, Rajput P, Panda Amit K, "FPGA Implementation of 16 bit BBS and LFSR PN Sequence Generator: A Comparative Study", In Proce. of the IEEE Student Conference on Electrical, Electronics and Computer Sciences 2012, 1-2 Mar 2012, NIT Bhopal, India.

3. Katti, R.S. Srinivasan, S.K., "Efficient hardware implementation of a new pseudo-random bit sequence generator" IEEE International Symposium on Circuits and Systems, 2009. ISCAS 2009.

4. Amit Kumar Panda, Praveena Rajput, Bhawna Shukla"FPGA Implementation of of 8,16 And 32 Bit LFSR with Maximum Length Feedback Polynomial using "VHDL" 2012 International Conference on Communication Systems and
Network Technologies.

5.Lember, A.W.L.Eastman:"High Speed generation of Maximal Length Sequences" IEEE trans. Computers, Short notes, VOL c-20, PP227-229,1981.

6. Ding Jun, Li Na, Guo Yixiong, "A high-performance pseudo-random number generator based on FPGA" 2009 International Conference on Wireless Networks and Information

7. Efficient Shift Registers, LFSR Counters, and Long Pseudo-Random Sequence Generators, Application Note, Xilinx Inc.