International Journal of Electronics Communications and Electrical Engineering
ISSN : 2277-7040 Volume 3 Issue 5 (May 2013)
http://www.ijecee.com/ https://sites.google.com/site/ijeceejournal/

Data Scrambler for Ultra-Wideband Communication Systems

Davinder Pal Sharma

Department of Physics, Faculty of Science & Technology
The University of the West Indies, St. Augustine, Trinidad and Tobago.
Davinder.Sharma@sta.uwi.edu

Abstract. Indoor communications of any digital data, whether it is high-speed signals carrying multiple HDTV programs or low-speed signals used for timing purposes, will be shared over a digital wireless network in the near future. Such indoor and home networking requires high data rates, very low cost and very low power consumption. Ultra Wide-band (UWB) system has enormous bandwidth to provide a promising solution to satisfying these requirements and becomes an attractive candidate for future wireless indoor networks. In such systems, synchronization plays very critical role to ensure correct and reliable system operation. Improper synchronization can introduce timing errors during transmission that can be eliminated using a device called scrambler. Simulation and implementation of data scrambler for UWB communication systems is discussed in the present paper. Model of the scrambler is build using Matlab to perform simulation. For FPGA (Field Programmable Gate Array) based implementation of UWB scrambler, Xilinx's System Generator for DSP (Digital Signal Processing) tool is used along with Xilinx's ISE Design Suite. Implementation and simulation results are then compared.

Keywords: Scrambler, UWB Communication Systems, FPGA based Implementation, Simulation.

1 Introduction

UWB communication systems are currently the focus of research and development in wireless personal area networks (WPANs). These systems are capable of transferring data at the rate of 110Mbps to 480Mbps in realistic multipath environment. They consume very little power and silicon area [1]. In such communication network, synchronization ensures that operations occur in a logically correct order and is a critical factor in ensuring correct and reliable system operations. As the physical size of a system increases or as the speed of the operation increases, synchronization plays an increasingly dominant role in the system's design. Synchronization is thus a critical part of communication system design. The complications that occur due to non-synchronization are referred to as timing errors and the scrambler is a device that can eliminate transmission errors in communication systems. Data scrambler is basically used to encode transmitted data before the data goes to a descrambler, where the data is returned to its original form to be recognized by the receiver [2].

In the subsequent sections of this paper modeling and simulation of the scrambler for UWB communication systems using Matlab and Xilinx's System Generator for DSP has been discussed. Implementation of the scrambler on Spartan 3E FPGA chip using Xilinx's ISE Design Suite has also been explained and results are compared.

1.1 UWB Data Scrambler

Data Scrambler is generally made up of linear sequential filters with feedback paths, counters, storage elements and peripheral logic in their discrete form. Basic structure of a data scrambler is shown in Fig. 1. In general, the serial data enters into a linear feedback shift register, where at each time step, the input causes the contents of the registers to shift sequentially. In other words, each stage in the register, delays the signal by one time unit .The delayed version of the output signal is then fed back and modulo-2-addition is performed with the input signal. Input and output relation of the scrambler in general is given by

$$y(n) = x(n) + \sum_{k=1}^{m} h_k\, y(n-\mathrm{k}) \qquad \textbf{(1)}$$

Scramblers are based upon maximum length shift register sequence or M sequences. Maximum possible sequence length before register repeats must be 2^n-1. Binary sequence of this maximum length is known as M-sequence or pseudorandom binary sequence because they pass several statistical tests for randomness. The auto-correlation function of such sequences resembles with the white noise.

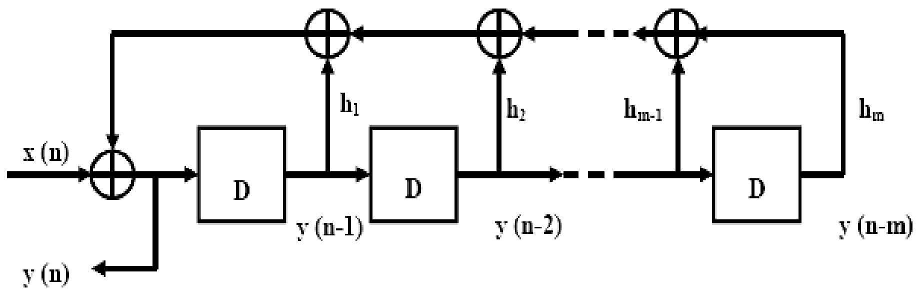


Fig. 1. Basic structure of a data scrambler

While implementing such devices, the design problem is to select the shift registers taps, which generate a M sequence. The theory behind it is based on finite fields so it involves algebraic polynomials and finite field arithmetic (modulo-2-addition). Polynomials, which generate M sequences, should be primitive. A polynomial y (x) of degree ‘n’ is primitive if it is irreducible i.e. has no factors except 1 and itself and if it divides x^k+1 for $k = 2^m-1$ and does not divide x^k+1 for $k< 2^m-1$. Characteristic polynomial for M sequence generator is given by

$$y(x) = 1 + \sum_{k=1}^{m} h_k\, x^k \qquad \textbf{(2)}$$

So mathematical operation performed by the scrambler is basically equivalent to dividing the input information sequence by a Generating Polynomial (GP). The polynomial resulting in the fewest feedback connection is often the most attractive for scrambling purpose [3-5].

IEEE recommended polynomial for scrambling in the UWB communication systems is [6]

$$y(x) = 1 + x^{-14} + x^{-15} \qquad \textbf{(3)}$$

2 Matlab Based Modelling and Simulation

Using the structure shown in Fig. 1. and eqn. (3), a model of the data scrambler for UWB communication system was created in Matlab using Simulink, Communication System, DSP System toolboxes and Xilinx’s System Generator for DSP [7-10]. Model of the UWB Scrambler is shown in Fig. 2.

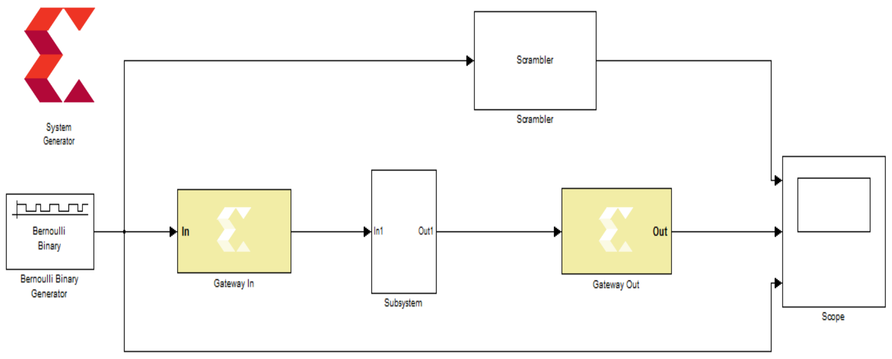


Fig. 3. Simulation model of UWB scrambler

Bernoulli Binary Generator block was used to generate test bit stream for the simulation. This block generates random binary numbers with probabilities **p** using a Bernoulli distribution with mean value **1-p** and variance **p (1-p).** Value of **p = 0.3** was chosen since this probability provides a satisfactory increase in transitions [11]. The input signal was given to the Simulink's Scrambler block (which is available in Communication System Toolbox) and then to the scope. The input signal was also sent straight to the scope for comparison purpose. Parameters like scrambling polynomial and initial states etc. were set on the scrambler block as per eqn. (3).

The input bit stream was also fed to the blocks made from Xilinx's System Generator for DSP for implementation of UWB scrambler on FPGA. To ensure that other blocks would be in sync with the Xilinx blocks, a Gateway In and Gateway Out blocks were included in the overall design. The Xilinx Gateway In block is the input into the Xilinx portion of the Simulink design. These blocks convert Simulink integer, double and fixed-point data types into the System Generator fixed-point type. Each block defines a top-level input port in the HDL design generated by System Generator. The System Generator token serves as a control panel for controlling system and simulation parameters, and it is also used to invoke the code generator for net listing. Once a System Generator token was added to a model, it became possible to specify how code generation and simulation should be handled. Subsystem block is Xilinx based UWB scrambler incorporated for the purpose of implementing UWB scrambler on FPGA and comparing implementation results with the simulation results. Details of Subsystem block are shown in Fig. 3.The delay blocks used in the subsystem were given same initial values as of Simulink's scrambler block. The output from the Simulink's scrambler block, the output from the Xilinx's Subsystem and the Bernoulli Binary Generator were all fed to same scope for comparison.

The entire model was then simulated and simulation results are shown in Fig. 4. Comparing the output obtained from the Scrambler block to the output obtained from the Subsystem (scrambler circuit built using Xilinx blocks) , it is clear that the output waveforms using two approaches are identical and randomization of input signal ensures that scrambling has been taken place.

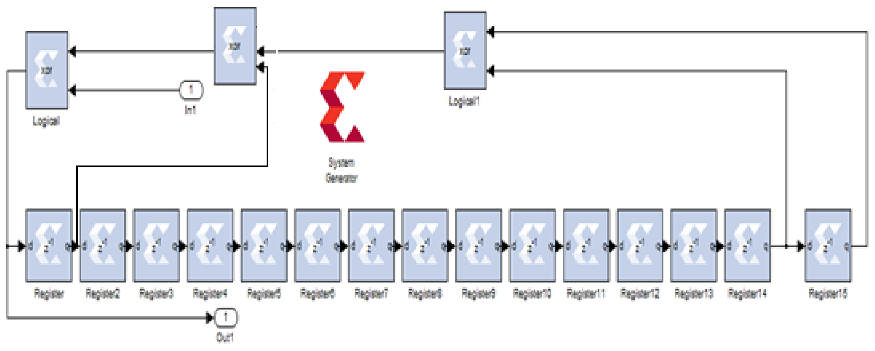


Fig. 3. Structural details of Subsystem block (practical UWB scrambler)

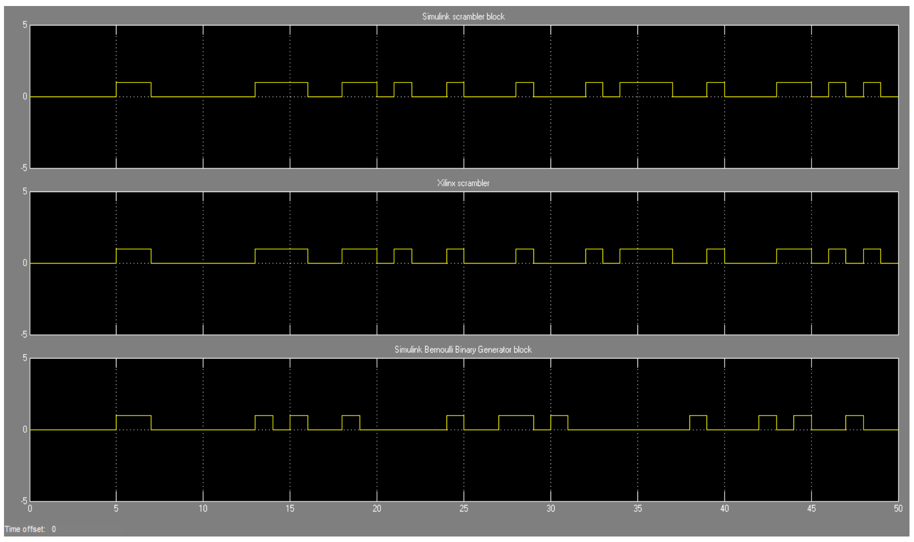


Fig. 4. Simulation results for UWB Scrambler

The increase in transitions show that the long sequences of null characters have been broken up, thus the signal has been made more random.

3 FPGA Based Implementation

To implement the scrambler on hardware we have many options. It can be implemented on an Applications Specific Integrated Circuit (APIC), Digital Signal Processor [12], Microprocessor, Microcontroller or on FPGA. The easier and efficient approach is to implement it on an FPGA. This is the most cost efficient approach and it facilitates the mistakes that may be made by inexperienced programmers. FPGA can carry out a wide range of possible tasks, which makes it the ideal choice for implementation of digital systems.

The subsystem shown in Fig. 3 was used to implement UWB scrambler on the FPGA. Spartan 3E FPGA Development kit as shown in Fig. 5 was used for FPGA based implementation of UWB scrambler. A bit file that actually configures the FPGA was created and uploaded to the FPGA. At this stage, the circuit was run through the FPGA and the output waveform obtained were recorded and compared to the input waveforms using I-Sim (one of the ISE design studio packages) software to see if scrambling occurred. The resulting waveforms are shown in Fig. 6. In order to show the transitions clearer, the time axis was set to nanoseconds. On comparing the output waveform with the input, scrambling is evident, as there is significant increase in transitions between the two waveforms.

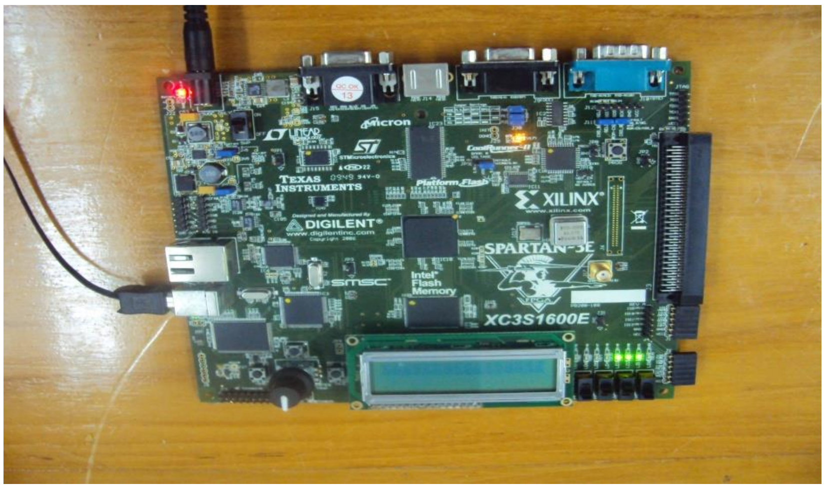


Fig. 5. Spartan-3E FPGA development kit

Once scrambling was obtained, the practical output waveform was then compared to the output waveforms obtained from the simulation. Results are shown in Fig. 7. From the results it can be seen that the waveform from the FPGA output is consistent with the waveforms obtained from the simulation. This confirm the successful implementation of UWB scrambler on Spartan 3E FPGA.

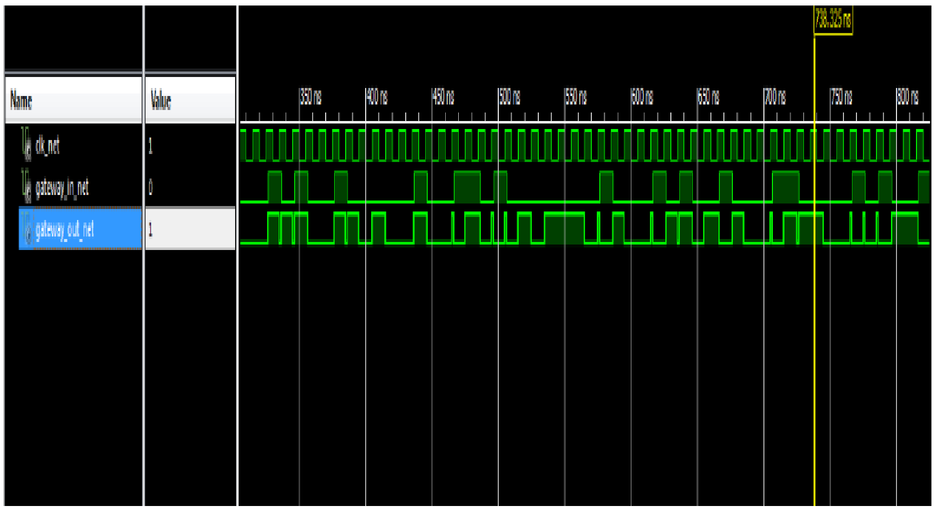


Fig. 6. Waveforms obtained from FPGA through I-Sim

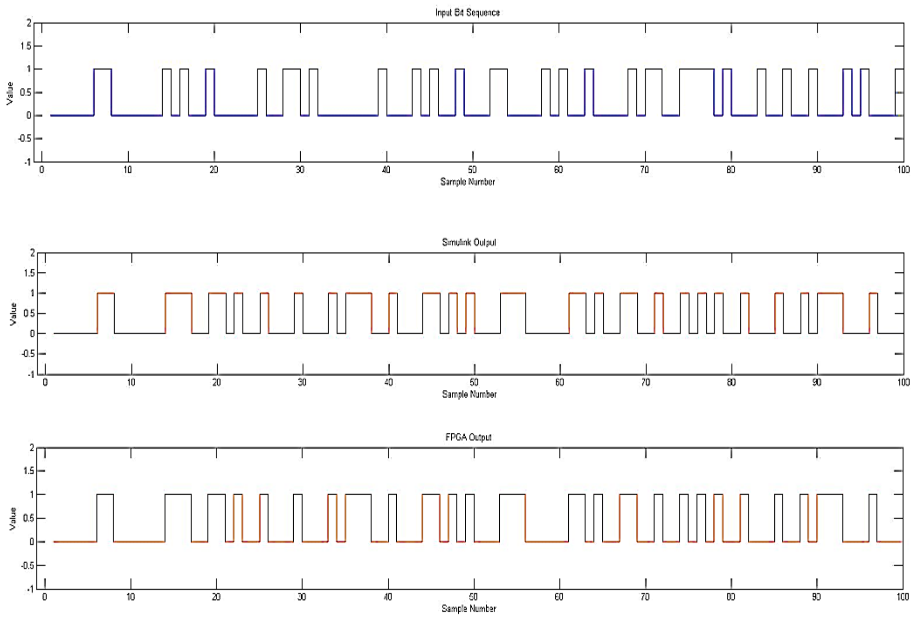
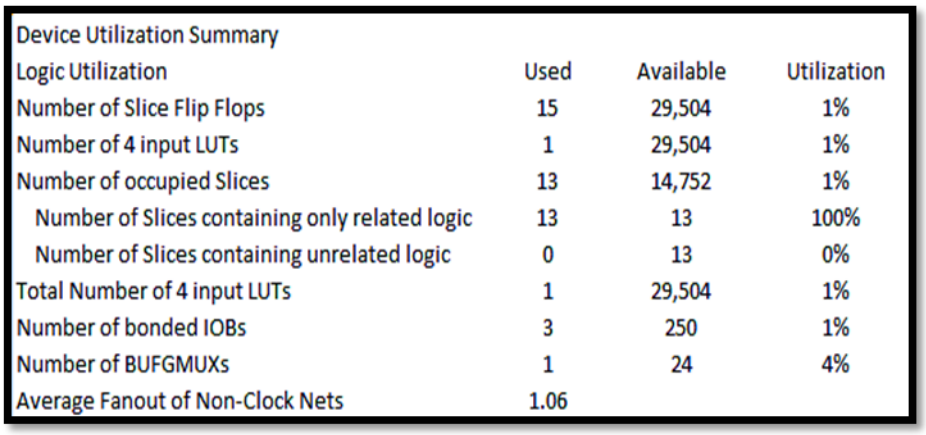


Fig. 7. Comparison of practical and simulation results

Device utilization summary for the current implementation is given in Table 1. This table shows that very little resources of FPGA were used for the present implementation and entire UWB communication system may be implemented on a suitable FPGA chip.

Table 1. Device Utilization Summary



Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slice Flip Flops	15	29,504	1%
Number of 4 input LUTs	1	29,504	1%
Number of occupied Slices	13	14,752	1%
Number of Slices containing only related logic	13	13	100%
Number of Slices containing unrelated logic	0	13	0%
Total Number of 4 input LUTs	1	29,504	1%
Number of bonded IOBs	3	250	1%
Number of BUFGMUXs	1	24	4%
Average Fanout of Non-Clock Nets	1.06		

Conclusion

The data scrambler for UWB communication system was successfully modeled and simulated on Matlab using its Simulink, Communication System and DSP System toolboxes. The scrambler circuit was also simulated using Xilinx's System Generator for DSP Toolbox for its practical implementation on Spartan-3E FPGA. Simulation results from Matlab and I-Sim, verifies the proper scrambling operation. Simulation results were then compared with implementation results and are found in good agreement. From this case study it is clear that FPGA based implementation of UWB communication systems is easy, efficient and cheaper. The entire UWB communication system can also be implemented on suitable FPGA.

References

1. Siriwongpairat, W. P., Liu, K.J.R.: Ultra-Wideband Communications Systems - Multiband OFDM Approach. John Wiley & Sons Inc., New Jersey (2007).
2. Mo, S. S., Gelman, A. D. : Scrambler design to reduce power spectral density of UWB signals in IEEE 802.15.3a. In: IEEE International Conference on Communication, pp. 3586–3590. IEEE Press, New York (2004).
3. Savage, J.E.: Some simple self –synchronizing digital data scramblers. The Bell System Technical Journal. 46, 449-487 (1967)
4. Cluzeau, M.: Reconstruction of a linear scrambler. IEEE Transactions on Computers. 56, 1283–1291 (2007)
5. Liu, X.B., Koh, S.N., Wu, X.W., Chui, C.C.: Reconstructing a linear scrambler with improved detection capability and in the presence of noise. IEEE Transactions on Information Forensics and Security. 7, 208-218 (2012).
6. IEEE 802.15: Working group for wireless personal area networks (WPANs). http://www.ieee802.org/15/.
7. Simulink User's Guide. The Math Works Inc., MA (2013). http://www.mathworks.com/help/pdf_doc/simulink/sl_using.pdf.
8. Communication System Toolbox User's Guide. The Math Works Inc., MA (2013) http://www.mathworks.com/help/pdf_doc/comm/comm.pdf
9. DSP System Toolbox User's Guide, The Math Works Inc., MA (2013) http://www.mathworks.com/help/pdf_doc/dsp/dsp_ug.pdf
10. Xilinx System Generator for DSP User's Guide**.** Xilinx Inc., USA (2012). http://www.xilinx.com/support/documentation/sw_manuals/xilinx14_1/sysgen_user.pdf
11. Sharma, D.P., Singh, J.: Simulation and spectral analysis of the scrambler for 56Kbps modem. The Journal of Signal Processing Systems. 67, 269-277 (2012)
12. Sharma, D.P., Singh, J.: DSP based implementation of scrambler for 56Kbps modem. Signal Processing – An International Journal. 4, 85-96 (2010)