DESIGNING OF STREAM CIPHER ARCHITECTURE USING THE CELLULAR AUTOMATA

[1]Brundha K A
MTech
Email:[1]brundha1905@gmail.com

Abstract— Pseudo-random number generators (PRNGs) are a key component of stream ciphers used for encryption purposes. While linear feedback shift registers (LFSRs) combined with nonlinear feedback shift registers (NFSRs) have typically been utilized for PRNGs, the use of cellular automata (CA) is another viable option. This paper explores the combination of LFSRs and CA as the key components of an efficient stream cipher design for implementation on Field Programmable Gate Arrays (FPGAs). The proposed stream cipher design builds upon a recent published design known as A2U2. The suite of statistical tests found in the DIEHARD program is utilized to evaluate the quality of random numbers from the proposed CA-based stream cipher. Comparisons with the A2U2 design indicate that the use of CA have the potential to improve the quality of the random numbers generated and hence increase the security of the cipher.

***Index Terms*— Cellular Automata, Linear/Non- linear feedback shift registers, stream cipher, FPGA encryption.**

I. INTRODUCTION

Stream ciphers represent an important class of encryption hardware that target applications with tight constraints on logic gates and memory or where high-throughput is necessary. For example, RFID tags are being used in a wide range of applications, many requiring secure transmission of identifying information and other data. Due to the tight limits on power and hardware resources, stream ciphers are attracting interest over the more complex block cipher designs for implementation in RFID tags. A key challenge in the design of a good stream cipher is to balance an efficient hardware implementation while making it difficult for an adversary to decrypt the transmitted data. The main component of the stream cipher is the key stream generator, which can be viewed as a pseudo-random number generator (PRNG). Important metrics for the performance of the PRNGs are speed, area, and power dissipation, while producing high-quality random numbers. A linear feedback shift register (LFSR), which is implemented from a cascade of flip-flops and a few XOR gates, typically forms the core of a PRNG. In addition, nonlinear feedback shift registers (NFSRs) must be included in a key stream generator design to remove the linearity in the encrypted cipher text, making it more difficult for an adversary to discover the secret key.

In this work, the impact of adding cellular automata
(CA) to the key stream generator is considered. The implementation of CA is relatively straightforward using Field Programmable Gate Arrays (FPGAs) due to their nearest neighbor interconnectivity and regularity in their physical layout [4]. The number of transistors on an integrated circuit has continued to exponentially increase as the semiconductor industry follows Moore’s law. This has allowed FPGA technology to implement more complex designs

that were formerly the exclusive domain of ASIC designs. While this paper will focus on the implementation of an efficient stream cipher using FPGA technology as a proof of concept, this technology is easily transferable to VLSI technology. The main contributions of this paper are the introduction of CA into stream cipher design, FPGA implementation and test of the stream cipher, and characterization results with DIEHARD and the entropy test.

This paper is organized as follows. The second section provides some background on the design of stream ciphers and the key components, the NFSRs and CAs. Previous work in the area of stream cipher design is also covered. The next section describes the research method used to design and test the stream ciphers. The fourth section contains the results and a discussion of their implication. Conclusions and future work are given in the final section.

II. Previous work

This section describes the basic concept of stream cipher design and overviews the key components, the linear feedback shift registers (LFSRs) and cellular automata (CA). A review of previous works in stream cipher design is also given.

The proposed stream is based upon the A2U2 stream cipher. The A2U2 stream cipher is a hardware-based stream cipher proposed for extremely resource limited devices such as RFID tags [1]. The KATAN [2] and A2U2 [1] ciphers utilize a pair of nonlinear feedback shift registers (NFSRs) in the main part of their design. The use of an NFSR instead of an LFSR improves the resistance of the cipher from various forms of cryptanalysis, such as correlation attacks and algebraic attacks [6]. As noted by Hortensius et al. the quality of the random number sequence generated by a cellular automata-based register is better than that of an LFSR [3]. The proposed stream cipher combines the A2U2's design principle with the cellular automata implementation. It consists of a 17 bit NFSR as the A2U2 stream cipher but replaces the shorter-length NFSR of the A2U2 with 9 bit maximal length cellular automata as depicted in Figure 2.

The proposed stream cipher is composed of five different blocks. The five blocks are: i) a counter, ii) a 17 bit long non-linear feedback shift register, iii) a 9 bit long cellular automata, iv) a key bit mixing mechanism, and v) a filter function. The overview of the proposed stream cipher is shown in Figure 1.

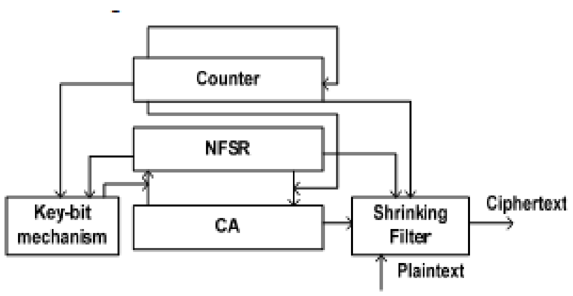


Figure 1: Overview of the stream cipher architecture

The proposed stream cipher uses the LFSR-based counter (see Figure 2) as in the A2U2 stream cipher because it has already been optimized to reduce the number of gates.

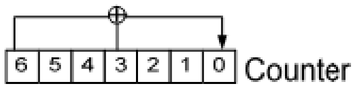


Figure 2: Counter used in proposed stream cipher

The feedback function of the counter is defined as:

$$C_f = C[6] \oplus C[3]$$

Where '$\oplus$' indicates the XOR operation and $C[i]$ represents the *ith* counter bit. Regarding the interconnection between the NFSR and CA, the feedback of the NFSR provides the feedback to the CA and vice versa, as shown in Figure 3. The feedback functions for the CA and the NFSR are represented by the following polynomial

$$B_t = N[16] \oplus \overline{(N[14] \cdot N[13])} \oplus N[11] \oplus \left(\overline{N[9] \cdot C[6]}\right) \oplus \left(\overline{N[6] \cdot N[5] \cdot N[4]}\right) \oplus \left(\overline{N[3] \cdot N[1]}\right)$$

$$A_t = CA[3] \oplus K_1$$

Where $C[i]$, $N[i]$, $CA[i]$ and K1 represent the ith counter bit, ith NFSR bit, ith CA bit, and the key bit generated by the key bit mechanism, respectively

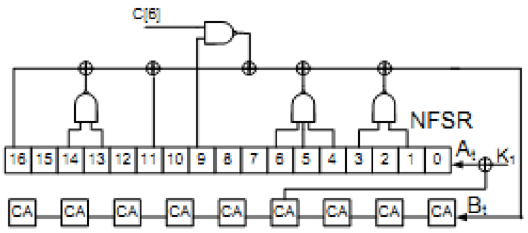


Figure 3: NFSR and the CA combination

An important part of the proposed stream cipher design is the CA implementation. Cellular automata with binary state values can be viewed as an array of cells where each cell can assume either the value 0 or 1. The CA will evolve in discrete time steps, where the next state of each cell interacts with their immediate neighbors based upon a local rule. A general configuration of a one dimensional CA with binary state values and a neighborhood consisting of the cell's own state and those of its immediate neighbors is shown in Figure 4. In a one dimensional CA, there can be a total of eight distinct neighborhood configurations with a total number of 256 distinct mappings from all the neighborhoods to the next state. Each mapping is defined as a rule of the CA. A pictorial representation of Rule 90 [7] is illustrated in Figure 5, where the top row represents the eight possible states for a three-cell neighborhood and the bottom row represents the next state for the cell of interest.

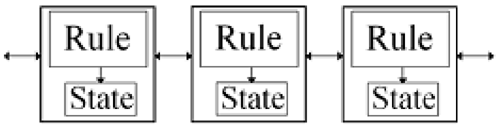


Figure 4: A one dimensional nearest-neighbor CA.

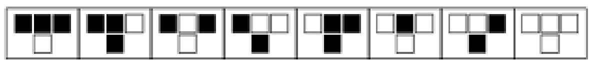


Figure 5: Representation of rule '90.'

As noted by Hortensius et al., if different rules are used in each cell (heterogeneous CA), higher quality random numbers can be generated from a CA than if a uniform (homogeneous) rule is applied to all cells [3]. Combinations of rules 90 and 150 (see Figure 6) were found to produce good random numbers with maximal length suitable for random number generators

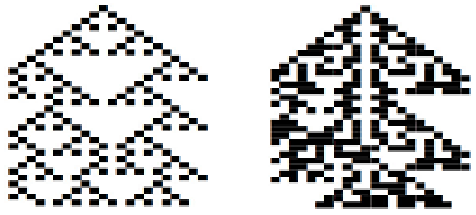


Figure 6: Space-time evolution patterns for a simple rule 90 and hybrid 90/150 CA

The other component of the proposed stream cipher is the key-bit mixing mechanism. The key-bit mechanism as shown in Figure 7, increases the complexity of the stream cipher making it harder to crypto-analyze.

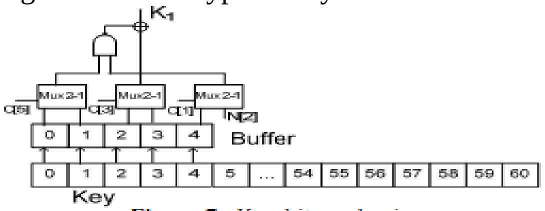


Figure 7: Key-bit mechanism

The output bit K1 is XORed with the CA bit 4 and provides the feedback to the NFSR. The key is generated using the same process as the A2U2 stream cipher. As illustrated in the Figure 7, at every round, five bits of the key are loaded into a buffer. Finally, these bits are combined with three bits of the counter and one bit of the NFSR as shown below

$$K_1 = \left(\overline{MUX_{C[5]}(B[0], B[1]) \cdot MUX_{C[1]}(B[4], N[1])}\right) \oplus MUX_{C[3]}(B[2], B[3])$$

where MUXZ(X1, X2) is defined as a multiplexer that uses two bits X1 and X2 as input and a selector bit Z. The last component of the proposed stream cipher is the filter function.

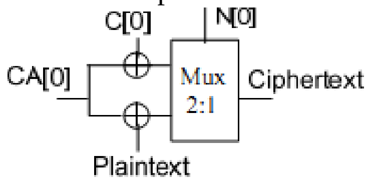


Figure 8: Filter function

The filter function ensures that only the part of the input string coming out from the CA-based register will be XORed with the plaintext based on a selector string provided by the NFSR. The filter function shown in Figure 8 is represented by the following equation,

$$F[x] = MUX_{N[0]}((CA[0] \oplus C[0])\ (CA[0] \oplus P))$$

where P represents the plaintext.

III. RESEARCH METHOD

It is This section describes the various simulation methods used to determine the quality of the random numbers that were produced by the A2U2 stream cipher and the proposed stream cipher using CA. Separate C programs were written to simulate the A2U2 stream cipher and the proposed stream cipher with CA. Logisim was used as a secondary simulator to confirm the results of the C program. The usefulness of Logisim lies in its graphical representation of the simulation, which makes some flaws more obvious and easier to fix than in C code. C code, however is itself sometimes easier to debug and is much faster. The A2U2 stream cipher and the proposed stream cipher both were implemented on an FPGA. The concept of entropy was also used to evaluate the quality of the random numbers. A separate C program was written to calculate the percent relative error based on the entropy for both the A2U2 stream cipher and proposed stream cipher. The entropy En for a subsequence of length *h* can be expressed in bits by,

$$E_h = -\sum_{j=1}^{M} p_j log_2 p_j$$

whereM=2^h , the number of possible subsequences of binary numbers of length *h* and *P*j is the measured probability that sequence *J* occurs. A higher entropy value implies a better PRNG. In our case, *h* was set to 8 and sequence lengths of 2^16 were generated. In theory, this allows a quick estimate of the quality of random numbers generated.

Finally, the results were verified using a Tektronix TLA 7012 logic analyzer (see Figure 9). The logic analyzer is a very useful device for visualizing the system behavior and to detect the hardware flaws that are commonly hard to determine from simulations.

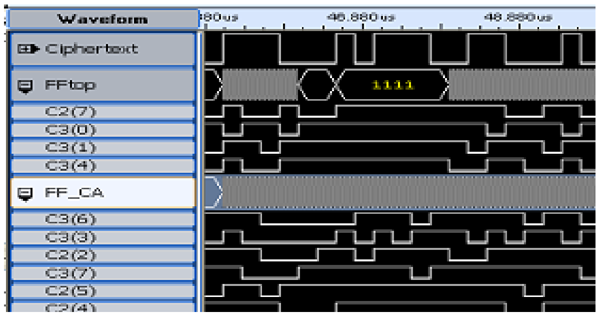


Figure 9: Testing the stream cipher with a logic analyzer

IV. RESULTS

The output of both the A2U2 stream cipher and the proposed stream cipher with CA were analyzed using the entropy tests and the DIEHARD tests. Both stream ciphers were initialized using six random seeds. The names of the DIEHARD tests are listed in Table I (for more details on each test, see [5]). All the tests return p-values. The p-values are numbers ranging from 0 to 1. Any test that returns a p-value equal to 1 or 0 is considered to have failed the tests. According to DIEHARD an “ideal” random number generator should have a uniform distribution of p-values.

Table I: List of DIEHARD Tests

1	Birthday Spacings
2	Overlapping 5-Permuatation
3	Binary Rank 31*31
4	Binary Rank 32*32
5	Binary Rank 6*8
6	Bitstream
7	OPSO
8	OQSO
9	DNA
10	Count-the-1's
11	Count-the-1's 2
12	Parking Lot
13	Minimum Distance
14	3D Spheres
15	Squeeze
16	Overlapping Sums
17	Runs
18	Craps

By using the equation shown below, the percent relative errors for the A2U2 and proposed stream cipher were calculated. In this case, a smaller relative errorε_t indicates a better quality random number sequence.

$$\varepsilon_t = \left|\frac{\mathit{True\ Value} - \mathit{Approximation}}{\mathit{True\ Value}}\right| \times 100\%$$

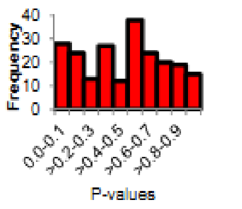
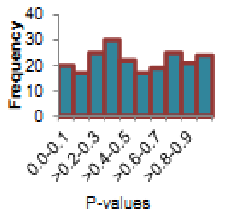
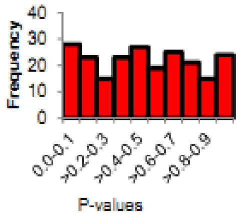
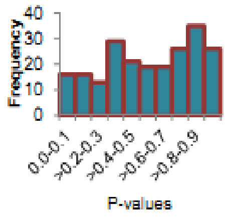
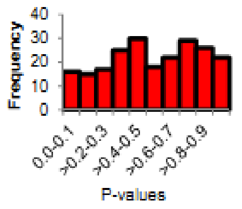
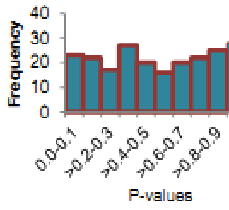
Table 2 shows the percent relative error for both stream ciphers using the equation shown above. Three random seeds were used to initialize the CA and the LFSR/NFSR to perform the entropy test.

Table 2: The percent relative errors

A2U2 Stream Cipher	The Proposed Stream Cipher
0.12071 %	0.12428 %
0.12723 %	0.13964 %
0.14554 %	0.16722 %

As the entropy test is inconclusive, DIEHARD tests were used to confirm the results. All the DIEHARD tests were run using 10 million bits to test the quality of the random number sequence. Table 3 compares the p-value distributions for the A2U2 stream cipher and the proposed stream cipher with CA using the DIEHARD test suite.

Table 3: The *p*-value distributions from the DIEHARD tests

A2U2 Stream Cipher	Stream cipher with CA
	
	
	

The A2U2 stream cipher and the proposed stream cipher both passed all DIEHARD tests. However, the proposed stream cipher returned p-values that are more equally distributed across the range [0,1). This indicates the higher quality of random numbers produced by the proposed stream cipher. Further analysis is required to establish the strength of the proposed stream cipher.

V. CONCLUSIONS AND FUTURE WORK

This paper has evaluated the performance of a CA-based stream cipher design suitable for implementation on FPGAs. The design was synthesized and tested using a Xilinx Spartan 3E FPGA. The DIEHARD suite of statistical tests was used to evaluate the quality of the random number produced. Results were compared with the A2U2 design. Future work includes cryptanalysis to study the strength of the proposed stream cipher, optimization of the CA design to improve the quality of randomness, and exploration of some of the hardware implementation techniques for preventing differential power analysis (DPA) attacks. For example, one of the effective techniques to prevent DPA attack might be the development of the secure system using complementary pass transistor logic (CPL). The proposed PRNG in this work might be useful in other applications requiring high quality random number sequences, such as particle swarm optimization engines implemented in reconfigurable hardware [8].

REFERENCES

[1] D. Mathieu, D. Ranasinghe, and T. Larsen, “A2U2: A Stream Cipher for Printed Electronics RFID Tags,” IEEEInternational Conference on RFID, 2011.

[2] C. de Canniere, O. Dunkelman, and M. Knezevic , “KATAN & KTANTAN – A Family of Small and Efficient Hardware Oriented Block Ciphers,” in Proc. 11thInt. Workshop on Cryptographic Hardware and Embedded Systems-CHES 2009, Switzerland, LNCS, vol. 5747, pp. 272-288.

[3] P. D. Hortensius et al., “Cellular automata-based pseudorandom number generators for built-in self-test,” IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems, vol. 8, no. 8, pp. 842-859, Aug. 1989.

[4] J. C. Cerda, C. D. Martinez, and J. M. Comer, D. H. K. Hoe, “An Efficient FPGA Random Number Generator using LFSRs and

Cellular Automata," IEEE 44th Southeastern Symposium on System Theory, March 2012.
[5]G. Marsaglia, DIEHARD, http://stat.fsu.edu/~geo/ diehard.html, 1996.
[6] N. T. Courtois and W. Meier, "Algebraic Attacks on Stream Ciphers with Linear Feedback," in Proc. Workshop Theory and Application of Cryptographic Techniques, Advances in Cryptology – EUROCRYPT '03, Warsaw, Poland, May 4-8, 2003, LNCS, vol. 2656, pp. 345-359, 2003.
[7] S. Wolfram, A New Kind of Science, Wolfram Media, Inc., IL, USA, 2002.
[8] R. W. Duren, R. J. Marks II, P. D. Reynolds, and M. L. Trumbo, "Real-Time Neural Network Inversion on the SRC-6e Reconfigurable Computer," IEEE Trans. on Neural Networks, vol. 18, no. 3, May 2007, pp. 889-901.