http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 13 (12): 1936-1944, 2014
ISSN 1812-5638 / DOI: 10.3923/itj.2014.1936.1944
© 2014 Asian Network for Scientific Information

Modeling Combo PR Generator for Stego Storage Self Test (SSST)

Sundararaman Rajagopalan, Yamini Ravishankar, Har Narayan Upadhyay,
J.B.B. Rayappan and Rengarajan Amirtharajan
School of Electrical and Electronics Engineering, SASTRA University, India

Abstract: Steganography, a protected envelope for information systems is reaching new horizons at software as well as hardware level. Due to the number of benefits that result in using reconfigurable hardware like FPGA for stego system development, some attention is needed in performing the stego memory testing. While Self test methodologies adopted for memories require attention due to the extensive memory requirements, testing the secret carrier stego memory modules occupies the center stage due to its higher importance of data protection. Normally block RAMs inside FPGA can store the cover and stego images. With the hardware pseudorandom pattern generators, the memory testing can be done effectively. In this regard, the present work focuses on the implementation and analysis of various combined multiple LFSR based pseudorandom sequence generation schemes for Stego Memory self testing on Cyclone II EP2C20F484C7 FPGA. Analysis of the different schemes for their suitability to stego memory arena is an important objective of this work and also sequence distribution analysis has been carried out to verify the distribution of pseudorandom sequences for N clock cycles. The synthesis reports for all the four cases undertaken in this work have also been reported.

Key words: Information hiding, hardware steganography, FPGA, LFSR

INTRODUCTION

Information sharing has crossed restriction boundaries in terms of payload, data rate, media and time. At this moment the wake up call for all of us is the proper delivery of our own message in a secured manner. Security approaches like cryptography, steganography (Hmood *et al*., 2010a, b; Rabah, 2004; Zaidan *et al*., 2010) and watermarking have proved their worth but that does not mean that the fool proof has been reached. Brute force attack, steganalysis and intruders' interruption has been a concern these days. Many methods have been proposed in the past on software based steganography (Al-Azawi and Fadhil, 2010; Al-Frajat *et al*., 2010; Janakiraman *et al*., 2012a-c; Qi *et al*., 2010; Cheddad *et al*., 2010; Amirtharajan and Balaguru, 2009; Zanganeh and Ibrahim, 2011; Thanikaiselvan *et al*., 2011; Thenmozhi *et al*., 2012). Hardware steganography (Amirtharajan and Rayappan, 2012a-d) is an emerging field and involves the hardware implementation of steganography in reconfigurable (Rajagopalan *et al*., 2012b) and embedded platforms and aids in better security. Farouk and Saeb (2004) have implemented a steganographic micro-architecture with FPGAs for the Prisoner's problem which is the basic example for steganography. A hardware architecture of the ConText steganographic technique implemented in Cyclone II FPGA of Altera family is given by Gomez-Hernandez *et al*. (2008). This implementation hides the information in the noisy regions and regions of abrupt gray level changes thus making it very difficult to detect the presence of data. Quadblock based Hardware stego architecture is reported by Rajagopalan *et al*. (2012a) and Rajagopalan and Upadhyay (2011) have implemented an image steganographic system on FPGA wherein the addresses for embedding the secret data are generated with the help of a Linear Feedback Shift Registers (LFSR) which introduce randomness of the generated address. Another hardware implementation of image steganography has been implemented by Amirtharajan *et al*. (2012) where the data is hidden in square sized blocks of the cover image and the traversal order in each block is defined by a quad block.

While security increases with hardware stego modules, memory testing becomes an important job as memory in ICs carry the hidden information. One of the important modules for testing a complex Integrated Circuit over the years is the Test Pattern Generator. As the testing is done in order to find out the faults in the design, there exists a demand for test pattern generators which will aid in finding the faults present in the system proposed. Before the invent of Built In Self Test (BIST), the test pattern generators were present outside the system and an Automatic Test Equipment (ATE) was needed to evaluate the equivalence of actual test pattern coming out of the sequence generator and the one

Corresponding Author: Sundararaman Rajagopalan, School of Electrical and Electronics Engineering, SASTRA University, India

desired. But BIST facilitates the presence of both test pattern generation schema and response analyser on the chip itself. Recent advances in audio, video, image processing and security approaches like steganography demand the extensive use of memory as the amount of information being processed is enormous. In this context, memory testing becomes an inevitable part of system testing. Whether it is a processor or FPGA or controller playing a role, the data processing operations especially memory read and memory write operations consume significant amount of time and power. Earlier works on Memory self Test approaches have deployed Pseudo random sequence generators for fault injection and fault detection by various methods. Hortensius *et al.* (1990), proposed a cellular automata based method which analysed the feasibility of various CA rules based sequence generator. Another work by Serra *et al.* (1990) analyses the properties of cellular automata.

A number of works have been proposed in the domain of LFSR as sequence generator which has been a work horse in the field of test pattern generation for IC testing. An LFSR is a Linear Feedback Shift Register with its input bit driven by a linear operation of few of its bits and is triggered by a clock input. The linear operation is mostly XOR operation. The maximal capacity of the LFSR is exploited only when the LFSR has a particular polynomial sequence which is implemented as the LFSR. All the other combinations of polynomials for a given degree might miss out few values. The maximal polynomial is capable of producing (2^n-1) values, provided the seed value is not all zeros for XOR operation where 'n' is the number of shift registers in the LFSR. The LFSRs have been widely exploited for their characteristic of producing Pseudo-Random numbers for various applications like in cryptography, memory testing, etc., Wang (2002) proposes a BIST with LFSR TPG, to reduce the switching activity employing ANDed outputs of LFSR sequence bits. An improved Test Compatability Class to group the LFSRs for efficient TPGs proposed by Nicolici *et al.* (2000), study out with Tabu search methodology to group the test patterns effectively. BIST for RAM in Seconds, proposed by Cheng *et al.* (2000) uses a TPG with March test for embedded memory testing. Pseudo exhaustive Pattern generation (Haridas and Devi, 2011) for BIST uses the efficient seed computation in order to avoid invalid test patterns. Dufaza and Zorian (1997) proposed a deterministic LFSR generation which is suitable for delay fault testing. This method however used lesser consumptions with n-LFSR compared to the techniques proposed by Starke (1984) and Craig and Kime (1985). The importance of random TPGs for BIST has been emphasized by Fagot *et al.* (1999), Gupta and Pradhan (1996), Krasniewski and Pilarski (1989) and Cheng and Lin (1995). The study by Jutman *et al.* (2008) discussesd the specific polynomial and seed generation for particular BIST applications. In this work, four different LFSR generation approaches have been considered where the effect of combining multiple LFSRs for stego memory self test has been discussed. Also sequence distribution analysis has been considered in this paper for examining the correlation between the overall pattern generated and which may be helpful in targeting the complete memory area for easy detection of memory faults thereby using an LFSR pattern which covers all the memory sections in a distributed fashion.

The rest of the paper is organized as follows. The next section discusses about combined LFSRs, followed by the sequence distribution analysis and discussion about synthesis report and the final section concludes the study.

COMBINED LFSRs AS PR-GENERATORS ON FPGA

We have worked on the very same concept of LFSR and tried to figure out the extent/nature of randomness induced in the output sequence when two or three LFSRs are generated by different clock frequencies and are selected in a sequence triggered by a different clock.

Basically, for the first two cases wherein the same 8-bit LFSR (8, 6, 5, 4) (i.e.,) $P_8(x) = 1+x^8+x^6+x^5+x^4$ is used, three LFSR sequences are generated with different seed values. In the last two cases, where different LFSRs are considered, only two sequences are considered since we have only two variations of the maximal LFSR for degree 8 polynomial. Functional simulation results have been considered for all the four cases. The input clock has a frequency of 50 MHZ and reset is given for one clock pulse. The output sequence is selected from each LFSR-one at a time at the rising edges of opclk. So for the first rising edge, value is selected from LFSR lp, for the next rising edge lq value is taken and so forth.

Case 1: Same frequency same LFSR: In this case, the same clock of 50 MHZ is used for generating the different LFSR sequences-lp, lq, lr with corresponding seed values of F0, 0F and 99. The three clocks-p1, q1, r1 are XORed to produce the output clock opclk which is also of the same frequency of 50 MHZ. The rising edge of the opclk is used for triggering the selection of the output values. Figure 1 shows the simulation results of same frequency same LFSR Combination.

Analysis: Upon analysis of the output sequence, it is observed that the values from each LFSR are selected for

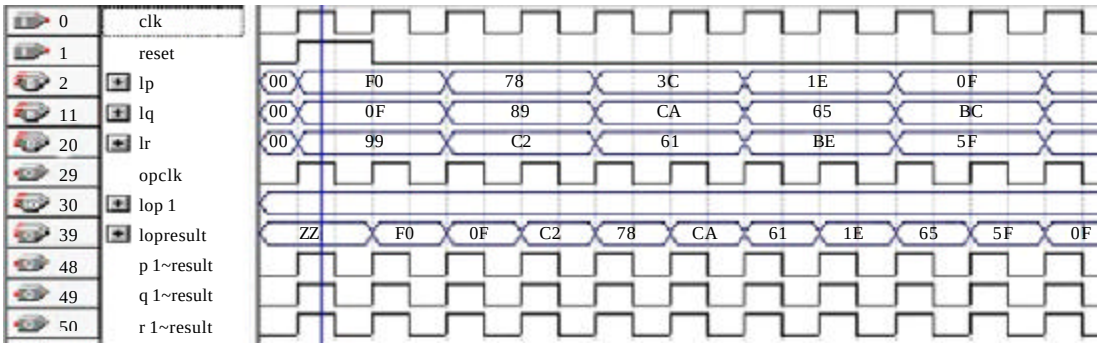

Fig. 1: Simulation results of same frequency same LFSR combination

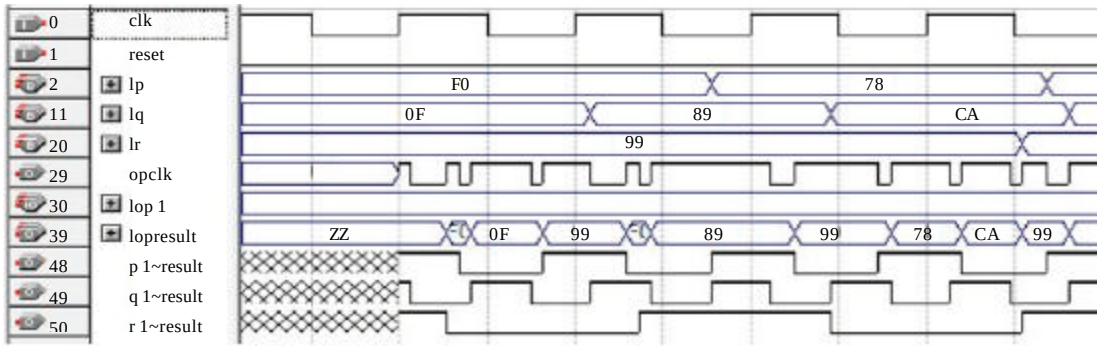

Fig. 2: Simulation results of different frequency same LFSR combination

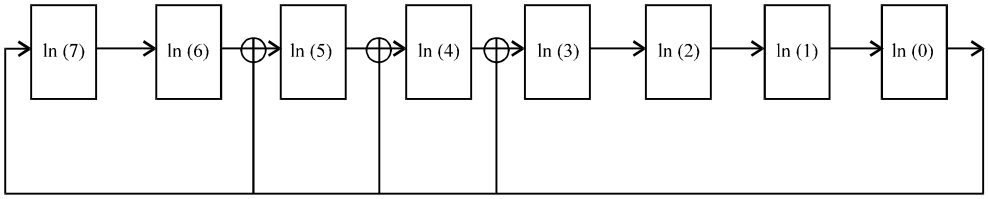

Fig 3 : Structure of (8, 4, 3, 2) 8-bit LFSR

each clock pulse and one value out of three is missed. This can possibly be overcome by taking the clock triggering the output selection to be of a slightly higher frequency.

Case 2: Different frequency same LFSR: In this case, three different clocks p,q,r are generated by using the PLL megafunction and the corresponding frequencies are 52.38, 73.33 and 22.9 MHZ. The corresponding LFSRs-lp, lq, lr are triggered at the rising edge of each clock with the seed values of F0, 0F and 99 available on reset. The output triggering clock is opclk and is generated by the XOR operation of the three clocks p,q and r. Figure 2 displays the Simulation results of Different frequency Same LFSR Combination.

Analysis: The analysis of the functional simulation shows that few of the values get repeated at the output sequence because the clock opclk is very random and changes very frequently compared to the input clocks.

Case 3 : Different frequency different LFSRs: In this case, two different frequencies of 52.38 and 73.33 MHZ are generated using the PLL megafunction in Quartus. Here, two different 8-bit LFSRs-lp (8, 6, 5, 4) and lq (8, 4, 3, 2) are used which are triggered at the rising edge of each clock with corresponding seed values as F0 and 0F and then the clocks are XORed to give the output clock opclk. Opclk is used to select the output sequence at its rising edges. Figure 3 shows the structure of (8, 4, 3, 2) and Fig. 4

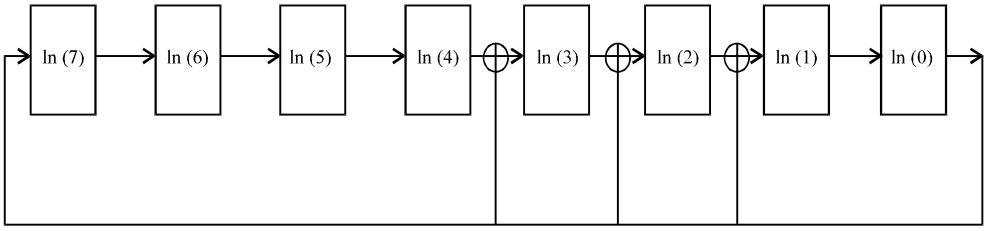

Fig. 4: Structure of (8, 6, 5, 4) 8-bit LFSR

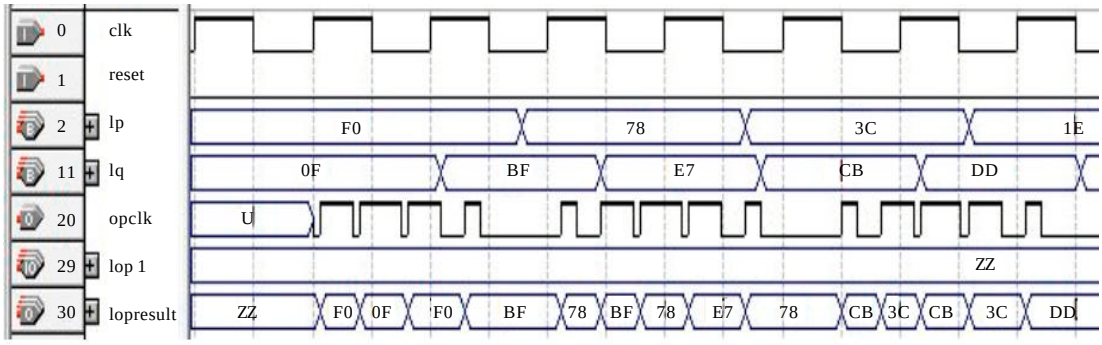

Fig. 5: Simulation results of different frequency different LFSR combination

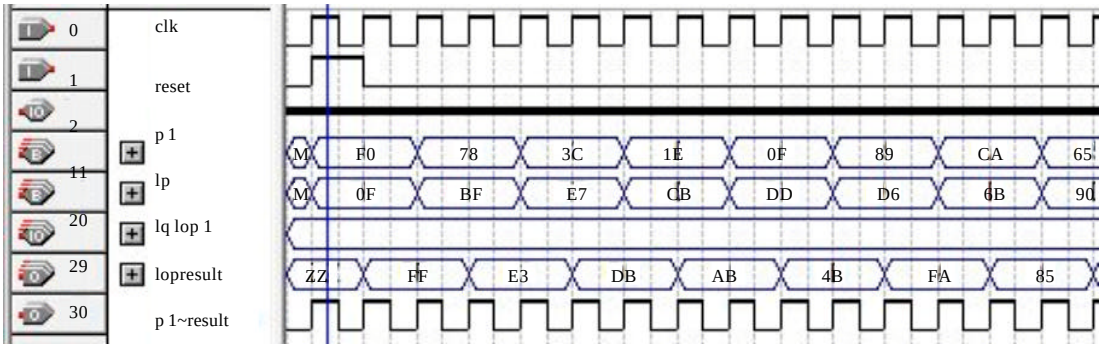

Fig. 6: Simulation results of same frequency different LFSR combination

shows the structure of (8, 6, 5, 4) 8-bit LFSRs. Figure 5 shows the simulation results of different frequency different LFSRs.

Analysis: From the functional simulation we observe that the opclk output changes frequently compared to the input clock signal and hence few values get repeated in the output lop. The opclk sequence appears to be periodic.

Case 4: Same frequency different LFSRs: For the last case considered, the same input clock frequency of 50 MHZ is used for LFSR triggering. Here, if we take the XOR of the two clock signals of same frequency and phase shift, we will get only a dc signal. Hence, for this case the output is sampled at 50 MHZ and the output values are taken as the XORed values of the corresponding LFSR (lp and lq) content at that instant. Figure 6 shows the simulation results for this case.

Analysis: An analysis of the simulation result shows that we get completely random values at the output. But the problem is that if the two values at lp and lq at two different clocks have the same combination of lsb and msb, then the value would be repeated. Also, in this way we will always miss out some values. If we take the values such that we select the one value from an LFSR for each rising edge of the clock, then we would be able to cover all the values without any repetitions in around 510 clock cycles.

COMPARISON OF THE CASES

Case 1 and case 2: In case 1, since the same frequency is used, after the end of 255×2 i.e., 510 clock cycles, the same set of values will be present in the LFSRs for the next cycle and this will get repeated. So the output sequence will get repeated when this happens. While in case 2, each LFSR gets values according to the frequency and the values would get repeated only at the clock cycle which would be the lcm of the three total clock cycle values. So we will be having a unique set of numbers.

Case 3 and case 4: The comparison is the same as above. For case 3, the values will get repeated only at the lcm value for clock cycle and in case 4 it will generate the same sequence from the 511th clock cycle since each LFSR consumes two clock cycles for producing a change in the value. Figure 7 displays the Generation of opclk for two input clocks (a) for case 3 and 4 (b) for case 1 and 2.

SEQUENCE DISTRIBUTION ANALYSIS

The purpose of analysing the sequence distribution is to study the various bit patterns which resulted from the LFSR output. During the address generation process of embedded BIST, even though we use primitive polynomial to cover all the cases, the LFSR output has to be checked for the coverage of addresses from various sections of the internal memories in quicker time (i.e.,) in lesser number of clock cycles. As the seed value of the LFSR decides the following sequences, the decision to use a particular seed becomes an important issue in this analysis. The first approach in this analysis splits the 256 patterns into four quadrants with each quadrant covering 64 patterns. Figure 8a-d show the plots of No. of clock cycles Vs No. of values range. The seed value is fixed initially. The first quadrant considers 64 clock cycles and during this time how many values fall in 0-63 band, 64-127 band and so on. Four seed values for 8 bit LFSR 00000001, 11111111, 00001111, 11110000 have been considered and the approximately equal distribution of the values falls in the non-extreme seeds which are 00001111 and 11110000.

In another approach, eight sub sections have been considered where values in 32 clock cycle band have been considered. Figure 9a-d show the sequence distribution in eight Quadrants with $P_8(x) = 1+x^8+x^6+x^5+x^4$.

The suitability of mid values as seed is more for proper distribution of bit combinations.

DATA INTEGRITY

This methodology of hardware steganography BIST has its main advantage as Data integrity. The data integrity of any stego-system can be ensured. For example, considering a 1-bit LSB substitution for embedding the secret data, values maybe embedded in the random locations generated by the combo LFSR schemes proposed in this study and if the proposed or any other hiding scheme is followed whenever data is embedded, the first 4 MSB bits of the pixel located by the LFSR pattern can be XORed and the resulting bit may be stored in the bit adjacent to LSB. Like this, the integrity mechanism can be strengthened. This helps in proper decoding and unless the proper seed value and frequency of the used clocks are known, the secret message cannot be deciphered by intruders. The random addressing makes it very difficult for any intruder to decipher the secret.

SYNTHESIS REPORT

These different cases were checked using Cyclone II EP2C20F484C7 FPGA. Table 1 shows the hardware consumption for all the four cases. Among the four cases, different frequency same LFSR case consumes a highest of 46 logic elements, next being the same frequency same LFSR which consumes 40 LEs. The third model different frequency different LFSR takes 27 LEs and the lowest number of LEs are consumed by same frequency different

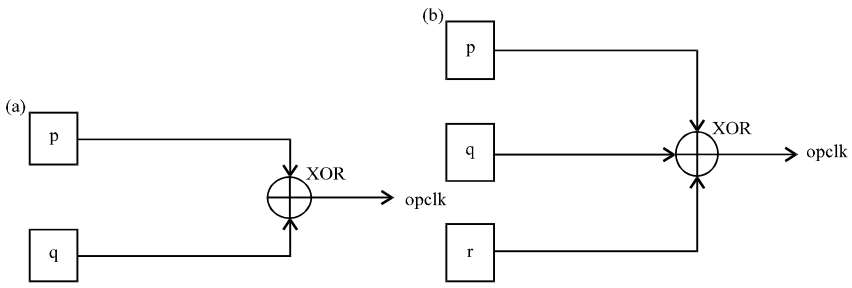

Fig. 7(a-b): Generation of opclk for two input clocks (a) For case 3 and 4 (b) For case 1 and 2

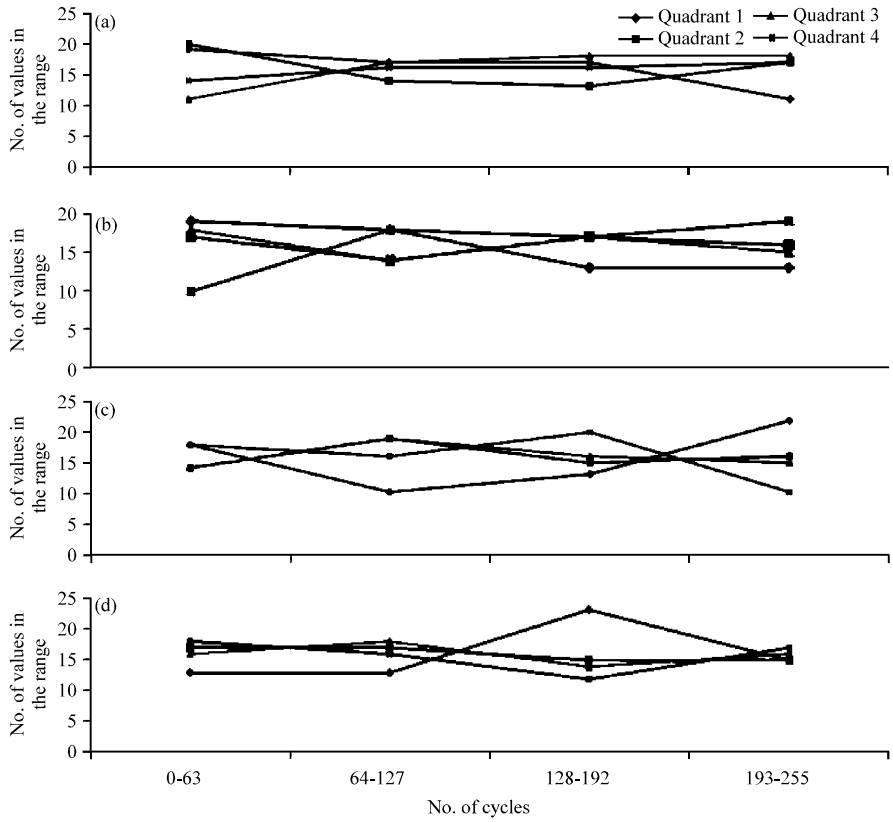

Fig. 8(a-d): Sequence distribution in four quadrants with P_8 (x) = $1+x^8+x^6+x^5+x^4$ (a) Seed value: "11110000", (b) Seed value: "00001111", (c) Seed value: "00000001" and (d) Seed value: "11111111"

Table 1: Comparison of the compilation reports for all the cases

Case	Same frequency same LFSR	Different frequency same LFSR	Different frequency different LFSR	Same frequency different LFSR
Total combinational functions	40	46	27	21
Dedicated logic registers	46	51	37	33
Total registers	46	51	37	33
Total pins	14	14	14	11
Total PLLs	0	1	1	0

LFSR model which is 21. It is pretty clear from Table 1 that cases 3 and 4 consume relatively lesser amount of logic elements when compared to case 1 and 2 as case 3 and 4 use a combination of two LFSRs and where 1 and 2 use three LFSR combinations.

RESULTS AND CONCLUSION

This study discussed about the combined LFSRs and their suitability for stego-memory BIST. Among the four cases, same frequency different LFSR consumes a much lesser 21 Logic Elements with a total registers of 33 as shown in Table 1. The cases considered in this study used different frequency LFSRs which can be generated with the help of DLLs and PLLs. The sequence distribution will play a vital role in choosing a proper LFSR combination for memory fault detection. When we analyse the four quadrant sequence distribution in Fig. 8a-d, the LFSR generation with seed values "00000001" and "11111111" both extremes show some variations in the uniformity and equal distribution of the sequences. In the LFSR generation with seed value "11111111", in the first 64 clock cycles applied to LFSR, around 23 pseudorandom patterns are falling in the range between 128-192 (i.e.,) between "10000000" and "11000000" which is the highest of all quadrants. The other quadrants have almost uniform distribution , as second 64 clock cycles yielded (17, 17, 15, 15) in the respective ranges (0-63, 64-127, 128-192, 193-255), third 64 clock cycles with a distribution of (16, 18, 14, 16) and

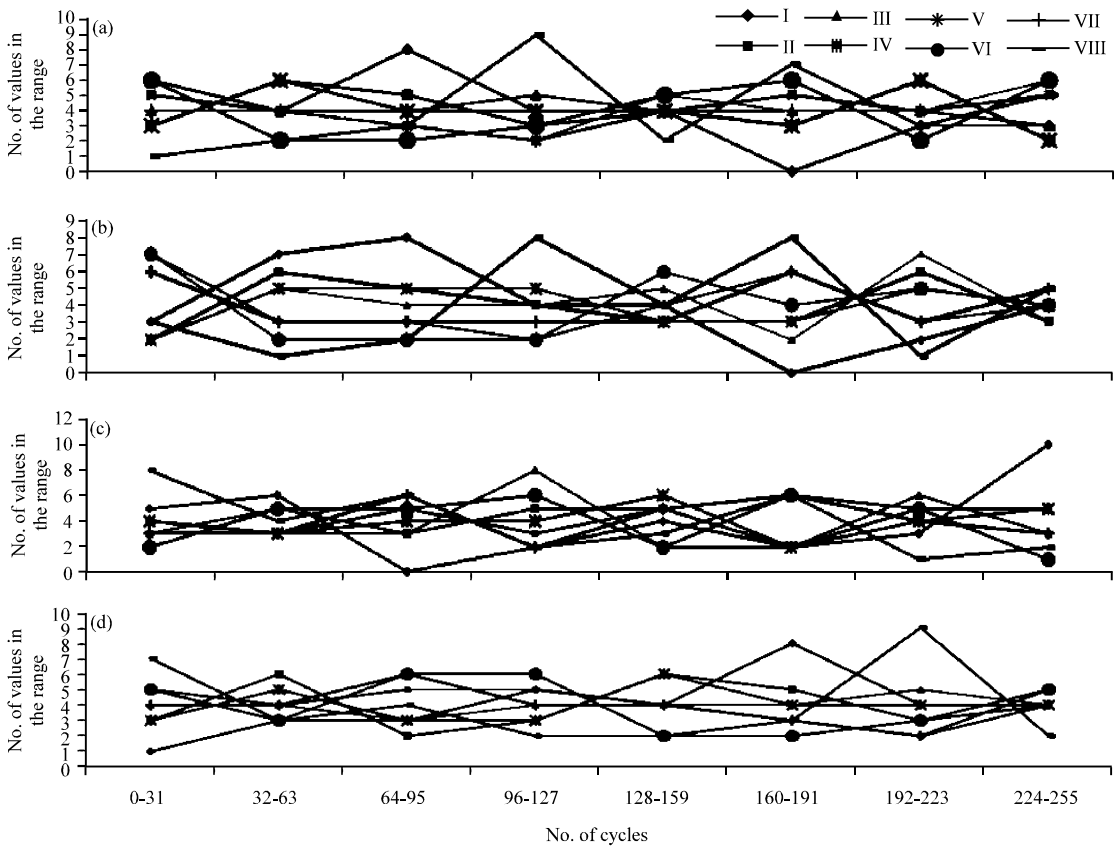

Fig. 9(a-d): Sequence distribution in eight quadrants with $P_8(x) = 1+x^8+x^6+x^5+x^4$ (a) Seed value: "11110000", (b) Seed value: "01111111", (c) Seed value: "00000001" and (d) Seed value: "11111111"

the fourth producing (18, 16, 12, 17) distribution. In the Eight quadrant sequence distribution analysis from Fig. 9a, where initial seed is "11110000" , between 160 and 191 clock cycles applied to LFSR, no pseudorandom pattern falls in the value between "00000000" and "00011111". The 8-bit LFSR started with the initial seed of "11111111", generates atleast one pattern to atmost nine patterns from all the eight ranges (i.e.,) 0-31, 32-63, 64-95, 96-127, 128-159, 160-191, 192-223 and 224-255 in all the eight quadrants of clock cycles. Therefore the LFSR $P_8(x) = 1+x^8+x^6+x^5+x^4$ with initial seed of "11111111" due to its near uniform distribution of sequences, takes a lead for stego memory address TPGs in fault detection. Also phase shift based LFSRs can be a part of TPGs where future study can be focussed. This method of integrating memory BIST model in hardware steganography gives the added advantage of ensuring the data integrity of the transmitted stego-image. The only consideration about this method is that care should be taken to keep the PSNR value high without a lot of deviation from the original values.

ACKNOWLEDGMENT

The authors wish to express their sincere thanks to DRDO, New Delhi for their financial support (ERIP/ER/1003836/M/01/1230). They also wish to acknowledge SASTRA University, Thanjavur for extending infrastructural support to carry out the study.

REFERENCES

Al-Azawi, A.F. and M.A. Fadhil, 2010. Arabic text steganography using kashida extensions with huffman code. J. Applied Sci., 10: 436-439.

Al-Frajat, A.K., H.A. Jalab, Z.M. Kasirun, A.A. Zaidan and B.B. Zaidan, 2010. Hiding data in video file: An overview. J. Applied Sci., 10: 1644-1649.

Amirtharajan, R. and J.B.B. Rayappan, 2012a. An intelligent chaotic embedding approach to enhance stego-image quality. Inform. Sci., 193: 115-124.

Amirtharajan, R. and J.B.B. Rayappan, 2012b. Brownian motion of binary and gray-binary and gray bits in image for stego. J. Applied Sci., 12: 428-439.

Amirtharajan, R. and J.B.B. Rayappan, 2012c. Inverted pattern in inverted time domain for icon steganography. Inform. Technol. J., 11: 587-595.

Amirtharajan, R. and J.B.B. Rayappan, 2012d. Pixel authorized by pixel to trace with SFC on image to sabotage data mugger: A comparative study on PI stego. Res. J. Inform. Technol., 4:: 124-139.

Amirtharajan, R. and R.J.B. Balaguru, 2009. Tri-layer stego for enhanced security-a keyless random approach. Proceedings of the IEEE International Conference on Internet Multimedia Services Architecture and Applications, December 9-11, 2009, Bangalore, India, pp: 1-6.

Amirtharajan, R., J. Qin and J.B.B. Rayappan, 2012. Random image steganography and steganalysis: Present status and future directions. Inform. Technol. J., 11: 566-576.

Cheddad, A., J. Condell, K. Curran and P.M. Kevitt, 2010. Digital image steganography: Survey and analysis of current methods. Signal Process., 90: 727-752.

Cheng, C., C.T. Huang, J.R. Huang, C.W. Wu, C.J. Wey and M.C. Tsai, 2000. BRAINS: A BIST compiler for embedded memories. Proceedings of the IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems, October 25-27, 2000, Yamanashi, Japan, pp: 299-307.

Cheng, K.T. and C.J. Lin, 1995. Timing-driven test point insertion for full-scan and partial-scan BIST. Proceedings of the IEEE International Test Conference, October 21-25, 1995, Washington, DC, USA., pp: 506-514.

Craig, G.L. and C.R. Kime, 1985. Pseudo-exhaustive adjacency testing: A BIST approach for stuck-open faults. Proceedings of the International Test Conference, November 19-21, 1985, Philadelphia, PA, USA., pp: 126-137.

Dufaza, C. and Y. Zorian, 1997. On the generation of pseudo-deterministic two-patterns test sequence with LFSRs. Proceedings of the European conference on Design and Test, March 17-20, 1997, Paris, France, pp: 69-76.

Fagot, C., O. Gascuel, P. Girard and C. Landrault, 1999. On calculating efficient LFSR seeds for built-in self test. Proceedings of the European Test Workshop, May 25-28, 1999, Constance, Germany, pp: 7-14.

Farouk, H.A. and M.M. Saeb, 2004. Design and implementation of a secret key steganographic micro-architecture employing FPGA. Proc. Des. Automat. Test Eur. Conf. Exhibit., 3: 212-217.

Gomez-Hernandez, E., C. Feregrino-Uribe and R. Cumplido, 2008. FPGA hardware architecture of the steganographic context technique. Proceedings of the 18th International Conference on Electronics, Communications and Computers, March 3-5, 2008, Puebla, pp: 123-128.

Gupta, S.K. and D.K. Pradhan, 1996. Utilization of on-line (concurrent) checkers during built-in self-test and vice versa. IEEE Trans. Comput., 45: 63-73.

Haridas, N. and M.N. Devi, 2011. Efficient linear feedback shift register design for pseudo exhaustive test generation in BIST. Proceedings of the 3rd International Conference on Electronics Computer Technology, April 8-10, 2011, Kanyakumari, Tamil Nadu, pp: 350-354.

Hmood, A.K., B.B. Zaidan, A.A. Zaidan and H.A. Jalab, 2010a. An overview on hiding information technique in images. J. Applied Sci., 10: 2094-2100.

Hmood, A.K., H.A. Jalab, Z.M. Kasirun, B.B. Zaidan and A.A. Zaidan, 2010b. On the Capacity and security of steganography approaches: An overview. J. Applied Sci., 10: 1825-1833.

Hortensius, P.D., R.D. McLeod and B.W. Podaima, 1990. Cellular automata circuits for built-in self-test. IBM J. Res. Dev., 34: 389-405.

Janakiraman, S., A.A. Mary, J. Chakravarthy, R. Amirtharajan, K. Thenmozhi and J.B.B. Rayappan, 2012a. Pixel bit manipulation for encoded hiding-An inherent stego. Proceedings of the International Conference on Computer Communication and Informatics, January 10-12, 2012, IEEE Explore, USA., pp: 1-6.

Janakiraman, S., R. Amirtharajan, K. Thenmozhi and J.B.B. Rayappan, 2012b. Firmware for data security: A review. Res. J. Inform. Technol., 4: 61-72.

Janakiraman, S., R. Amirtharajan, K. Thenmozhi and J.B.B. Rayappan, 2012c. Pixel forefinger for gray in color: A layer by layer stego. Inform. Technol. J., 11: 9-19.

Jutman, A., A. Tsertov and R. Ubar, 2008. Calculation of LFSR seed and polynomial pair for BIST applications. Proceedings of the IEEE 11th Workshop on Design and Diagnostics of Electronic Circuits and Systems, April 16-18, 2008, Bratislava, Slovakia, pp: 1-4.

Krasniewski, A. and S. Pilarski, 1989. Circular self-test path: A low-cost BIST technique for VLSI circuits. IEEE Trans Comput. Aided Des., 8: 46-55.

Nicolici, N., B.M. Al-Hashimi, A.D. Brown and C. Alan, 2000. BIST hardware synthesis for RTL data paths based on test compatibility classes. IEEE Trans. Comput. Aided Des. Integr. Circuits Syst., 19: 1375-1385.

Qi, K., D.F. Zhang and D. Xie, 2010. A high-capacity steganographic scheme for 3D point cloud models. Inform. Technol. J., 9: 412-421.

Rabah, K., 2004. Steganography-the art of hiding data. Inform. Technol. J., 3: 245-269.

Rajagopalan, S. and H.N. Upadhyay, 2011. Stego system on chip with LFSR based information hiding approach. Int. J. Comput. Appl., 18: 24-31.

Rajagopalan, S., R. Amirtharajan, H.N. Upadhyay and J.B.B. Rayappan, 2012a. Survey and analysis of hardware cryptographic and steganographic systems on FPGA. J. Applied Sci., 12: 201-210.

Rajagopalan, S., S. Janakiraman, H.N. Upadhyay and K. Thenmozhi, 2012b. Hide and Seek in Silicon- Performance Analysis of Quad Block Equisum Hardware Steganographic Systems. Procedia Eng., 30: 806-813.

Serra, M., T. Slater, J.C. Muzio and D.M. Miller, 1990. The analysis of one-dimensional linear cellular automata and their aliasing properties. IEEE Trans. Comput. Aided Des. Integr. Circuits Syst., 9: 767-778.

Starke, C.W., 1984. Built-in test for CMOS circuits. Proceedings of the International Test Conference on the Three Faces of Test: Design, Characterization, Production, October 16-18, 1984, Washington, USA., pp: 309-314.

Thanikaiselvan, V., S. Kumar, N. Neelima and R. Amirtharajan, 2011. Data battle on the digital field between horse cavalry and interlopers. J. Theor. Applied Inform. Technol., 29: 85-91.

Thenmozhi, K., P. Praveenkumar, R. Amirtharajan, V. Prithiviraj, R. Varadarajan and J.B.B. Rayappan, 2012. OFDM+CDMA+Stego = Secure Communication: A Review. Res. J. Inform. Technol., 4: 31-46.

Wang, S., 2002. Generation of low power dissipation and high fault coverage patterns for scan-based BIST. Proceedings of the International Test Conference, December 10, 2002, USA., pp: 834-843.

Zaidan, B.B., A.A. Zaidan, A.K. Al-Frajat and H.A. Jalab, 2010. On the differences between hiding information and cryptography techniques: An overview. J. Applied Sci., 10: 1650-1655.

Zanganeh, O. and S. Ibrahim, 2011. Adaptive image steganography based on optimal embedding and robust against chi-square attack. Inform. Technol. J., 10: 1285-1294.