LFSR Based Watermark and Address Generator for Digital Image Watermarking SRAM

S. Bhargav Kumar[#1], S.Jagadeesh[*2], Dr.M.Ashok [#3]

[#1] P.G. Student, M.Tech. (VLSI), Department of Electronics and Communication Engineering, Sri Sai Jyothi Engineering College, Gandipet, Hyderabad-75, (A. P.), India.

[#3]Professor, Department of Computer Science and Engineering, Sri Sai Jyothi Engineering College, Gandipet, Hyderabad-75, (A. P.), India.

[*2]Associate Professor & HOD, Department of Electronics and Communication Engineering, Sri Sai Jyothi Engineering College, Gandipet, Hyderabad-75, (A. P.), India.

Abstract— In digital image watermarking authentication methods and techniques, the original image will be watermarked with a text, image, audio or any signature. To overcome the uneven and enormous distribution of multimedia content in the internet, we propose a new method of watermarking technique using pseudo random number generator LFSR (Linear Feedback Shift Register). Two LFSR based implementations of pseudo random number generator are designed for embedding and generating an address in image watermarking applications are presented. The first LFSR is axa size bit array watermark generator, which will be embedded in the bxb size bit array original image. After embedding, the watermarked bit array image will be stored in the image watermarking SRAM (Static Random Access Memory) memory location using a cxc size bit array memory address location which will be generated by the second LFSR. In our paper we used LFSR, to generate a unique random number watermark array for the original image. The embedded watermarked image will contain the combinations of LFSR axa size array bit stream in such a way that, if this bit stream doesn't match with the LFSR used at the watermark extraction process, the watermark won't be revealed, providing a secured authentication to the original image. In watermark extraction process, we used the same LFSR of axa size bit array and extracted the watermark bit stream. Our proposed method of watermarking is simulated and synthesized using Active-HDL Version 7.2SE design tools and ModelSim XE III 6.4b. Our proposed method showed better results compared to other conventional methods of digital image watermarking.

Keywords— digital image watermarking, authentication, revealed, pseudo random number generator, LFSR, image watermarking SRAM, bit array, address generator, FPGA, bit stream, watermark extraction, Active-HDL, ModelSim.

I. Introduction

The need for data hiding is to covert communication using images; mostly secret message is hidden in a carrier image providing the automatic copyright protection. In data hiding, the two important sub-categories are steganography and watermarking. Steganography uses some basic principle of cryptography to hide information both text and audio in undetectable way. Watermarking is the technique of hiding a message about an image, audio clip, or other signature within the image or audio itself. Comparatively with steganography, watermarking technique is invisible and robust to additive noise. A digital watermark is a digital signal or pattern inserted into a digital image. In digital image watermarking, the message is related to the cover and is often used between the parties who know the presence of the hidden data. In digital image watermarking, the techniques evolved will hide cover image in original image and at the reverse process the cover image will be used. Due to uneven distribution of information in the internet, the hidden image may be retrieved by the third party. To overcome this, we are implementing a new technique of digital image watermarking.

In steganography, the text string will be hidden in an image or audio. In [1] the text will be hidden based on the LFSR (Linear Feedback Shift Registers). LFSR is a pseudo random number generator. An LFSR is a group of shift registers depending on the bit size, when clocked, advances the signal through the registers from one bit to the next most-significant bit. To generate different values of the LFSR some of the outputs are combined in exclusive-OR configuration which forms a feedback mechanism (a bit pattern generation technique). A linear feedback shift register can be formed by performing exclusive-OR on the outputs of two or more of the flip-flops together and feeding those outputs back into the input of one of the flip-flops. Linear feedback shift registers make extremely good pseudorandom pattern generators [2]. When the outputs of the flip-flops are loaded with an initial or seed value (anything except all 0s) and when the LFSR is clocked, it will generate a pseudorandom pattern of 1s and 0s. The pattern of the LFSR will depend on the number of shift registers used and the seed value. The main advantage of LFSR is the random pattern generation, where the pattern will be non-repetitive till the seed value again generated from the LFSR.

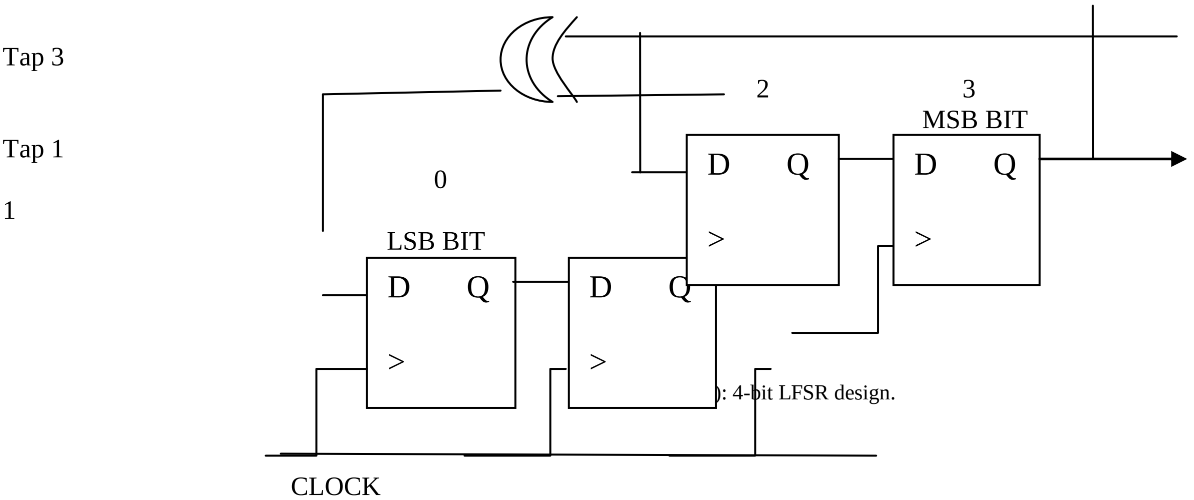
In our paper, we are implementing two LFSRs, one for digital image watermarking [3] of size axa and the other for address generator of size cxc. Here the first LFSR will generate the cover image of axa size bit array which will be watermarked with bxb size bit array. After watermarking, the watermarked bit array will be stored in the SRAM (Static Random Access Memory) address locations using another LFSR of cxc size bit array. LFSR's will generate random numbers, which will be hidden in the image. If these random numbers won't match at the extraction process, the original image won't be retrieved [7]. This is the major advantage of our method of watermarking.

In our paper, bit wise operation is made between the LFSR generated axa size bit array and the original image bxb size bit array. In our technique, we had generated the size of watermarking LFSR as 16x16 and the original image size as 256x256. And for address generator LFSR, we had generated the 8x8 size LFSR. Depending on the seed value of the LFSR we can implement watermarking technique for different sizes of images. LFSR based watermarking showed a secured authenticated copyright protection to the digital images.

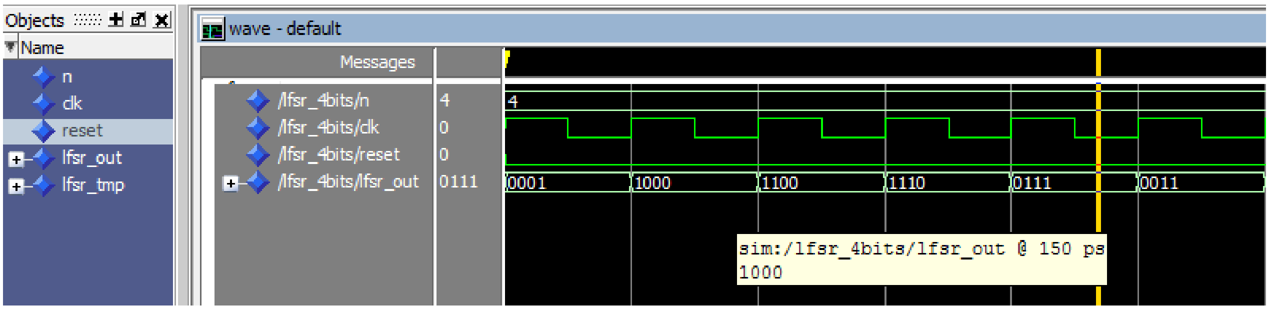
The rest of the paper as follows. Section 2 will discuss about the LFSR and its implementation. In section 3, the proposed LFSR's implementation for image watermarking and address generation will be explained in steps wise. In section 4, is about the complete results of our proposed method. Section 5 will conclude our paper.

II. Linear Feedback Shift Register

In this section, we discuss about the LFSR and its pseudo random generation technique [2] and [7]. Figure 1 (a) shows a 4-bit LFSR design. A 4-bit LFSR is a 4-bit length shift register with feedback to its input. The feedback is formed by XORing the outputs of selected stages of the shift register referred to as 'taps' and then inputting this to the least significant bit (stage 0). Each stage has a common clock.



): 4-bit LFSR design.



```
n
clk
reset
lfsr_out
lfsr_tmp

/lfsr_4bits/n          4     4
/lfsr_4bits/clk        0
/lfsr_4bits/reset      0
/lfsr_4bits/lfsr_out   0111  0001  1000  1100  1110  0111  0011

sim:/lfsr_4bits/lfsr_out @ 150 ps
1000
```

Figure 1(b): The sequence of 4-bit LFSR simulated in ModelSim XE III.

This has taps at stages 1 and 3 with XOR feedback. Assume that the example LFSR above is set to 0101 as seed value. After the initial state, the bit shifted into stage0 on each clock edge is the XOR of stage3 and stage1. The LFSR passes through 15 states 2n-1(16-1 = 15) and so is of maximal length. The sequence will then repeat from the initial state for as long as the LFSR is clocked. Figure 1(b) shows the sequence of 4-bit LFSR simulated in ModelSim XE III.

By setting the tap values to either 8-bit or 16-bit, we can generate 127 or 255 respectively the bit combinations. We are using this bit combination in our project.

In our paper, we are implementing 16 bit LFSR for watermarking and 8 bit LFSR for address generation.

A. Digital Image Watermarking Using LFSR

In this section, the method of digital image watermarking was implemented using a 16x16 size bit array watermarking [6] LFSR. In our method we used the seed value of the watermarking LFSR as "1011010010110100". Figure 3(a) shows the sequence of 16-bit digital image watermarking LFSR simulated in ModelSim XE III.

Steps wise implementation of LFSR watermark generator based digital image watermarking are as follows:

1. Original gray scale image 256x256 size is taken and is converted into 16 bit pattern representation of size (16x16) x (16x16).

III. Proposed Method Of Digital Image Watermarking

2. Now a 16x16 watermarking LFSR is designed with the seed value of "1011010010110100". This watermarking LFSR will generate 16x16 combinations of 16x16 size bit array.
3. So the watermarking LFSR bit pattern representation will be in (16x16) x (16x16) bit pattern.
4. Using AOI logic based watermark generator [4] see figure 2(a), the bit patterns of both original image and watermarking LFSR bits will be watermarked.
5. LFSR watermarked bits will result in (16x16) x (16x16) pattern representation.
6. The LFSR watermarked bit pattern will be reshaped in to 256x256 row and column representation.
7. This reshaped row and column matrix will be retrieved as LFSR watermarked image which contains a gray scale watermarked image.

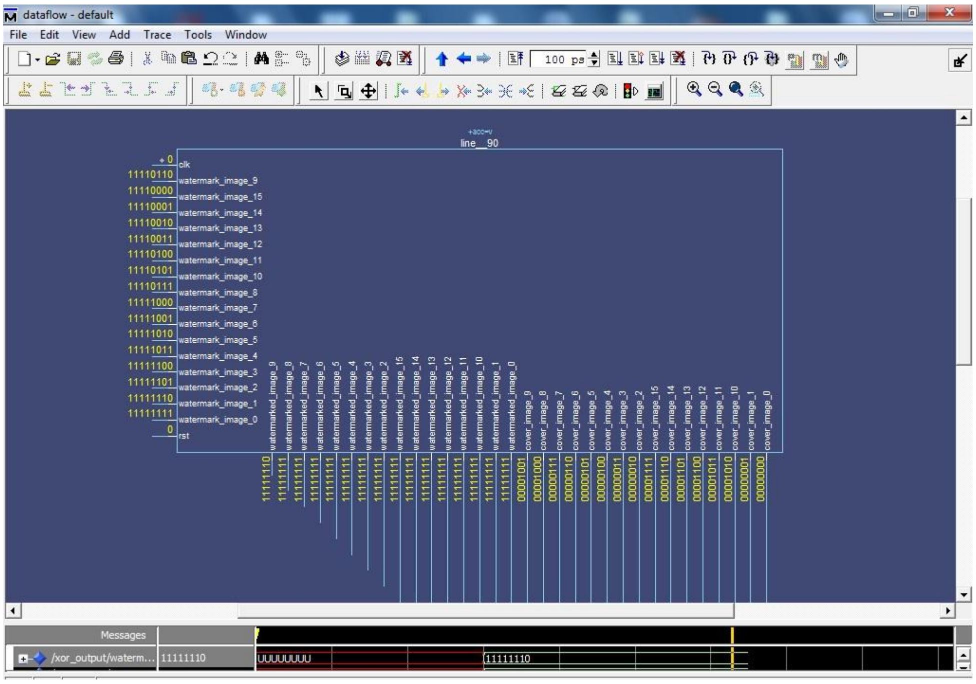

dataflow - default
File Edit View Add Trace Tools Window
100 ps
line__90
0 clk
11110110 watermark_image_9
11110000 watermark_image_15
11110001 watermark_image_14
11110010 watermark_image_13
11110011 watermark_image_12
11110100 watermark_image_11
11110101 watermark_image_10
11110111 watermark_image_8
11111000 watermark_image_7
11111001 watermark_image_6
11111010 watermark_image_5
11111011 watermark_image_4
11111100 watermark_image_3
11111101 watermark_image_2
11111110 watermark_image_1
11111111 watermark_image_0
0 rst
watermarked_image_9 11111110
watermarked_image_8 11111111
watermarked_image_7 11111111
watermarked_image_6 11111111
watermarked_image_5 11111111
watermarked_image_4 11111111
watermarked_image_3 11111111
watermarked_image_2 11111111
watermarked_image_15 11111111
watermarked_image_14 11111111
watermarked_image_13 11111111
watermarked_image_12 11111111
watermarked_image_11 11111111
watermarked_image_10 11111111
watermarked_image_1 11111111
watermarked_image_0 11111111
cover_image_9 00001001
cover_image_8 00001000
cover_image_7 00000111
cover_image_6 00000110
cover_image_5 00000101
cover_image_4 00000100
cover_image_3 00000011
cover_image_2 00000010
cover_image_15 00001111
cover_image_14 00001110
cover_image_13 00001101
cover_image_12 00001100
cover_image_11 00001011
cover_image_10 00001010
cover_image_1 00000001
cover_image_0 00000000
Messages
/xor_output/waterm... 11111110
UUUUUUUU
11111110

Figure 2(a): AOI logic based watermark generator.

B. LFSR Based Address Generator

The resulted watermarked image will be stored in the memory locations of a SRAM for further process. To store in the desired and specific memory locations, we are implementing a LFSR based address generator [5]. The design of address generator using LFSR will be discussed in this section.

The LFSR based watermarked image will be sized in to 8-bit 256 rows and 256 columns. The 8-bit pattern of the image will be stored in the 8-bit address locations of SRAM. To generate 8-bit address locations we are implementing 8x8 size bit array LFSR. Figure 3(d) shows the sequence of 8-bit address generating LFSR simulated in ModelSim XE III.

Steps wise implementation of LFSR address generation and allocating memory locations to the watermark bits are as follows:

1. The LFSR watermarked image of size 256x256 will be taken in 8-bit pattern representation.
2. LFSR address generator will generate 8-bit random numbers, which will be used here as the memory locations of SRAM.
3. The seed value of the LFSR address generator we implemented is “10110001”.
4. The size of SRAM is 256KB so the memory locations will be of 8-bit combinations.
5. As the LFSR generates the memory locations, the 8-bit pixel values of the watermarked image will be stored in the SRAM memory locations.
6. In SRAM, serially the allocations of 8-bits in those specific memory locations will be done.
7. If any fault in the memory allocation is found, will be rectified.

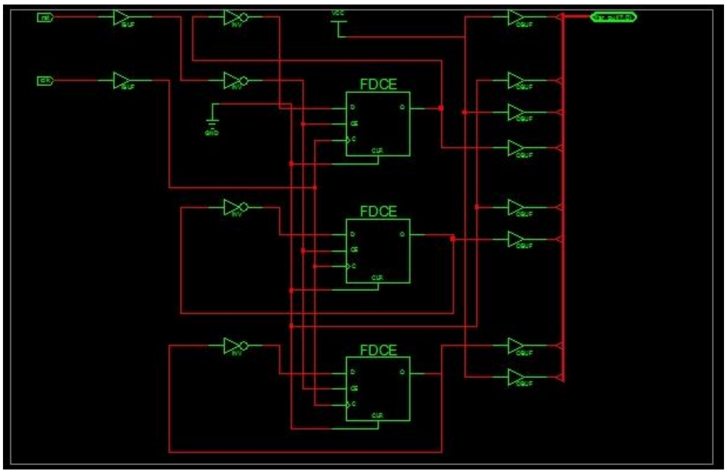
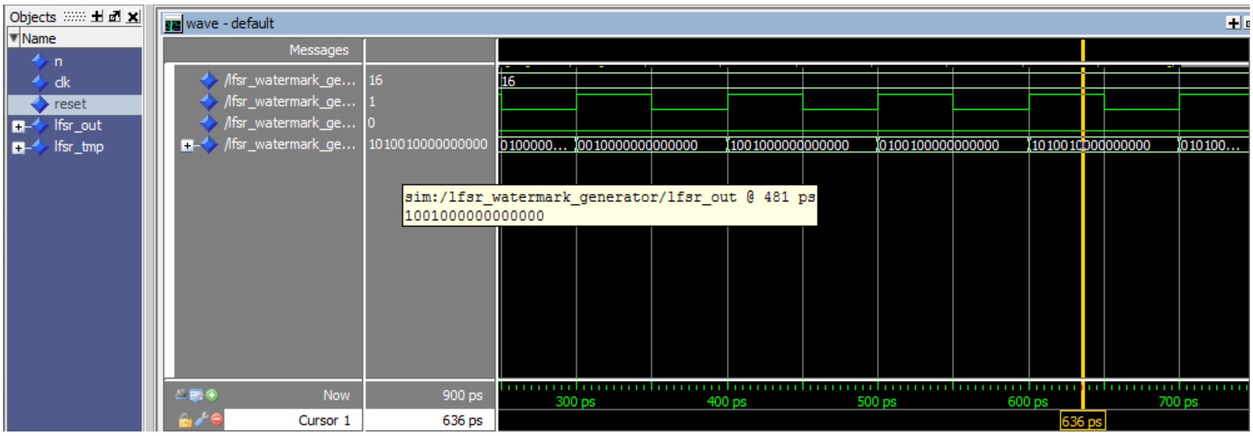


Figure 2(b): Technology view of Image watermark address generator.

IV. RESULTS

The digital image watermarking LFSR generates the sequence of random bit patterns ranging from initial value to maximum 255 bit patterns bit values. These patterns will be taken to watermark with the original (cover) image. The watermark generator we implemented in our paper is an AOI logic circuit, which will perform the combined cover and watermarked bit manipulation logic operation on cover image (c_i) bit patterns and watermark image (w_i)bit patterns.

In the watermark embedding AOI logic circuit the enable pin En_1 must be in active mode and clock pin Clk_1 must be a positive edge triggered.



```
sim:/lfsr_watermark_generator/lfsr_out @ 481 ps
1001000000000000
```

Figure 3(a): The sequence of 16-bit digital image watermarking LFSR simulated in ModelSim XE III.

Figure 3(b): Gray scale image taken for implementing our method.

The 16x16 size bit array of LFSR watermark generator output bit array will be watermarked with the original gray scale image of 256x256 size bit arra shown in the figure 3(b).

A watermarking generator has been designed using AOI logic in VHDL program, which will take the bits from original 256x256 size image and performs AOI bit watermarking with the LFSR watermark generated 16x16 size bit pattern. The watermarked bits will be scaled down to generate the watermarked image, where we can find the robustness [3] and [6] between the original and watermarked image.

Figure 2(a) shows the AOI logic based watermark generator.

Our proposed method can also be implemented for the combinations of:

TABLE 1: THE DIFFERENT COMBINATIONS OF SEED VALUES FOR THE GIVEN IMAGE SIZE.

S.No.	Original Image size	LFSR seed value
1	16x16	“1010101010101010”
2	32x32	“1011111010111110”
3	64x64	“1011100011100100”
4	128x128	“1100011010110001”

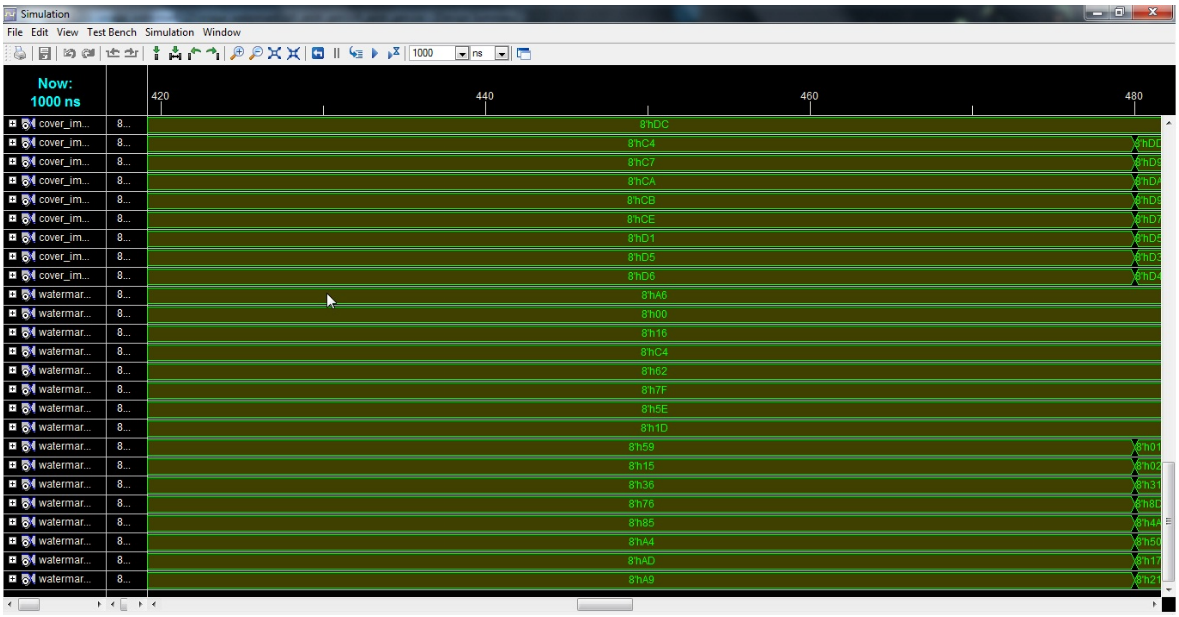

Simulation
File Edit View Test Bench Simulation Window
1000 ns
Now: 1000 ns
420
440
460
480
cover_im... 8... 8'hDC
cover_im... 8... 8'hC4 8'hDD
cover_im... 8... 8'hC7 8'hD9
cover_im... 8... 8'hCA 8'hDA
cover_im... 8... 8'hCB 8'hD9
cover_im... 8... 8'hCE 8'hD7
cover_im... 8... 8'hD1 8'hD5
cover_im... 8... 8'hD5 8'hD3
cover_im... 8... 8'hD6 8'hD4
watermar... 8... 8'hA6
watermar... 8... 8'h00
watermar... 8... 8'h16
watermar... 8... 8'hC4
watermar... 8... 8'h62
watermar... 8... 8'h7F
watermar... 8... 8'h5E
watermar... 8... 8'h1D
watermar... 8... 8'h59 8'h01
watermar... 8... 8'h15 8'h02
watermar... 8... 8'h36 8'h31
watermar... 8... 8'h76 8'h8D
watermar... 8... 8'h85 8'h4A
watermar... 8... 8'hA4 8'h50
watermar... 8... 8'hAD 8'h17
watermar... 8... 8'hA9 8'h21

Figure 3(c): Watermarked bits using AOI logic based watermark generator.

In AOI logic circuit, the implementation [1] and [5] will be done based on the logic operation on cover image (c_i) bit patterns and watermark image (w_i) bit patterns. The bit patterns are taken individually form original and watermarked image. In the equation 1, the subscripts are : W_d is the watermarked bit pattern, w_i is watermarking image bit, c_i is cover or original image bit, En_1 is enable input and Clk_1 is clock input.

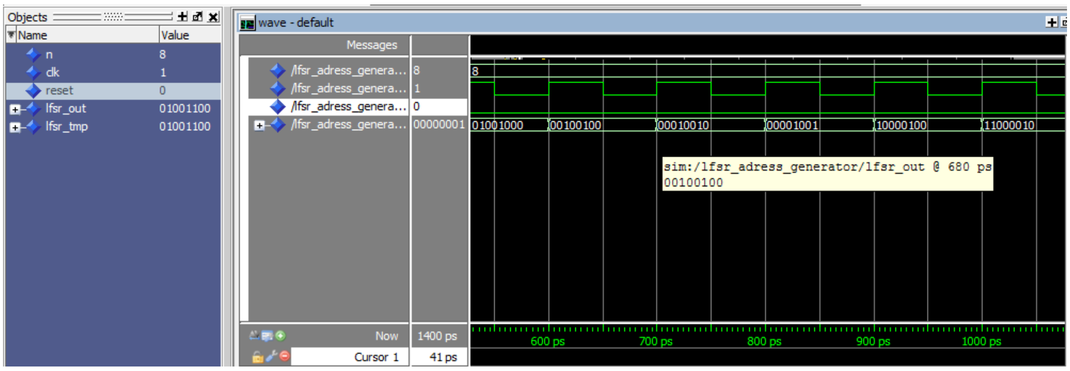
$$W_d = w_i * En\_1*Clk\_1+c_i'* En\_1 * Clk\_1 +w_ic_i *En\_1*Clk\_1+w_i'c_i *En\_1*Clk\_1.......................(1)$$

In the above simulated results, the watermarking bits which are generated will be of (16x16) x(16x16) patterns.

These bit patterns will be reshaped into 256x256 original image sizes.

In the next process of address generation, the 256x256 rows and columns image pixel bit values will be reshaped into 8x8 bit matrix.

These reshaped watermarked pixel bit values will be stored in the memory location of SRAM, defined by the LFSR address generator.



Name	Value
n	8
clk	1
reset	0
lfsr_out	01001100
lfsr_tmp	01001100

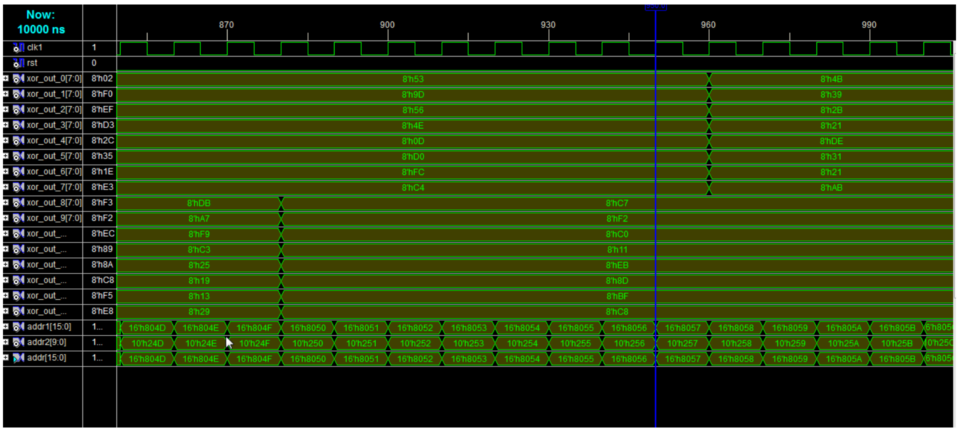
Messages	
/lfsr_adress_genera...	8
/lfsr_adress_genera...	1
/lfsr_adress_genera...	0
/lfsr_adress_genera...	00000001

01001000 00100100 00010010 00001001 10000100 11000010

sim:/lfsr_adress_generator/lfsr_out @ 680 ps
00100100

Now 1400 ps
Cursor 1 41 ps

Figure 3(d): The sequence of 8-bit address generating LFSR simulated in ModelSim XE III.



Now:
10000 ns
870
900
930
960
990
clk1 1
rst 0
xor_out_0[7:0] 8'h02 8'h53 8'h4B
xor_out_1[7:0] 8'hF0 8'h9D 8'h39
xor_out_2[7:0] 8'hEF 8'h56 8'h2B
xor_out_3[7:0] 8'hD3 8'h4E 8'h21
xor_out_4[7:0] 8'h2C 8'h0D 8'hDE
xor_out_5[7:0] 8'h35 8'hD0 8'h31
xor_out_6[7:0] 8'h1E 8'hFC 8'h21
xor_out_7[7:0] 8'hE3 8'hC4 8'hAB
xor_out_8[7:0] 8'hF3 8'hDB 8'hC7
xor_out_9[7:0] 8'hF2 8'hA7 8'hF2
xor_out_... 8'hEC 8'hF9 8'hC0
xor_out_... 8'h89 8'hC3 8'h11
xor_out_... 8'h8A 8'h25 8'hEB
xor_out_... 8'hC8 8'h19 8'h8D
xor_out_... 8'hF5 8'h13 8'hBF
xor_out_... 8'hE8 8'h29 8'hC8
addr1[15:0] 16'h804D 16'h804E 16'h804F 16'h8050 16'h8051 16'h8052 16'h8053 16'h8054 16'h8055 16'h8056 16'h8057 16'h8058 16'h8059 16'h805A 16'h805B
addr2[9:0] 10'h24D 10'h24E 10'h24F 10'h250 10'h251 10'h252 10'h253 10'h254 10'h255 10'h256 10'h257 10'h258 10'h259 10'h25A 10'h25B
addr[15:0] 16'h804D 16'h804E 16'h804F 16'h8050 16'h8051 16'h8052 16'h8053 16'h8054 16'h8055 16'h8056 16'h8057 16'h8058 16'h8059 16'h805A 16'h805B

Figure 3(e): The sequences of 8-bit watermarked bits are stored in 8-bit memory address locations generated by 8-bit LFSR, simulated in ModelSim XE III.

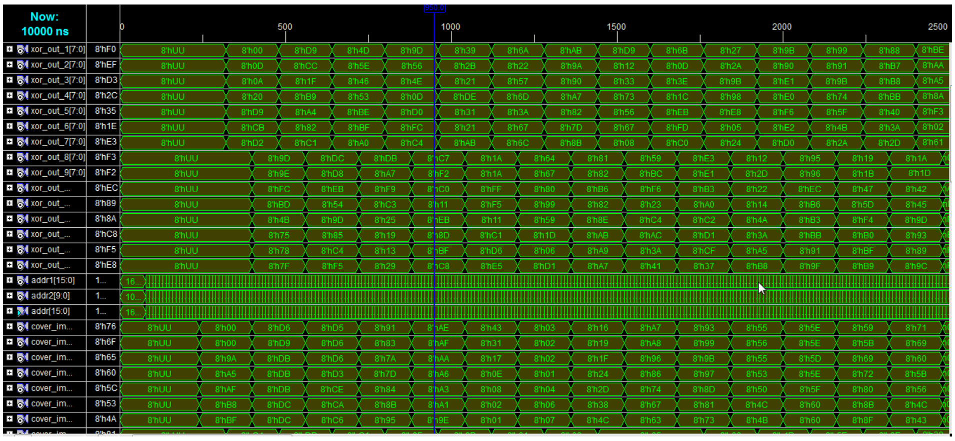


Figure 3(f): The cover image extracted from the watermarked image, simulated in ModelSim XE III.

In the above simulated results, the watermarked image 256x256 bits will undergo watermark extraction process using the same AOI logic circuit. The extracted watermark image bit values will be 8x8 matrixes of 24x24 dimensions. To convert these bit values, we used VB Script, which extracts the bit values serially and represents an image matrix. This image matrix is our original image. The extracted image shows better robustness to the noise.

Our method shown best results in generating the watermark and computing the address locations. We calculated PSNR to know the comparisons between the different LFSR seed values.

TABLE 2: PSNR VALUES OF DIFFRENT SEED VALUES

Original Image size	LFSR seed value	PSNR (dB)
16x16	"1010101010101010"	40.21
32x32	"1011111010111110"	38.24
64x64	"1011100011100100"	41.10
128x128	"1100011010110001"	42.23

V. CONCLUSIONS

The LFSR based digital image watermarking and address generation is discussed in this paper. The pseudo random generator LFSR having advantage to generative non-repetitive numbers till the seed value reached, had been shown a tremendous change in the watermarking technique.

In our paper, LFSR of 16x16 size bit array watermark generator shown good results in image watermarking and as 8x8 size bit array address generator shown good results in storing the watermarked image in the relevant memory locations.

Our method, improved the watermarked technique and offer high speed address generator for image watermarking.

VI. FUTURE SCOPE

LFSR's used in this paper improved the robustness of the image and increased the protection to the original image. The proposed technique of watermarking and address generation using LFSR can be further extended to test pattern generators for image watermarking SRAM. To test the SRAM and to find the faults in the SRAM memory locations we can use the LFSR. And this paper can be extended further for Memory Built in Self Test (MBIST) and Memory Built in Self Repair (MBISR).

REFERENCES

[1] R.Sundararaman and Dr. Har Narayan Upadhyay, *Stego System on Chip with LFSR based Information Hiding Approach*, International Journal of Computer Applications (0975 – 8887), Volume 18– No.2, March 2011, pp.24-31.

[2] W.A.S Wijesinghe, M.K Jayananda and D.U.J Sonnadara, *Hardware Implementation of Random Number Generators*, Proceedings of the Technical Sessions, 22 (2006) 28-38, Institute of Physics – Sri Lanka, pp.28-38.

[3] Nebu John Mathai, Student Member, IEEE, Deepa Kundur, Member, IEEE, and Ali Sheikholeslami, Member, IEEE, "*Hardware Implementation Perspectives of Digital Video Watermarking Algorithms*," *IEEE Transactions On Signal Processing, Vol. 51, No. 4, , pp. 925–938, April 2003*.

[4] B. Rajan and S.Ravi, "*FPGA Based Hardware Implementation of Image Filter With Dynamic Reconfigurable Architecture*," in *IJCSNS International Journal of Computer Science and Network Security, VOL.6 No.12, December 2006*, p. 121-127.

[5] Deepthi P.P. and P.S. Sathidevi, "*Hardware Stream Cipher Based on LFSR and Modular Division Circuit*," International Journal of Electrical and Computer Engineering 3:12 2008, pp.791-799.

[6] Saraju P. Mohanty, Renuka Kumara C, and Sridhara Nayak, "*FPGA Based Implementation of an Invisible-Robust Image Watermarking Encoder*," CIT 2004, LNCS 3356, pp. 344–353, 2004.

[7] Xiangxue Li, Dong Zheng, and Kefei Chen, "*LFSR-Based Signatures with Message Recovery*," International Journal of Network Security, Vol.4, No.3, pp.266–270, May 2007.