Hindawi
VLSI Design
Volume 2018, Article ID 4804729, 8 pages
https://doi.org/10.1155/2018/4804729

Research Article

Crypto-Stego-Real-Time (CSRT) System for Secure Reversible Data Hiding

Latika Desai and Suresh Mali

Dr. D. Y. Patil Institute of Technology, Savitribai Phule Pune University, Pune, India

Correspondence should be addressed to Latika Desai; latikadesai@gmail.com

Received 17 May 2018; Revised 16 August 2018; Accepted 1 September 2018; Published 27 September 2018

Academic Editor: Maurizio Martina

Due to demand of information transfer through higher speed wireless communication network, it is time to think about security of important information to be transferred. Further, as these communication networks are part of open channel, to preserve the security of any Critical Information (CI) is really a challenging task in any real-time application. Data hiding techniques give more security and robustness of important CI against encryption or cryptographic software solutions. However, hardwired approach exhibits better solution not only in terms of reduction of complexity but also in terms of adaptive real-time output. This paper demonstrates frequency, Discrete Cosine Transform (DCT) domain Steganographic data hiding hardware solution for secret communication called Crypto-Stego-Real-Time (CSRT) System. The challenge is to design a secure algorithm keeping reliability of minimum distortion of original cover signal while embedding considerable amount of CI. Field Programmable Gate Array (FPGA) implementation shown in this paper is more secure, robust, and fast. Pipelining process while embedding enhances the speed of embedding, optimizes the memory utilization, and gives better Peak Signal to Noise Ratio (PSNR) and high robustness. Practically implemented hardware Steganographic solutions shown in this paper also give better performance than that of the current state-of-the-art hardware implementations.

1. Introduction

In the today's Internet era, there is a need of protection of Critical Information (CI) like Personal Identification Number (PIN) of Automated Teller Machine (ATM) card, One Time Password (OTP), bank transactions, etc. while communicating CI in open channel system (internet). To avoid unauthorized access, one can encrypt CI itself before it actually gets transferred from one location to another. However, such encrypted CI is still specifically available to any hacker and will be in a position to extract CI. Therefore, to make highest security of CI, Steganographic data hiding techniques are used to keep the communication of CI itself hidden from the hacker. Many such techniques have been proposed earlier, which comes under two categories, namely, spatial domain and frequency domain. Most of the researchers are focused on the software-data hiding through different approaches like Least Significant Bit (LSB), 2/3 LSB, n-bit LSB, DCT, and Discrete Wavelet Transform (DWT). Unfortunately, software-data hiding techniques are generic, complex, and slow. Therefore these techniques are not suitable for real-time applications such as ATM where uploading authentication credentials (as CI) through Internet is always essential. Authenticating mobile wallets, online Stock trading, and many more real-time e-transactions applications are to be supported by cryptographic and steganographic security systems.

Researchers are constantly trying to improve the capabilities of reversible Steganographic data hiding methodologies in terms of parameters such as embedding capacity, imperceptibility, security, time complexity, and robustness. Implementation of embedding function (f_{Em}) and extraction function (f_{Ex}) may be in either software or hardware platform. Hardware implementation has always offered advantages over software realization [1] in terms of low execution time, low power consumption, high reliability, and real-time performance. Implemented hardware can also be made compatible with existing consumer electronics communicating devices. Table 1 also narrates many more advantages of hardware approaches by [1–3].

Table 1: Hardware versus software based implementation of data security.

Software Implementation	Hardware Implementation
Generalized design under PC environment	Customized designed using FPGAs (ES)
Complex Algorithms can be possible	Simplified Algorithms with hardware
Not a Real time solution	Real time solution with hardware
More power requires for PC	Less power required for embedded system
Easy to modify the design	Difficult to modify the design
More area & cannot execute without PC.	Less area & can execute without PC.
Not Portable	Portable
The Cost of PC, OS and language support	Low-cost due to specific hardware involved
Less secured	More secured
Easy to copy the implemented code.	Difficult to copy the implemented hardware.
All operations are sequential	Parallel operations are possible

2. Overview of Reversible Data Hiding Techniques

Reconfigurable hardware architecture supports algorithm change and multiple keys, as well as different CI size. It speeds up the process of embedding due to availability of cache memory. Multiprocessor System on Chip (MPSoC) architectures demonstrated by Maity et al. [4] exhibits high intrusion protection. It gives real-time transmission using channel coding and biphase modulation. However, the computation requirement for embedding is quite high. The Adaptive Random Inverted Key (ARIK) LSB substitution method proposed by Balakrishnan et al. [5] using cyclone II FPGA board has improved performance with respect to imperceptibility of Stego. However, the designed architecture was operated at a frequency of 50 MHz, occupies 10513 logic elements, and consumes 92 mw of power at the embedding stage. Effective integration of conventional cryptography, encryption, and Steganography has been presented by Mali et al. [6] using Adaptive Energy Thresholding (AET) technique. The approach enhances the parameters security, robustness, and payload against different attacks. Being implemented in software, the approach cannot be applied in real-time environment. Gomez-Hernandez et al. [7] worked on context techniques used for embedding, where the loss of information during recovery process has been avoided using simple and repetitive operations with context technique. However, the performance would have been better with parallelism in terms of area and time.

Mohd et al. [8] demonstrate the FPGA hardware implementation in spatial domain Steganography. Simulation, synthesis, and analysis show that random embedding increases utilization of LEs. However, spatial domain techniques are always more vulnerable than that of frequency domain techniques. Adaptive randomization in reconfigurable hardware architecture using Integer Wavelet Transform (IWT) has been proposed by Ramalingam et al. [9]. Although it gives PSNR up to 60, the design consumed 34% of the LEs, 22% of the dedicated logic register, and 2% of the embedded multiplier on FPGA. The main drawback of the IWT based data hiding is the computational overhead. The performance of the algorithm implemented in an FPGA-based hardware by Mohd et al. [10] has been examined with respect to resource utilization, timing, and energy. Because of higher complex computation, the implementation has a higher cost and slower speed.

The spatial domain hardware approach has also been introduced by Mahmoudpour and Mirzakuchaki et al. [11] through randomizations using Linear Feedback Shift Register (LFSR) for getting better security against attack. However, the maximum value of PSNR in this approach is 51.217. Hardware platform has also been used by Rajagopala et al. [12]. However, being a spatial domain embedding approach, it is less secure and non_robust even if PSNR is up to 60.98. Reversible Steganography spatial domain implemented using LFSR has been demonstrated by Mahmood et al. [13]. The different LFSR and seeds are preferred for better security. However, complex logic consumes more hardware and the maximum value of PSNR is 51.21 for even 10% of the pay load of embedding.

The reversible data hiding implementation proposed by Sundararaman et al. [14] is the spatial domain approach with LFSR. Five different polynomials are used for five different covers with different sizes. Because of the limited use of bitwise Steganographic operation, this hardware approach is less secure. The demonstrated approach has maximum PSNR 51.27. Also, Synthesis report states that for 64 x 64 size cover LEs used are 11493. Intermediate Significant Bit (ISB) replacement technique demonstrated by Shabir et al. [15] suggests a fix location for embedding. Although it provides capacity up to 25%, the maximum PSNR achieved due to ISB process is 37.97. The transform domain DCT based approach for data hiding implemented in MATLAB by Rahman et al. [16] uses sequential embedding of secret bits in the LSBs of DCT coefficients. The experimental results show that the use of middle frequency DCT coefficients has given better results of PSNR as compared to low frequency DCT coefficients. However, being a software implementation, it is not suitable for real-time applications.

Anderson and Petitcolas [17] show that considerations of entropy give us better results as it gives us some quantitative leverage. The embedding of information is in parity checks

Table 2: Analysis of various data hiding techniques.

Approach	Imperceptibility	Capacity	Robustness	Time Complexity
Spatial Domain[31]	Medium	High	Low	Low
Frequency Domain[32]	**High**	**Low**	**High**	**Low**
Spread Spectrum [33]	High	High	Medium	High
Matrix Embedding [34]	High	High	Low	Medium
Difference Expansion[35]	High	High	Low	Low
Optimization Technique [36]	High	Medium	High	High
Histogram Modification [37]	High	Medium	Low	Low

rather than in the data directly. This approach gives improved efficiency and also allows us to do public key Steganography. The Steganography approach introduced by Odeh et al. [18] is a real-time hardware engine, where text is embedded. By taking care of a real-time application proposed work is faster to maintain security in communication over Internet. However, because of lack of parallel processing, the speed of embedding is low. If we consider different applications like covert communication, fingerprinting, and copyright protection as well as many more, the most important parameters observed are the security, robustness, invisibility, time complexity, and area and power dissipation. Among all these parameters security is the major aspect and is commonly used in all these different applications announced by Fedrich [19].

The FPGA-based microarchitecture defined by Farouk et al. [20] with the secret key feature is suitable for real-time application. Selecting the hiding bits in a pseudorandom manner as a function of a secret key has been used to increase obscurity. The receiver needs only the modulated cover and the secret key to recover the message, i.e., no original cover is required.

The importance of DCT in terms of scalability and minimal distortion demonstrated by Cariccia et al. [21] as well as Renda et al. [22] is quite noticeable. Because of the ability of DCT algorithms to produce high throughput, low power dissipation, and reduced chip area with primary objective of security, they are more popular amongst the researchers.

Cryptographic different approaches are demonstrated through [23–29] using Virtex 5 FPGA platform. While securing the data, with AES (Advanced Encryption Standard) as well as Data Decryption Systems Standard (DES) and Triple DES (TDES) using FPGA, different approaches are implemented to enhance the performance for secure communication. However, the area utilization is varying from 260 LEs to 9276 LEs. Watermarking method using DWT [30] with 344.329 MHz frequency has also shown considerably less area utilized with more delays in the process of embedding. Comparison of software and hardware implementation of such systems is given in Table 1. This table shows that the hardware deployment gives more speed, the secrecy, and the robustness with minimum cost as compared to software counterpart. Brief analysis of various techniques [31–37] to develop such systems is given in Table 2. The frequency domain techniques demonstrated in [32–37] as well as [21, 22] show better performance in terms of imperceptibility, capacity, robustness, and process.

Analysis of the state-of-the-art techniques leads to some interesting findings: frequency domain gives high robustness and high imperceptibility. However, these techniques are very complex and time consuming while being implemented in hardware. Further, data hiding Capacity, Security, and Robustness parameters are mutually exclusive to each other and therefore optimizing them in real-time hardware environment is really a tough task. The support of parallel processing helps in compensating for the time needed for such complex computations.

3. Proposed Methodology

Any non-Critical Information called cover data (C) acts as a carrier of Critical Information (CI). A Secrete Key (K) is used by the Steganographic embedding function (f_{Em}) to hide CI and gives Stego data (S) as an output (device at Transmitting end D_T) as shown in

$$S \xrightarrow{D_T} f_{Em}(C, CI, K) \tag{1}$$

where S is Stego data, C is cover data, CI is Critical Information, and K is Secrete Key.

The same Secrete Key (K) is used by the Steganographic extraction function (f_{Ex}) to extract CI′ (as a device at receiving end D_R) as shown in

$$CI' \xrightarrow{D_R} f_{Ex}(S, K) \tag{2}$$

where S is Stego data, CI′ is Extracted Critical Information, and K is Secrete Key.

Typical generalized hardware Steganographic data hiding mechanism is as shown in Figure 1.

Proposed hardware based reversible data hiding system to implement data hiding system consists of both cryptographic and Steganographic approach and therefore is called Crypto-Stego-Real-Time (CSRT) System. Figure 2 outlines the proposed methodology.

As a part of cryptography, the encryption process is converting CI from plain text into unintelligible ciphertext. On the receiving side of the process, decryption is used to convert this unintelligible ciphertext back into plaintext CI′ as an extracted CI. If CI consists of M *"Characters"* in CI, stored in the Message Cache, it can be represented as

$$CI = [X_0, X_1, X_2, \ldots, X_{M-1},] \tag{3}$$

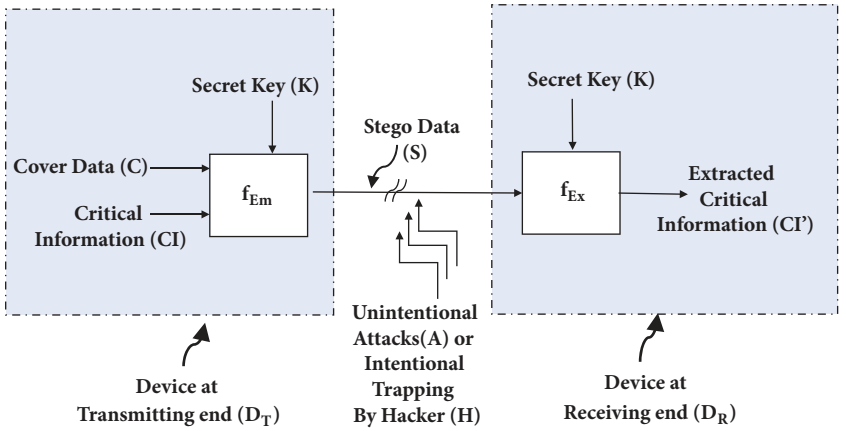

Figure 1: Steganographic data hiding mechanism.

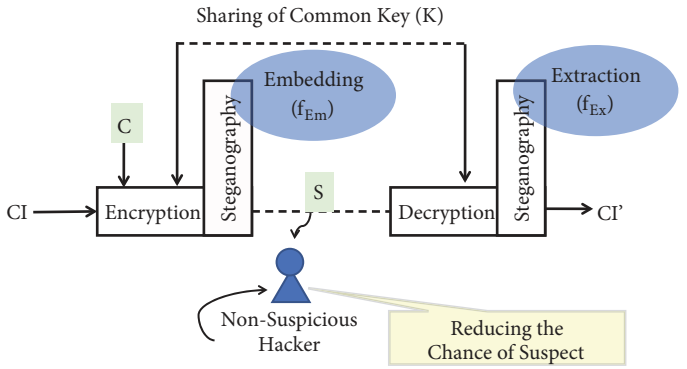

Figure 2: Proposed Crypto-Stego-Real-Time (CSRT) System.

where M is number of characters in CI stored in "Message Cache" at 'A$_0$ to A$_{M-1}$ address locations in a sequence.

The process of encryption typically carried out using *"Randomly Selected Set of Addresses"* stored in a *"LOOK-UP Table"* is randomly selecting any address location (A$_m$) of Message Cache and hence, at any given time, one of the characters stored in Message Cache get selected as "Y$_m$" and can be written as follows: there is a *"LOOK-UP Table"* consisting of random numbers which eventually act as an addresses to locate any random character at *"Message Cache"*. Obviously, the number of locations in *"LOOK-UP Table"* is 8 times more than that of the number of locations in *"Message Cache"*, i.e., 8∗M. At any given time, one of the characters stored in *"Message Cache"* gets selected as *"Randomly Selected Character"* given by

$$Y_m = [A_m] = [[L_i]] = [[Cnt]] \tag{4}$$

where Y$_m$ is Randomly Selected Character and A$_m$is Randomly Selected Address for "*Message Cache*" and eventually content of sequential location of *"LOOK-UP Table"* 0 to M-1 excluding three *"Least Significant Bits"* of content of L$_i$.

The *"Clock Generator"* of CSRT selects sequential locations of *"LOOK-UP Table"* and is given as

$$L_i = Cnt \tag{5}$$

where Cnt is output of the *COUNTER*.

The format of content of L$_i$ while selecting the character and specific bit of the character is as shown in Figure 3. As there are 8 bits in each character, the randomization is also applicable to these bits.

The proposed algorithm works on 8 bytes of cover at a time as follows:

$$C = [B_7, B_6, B_5, \ldots, B_0] \tag{6}$$

where B$_7$ to B$_0$ = 8 bytes of the cover given to **CSRT** at any given time whose DCT can be written as

$$D = DCT\,(C) \tag{7}$$

where D is A set of 8 DCT coefficients of 8 respective bytes of the cover given in (6) which can also be written as

$$D = C_7, C_6, C_5, \ldots, C_0 \tag{8}$$

where every DCT coefficient (C) consists of signed binary representation. The bit selected (shown in Figure 3) is embedded in C$_3$. Embedding of the bit in C$_3$ is nothing but making LSB of C$_3$ as that of bit to be embedded as

$$\begin{aligned} \text{LSB of } C_3 &= \text{Selected Bit of } Y_m \\ &= \text{Selected Bit of } [[L_i]] \end{aligned} \tag{9}$$

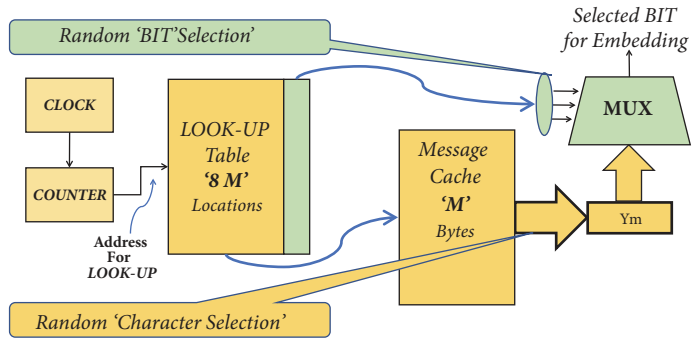

Figure 3: Randomization of characters and bits of "*Message Cache*".

After modifying the LSB of $\mathbf{C}_3$ inverse DCT (IDCT) of D is taken to get Stego as follows:

$$S = IDCT\,(D) \tag{10}$$

where is S is the Stego block of data consisting of 8 bytes having an embedded bit of information in LSB of $\mathbf{C}_3$.

4. Embedding Algorithm

(1) Accept CI into *Message Cache*
(2) Accept *LOOK-UP Table* as an *Embedding Key* (K)
(3) Accept 8-byte *cover data* (C)
(4) Compute DCT of 8 bytes of cover data
(5) Randomly select byte (Y_m) from *Message Cache*
(6) Select a bit using 3 LSBs of contents of selected byte *LOOK-UP Table*
(7) Embed the selected bit at DCT coefficient C3
(8) Compute IDCT of 8 bytes of cover data to get *Stego data*
(9) Repeat Step-3 to Step-8 for all the bits of all the characters *Message Cache*
(10) Stop.

Although there are multiple steps in the proposed algorithm, it is possible to have parallel processing of Step-(3) to Step-8. If we take *Pre_Em = Step-(3) and Step-(4), Emd = Step-(5) and Step-(6) and Post-Em = Step-(7) and Step-(8)*, it is possible to reduce the total time required for embedding all the bits of *Message Cache* as shown in Figure 4.

5. Experimental Set Up

The logic necessary to implement the algorithm is downloaded from the PC to Configurable FPGA board using Virtex-5, XC5VLX50T, FFG1136C Board. Critical Information (CI) and *Embedding Key* (K) are transferred to the FPGA RAM at *Message Cache* and *LOOK-UP Table, respectively.* The *cover data* (C) is then transferred on byte by byte basis. After the embedding process, it gets validated by transferring *Stego data* (S) back to PC. While extracting the Critical Information (CI′) the *Stego data* (S) sent from the PC to FPGA board on byte by byte basis and the extracted CI' stored in FPGA RAM at *Message Cache* is validated by receiving it on PC.

6. Result and Discussion

This section declares the analysis of different parameters like PSNR, area, and the time. The different methodologies are explained in the literature such as LSB, MSB, DWT, IWT, and proposed approach [CSRT]. The PSNR values are shown through Figure 5. The proposed hardware approach gives 73.77 dB PSNR value which is approximately 10% more than mentioned approaches.

The analysis of the proposed hardware approach for the area in terms of Slice Registers, Slice Look-Up-Tables (LUTs), and slice LUT-FF pair used in various approaches is demonstrated through Figures 6, 7, and 8 respectively. The proposed CSRT method uses only 241 Slice Registers and 1202 LUTs.

To understand the performance, speed of execution of algorithm of embedding of CI has been considered. This time varies from 84.48 μs to 1351.68 μs while embedding of CI from 128 bytes to 2 K bytes in 1K bytes to 16 K bytes of cover data; Figure 9 gives an idea of reduction of processing speed of the proposed CSRT method.

Security of the system is very high as the cryptographic Encryption and Steganographic data hiding processes has been developed in FPGA hardware. Although hidden data are reversible and can be recovered by the intended user, it should not be detectable by any hacker (most of the hackers are nonsuspicious) as shown in Figure 2. The sharing of a common key is important for the intended extractor. This key may be predecided from look up table which eventually randomly select the bits of the characters from message cache for embedding. A person who knows the algorithm and look-up table must be the person who will be able to extract the Critical Information (CI). For example, for 128 bytes of CI, there will be 1 K bytes of look-up table and the attacker has to try 2^{1024} and check permutations and combinations of look-up table provided that the embedder knows the algorithm. As CI increases, the size of the look-up table also increases, and hence the permutations and combinations of look-up table. Although this dependability is the drawback of the system,

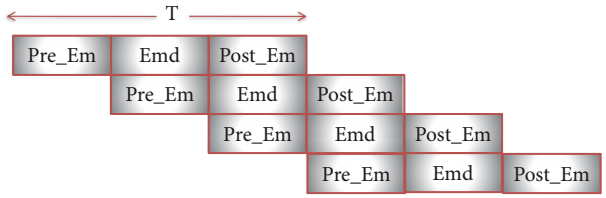

Figure 4: Parallel processing of steps of CSRT algorithm.

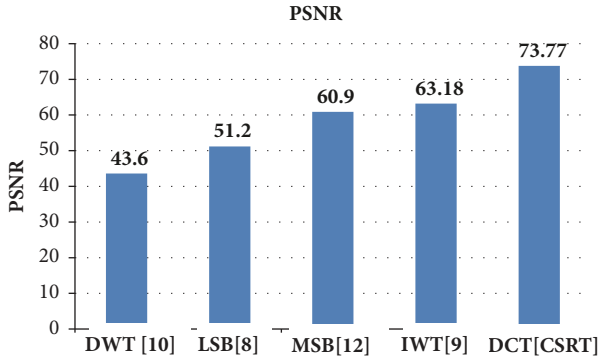

Figure 5: PSNR with different hardware approaches.

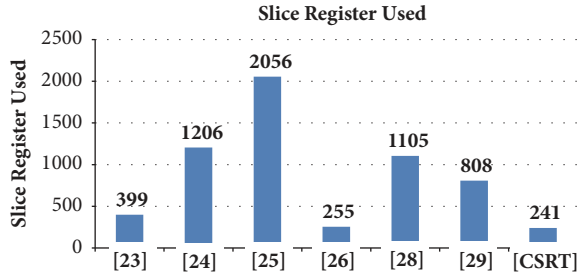

Figure 6: Slice registers used with different approaches.

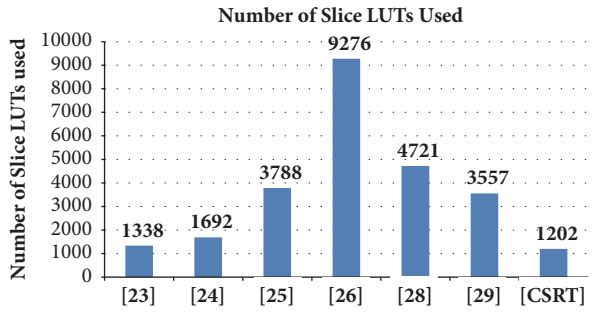

Figure 7: Number of slice LUTs used with different approaches.

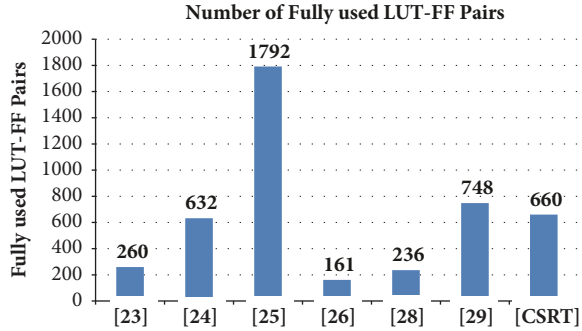

Figure 8: Fully used slice LUT-FF pairs with different approaches.

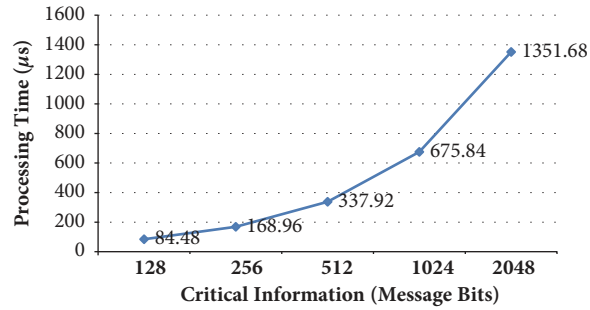

Figure 9: Critical Information (CI) processing time.

the system is reconfigurable and, therefore, one can design the look-up table of any size based on size of the CI.

7. Conclusion

The Crypto-Stego-Real-Time (CSRT) System presented in this paper is highly secured due to its implementation in hardware. As the encryption of CI has been done in the hardware using cache memory, the random addresses of *cache memory* are used for scribbling CI to be embedded in the cover bit by bit fashion. One cannot extract CI unless he/she knows the algorithm and look-up table as a key of embedding. Pipelining process while embedding enhances the speed of embedding and optimizes the memory utilization. The system PSNR (73.77), Number of Slice LUTs used (1202), and execution time (1351.68 μS for 2 K bits of CI) show an evidence of better performance in terms of throughput of the system. The drawback of the system is its Embedding Key which varies with CI. As CI increases, the size of the key also increases. However, the system is reconfigurable and, therefore, one can design the look-up table of any size based on the size of the CI. The overall performance of the presented CSRT system is better than current state-of-the-art methods and, therefore, it is most suitable for any real-time applications.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

We are thankful to Dr. D. Y. Patil Institute of Technology, Pimpri, Pune, for encouragement and support.

References

[1] S. Debnath, M. Kalita, and S. Majumder, "A Review on Hardware Implementation of Steganography," in *Proceedings of the Devices for Integrated Circuit*, pp. 149–152, Kalyani, India, March 2017.

[2] M. Faheem, S. Jamel, A. Hassan, Z. A., N. Shafinaz, and M. Mat, "A Survey on the cryptographic encryption algorithms," *International Journal of Advanced Computer Science and Applications*, vol. 8, no. 11, pp. 333–344, 2017.

[3] https://www.infosecurity-magazine.com/magazine-features/tales-crypt-hardware-software/.

[4] S. P. Maity and M. K. Kundu, "Distortion free image-in-image communication with implementation in FPGA," *Elsevier International Journal of Electronics and Communications*, vol. 67, pp. 438–447, 2012.

[5] R. Balakrishnan, A. Rengarajan, and J. B. B. Rayappan, "Multiplexed stego path on reconfigurable hardware: a novel random approach," *Elsevier Computers and Electrical Engineering*, vol. 55, pp. 153–163, 2016.

[6] S. N. Mali, P. M. Patil, and R. M. Jalnekar, "Robust and secured image-adaptive data hiding," *Digital Signal Processing*, vol. 22, no. 2, pp. 314–323, 2012.

[7] E. Gómez-Hernández, C. Feregrino-Uribe, and R. Cumplido, "FPGA hardware architecture of the steganographic context technique," in *Proceedings of the IEEE Computer Society International Conference on Electronics, Communications and Computers*, pp. 123–128, Mexico, March 2008.

[8] B. J. Mohd, T. Hayajneh, S. Abed, and A. Itradat, "Analysis and modeling of FPGA implementations of spatial steganography methods," *Journal of Circuits, Systems and Computers*, vol. 23, no. 2, pp. 1–25, 2014.

[9] B. Ramalingam, R. Amirtharajan, and J. B. B. Rayappan, "Stego on FPGA: An IWT Approach," *Hindawi Publishing Corporation Scientific World Journal*, pp. 1–9, 2014.

[10] B. J. Mohd, T. Hayajneh, and A. N. Quttoum, "Wavelet-transform steganography: Algorithm and hardware implementation," *International Journal of Electronic Security and Digital Forensics*, vol. 5, no. 3-4, pp. 241–256, 2013.

[11] S. Mahmoudpour and S. Mirzakuchaki, "Hardware Architecture for a Message Hiding Algorithm with Novel Randomizers," *International Journal of Computer Applications*, vol. 37, no. 7, pp. 46–53, 2012.

[12] S. Rajagopala, P. J. Prabhakar, M. S. Kumar et al., "MSB Based Embedding with Integrity: An Adaptive RGB Stego on FPGA Platform," *Information Technology Journal*, vol. 13, no. 12, pp. 1945–1952, 2014.

[13] A. Mahmood, N. Kanai, and S. Mohmmad, "An FPGA implementation of secured steganography communication system," *Tikrit Journal of Engineering Sciences*, vol. 19, no. 4, pp. 14–23, 2012.

[14] R. Sundararaman and H. Narayan Upadhyay, "Stego system on chip with lfsr based information hiding approach," *International Journal of Computer Applications*, vol. 18, no. 2, pp. 24–31, 2011.

[15] A. Shabir, A. Parah Javaid, and G. M. Bhat, "Data hiding in intermediate significant bit planes, a high capacity blind steganographic technique," in *Proceedings of the IEEE International Conference on Emerging Trends in Science, Engineering and Technology*, pp. 192–197, 2012.

[16] S. A. El_Rahman, "A comparative analysis of image steganography based on DCT algorithm and steganography tool to hide nuclear reactors confidential information," *Computers and Electrical Engineering*, pp. 1–20, 2016.

[17] R. J. Anderson and F. A. P. Petitcolas, "On the limits of steganography," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 4, pp. 474–481, 1998.

[18] A. Odeh, K. Elleithy, and M. Faezipour, "Fast real-time hardware engine for multipoint text steganography," in *Proceedings of the 2014 IEEE Long Island Systems, Applications and Technology Conference, LISAT 2014*, pp. 1–5, May 2014.

[19] J. Fridrich, *Steganography in Digital Media: Principles, Algorithms, and Applications*, Cambridge University Press, New York, NY, USA, 2009.

[20] H. A. Farouk and M. Saeb, "Design and implementation of a secret key steganographic micro-architecture employing FPGA," in *Proceedings of the ASP - DAC 2004 Asia and South Pacific Design Automation Conference - 2004*, pp. 577-578, Japan, January 2004.

[21] F. Cariccia, P. Cariccia, M. Martina, A. Molino, and F. Vacca, "Multimedia SoC: A systolic core for embedded DCT evaluation," in *Proceedings of the IEEE Conference Paper*, pp. 1749–1753, USA, November 2002.

[22] G. Renda, M. Masera, M. Martina, and G. Masera, "Approximate Arai DCT architecture for HEVC," in *Proceedings of the 1st New Generation of CAS, NGCAS 2017*, pp. 133–136, Italy, September 2017.

[23] M. H. Rais and S. M. Qasim, "Virtex-5 Fpga implementation of advanced encryptionstandard algorithm," in *Proceedings of the 3rd Global Conference on Power Control and Optimization*, pp. 201–206, May 2010.

[24] P. Ghosal, M. Biswas, and M. Biswas, "Hardware Implementation of TDES CryptoSystem with on chip verification in FPGA," *Journal of Telecommunications*, vol. 1, no. 1, pp. 113–117, 2010.

[25] K. Rahimunnisa, P. Karthigaikumar, S. Rasheed, J. Jayakumar, and S. Sureshkumar, "FPGA implementation of AES algorithm for high throughput using folded parallel architecture," *Security and Communication Networks*, vol. 7, no. 11, pp. 2225–2236, 2014.

[26] U. Farooq and M. F. Aslam, "Comparative analysis of different AES implementation techniques for efficient resource usage and better performance of an FPGA," *Journal of King Saud University - Computer and Information Sciences*, vol. 29, no. 3, pp. 295–302, 2017.

[27] A. Gielata, P. Russek, and K. Wiatr, "AES Hardware Implementation in FPGA for Algorithm Acceleration Purpose," in *Proceedings of the ICSES 2008 International Conference on Signals and Electronic Systems, ICSES'08*, pp. 137–140, Poland, September 2008.

[28] S. Sadoudi, C. Tanougast, M. S. Azzaz, and A. Dandache, "Design and FPGA Implementation of a wireless hyperchaotic communication system for secure real-time image transmission," *Journal on Image and Video Processing Springer*, pp. 1–18, 2013.

[29] R. R. Farashahi, B. Rashidi, and S. M. Sayedi, "FPGA based fast and high-throughput 2-slow retiming 128-bit AES encryption algorithm," *Microelectronics Journal*, vol. 45, pp. 1014–1025, 2014.

[30] P. P. Karthigai and A. K. Baskaran, "FPGA implementation of high speed low area DWT based invisible image watermarking algorithm," in *Proceedings of the International Conference on Communication Technology and System Design Elsevier*, vol. 30, pp. 266–273, October 2012.

[31] T.-T. Quach, "Optimal cover estimation methods and steganographic payload location," *IEEE Transactions on Information Forensics and Security*, vol. 6, no. 4, pp. 1214–1222, 2011.

[32] H.-L. Yeh, S.-T. Gue, P. Tsai, and W.-K. Shih, "Wavelet bit-plane based data hiding for compressed images," *AEÜ - International Journal of Electronics and Communications*, vol. 67, no. 9, pp. 808–815, 2013.

[33] A. Valizadeh and Z. J. Wang, "Correlation-and-bit-aware spread spectrum embedding for data hiding," *IEEE Transactions on Information Forensics and Security*, vol. 6, no. 2, pp. 267–282, 2011.

[34] Q. Mao, "A fast algorithm for matrix embedding steganography," *Digital Signal Processing*, vol. 25, no. 1, pp. 248–254, 2014.

[35] O. M. Al-Qershi and B. E. Khoo, "High capacity data hiding schemes for medical images based on difference expansion," *The Journal of Systems and Software*, vol. 84, no. 1, pp. 105–112, 2011.

[36] A. Khamrui and J. K. Mandal, "A Genetic Algorithm-Based Steganography using Discrete Cosine Transformation (GAS-DCT)," in *Proceedings of the International Conference on Computational Intelligence: Modeling Techniques and Applications*, vol. 10, pp. 105–111, 2013.

[37] N. A. Saleh, H. N. Boghdady, S. I. Shaheen, and A. M. Darwish, "High capacity lossless data embedding technique for palette images based on histogram analysis," *Digital Signal Processing*, vol. 20, no. 6, pp. 1629–1636, 2010.