IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 26, 2020, accepted March 22, 2020, date of publication April 8, 2020, date of current version April 23, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2986134

Parallel Hardware Implementation of Efficient Embedding Bit Rate Control Based Contrast Mapping Algorithm for Reversible Invisible Watermarking

SUBHAJIT DAS[1], ARUN KUMAR SUNANIYA[1], RESHMI MAITY[2], AND NILADRI PRATAP MAITY[2], (Senior Member, IEEE)

[1]Department of Electronics and Instrumentation Engineering, National Institute of Technology, Silchar 788010, India
[2]Department of Electronics and Communication Engineering, Mizoram University, (A Central University, Government of India), Tanhril 796004, India

Corresponding author: Subhajit Das (subhajitdas151@gmail.com)

ABSTRACT This paper presents an improved reversible contrast mapping (RCM) algorithm for reversible invisible watermarking (RIW) in both software and hardware platforms. Based on well-known parameters for RIW like distortion, embedding bit rate, payload size and data hiding capacity, an efficient embedding bit rate control based contrast mapping (EBCRCM) algorithm is proposed. An adaptive linear contrast mapping on pixel intensity value is asserted with RCM that controls the embedding bit rate without changing the embedding capacity to maintain distortion. Xilinx system generator (XSG) and VIVADO tool construct the novel VLSI architecture that needs 173.362 ns latency for 100 MHz clock with throughput 46.146 Mbps and 5.8 ns critical path for single cycle of embedding process. The proposed algorithm is verified in MATLAB tool based software platform by taking different types of multimedia data like gray-scale images, color images and video signals. Implementation of low hardware resources based VLSI architecture through zed-board in real time field programmable gate array (FPGA) platform confirms the capability of high speed, low cost and real-time use. 100% agreement is observed from software simulations and hardware platforms.

INDEX TERMS Reversible image watermarking, reversible contrast mapping, Xilinx system generator, FPGA.

I. INTRODUCTION

The modern world depends on digital medium as well as digital multimedia data for contemporary communication system. The proficient choice to provide authenticity as well as ownership of these digital multimedia data is digital watermarking (DW) [1]. A watermark which can be a pattern of bits or any type of digital information like digital signature, digital finger print data are embedded into the original cover information during data embedding. The digital watermark is robust in nature [2]–[4]. In other words, DW is used when extraction of the watermark is not required. In case of some DW algorithms, the watermark may be extracted back but it causes harm to the cover data. Clearly some information about the cover data is lost during data extraction. Moreover for some applications like medical imaging, legal domains information based military data transmissions; the small imperceptible distortion that occurs due to the watermarking process is not acceptable. To overcome these problems and recover back both the original and watermark data without any loss, the reversible watermarking is taken in account [3], [4]. Some important parameters need to be explored first. The efficiency and superiority of an algorithm for RIW are found out based on these parameters. They are distortion, data hiding capacity, payload size and embedding bit rate. Distortion is a change or exaggeration that makes something come into view different from the way it actually is. It is a very important feather in RIW. It can provide the detail of embedding possibility in multimedia indirectly. The two unique and basic

The associate editor coordinating the review of this manuscript and approving it for publication was Mehul S Raval.

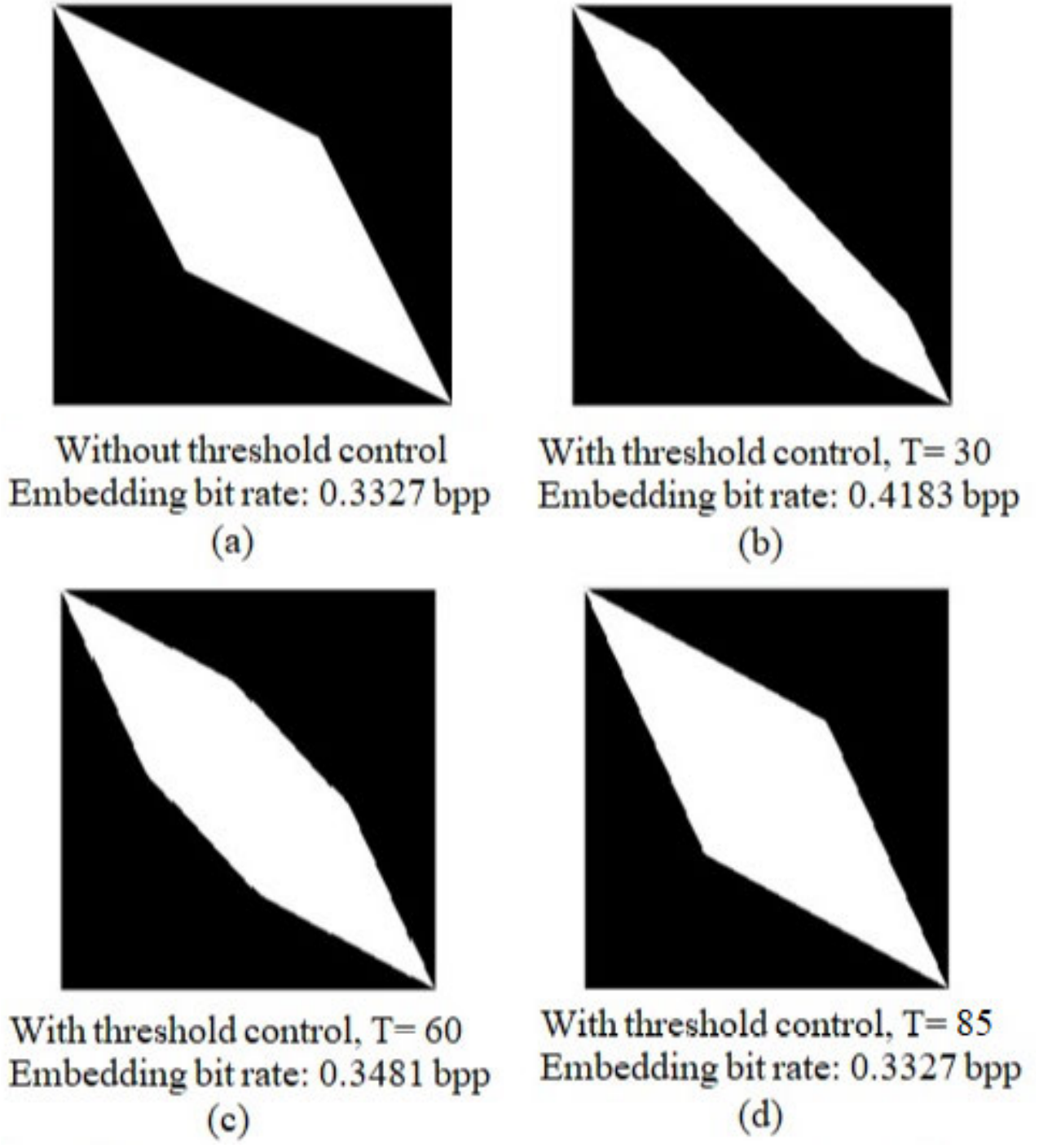

FIGURE 1. Controlling over the payload capacity using threshold.

parameters used for measuring the distortions are peak signal to noise ratio (PSNR) and structural similarity index matrix (SSIM).

The PSNR can be explored as the maximum value of signal power to Mean Signal Error (MSE). It is noted that PSNR is indirectly proportional to MSE. If the disparity between original and watermarked multimedia data is increased then PSNR will be reduced which in turn provides a bad visual quality based watermarked image. PSNR will be affected by embedding capacity and embedding bit rate. PSNR should be maintained in a limited range. It is found that the PSNR between the original and watermarked data vary in between 36 dB to 38 dB to maintain its blind property and provide the best visual quality based invisible watermarked data [1]–[5].

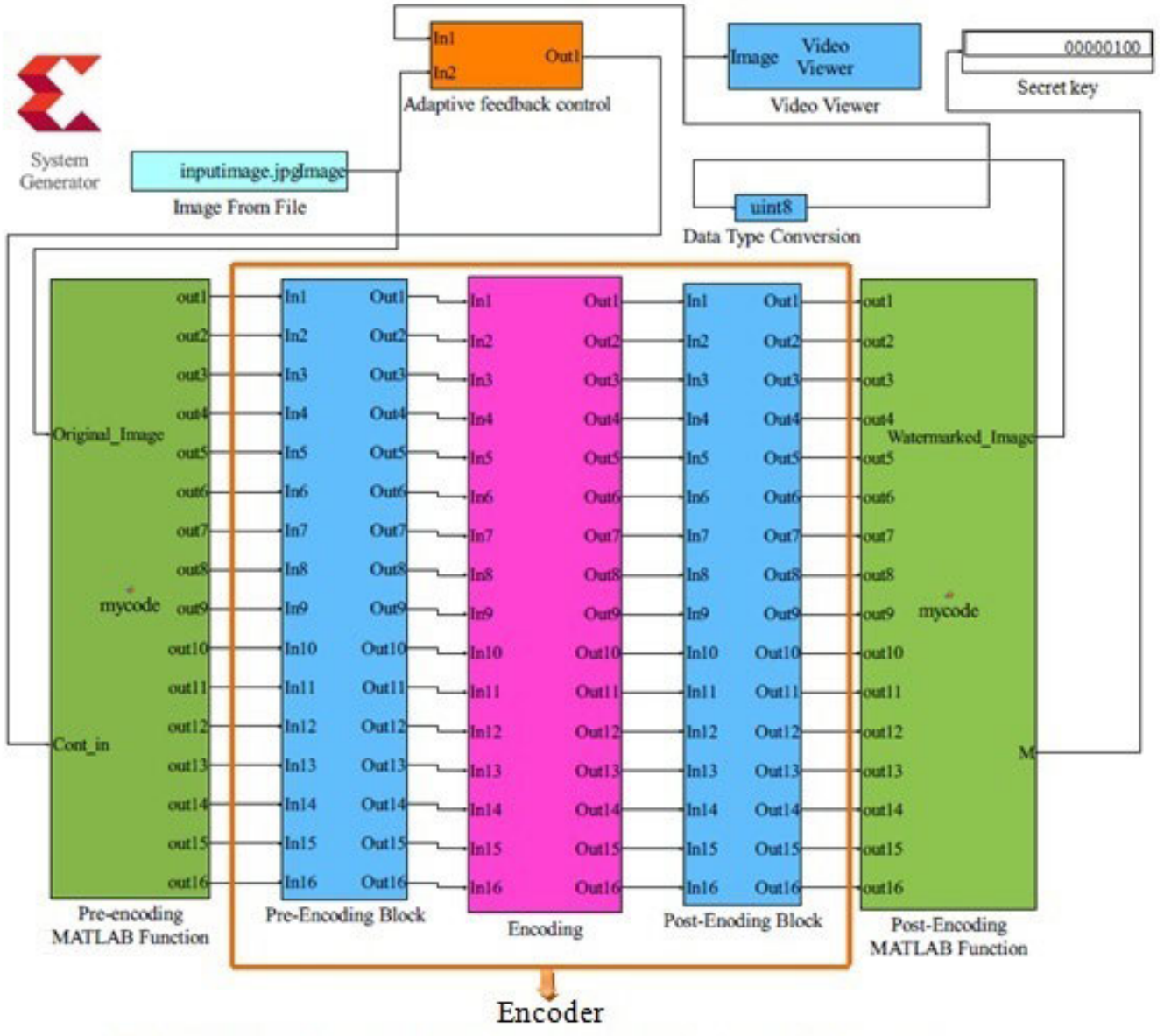

FIGURE 2. The basic building block of the encoding process.

The visual quality can be increased by increasing the value of PSNR above the range but to do this either the embedding capacity or embedding bit rate needs to be reduced below their desired range. This decreases the overall efficiency, superiority and performance of an algorithm for RIW which is practically not acceptable.

SSIM is another proficient choice to measure the superiority of an algorithm. SSIM is the measurement of the perceptual difference between two similar multimedia data like images. When both the original and watermarked images are purely same with each other, SSIM is equal to 1. In other words, bits and properties of original data have not been changed after embedding. We can say that embedding process has not happened. To maintain the desired range for PSNR, the SSIM between original and watermarked image should be varied in between 0.85 to 0.95. [1]–[6].

The second most important property of RIW is the embedding bit rate. Bit per pixel (bpp) is used as the unit of embedding bit rate. It provides the information regarding the number of bits that are changed during data embedding. The target value of embedding should be close to 4 intended for 8 bit gray-scale image data. It is found that the lower 4 bits perform the basic role of changing the embedding bit rate.

The data hiding capacity is the last foremost significance for RIW. It defines the total available space on original data for data embedding. If the full size of the original data is used for data embedding, the data hiding capacity is equal to 1. By decreasing the space with respect to the original

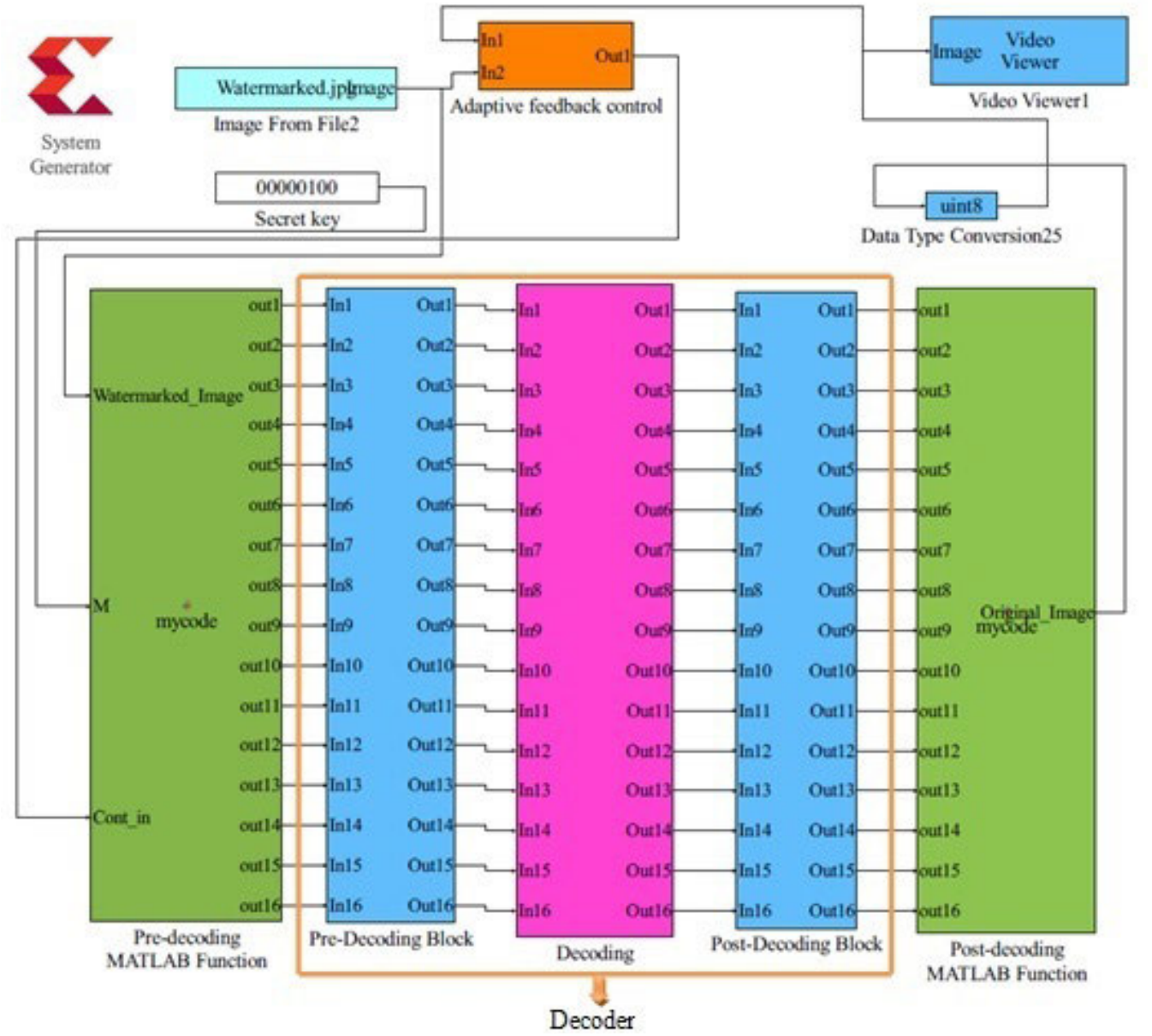

FIGURE 3. The basic building block of the decoding process.

data, the data hiding capacity will be reduced. The distortion can be controlled by varying the data hiding capacity through an algorithm. It is concluded that it is better to keep the average data hiding capacity in between 0.5 to 0.8. In other words it is the ratio between total number of embedding bits and total number bits of the input multimedia data. Here the total number of bits used for embedding process is known as payload size.

Some related works on RIW are discussed briefly. It is noted that most of these exiting algorithms are application specific. They have not followed the desired range for above mentioned propertied at all times. Lossless data compression by least significant bit (LSB) shifting [7] is early RIW algorithm that affords high embedding capacity under elevated mathematical complexity but the strength of encryption or authenticity is very low. While the histogram bin exchange [8] and its modified version [9]–[11] eliminated these drawbacks but suffer from low embedding capacity as well as low embedding bit rate. Alternatively to get better overall performance, the prediction error expansion (PEE) [12]–[16] is a preferable choice but it suffers from the requirement of large payload size for embedding prediction error information which in turn causes increased distortion. The most proficient approach to realize reversibility is to build up a linear transformation based lossless forward and reverse mapping of data. By following these concepts two well-known exiting works have been found. They are difference expansion (DE) [17]–[24] and reversible contrast

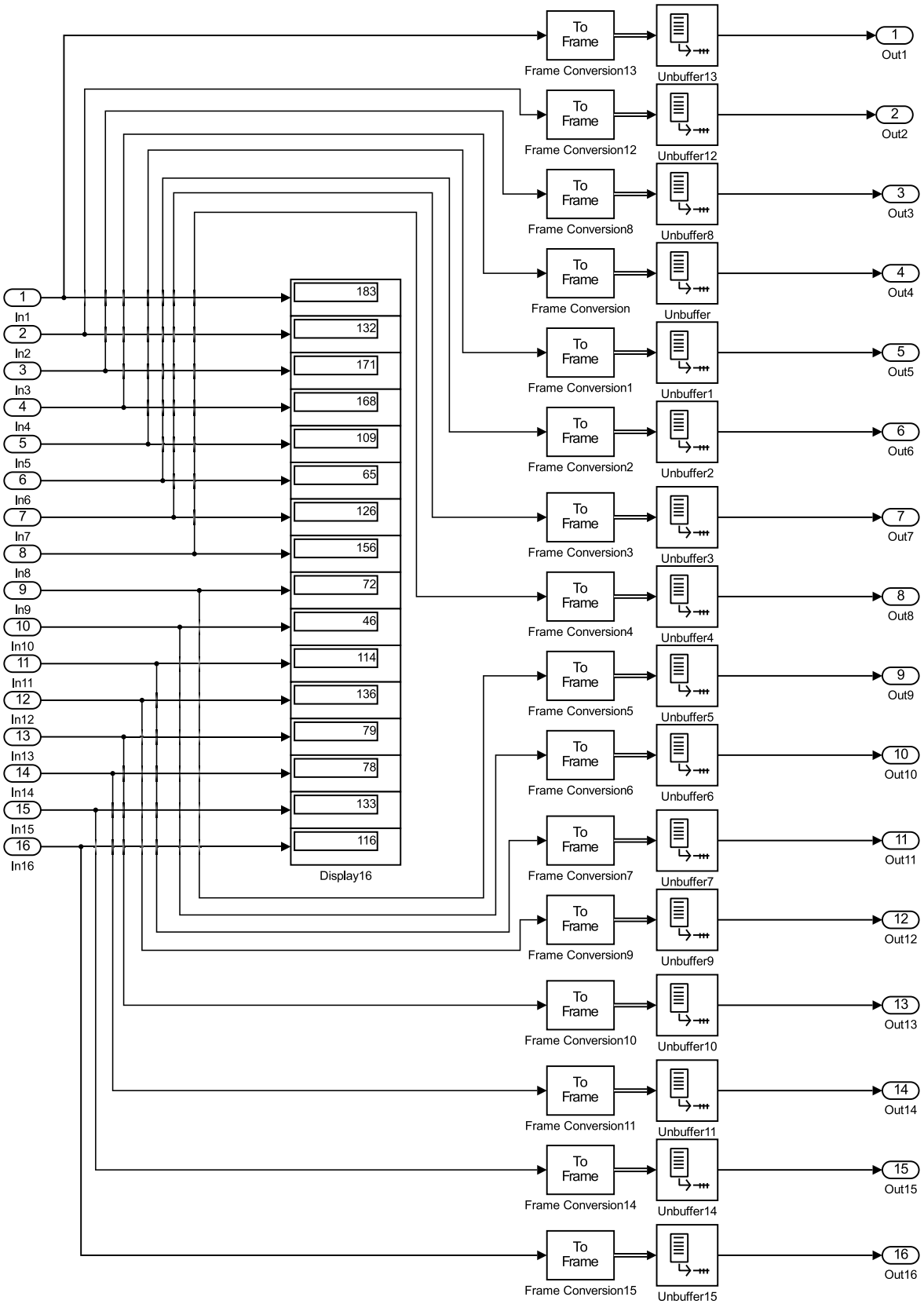

FIGURE 4. The inside structural model of pre-encoding function block.

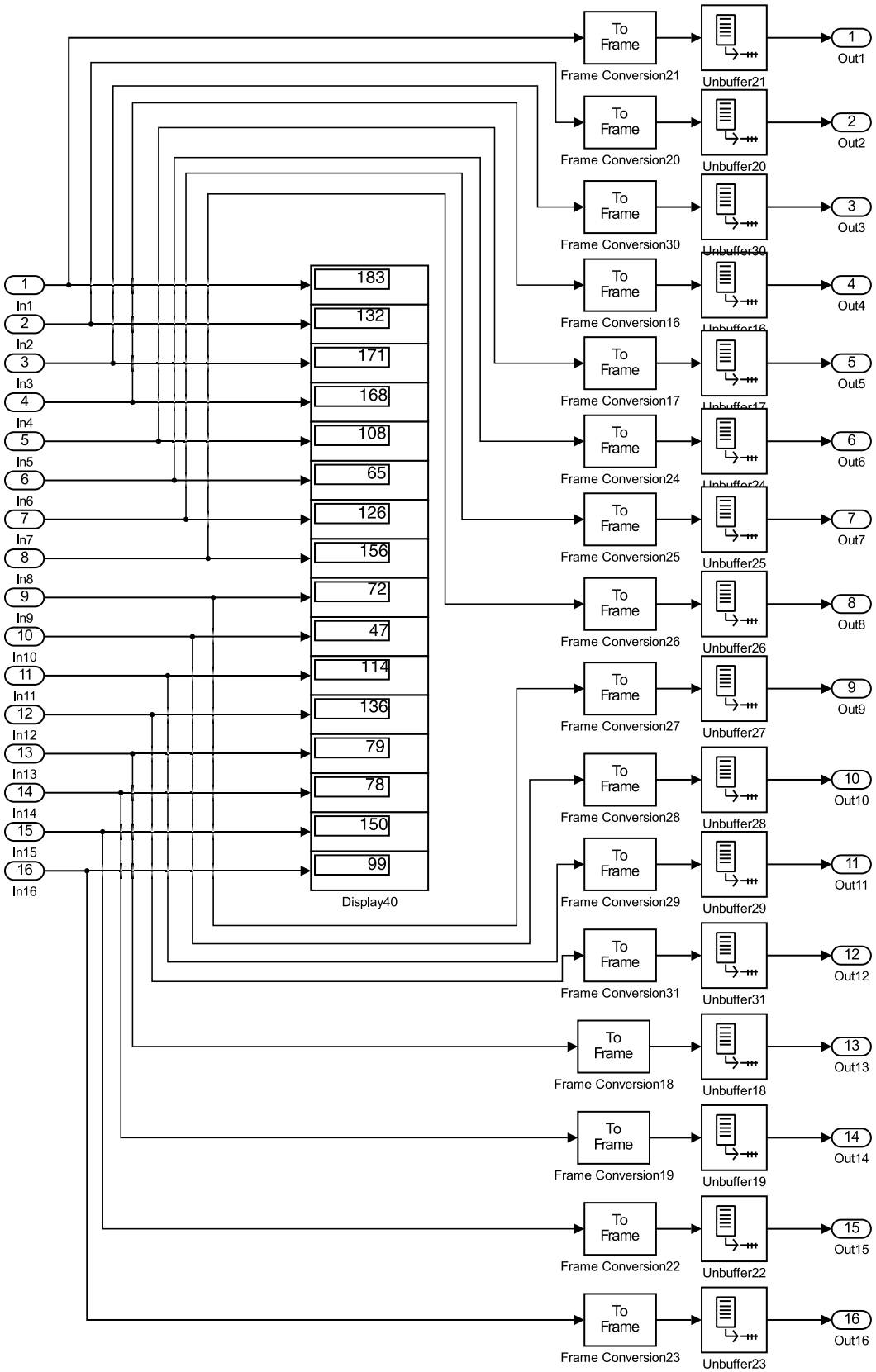

FIGURE 5. The inside structural model of pre-decoding function block.

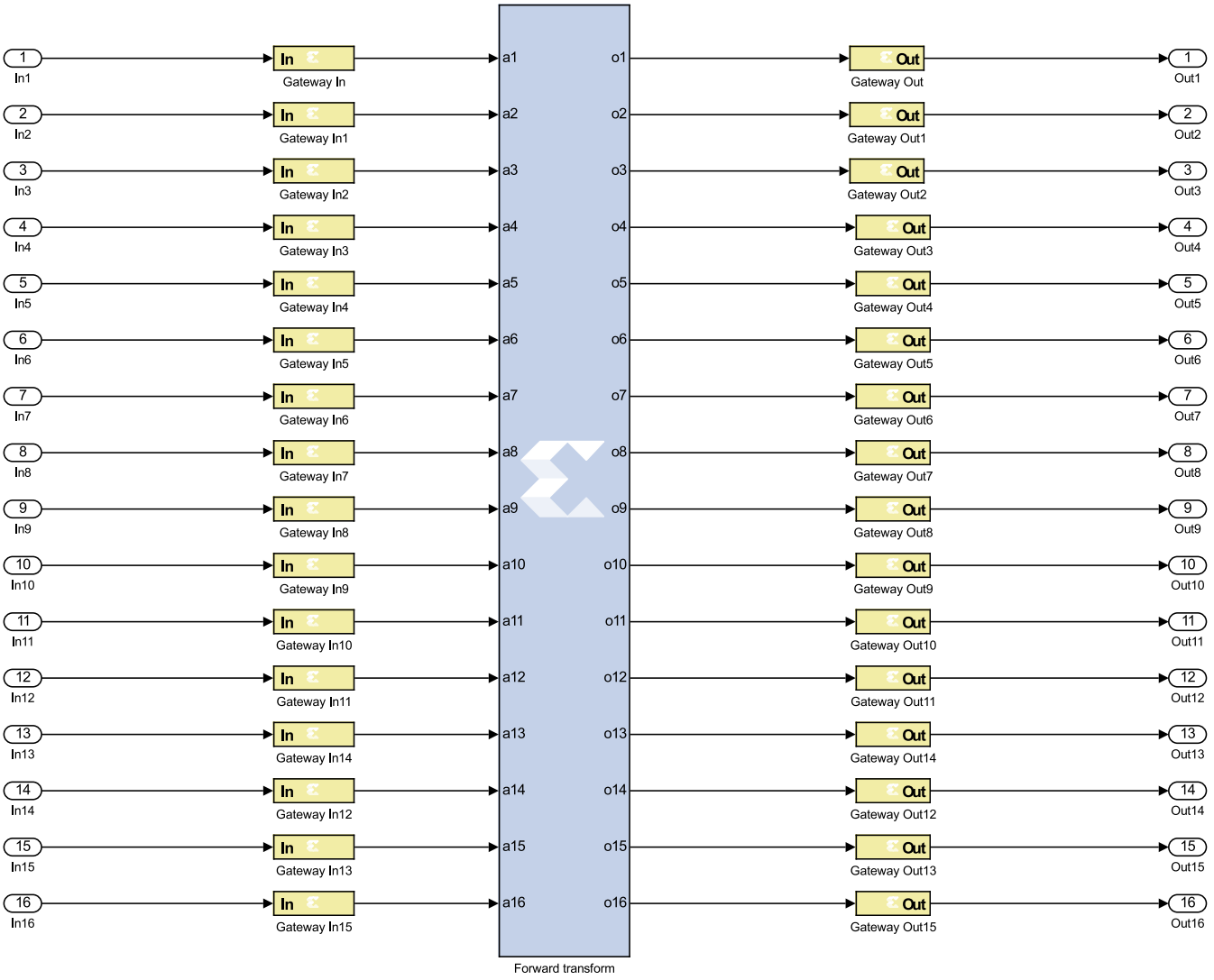

FIGURE 6. The inside structural model of encoding function block.

mapping (RCM) [25]–[28]. Both these spatial domain based algorithms improve the payload size as well as the embedding bit rate [23]–[28]. In DE algorithm, the linear transformation is applied only on the pixel intensity values that caused the need of location map control mechanism. In other hand to reduce the arithmetic complexity, the linear transformation in RCM algorithm is applied on both pixel intensity values as well as the pair of pixel locations. In the way under a low mathematical complexity RCM provides higher embedding bit rate than DE in single iteration. That makes RCM most attractive algorithm for RIW particularly for real-time implementation.

The need of hardware based real time implementation RIW is increasing day by day when various types of attacks have been considered. There are basically geometrical attacks in watermarking that affect the efficiency and superiority of an algorithm. These attacks happen during data transmission over unsecured medium. It can be classified into two types. They are local and global attacks. The local attack is mainly active on a part or portion of the whole data. In other hand, the global attack affects on whole multimedia data through pixel by pixel. To recover back both the cover and watermark data from attacked watermarked data an additional watermark synchronizer function can be used [29]. RIW is fragile in nature. Due to this reason software implementation of RIW is easily accessed and corrupted by attackers. Also software implementation cannot provide efficiency and superiority of an algorithm in terms of power, resources and simulation time. To make the algorithm robust in nature and improve the efficiency, real time hardware implementation is a most acceptable solution. The hardware implementations of DE, RCM, histogram bin exchange, wavelet transform and channel coding algorithms have been observed [30]–[34]. The comparative results between these and proposed algorithm based on hardware resources utilization are presented.

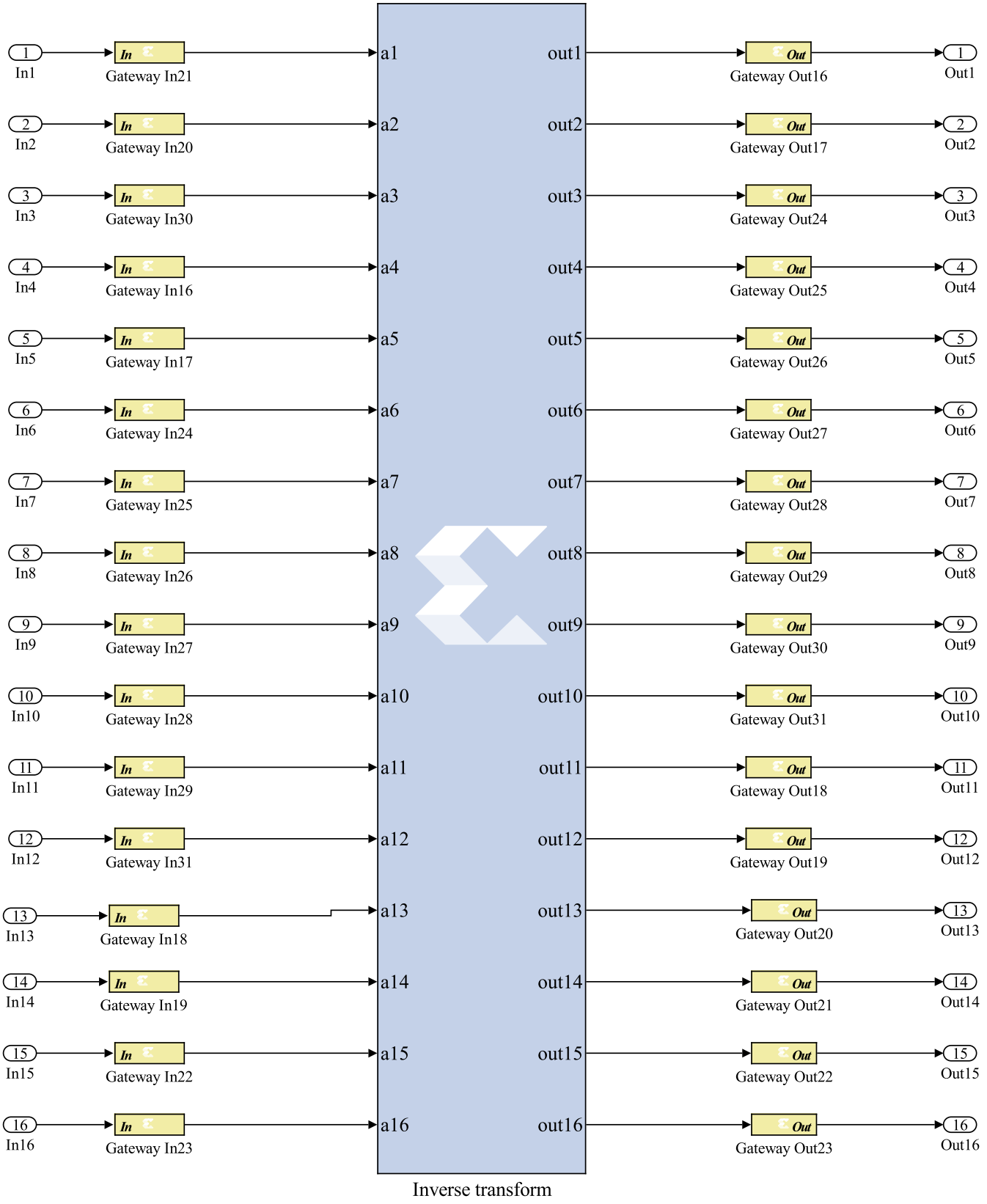

FIGURE 7. The inside structural model of decoding function block.

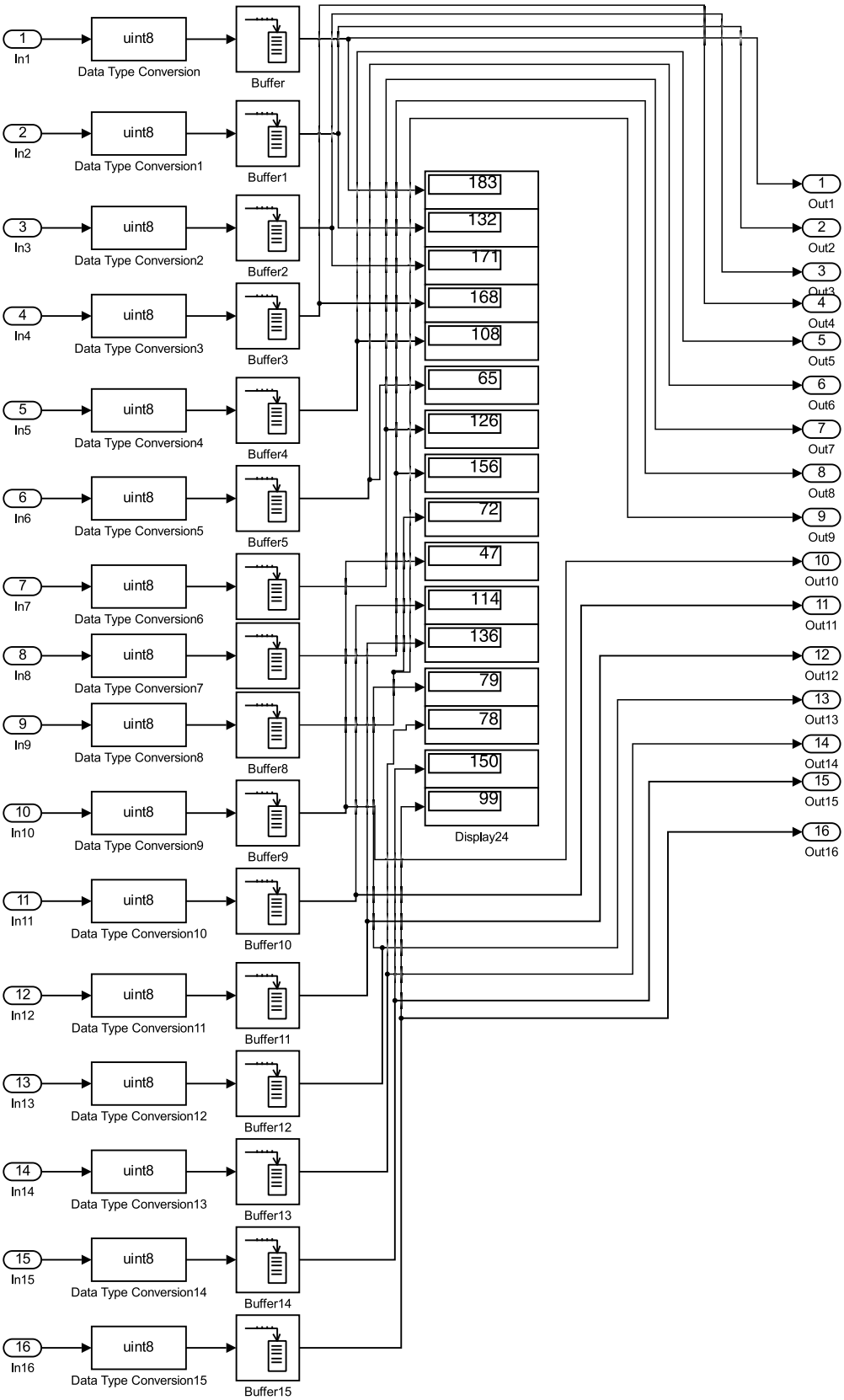

FIGURE 8. The inside structural model of post-encoding function block.

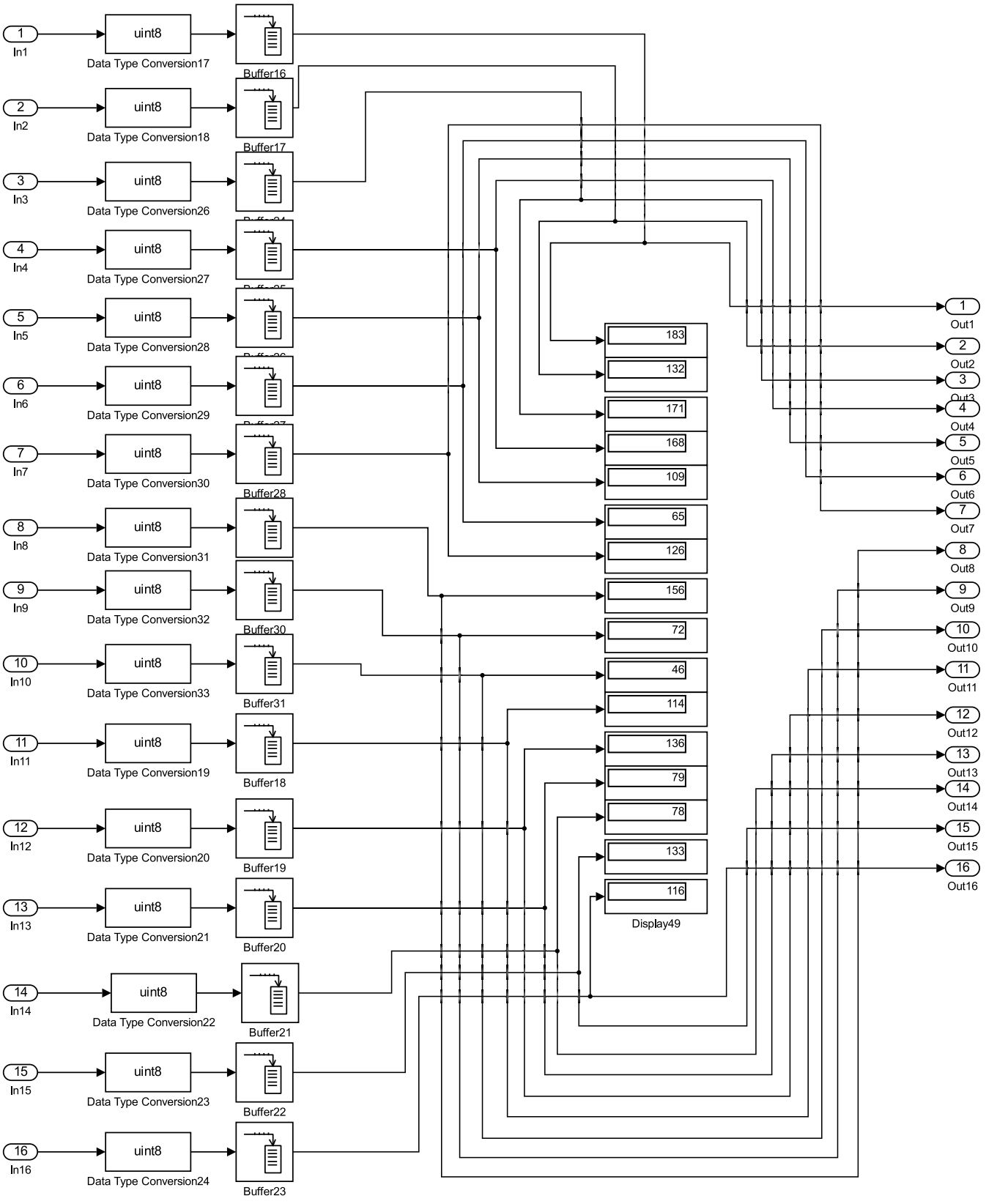

FIGURE 9. Inside structural model of post-decoding function block.

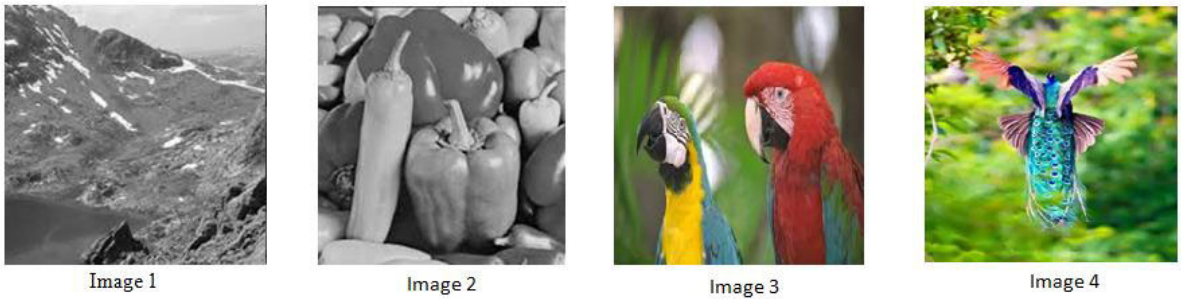

FIGURE 10. The four input test images.

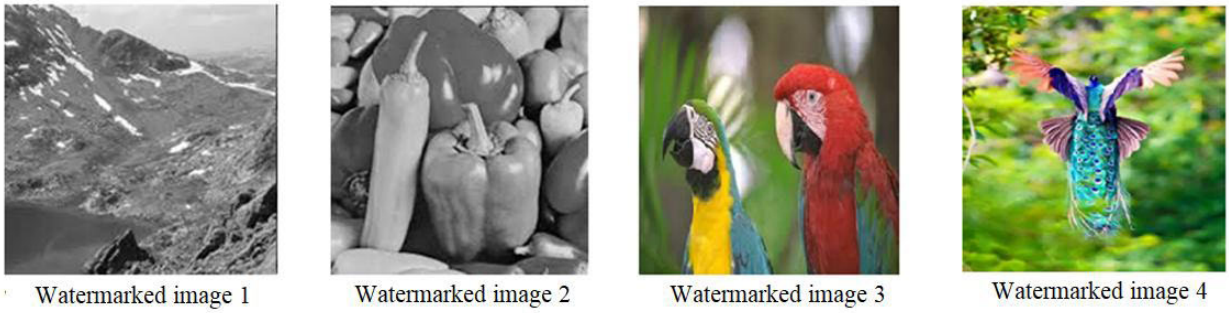

FIGURE 11. The watermarked images of the respective test images.

The major goal of this work is to propose a novel and efficient embedding bit rate control RIW algorithm which is adaptively based on well-known parameters using RCM algorithm. Given attention on the desired range of these parameters, modified linear transformations are used over selected neighbour pixels and verified in software environment. The outcome of the algorithm will provide a good visual quality watermarked image as well as the value of respective parameters will vary in their desired range. In this way, unlike application specific algorithms the proposed algorithm represents a novel algorithm for RIW. The FPGA based hardware verification and real time implementation of the proposed algorithm is also presented. The parallel processing based hardware implementation is also used to achieve high speed which is immensely needed for real-time applications.

The manuscript is prearranged as follows: The proposed EBCRCM algorithm is reported in section 2. The embedding and decoding process are explored in section 3. Section 4 and section 5 dispense the block diagrams to implement the VLSI architecture for proposed EBCRCM algorithm using XSG and VIVADO tool respectively. The analysis and innovative outcome results are discussed in the last section. This section also highlights the performance based on inevitability and comparative analysis with other algorithms.

II. THE PROPOSED EBCRCM BASED RW

In RCM algorithm [25], linear transforms are applied on pixels locations and selected pixel intensity values. The algorithm fails to maintain the desired range of all parameters. It leads to destroy the essential nature of RIW. The contrast is used as a constant and it is equal to 3 without depending on the parameters. Due to this reason most of the parameters values fall outside their desired range. Fig. 1 shows the control over payload capacity using a threshold value adaptively. The white space is used for further data embedding process. The performance can be improved by modifying the linear transform on pixel locations in RCM [35], [36]. However the implementation of the algorithm in real time is very difficult as it requires huge number of control units and finite state machines to control the embedding capacity over 2D multimedia data.

In proposed EBCRCM algorithm, the same forward linear transforms are used on pixel locations but modified forward transforms are applied on the pixel intensity values. The linear forward transforms are $i' = 2i - j$ and $j' = 2j - i$. Here (i, j) is the pair of pixel locations. These forward transforms are used to set the embedding space and payload capacity also. It can be controlled by a predefined threshold value as shown in Fig. 1. In EBCRCM algorithm the threshold value T is predefined and set to 20 as from the analysis it is found that the best possible PSNR within its desired range can be achieved. The equations of modified forward transformation are specified in (1) and (2). Here n is a constant which is used to control the contrast. The n is varied from 3 to 30. The n is bounded by a predefined contrast limit. The contrast limit is predefined based on the size of original image. For

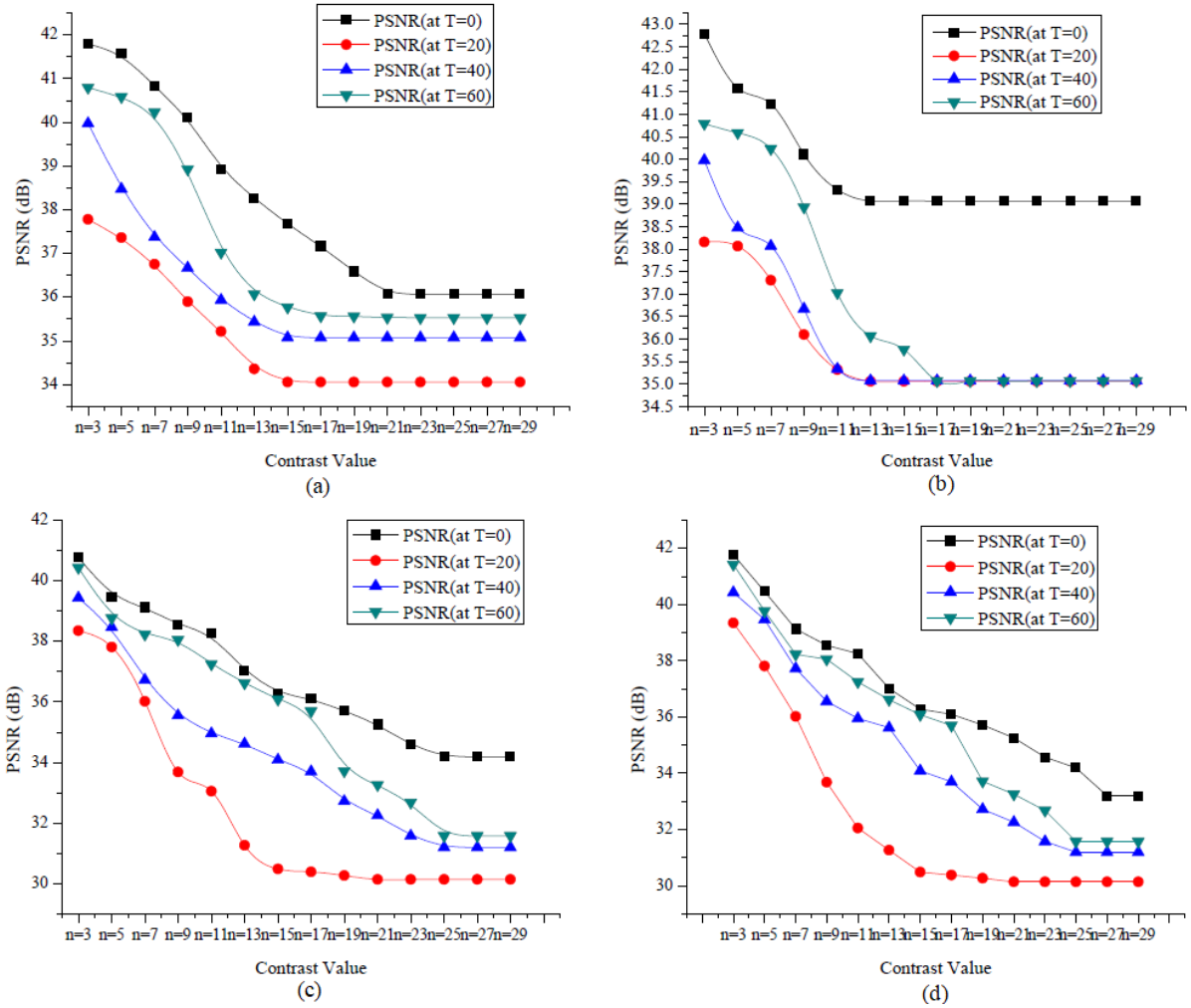

FIGURE 12. The graphical representation of PSNR with respect to the contrast value and threshold value for (a) Image 1, (b) Image 2, (c) Image 3 and (d) Image 4.

256 × 256 image the predefined contrast limit is 18. The gray level dynamic range of the input image is denoted as $[0, L]$. Here $L = 255$ for eight bits per pixel. The pixel intensity values of two selected neighbor pixels are p and q. The transformed intensity values of the pair (p', q') are given as follows:

$$p' = \left\lfloor \frac{n+1}{2} \right\rfloor p - \left\lfloor \frac{n-1}{2} \right\rfloor q \tag{1}$$

$$q' = \left\lfloor \frac{n+1}{2} \right\rfloor q - \left\lfloor \frac{n-1}{2} \right\rfloor p \tag{2}$$

The above two transformations should satisfy the following conditions to overcome the overflow and underflow. The two boundary conditions are

$$0 \le p' = \left\lfloor \frac{n+1}{2} \right\rfloor p - \left\lfloor \frac{n-1}{2} \right\rfloor q \le L \tag{3}$$

$$0 \le q' = \left\lfloor \frac{n+1}{2} \right\rfloor q - \left\lfloor \frac{n-1}{2} \right\rfloor p \le L \tag{4}$$

The modified inverse transform can be given by

$$p = \left\lfloor \left[\frac{n+1}{2n} \right] p' + \left[\frac{n-1}{2n} \right] q' \right\rfloor \tag{5}$$

$$q = \left\lfloor \left[\frac{n+1}{2n} \right] q' + \left[\frac{n-1}{2n} \right] p' \right\rfloor \tag{6}$$

Here a linear adaptive feedback control mechanism is asserted with the modified transformations. The adaptive feedback controls the distortion and embedding bit rate through the variation of the contrast. For an application, if distortion is not important then the data hiding capacity can be controlled adaptively by varying the predefined threshold T to get higher embedding bit rate which is shown in Fig. 1. Taking into consideration distortion as well as to implement less hardware resources based VLSI architecture, in EBCRCM algorithm the T is kept as a constant and the embedding bit rate is controlled by adaptively varying contrast value.

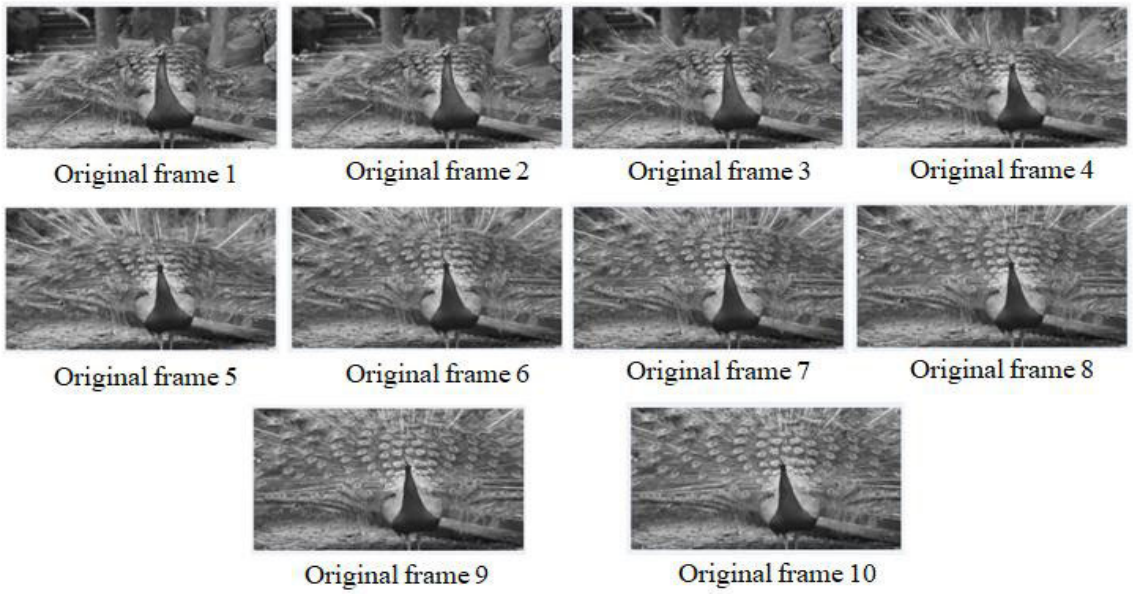

FIGURE 13. The original frames of the input video signal.

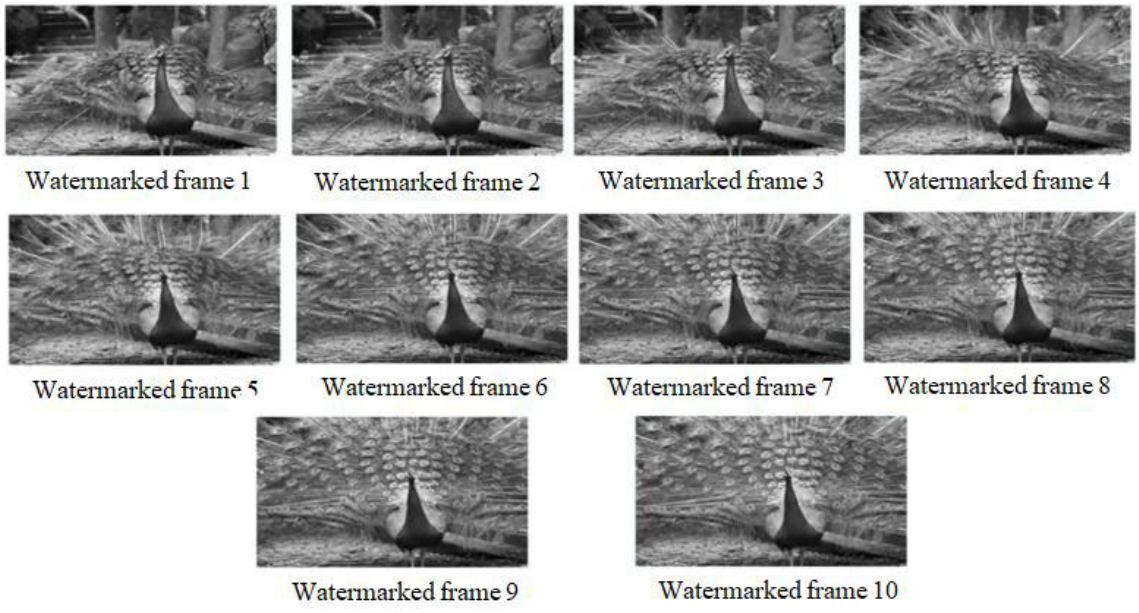

FIGURE 14. The watermarked frames of the input video signal.

III. OVERVIEW OF EBCRCM

EBCRCM method is verified by MATLAB R2015b using three types of multimedia data. They are gray-scale images, color images and video.

A. PROPOSED EBCRCM ALGORITHM FOR GRAY-SCALE IMAGE

The gray level intensity value of image pixel indicates the amount of light present on that particular pixel. It has 0 to 255 gray levels for 8 bits image. With some modification this algorithm is also applicable for color and video signals.

1) ALGORITHM 1: ENCODING PROCESS

Step 1: The original image is read and the predefined threshold T is defined. The contrast value n is set to 3.

Step 2: A pixel location value of image (i, j) is chosen and (i', j') is obtained. This will be derived from $i' = 2i - j$ and $j' = 2j - i$ followed by checking the conditions $1 < i' < sx$ and $1 < j' < sx$, where sx is image length. For 256×256

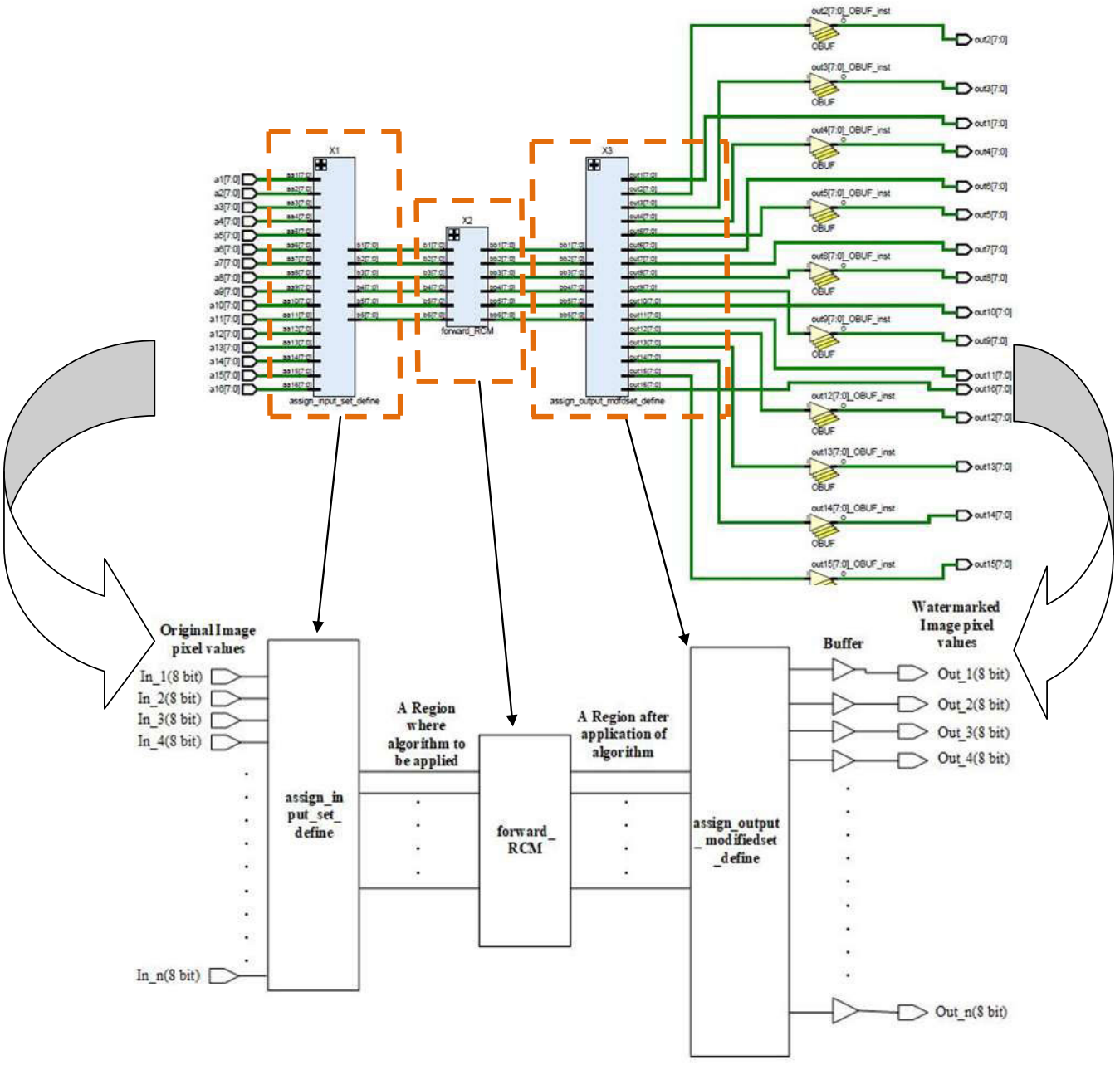

FIGURE 15. The VLSI architecture for Forward transform of encoding process.

image sx is equal to 256. The number of pixels in this region is counted and the pixel intensity values are stored.

Step 3: A pixel pair (p, q) from stored values is taken and checked whether $|n(p-q)| < T$. The intensity value of pixel locations p and q are denoted as m and n respectively. In this step, a location map M is created which will act as a secret key while retrieving the original image from the watermarked image. If condition $|n(p-q)| < T$ is satisfied and if both pixel intensity values m and n are odd then LSB of p is to be reset and q is kept unchanged. In this case, M is set to 0. Otherwise the pair is transformed according to (1) and (2). After transferring, the transformed pixel pair should follow the conditions which are given in (3) and (4). If it is satisfied then M is set to 1 otherwise it is set to 0. After getting the selected transformed pixel pairs by checking the boundary conditions in (3) and (4), the secret bit is then embedded into the LSB position of the second transformed pixel and transformation information or the location map M is embedded in the LSB of the first transformed pixel. The secret bits are randomly generated. At the end of this step, we will be able to find the transformed pair (p', q') and location array M.

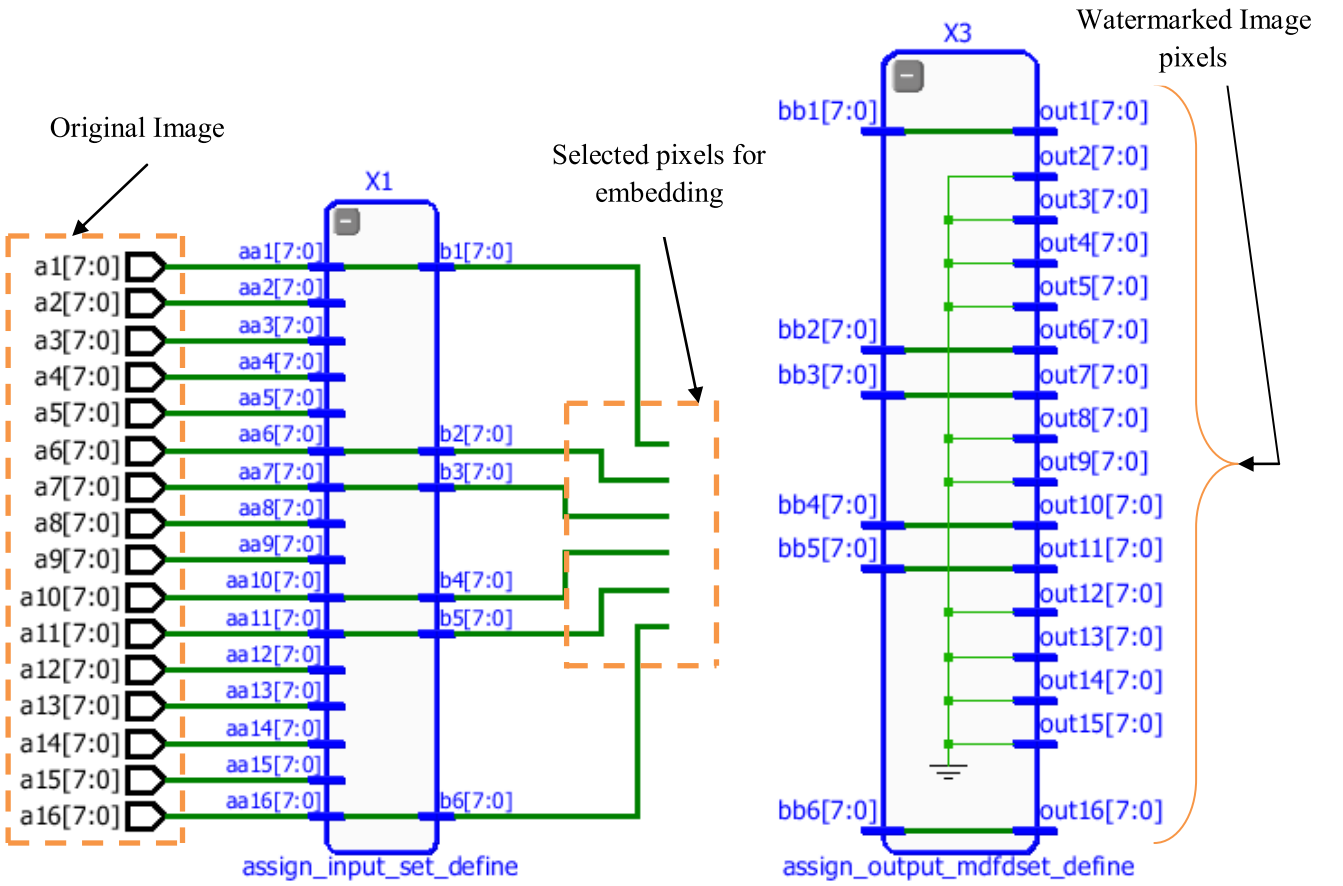

FIGURE 16. The VLSI architecture of assign input set define block and assign output modified set define block for encoder.

Step 4: If $|n(p-q)| > T$, the pixel pairs are kept unchanged.

Step 5: The unchanged pixel pairs are rearranged with transformed pairs (p', q') as the output watermarked pixel $W(p', q')$.

Step 6: To insert these transformed values in place of original values, pixel position values of image size (ii, jj) are taken. The (ii', jj') are obtained using the linear transforms $ii' = 2ii - jj$ and $jj' = 2jj - ii$ followed by checking the two boundary conditions, $1 < ii' < sx$ and $1 < jj' < sx$. If both conditions are satisfied then the value of transformed pair is to be inserted into original image. In this way the transformed image is generated from the original image.

Step 7: After getting watermarked image, the relative parameters are measured. If these parameters do not fall into desired range, the value of n is increased by 2 using a feedback control part. T is varied up to 20 as maximum difference of 16 between original and transformed pixel intensity value is taken to vary the contrast to maintain the distortion. When high PSNR within desired range is detected, we get the preferred watermarked image and randomly generated watermark bits. M is preserved as an encrypted key for the data extraction process and final n is preserved.

2) ALGORITHM 2: DECODING PROCESS

Step 1: The watermarked image is read and the predefined threshold T and location map matrix M and n are obtained.

Step 2: From pixel position values of image size (i', j') we get (i, j). This will be derived from the following equations: $i' = 2i-j$ and $j' = 2j-i$ followed by checking two conditions. They are $1 < i' < sx$ and $1 < j' < sx$, where sx is image length. The number of pixels of this region are counted and stored.

Step 3: The condition $|n(p'' - q'')| < T$ for the pixel pair (p', q') is checked. During this operation location matrix M and the secret key n is checked. If M for current set is 1 then the pair is transformed according to Eq. 8 and Eq. 9. If it is 0 then LSB of p' is set to 1 and q' is as it is.

Step 4: If $|n(p'' - q'')| < T$, the pixel pairs are kept unchanged.

Step 5: The transformed pair as well as the changes in (p', q') are stored as the original pixel pair (p, q).

Step 6: To insert these transformed values in place of watermarked values, pixel position values of size of image i.e. (ii', jj') are taken, (ii, jj) are obtained using $ii = \left\lfloor \frac{2}{3}ii' + \frac{1}{3}jj' \right\rfloor$ and $jj = \left\lfloor \frac{1}{3}ii' + \frac{2}{3}jj' \right\rfloor$ followed by checking the boundary conditions $1 < ii < sx$ and $1 < jj < sx$. If

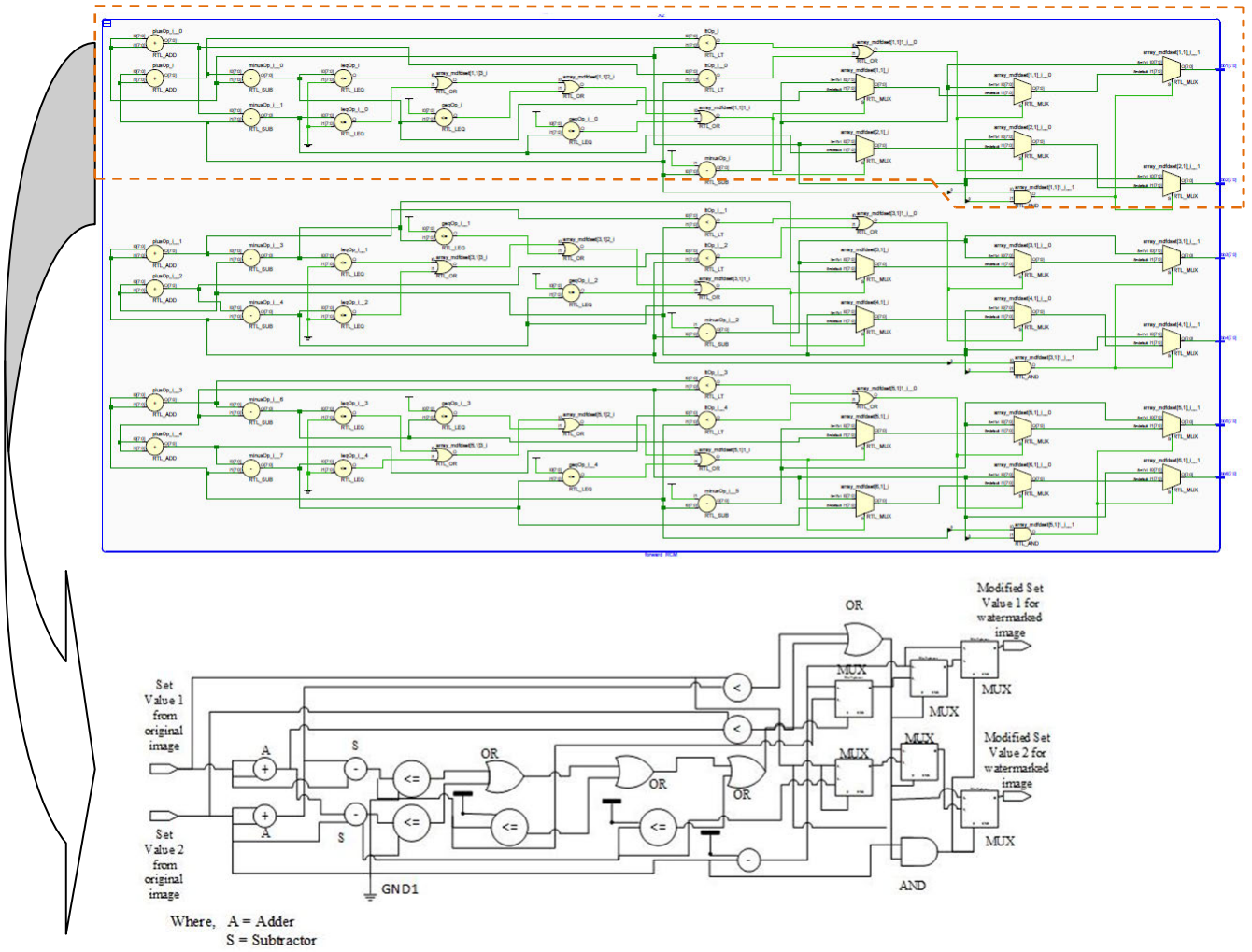

FIGURE 17. The inside schematic architecture of the forward RCM.

both conditions are satisfied then the value of transformed pair is to be inserted into watermarked image. In this way the original image is recovered back from the watermarked image.

3) PROPOSED EBCRCM ALGORITHM FOR THE COLOR IMAGE

To describe any particular color image, three independent quantities are required. RGB images are taken as the input images. The color image contains the combination of these three colors. Each intensity value has these three components. Each component is taken at a time and the encoding-decoding algorithm has been applied on it. At the end all three components are combined to collect the watermarked - original image. To convert three components to one we need to reshape the image using size vector. The remaining steps for the encoding and decoding will be the same as algorithm 1 and algorithm 2. After the conversion we need to restore the three component values such that we can get the color image.

4) PROPOSED EBCRCM ALGORITHM FOR THE VIDEO

The proposed EBCRCM is also verified using video signal. A video is a series of images or frames which is played in the sequence at a specified frame rate. We can process the video after converting it into frames. Each frame will work as an input image. After processing, the required outcomes which are in the form of images, they will be again converted into video which have same frame rate of the input video.

IV. THE BLOCK MODEL OF EBCRCM BASED RIW USING XSG

A 4×4 sized 8 bits gray image having 16 gray label intensity values as shown in (7) is considered. By taking two neighbors pixel locations there are total 16 cases. In other words the case 1 has been formed by taking the first two neighbor pixels [1, 1]. Then case 2 as [1, 2], case 3 as [1, 3] and so on. By applying liner transformation and the boundary conditions, (3) and (4) on the pixel locations, we found that 12 cases are satisfied. They are case 1, case 2, case 5, case 6, case 7, case 8, case 10, case 11, case 12, case 14, case 15 and

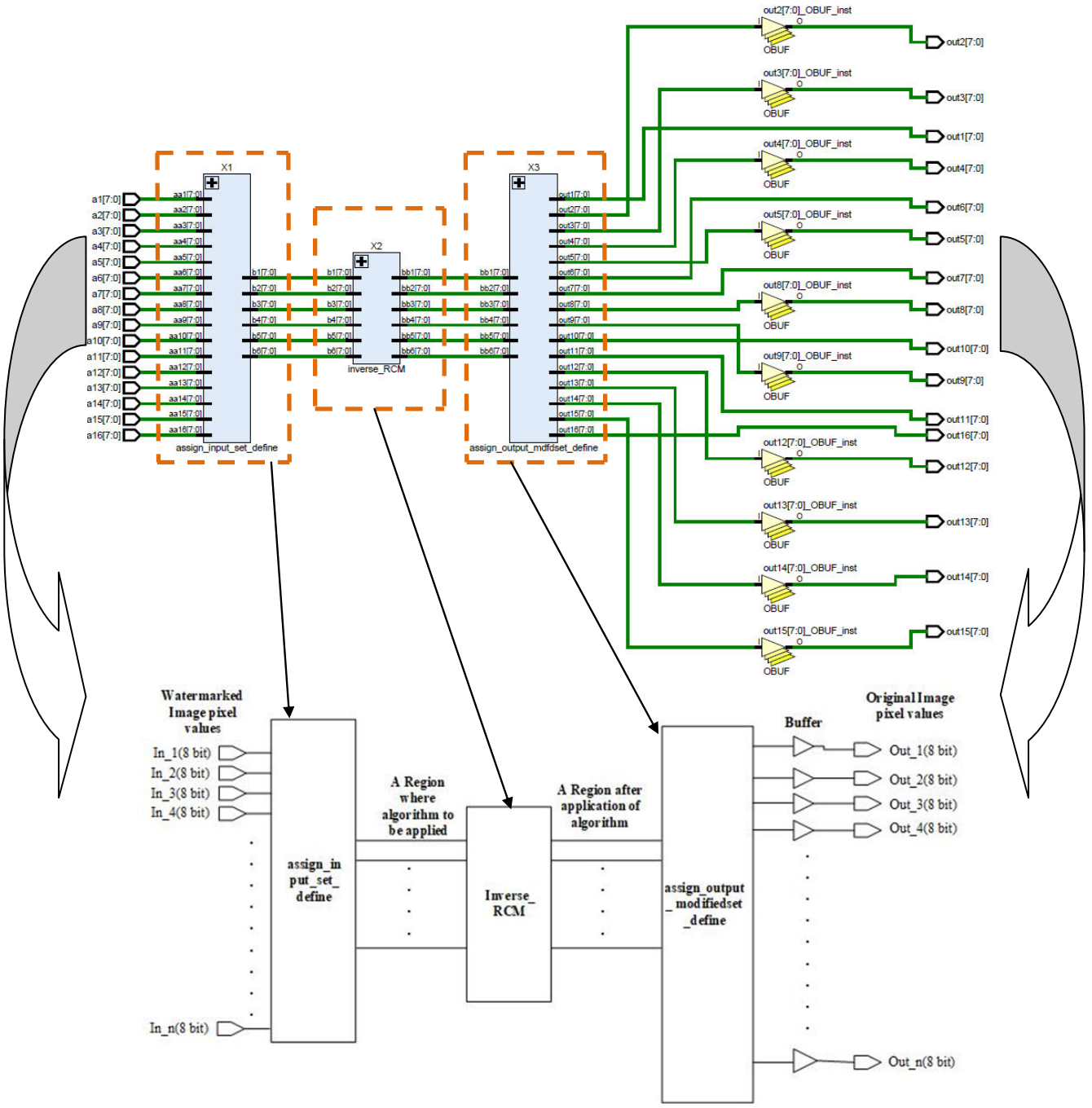

FIGURE 18. The VLSI architecture for Inverse transform of encoding process.

case 16. According to the proposed method, the pair of pixels for integer transformation is formed by taking the intensity values of those cases which are satisfied by the boundary conditions. As there are total 12 satisfied cases, 6 numbers pairs of pixels are available for transformation. The first pair of pixels is then formed by the intensity values of case 1 and case 2 and denoted as (183, 132). Similarly the second pair of pixels is (109, 65), third is (126, 156) and so on. The odd pixel pairs are avoided for transformation as the invertible property is not satisfied by them. For odd pair of pixels, the LSB of first pixel is set to 0. Here (109, 65) is an odd pair of pixels. Setting the LSB of first pixel with 0, the pair become (108, 65). After transformation of selected fourth pair of pixels (46,114), they have failed to satisfy the boundary conditions given in (3) and (4). According to the algorithm, in this case if the first pixel intensity value is odd, then LSB of it is set to 0 otherwise it is set with 1. The last selected pair of pixels is processed for transformations by

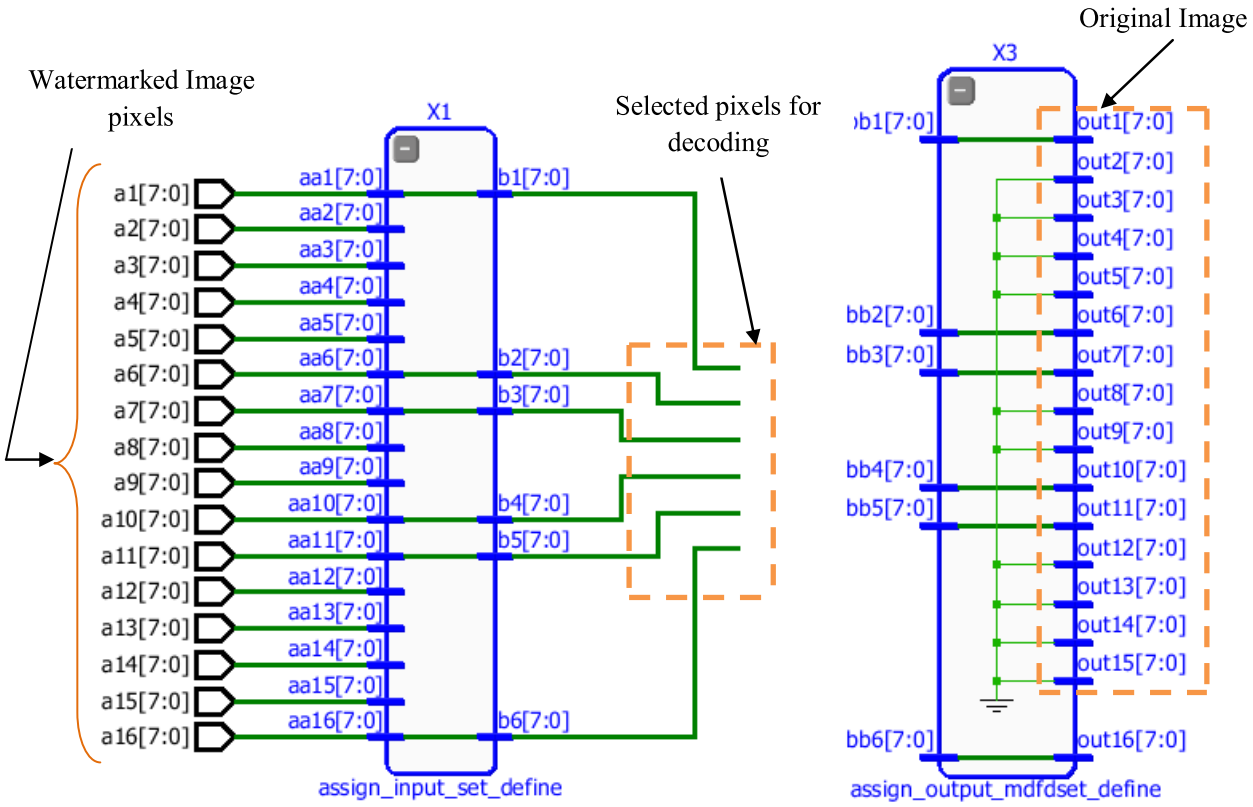

FIGURE 19. The VLSI architecture of assign input set define block and assign output modified set define block for decoder.

following (1) and (2).

$$A = \begin{bmatrix} 183 & 132 & 171 & 168 \\ 109 & 65 & 126 & 156 \\ 72 & 46 & 114 & 136 \\ 79 & 78 & 133 & 116 \end{bmatrix} \tag{7}$$

According to the transformation a secret key is created and the value n is preserved. In the similar way, the watermarked pixel values for remaining pair of pixels have been calculated. By following the boundary conditions to overcome the underflow and overflow problems given in (3) and (4), the watermarked image is denoted as W. The resultant watermarked gray level intensity values are shown in (8)

$$W = \begin{bmatrix} 183 & 132 & 171 & 168 \\ 108 & 65 & 126 & 156 \\ 72 & 47 & 114 & 136 \\ 79 & 78 & 150 & 99 \end{bmatrix} \tag{8}$$

Similarly for the decoding process, six pairs of pixels have been considered by taking the intensity values of selected neighborhood pixels from watermarked image. Based on M and n, the LSB is changed prior to inverse modified transform. The first pair of pixels p and q is obtained by modified inverse transformations given in (5) and (6). This modified inverse integer transforms are applied for the remaining cases to find out the decoded image. It has been found that the gray level intensity values of the decoded image are exactly same as the original cover image. It is concluded that the proposed algorithm preserves the reversibility properties by recovering the original cover image without any loss of information. We verified the distortion control method and satisfied with the conclusions of [25]. The relative parameters are checked using the feedback control path. The value of n is increased by 2 and repeated until it meets the goal.

The basic block model diagram of XSG contains several functional blocks named image from file, encoder, decoder and video viewer. The first function is used to store the input cover image. The data type and size of the cover image are set to uint8 and (4×4) respectively. The basic block model diagram of embedding and decoding processes are shown in Fig. 2 and Fig. 3 respectively. Initially the pre-processing blocks for both encoder and decoder have been passed through an m-code block to obtain serial order input data. The pre encoding and decoding MATLAB function blocks perform the conversion process of 2D to 1D data. In other words the 4×4 sized image data is stored as (1×16) 1D array form. Whereas revert operation is performed by post-encoding and post-decoding MATLAB function blocks. The first sub-blocks of encoder and decoder are named as pre-processing of encoder and decoder respectively as shown in Fig. 4 and Fig. 5 respectively. The mode of the output is set to sampling mode by 'frame conversion' function block.

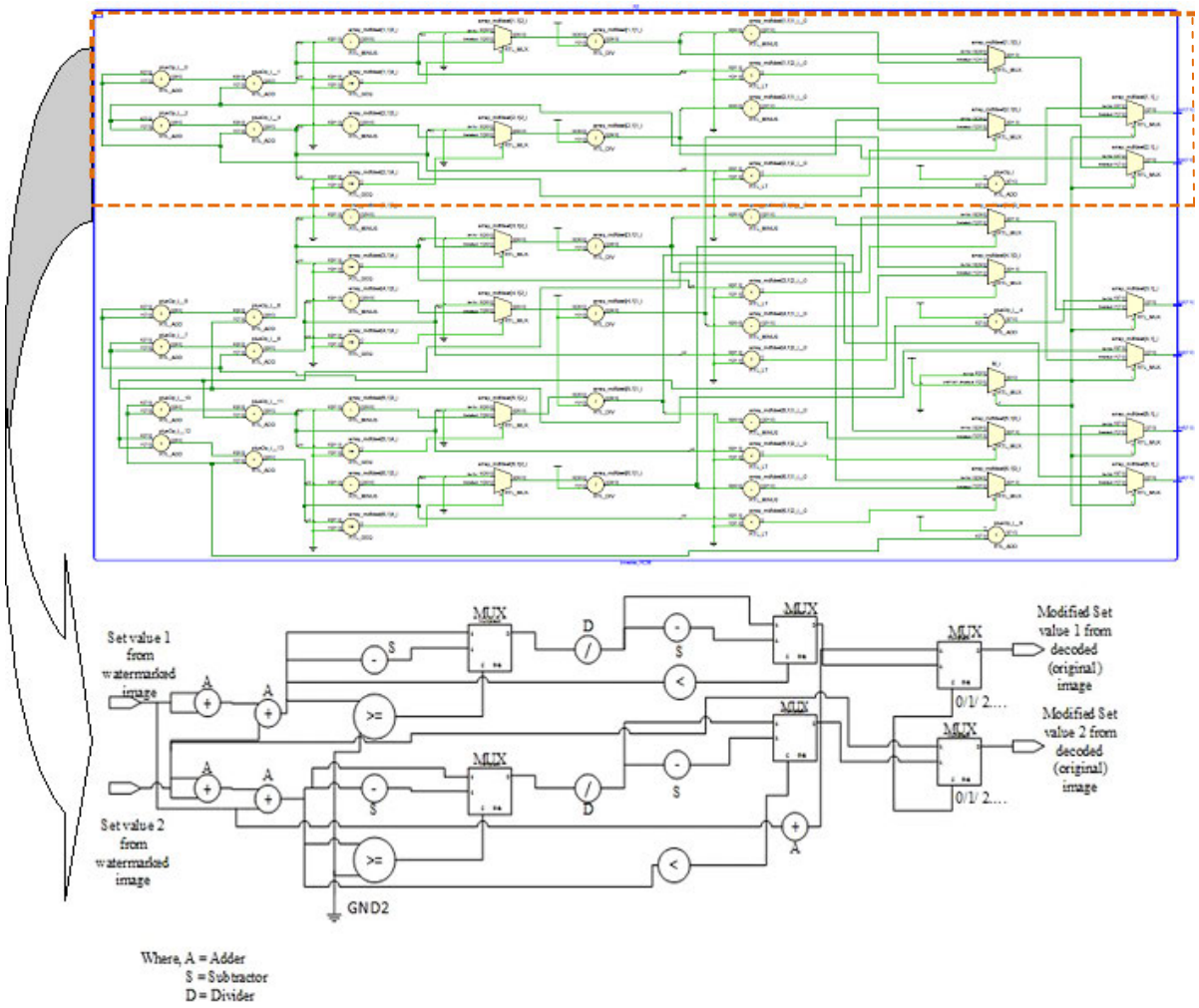

FIGURE 20. The VLSI architecture of assign input set define block and assign output modified set define block.

The last steps of preprocessing block is to un-buffer the input data into row wise which have been done by 'un-buffer' function block.

In XSG, hardware description language (HDL) codes are read through the function block named ''black box'' for processing the arithmetic operations. The black box for embedding and decoding processes are named as ''Forward transform'' and ''Inverse transform'' respectively as in Fig. 6 and Fig. 7. Here two main function blocks named 'Gateway In' and 'Gateway Out' are used in between the preprocessing and post-processing blocks for connection between Xilinx FPGA based environment and the Simulink model.

The last sub-block is called post-processing for both embedding and decoding parts. The working principle of the post-processing block is exactly opposite of the preprocessing sub-block. The post-processing sub-block for embedding is presented in Fig. 8. And the post-processing sub-block for decoding is presented in Fig. 9. From Fig. 8, we notice that the outcome intensity values of the watermarked image are accurately identical with calculated intensity values of the watermarked image which has been revealed in (8). Fig. 9 shows the intensity values of decoded image pixels after decoding process. We notice that they are identical with the intensity values of the original input cover image which have been shown in (7). So we can say that the algorithm preserves the properties of reversible watermarking.

V. RESULT AND ANALYSIS

In this section software and hardware implementation of the proposed method are presented. If n is represented as the number of bits then the unit of PSNR is dB and it can be given by $PSNR = 10\log_{10}\frac{(2^n-1)^2}{\sqrt{MSE}}$ and SSIM can be defined as $SSIM(f_1, f_2) = L(f_1, f_2)C(f_1, f_2)S(f_1, f_2)$ [6]. Here L is the luminance comparison function, C is the contrast comparison function and S is the structural comparison function. $L = \frac{2\mu_{f_1}\mu_{f_2}+C_1}{\mu_{f_1}^2+\mu_{f_2}^2+C_1}$, $C = \frac{2\sigma_{f_1}\sigma_{f_2}+C_2}{\sigma_{f_1}^2+\sigma_{f_2}^2+C_2}$ and $S(f_1, f_2) = \frac{\sigma_{f_1 f_2}+C_3}{\sigma_{f_1}\sigma_{f_2}+C_3}$. The same sized reference and original multimedia data are f_1 and f_2 respectively. The luminance of the two data are denoted

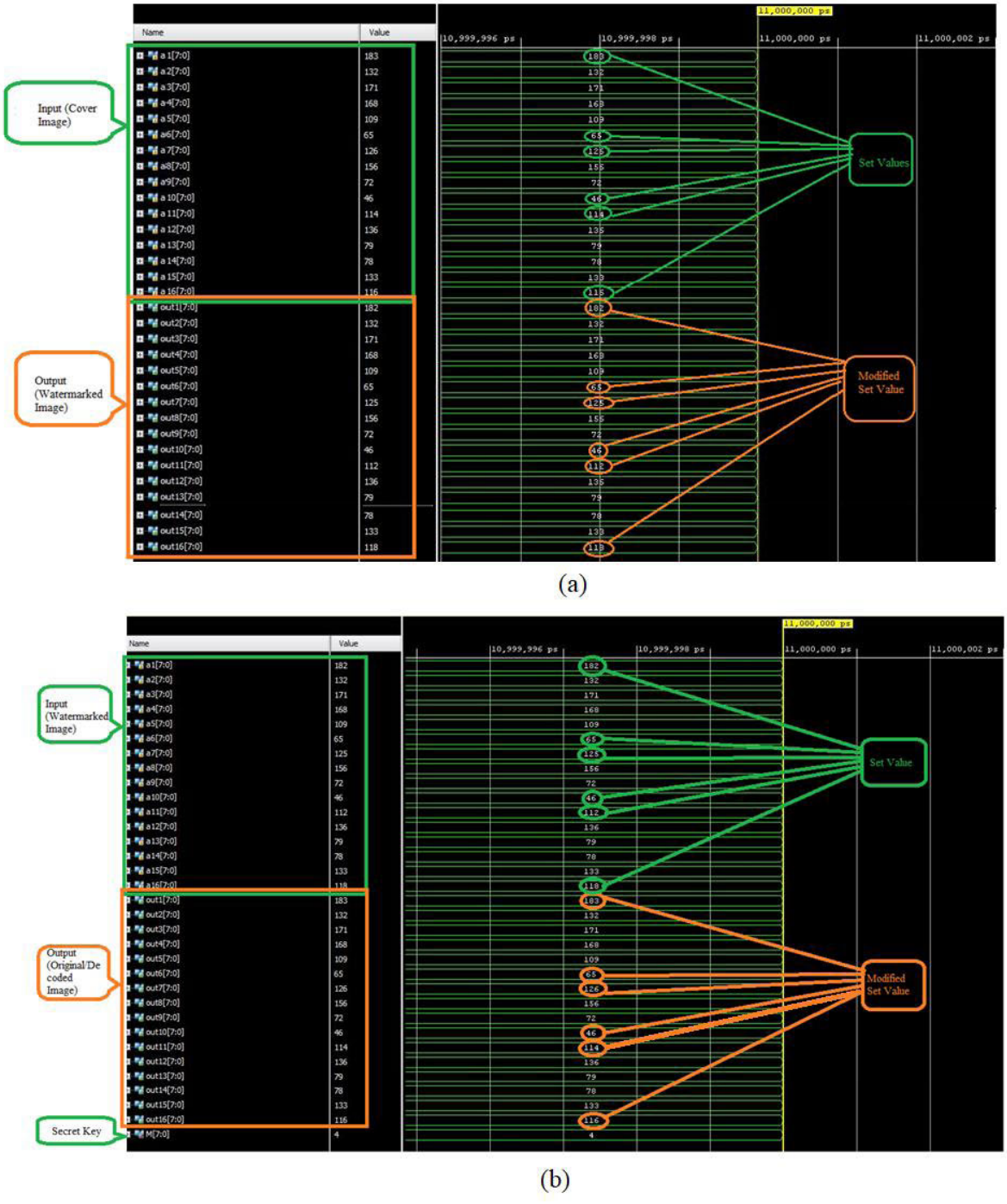

FIGURE 21. The VLSI architecture of assign input set define block and assign output modified set define block.

as μ_{f_1} and μ_{f_2} respectively. σ_{f_1} and σ_{f_2} are the standard deviation of these respective data. $\sigma_{f_1 f_2}$ is used to represent the covariance of these two input data. C_1, C_2 and C_3 are positive constants and used to avoid the null denominator.

A. SOFTWARE IMPLEMNENTATION

The illustration quality of the watermarked image and the quantity of the concealed information can be obtained by PSNR and SSIM by taking 40 number of test images. Two gray images and two color images are in Fig. 10. The gray images are indicated as Image 1 and Image 2 and the color images as Image 3 and Image 4. Corresponding watermarked images of the four test images are in Fig. 11. They are denoted as watermarked image 1, watermarked image 2, watermarked image 3 and watermarked image 4. The comparative analysis between PSNR with variation in contrast at a constant

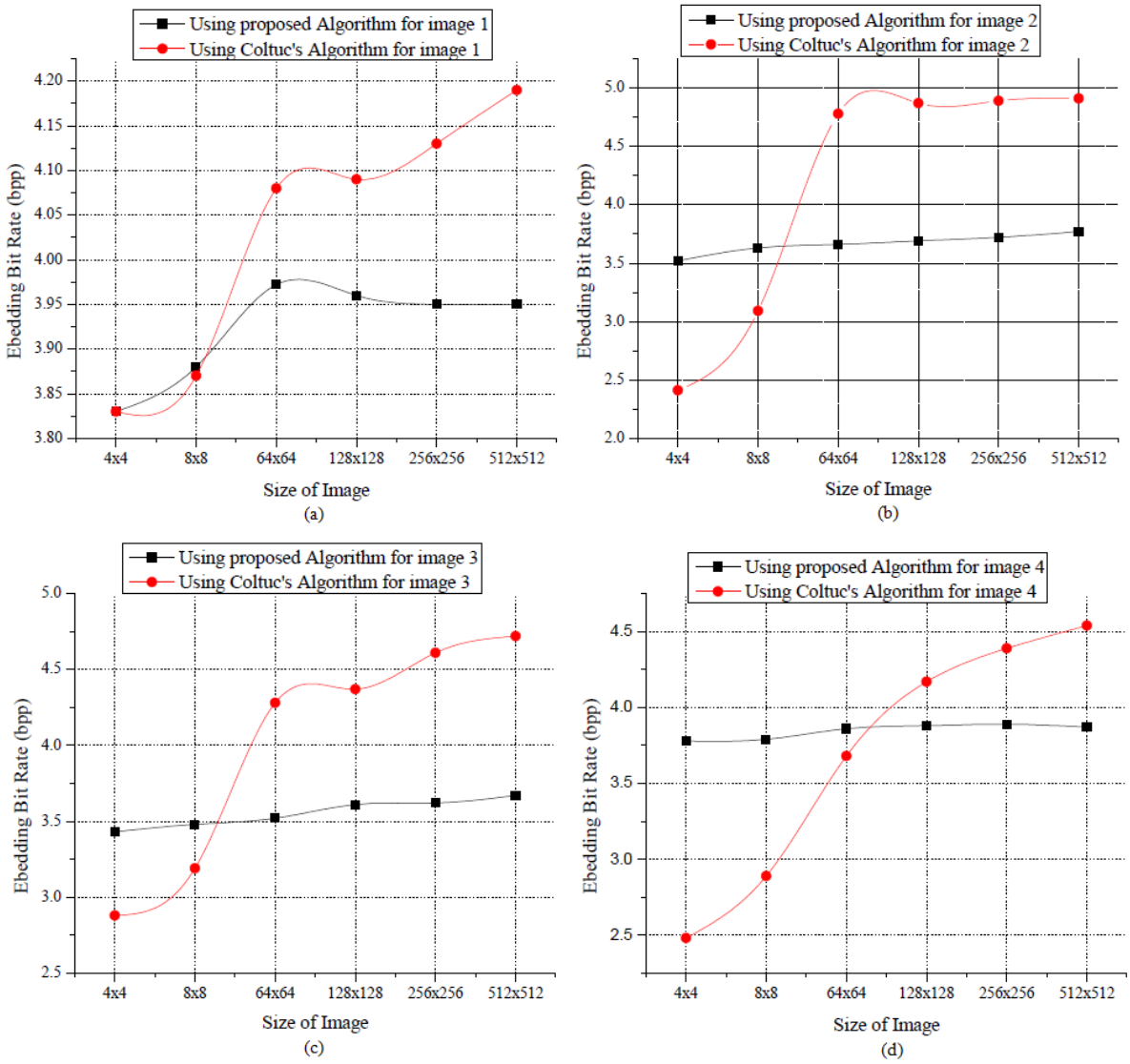

FIGURE 22. The graphical representation of PSNR with respect to the contrast value and threshold value for (a) Image 1, (b) Image 2, (c) Image 3 and (d) Image 4.

threshold for each test image is shown in Fig.12. From Fig. 12(a) and Fig. 12(b) it is observed that for the gray scale images, the desired PSNR based watermarked images are found at n = 3 and n = 7 at T = 20. Whereas for the color images, the desired watermarked images are found at n = 5 at T = 20. So it is concluded that it is better to process the proposed algorithm by taking constant threshold T = 20.

1) VIDEO SIGNAL VERIFICATION

A video signal is verified by taking a frame at a time. Figure 13 shows the original ten frames of an input gray scale video signal. From original frames we get the watermarked frames after encoding as shown in Fig. 14. It has an average PSNR value of 37.6583 dB and SSIM value between original and watermark frame is 0.7445. Whereas SSIM value between original and decoded frame is 1.

B. HARDWARE IMPLEMNENTATION

The VLSI architecture can be obtained from structural modeling in VIVADO 2016.2 design suite. The hardware co-simulation is carried out at Zynq-7000 (Zed-board) technology with XC7Z030 based target device.

1) ENCODER

The VLSI architecture of forward transform for encoding function block of XSG is shown in Fig.15. It has three components named as assign input set define block, forward RCM block and assign output modified set define block. The inputs of the first component are the pixel values of the input image which are coming from the Gateway In via pre-processing encoding block of XSG. In this block the incoming values are converted into an array to find the set values of the input image pixels which are applicable for embedding process.

After converting the 1D pixel value to the array, the region of the conversion is to be decided and the selected set values will be the inputs of the second component. The Register transfer logic (RTL) layout of the assign input set define block is shown in Fig. 16. This component is used to determine the selected pixels those are compatible for embedding processing. Here only six selected pixels are processed though the output ports of this component.

The assign output modified set define block works reverse of input block. This block will take transformed set value which is converted by the forward RCM block using proposed EBCRCM algorithm. In this block transformed set value will be restored in to 1D array. This array is again rearranged with remaining disapproved set of pixels to get the resultant watermarked image.

The heart of this VLSI architecture is the forward RCM block. This component applies the modified forward transforms on the selected pixels followed by the boundary conditions. The schematic architecture of the forward RCM is given in Fig. 17. It will take two pixel intensity values at a time for processing. It is found that for processing a single case, it requires 2 adders, 3 sub-tractors, 5 basic logic gates, 6 comparators and 6 multiplexers. The output of this block gives the transformed set of pixel values that will act as inputs to the assign output modified set define block.

2) DECODER

The inverse transform block of XSG consists of three sub-blocks named as assign input set define block, inverse RCM block and assign output modified set define block. The VLSI architecture of decoder is shown in Fig. 18. The working principles of first and third components of decoder are same as the first and third components of the encoder. The VLSI architectures of assign input set define block and assign output modified set define block for decoder are shown in Fig. 19.

Like embedding process, the heart of the decoder's VLSI architecture is Inverse RCM block. The layout of the Inverse RCM block is in Fig. 20. The input of the block will be set value which is defined by first component. It is found that to process a single case it requires 5 adders, 4 subtractors, two dividers, 4 comparators and 6 multiplexers. A little bit higher resources are required for the decoding process as compared with encoding process.

After full synthesis operation the comparative micro-statistic results between encoder and decoder is given in Table 1. The device consumption result for different sizes of input image is given in Table 2.

The simulation outcome for encoding is shown in Fig. 21 (a) where the timing waveforms and the values of input cover image, output watermarked image and the secret key are given away. The selected and transformed sets of pixel pairs are marked by green and orange colours respectively. Fig. 21(b) indicates the simulation results of input watermarked image and output decoded image of original cover image. In Fig. 21(b) the selected set of pixels pairs for inverse transform are marked by green colour and the resultant set of pixel pairs after transformation are marked by orange colour. For encoding process a 4 × 4 sized gray scale image, 392 numbers of slice LUT (look up table) with 481 numbers of LUT–flip-flop pairs are used.

After synthesis 44 numbers of bonded IOBs are used. A LUT–input-output pair represents one LUT paired with one input-output within a slice. The critical path is formed by adding the delay of six multiplexers (0.451 ns × 6), one maximum delay among the resources (multiplier: 2.352 ns, adder, subtractor, divider) and the register delay (0.487 ns + 0.255 ns). Thus, the maximum path delay for single cycle is 5.8 ns. For entire embedding process, we require 7 cycles, so the latency is 5.8 ns × 7 cycles × 4.27 ns = 173.362 ns with a 100MHz clock. The average fan-out of non-clock nets used is 1. The throughput obtained is the inverse of latency multiplied by the number of bits. The throughput is 46.146 Mbps for a pixel of 8 bits.

TABLE 1. The micro statistics result of encoder process.

Name of the Process	Cover image size	Resources	Numbers of resources
ENCODER	(4×4)	Comparator (32 bit)	18
		Adder/Subtractor (8 bit)	15
		Divisor	0
		Accumulator (10 bit)	15
		Multiplexer (1 bit 64:1-1; 8 bit 4:1-2)	20
		Latches (1 bit-2)	4
		Register (1 bit -32:8 bit-120)	45
		Tristate (1 bit-1,8 bit-2)	9
		Block RAMs (32 × 1 bit-1, 64 × 8 bit-2)	3
DECODER	(4×4)	Counter (5 bit-1:6 bit-1)	6
		Comparator (32 bit)	12
		Adder/Subtractor (8 bit)	27
		Divisor	6
		Accumulator (10 bit)	18
		Multiplexer (1 bit 64:1-1; 8 bit 4:1-2)	20
		Latches (1 bit-2)	7
		Register (1 bit -32:8 bit-120)	54
		Tristate (1 bit-1,8 bit-2)	12
		Block RAMs (32 × 1 bit-1, 64 × 8 bit-2)	4
		Counter (5 bit-1:6 bit-1)	6

TABLE 2. The device consumption results.

Quantity of	4 × 4		64 × 64		128 × 128	
	E	D	E	D	E	D
BRAMs	3	4	3	4	3	4
Slice Flip-Flop	392	377	439	493	587	743
4 input LUTs	834	1142	862	1213	875	1221
Bonded IOBs	44	44	44	44	44	44
Slices	481	762	526	711	754	931
GCLKs	1	1	1	1	1	1

*E and D are stands for Encoding and Decoding process

TABLE 3. The device consumption results.

Proposed by	Platform of the research work	Device	Operational frequency
Mohanty et al. (2009) [5]	Spatial domain	Virtex, XCV50BG256-6	545.39 MHz
Karthigaikumar et. al (2011) [32]	Connectivity preserving criteria	Virtex-E (xcv50e-8-cs144)	82 MHz
Hazra et. al. (2018) [31]	Histogram bin shifting	Virtex, XC2VP30, Spartan-3E, XC6slx45tfgg484-3	445.82 MHz
Phadikar et. al. (2019) [33]	DWT lifting	Xilinx Zynq (XC7Z020-CLG484-1)	80 MHz
Phadikar et. al. (2019) [34]	DCT	Virtex 7 (XC7VX330T-FFG1157-3)	60 MHZ
Proposed Work	Reversible Contract Mapping	Xilinx, Zynq, Zed-board, XC7Z030.	100 MHz

TABLE 4. The device consumption results.

Proposed by	Image Size	Power(mW)	Throughput
Mohanty et al. (2009) [5]	512×512	2.054	NA
Karthigaikumar et. al (2011) [32]	128×128	113.24	36.14 Mb/s
Hazra et. al. (2018) [31]	256×256	1.319	NA
Phadikar et. al. (2019) [33]	512×512	78.48	23.827 Mb/s
Phadikar et. al. (2019) [34]	512×512	78.51	1.34 GB/s
Proposed Work	256×256	36.82	46.146 Mb/s
	512×512	39.21	52.724 Mb/s

C. COMPARATIVE ANALYSIS

Comparative analysis between proposed and one of the most referred algorithm [25] based on embedding bit rate is explored. From Fig. 22 it is noticed that for proposed method the embedding bit rate is controlled through adaptive linear transformation in such way so that it always vary in it desired range. In this way the distortion on the watermarked image is maintained for fulfilling the requirements of RIW process. Comparison of hardware implementations for different watermarking algorithms based on operation frequency is exposed in Table 3.

The comparative analysis of hardware implementations for different watermarking algorithms based on power and throughput is given in Table 4. From Table 4 it is concluded the due to high throughput the proposed algorithm required very less power to complete both encoding and decoding process. It is also demonstrated that comparatively less hardware resources are required for proposed EBCRCM algorithm.

VI. CONCLUSION

In this paper a new embedding bit rate control based contrast mapping algorithm is proposed for reversible image watermarking. The proposed method is verified using MATLAB and Xilinx system generator tools. Less hardware resources and power based VLSI structural design for real time implementation in FPGA environment is presented. The Zynq-7000 (Zed-board) technology with XC7Z030 is used to develop the architecture. A little bit higher resources are required for decoder with a different pipeline cycle. The EBCRCM is compared with other well-known methods based on the value of power, throughput and operational frequency. The proposed architecture can operate at a frequency of 100MHz. It is also concluded that Xilinx system generator and VIVADO based HDL tool can be worked alongside to reduce the overall complexity and hardware resources for implementing the EBRCM based RIW algorithm through Zynq-7000 processing device.

REFERENCES

[1] K. Thongkor, T. Amornraksa, and E. J. Delp, "Digital watermarking for camera-captured images based on just noticeable distortion and Wiener filtering," *J. Vis. Commun. Image Represent.*, vol. 53, pp. 146–160, May 2018.
[2] X. Y. Wang, Q. L. Shi, S. M. Wang, and H. Y. Yang, "A blind robust digital watermarking using invariant exponent moments," *AEU-Int. J. Electron. Commun.*, vol. 70, no. 4, pp. 416–426, Apr. 2016.
[3] N. M. Makbol and B. E. Khoo, "A new robust and secure digital image watermarking scheme based on the integer wavelet transform and singular value decomposition," *Digit. Signal Process.*, vol. 33, pp. 134–147, Oct. 2014.
[4] S. P. Mohanty, E. Kougianos, and N. Ranganathan, "VLSI architecture and chip for combined invisible robust and fragile watermarking," *IET Comput. Digit. Techn.*, vol. 1, no. 5, pp. 600–611, 2007.
[5] S. P. Mohanty, N. Memon, and K. S. Chatha, "Circuits and systems for real-time security and copyright protection of multimedia," *Comput. Electr. Eng.*, vol. 35, no. 2, pp. 231–234, Mar. 2009.
[6] A. Hore and D. Ziou, "Image quality metrics: PSNR vs. SSIM," in *Proc. 20th Int. Conf. Pattern Recognit.*, Aug. 2010, pp. 2366–2369.
[7] M. U. Celik, G. Sharma, A. M. Tekalp, and E. Saber, "Lossless generalized-LSB data embedding," *IEEE Trans. Image Process.*, vol. 14, no. 2, pp. 253–266, Jan. 2005.
[8] B. Chen, W. Zhang, and N. Yu, "Reversible watermarking in JPEG images based on modified RZL codes and histogram shift," *Wuhan Univ. J. Natural Sci.*, vol. 18, no. 2, pp. 126–132, Apr. 2013.
[9] C.-C. Lin, W.-L. Tai, and C.-C. Chang, "Multilevel reversible data hiding based on histogram modification of difference images," *Pattern Recognit.*, vol. 41, no. 12, pp. 3582–3591, Dec. 2008.
[10] W. Zhang, X. Hu, X. Li, and N. Yu, "Recursive histogram modification: Establishing equivalency between reversible data hiding and lossless data compression," *IEEE Trans. Image Process.*, vol. 22, no. 7, pp. 2775–2785, Jul. 2013.
[11] T. Zong, Y. Xiang, and I. Natgunanathan, "Histogram shape-based robust image watermarking method," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2014, pp. 878–883.
[12] L. Luo, Z. Chen, M. Chen, X. Zeng, and Z. Xiong, "Reversible image watermarking using interpolation technique," *IEEE Trans. Inf. Forensics Security*, vol. 5, no. 1, pp. 187–193, Mar. 2010.
[13] H.-C. Wu, C.-C. Lee, C.-S. Tsai, Y.-P. Chu, and H.-R. Chen, "A high capacity reversible data hiding scheme with edge prediction and difference expansion," *J. Syst. Softw.*, vol. 82, no. 12, pp. 1966–1973, Dec. 2009.
[14] W. Hong, T.-S. Chen, and C.-W. Shiu, "Reversible data hiding for high quality images using modification of prediction errors," *J. Syst. Softw.*, vol. 82, no. 11, pp. 1833–1842, Nov. 2009.
[15] D. Coltuc, "Improved embedding for prediction-based reversible watermarking," *IEEE Trans. Inf. Forensics Security*, vol. 6, no. 3, pp. 873–882, Sep. 2011.
[16] Y. Huo, S. Xiang, S. Liu, X. Luo, and Z. Bai, "Reversible audio watermarking algorithm using non-causal prediction," *Wuhan Univ. J. Natural Sci.*, vol. 18, no. 5, pp. 455–460, Oct. 2013.
[17] J. Tian, "Reversible data embedding using a difference expansion," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 8, pp. 890–896, Aug. 2003.
[18] A. M. Alattar, "Reversible watermark using the difference expansion of a generalized integer transform," *IEEE Trans. Image Process.*, vol. 13, no. 8, pp. 1147–1156, Aug. 2004.

[19] D. M. Thodi and J. J. Rodriguez, "Expansion embedding techniques for reversible watermarking," *IEEE Trans. Image Process.*, vol. 16, no. 3, pp. 721–730, Mar. 2007.
[20] X. Wang, C. Shao, X. Xu, and X. Niu, "Reversible data-hiding scheme for 2-D vector maps based on difference expansion," *IEEE Trans. Inf. Forensics Security*, vol. 2, no. 3, pp. 311–320, Sep. 2007.
[21] H. Joong Kim, V. Sachnev, Y. Qing Shi, J. Nam, and H.-G. Choo, "A novel difference expansion transform for reversible data embedding," *IEEE Trans. Inf. Forensics Security*, vol. 3, no. 3, pp. 456–465, Sep. 2008.
[22] Y. Hu, H.-K. Lee, and J. Li, "DE-based reversible data hiding with improved overflow location map," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 2, pp. 250–260, Feb. 2009.
[23] C.-C. Lin, S.-P. Yang, and N.-L. Hsueh, "Lossless data hiding based on difference expansion without a location map," in *Proc. Congr. Image Signal Process.*, vol. 2, May 2008, pp. 8–12.
[24] I.-C. Dragoi and D. Coltuc, "Local-prediction-based difference expansion reversible watermarking," *IEEE Trans. Image Process.*, vol. 23, no. 4, pp. 1779–1790, Apr. 2014.
[25] D. Coltuc and J.-M. Chassery, "Very fast watermarking by reversible contrast mapping," *IEEE Signal Process. Lett.*, vol. 14, no. 4, pp. 255–258, Apr. 2007.
[26] W. Hong, J. Chen, and T.-S. Chen, "Blockwise reversible data hiding by contrast mapping," *Inf. Technol. J.*, vol. 8, no. 8, pp. 1287–1291, Aug. 2009.
[27] P. Fei, C. Li, and L. Min, "A reversible watermark scheme for 2D vector map based on reversible contrast mapping," *Secur. Commun. Netw.*, vol. 6, no. 9, pp. 1117–1125, Sep. 2013.
[28] Y.-H. Kao, W.-B. Lee, T.-Y. Hsu, C.-Y. Lin, H.-F. Tsai, and T.-S. Chen, "Data perturbation method based on contrast mapping for reversible privacy-preserving data mining," *J. Med. Biol. Eng.*, vol. 35, no. 6, pp. 789–794, Dec. 2015.
[29] V. Licks and R. Jordan, "Geometric attacks on image watermarking systems," *IEEE Multimedia Mag.*, vol. 12, no. 3, pp. 68–78, Jul. 2005.
[30] S. Das, R. Maity, and N. P. Maity, "VLSI-based pipeline architecture for reversible image watermarking by difference expansion with high-level synthesis approach," *Circuits, Syst., Signal Process.*, vol. 37, no. 4, pp. 1575–1593, Apr. 2018.
[31] S. Hazra, S. Ghosh, S. De, and H. Rahaman, "FPGA implementation of semi-fragile reversible watermarking by histogram bin shifting in real time," *J. Real-Time Image Process.*, vol. 14, no. 1, pp. 193–221, Jan. 2018.
[32] P. Karthigaikumar and K. Baskaran, "FPGA and ASIC implementation of robust invisible binary image watermarking algorithm using connectivity preserving criteria," *Microelectron. J.*, vol. 42, no. 1, pp. 82–88, Jan. 2011.
[33] A. Phadikar, G. K. Maity, T.-L. Chiu, and H. Mandal, "FPGA implementation of lifting-based data hiding scheme for efficient quality access control of images," *Circuits, Syst., Signal Process.*, vol. 38, no. 2, pp. 847–873, Feb. 2019.
[34] A. Phadikar, H. Mandal, and T.-L. Chiu, "Parallel hardware implementation of data hiding scheme for quality access control of grayscale image based on FPGA," *Multidimensional Syst. Signal Process.*, vol. 31, no. 1, pp. 73–101, Jan. 2020.
[35] D. Maiti, S. P. Maity, and H. Maity, "Modification in contrast mapping: Reversible watermarking with performance improvement," in *Proc. Int. Conf. Signal Process. Commun. (SPCOM)*, Jul. 2012, pp. 1–5.
[36] S. P. Maity and H. K. Maity, "Optimality in distortion control in reversible watermarking using genetic algorithms," *Int. J. Image Graph.*, vol. 17, no. 3, Jul. 2017, Art. no. 1750013.

SUBHAJIT DAS received the B.Tech. degree in electronics and communication engineering from Mizoram University (A Central University), Aizawl, India, in 2013, and the M.Tech. degree in embedded system from Sambalpur University, Sambalpur, India, in 2015. He is currently pursuing the Ph.D. degree in engineering with the Department of Electronics and Instrumentation Engineering, National Institute of Technology, Silchar, India.

From 2016 to 2017, he was a Lab Engineer with the National Institute of Technology, Aizawl, Mizoram, India. He has published a number of articles in IEEE conference and other reputed journals. His research interests include reversible watermarking, image processing, and VLSI design.

ARUN KUMAR SUNANIYA received the M.Tech. degree in microelectronics and VLSI design from SGSITS, Indore, India, and the Ph.D. degree in electronics and communication engineering from the National Institute of Technology, Bhopal.

From 2009 to 2012, he was an Assistant Professor with the Department of Electronics and Communication Engineering, Sagar Institute of Research and Technology, Bhopal, India. From 2012 to 2014, he was an Associate Professor with the Department of Electronics and Communication Engineering, Corporate Institute of Science and Technology, Bhopal. Since 2014, he has been an Assistant Professor with the Electronics and Instrumentation Engineering Department, National Institute of Technology, Silchar, India. He is the author of more than 30 articles. His research interests include low-power VLSI, hybrid and CMOS devices, and digital and analog integrated circuits.

RESHMI MAITY received the B.Tech. and M.Tech. degrees in radio physics and electronics from the University of Calcutta, Kolkata, India, in 2004 and 2006, respectively, and the Ph.D. degree in electronics and communication engineering from the National Institute of Technology, Silchar, India, in 2016.

From 2004 to 2008, she was an Assistant Professor with the JIS College of Engineering (now JIS University), Kolkata. From 2008 to 2018, she was an Assistant Professor with the Department of Electronics and Communication Engineering, Mizoram University (A Central University), Aizawl, India. Since 2018, she has been an Associate Professor with the Department of Electronics and Communication Engineering, Mizoram University (A Central University). She is the author or coauthor of more than 100 archival refereed publications. Her research interests include VLSI design, nanoelectronics, and MEMS.

NILADRI PRATAP MAITY (Senior Member, IEEE) was born in Girisha, West Bengal, India, in 1976. He received the M.Tech. degree in electronics design and technology from Tezpur University (A Central University), Tezpur, India, in 2004, and the Ph.D. degree in electronics and communication engineering from the National Institute of Technology, Silchar, India.

From 2007 to 2017, he was an Assistant Professor with Mizoram University (A Central University), Aizawl, India. Since 2017, he has been an Associate Professor with the Department of Electronics and Communication Engineering, Mizoram University (A Central University). He is currently the Head of the Department of Electronics and Communication Engineering. He is the author of more than 120 articles. His research interests include VLSI design, MOS device modeling, and MEMS.