Hindawi
Security and Communication Networks
Volume 2019, Article ID 1294267, 17 pages
https://doi.org/10.1155/2019/1294267

*Research Article*

# FPGA Implementation of Digital Images Watermarking System Based on Discrete Haar Wavelet Transform

**Mohamed Ali Hajjaji**,[1] **Mohamed Gafsi**,[1,2] **Abdessalem Ben Abdelali**,[1] **and Abdellatif Mtibaa**[1,2]

[1] *Université de Monastir, Laboratoire d'Electronique et de Microelectronique, LR99ES30, 5000, Monastir, Tunisia*
[2] *Université de Monastir, Ecole Nationale d'Ingénieurs de Monastir, 5000, Monastir, Tunisia*

Correspondence should be addressed to Mohamed Ali Hajjaji; daly_fsm@yahoo.fr

Received 14 August 2018; Revised 11 November 2018; Accepted 9 December 2018; Published 3 January 2019

Academic Editor: Stelvio Cimato



In this paper we propose a novel and efficient hardware implementation of an image watermarking system based on the Haar Discrete Wavelet Transform (DWT). DWT is used in image watermarking to hide secret pieces of information into a digital content with a good robustness. The main advantage of Haar DWT is the frequencies separation into four subbands (LL, LH, HL, and HH) which can be treated independently. This permits ensuring a better compromise between robustness and visibility factors. A Field Programmable Gate Array (FPGA) that is based on a very large scale integration architecture of the watermarking algorithm is developed to accelerate media authentication. A hardware cosimulation strategy using the Matlab-Xilinx system generator (XSG) was applied to prove the validity of the suggested implementation. The hardware cosimulation results show the effectiveness of the developed architecture in terms of visibility and robustness against several attacks. The proposed hardware system presents also a high performance in terms of the operating speed.

## 1. Introduction

Digital watermarking is a technique of hiding information on a digital support such as images, video, or audio for authentication control, copyright protection, integrity verification, etc. The hidden information is called a watermark and the marked documents are named watermarked data. Distortion caused by the hidden watermark on the host data should be made as low as possible. The watermarked and original images must be perceptually equivalent so that the embedded watermark can remain imperceptible by a Human Visual System (HVS). The Peak Signal to Noise Ratio (PSNR) parameter is used for the imperceptibility measure. Even if the distortion, caused by the watermark, is small, it can be undesirable in some image types such as the medical and military ones. For these types of applications, the PSNR value must be greater than 40 dB [1]. In this case, watermarking in the transform domain is recommended. In fact, transform spaces such as Discrete Cosine Transform (DCT), Discrete Wavelet Transform (DWT), and Karhunen Loeve Transform (KLT) provide a special authentication to host images. They are especially used in telemedicine, e-healthcare, legal domains, telesurgery, etc.

The performance of a watermarking system is generally subject to the following requirements.

*(i) Imperceptibility*. The watermark should not affect the quality of the original image after any watermarking operation. Cox et al. [2] defined the imperceptibility as a visual similarity between the original and watermarked images. The watermark has to be inserted in a way that it still is completely invisible to HVS [3]. Indeed, the insertion process must not damage the host image. However, not only the image but also the watermark should not be distorted. This latter must be invisible, but also easy to extract.

*(ii) Capacity*. The ability of a watermarking system refers to the ratio of the amount of data to be hidden according to

the size of the host document [4]. Sometimes the size of the watermark is limited just to 1 bit.

*(iii) Robustness*. Robustness is the resistance of the watermark system against intentional transformations on a watermarked image [5]. These transformations can be of a given geometric type such as rotation and cropping and they include all types of image degradation caused by lossy compression, high-pass filter, low-pass filter, etc.

To these requirements, we can add the computational complexity. In fact, execution time can be an important factor for many applications. Watermarking algorithms with a low computation cost can be used to reduce the execution time. However, this can highly affect the system performance. Elsewhere, the algorithm can be adapted for hardware implementation to accelerate the processing while maintaining the techniques effectiveness [6]. In the related literature, software implementation of the watermarking algorithms is largely applied in contrast to hardware implementation, despite the performance that can be achieved by applying this type of development [7]. In a software implementation, the algorithm's operations are performed as a code running on a microprocessor [8]. The main drawback of this type of implementation [8] is the limited means for improving the system speed and the hardware performances. Although it might be faster to implement an algorithm in software, there are a few compelling reasons for a move to hardware implementation. In this kind of implementation the algorithm's operations are fully implemented in a custom-designed circuitry. This investigates great advantages such as hardware area and consumption decrease and mainly speed increase [7–9].

In the given literature, a number of hardware designs for conventional watermarking algorithms have been reported. The Very Large Scale Integration (VLSI) architecture for a conventional watermarking algorithm in the spatial domain proposed by Gerimella et al. [10] might be considered as a noteworthy early work. Later, Mohanty et al. [11] proposed a watermarking hardware architecture that can insert two visible watermarks into digital images using a spatial domain watermarking technique. Mohanty et al. [12] put forward a VLSI architecture that could insert invisible or visible watermarks into digital images in the DCT domain. Mohanty et al. [13] developed two versions (low-power, high-performance) of watermarking hardware module. The DC component and the three low frequency components are considered for insertion in the DCT domain. Maity et al. [14] suggested a fast Walsh transform (FWT) based on a Spread Spectrum (SS) image watermarking scheme that would serve for authentication in data transmission. In [15], Korrapati Rajitha et al. proposed an FPGA implementation of a watermarking system using the Xilinx System Generator (XSG). Insertion and extraction of information were applied in the spatial domain. In [16], Rohollah Mazrae Khoshki et al. put forward a hardware implementation of a watermarking system based on DCT. Their work was developed using Matlab-Simulink followed by Altera DSP Builder (integrated with Simulink Embedded coder) for Auto-Code generation. In [17], Rahate Kunal B. et al. suggested a hardware implementation of a fragile watermarking system operating in the spatial domain. Their proposed watermarking scheme was imperceptible and robust against geometric attacks, but fragile against filtering and compression. Hirak Kumar Maitya et al. [6] put forward a hardware implementation of reversible watermarking in the spatial domain by using a reversible contrast mapping technique. The principal advantage of the proposed work was the operation frequency (more than 98.76 MHz). In [18], Sakthivel and S.M. et al. put forward a VLSI architecture of a digital image watermarking system. Their embedding process was based on the Pixel Value Search Algorithm (PVSA) applied in the spatial domain. The system was implemented using verilog Hardware Description Language (HDL) and the Altera Quartus-II 11.0 tool with Matlab R-2012b. The presented results showed that the proposed system was not highly fast with an average quality of the watermarked image and the extracted watermark resulting in different attacks. In [19], Manas N. et al. suggested a hardware implementation of a watermarking algorithm based on phase congruency and singular value decomposition. Their idea consisted in embedding watermark data in the host image using the Singular Value Decomposition (SVD) in the congruency phase mapping points applied in the spatial domain. Their system was implemented using the Xilinx ISE 14.3 tool and a Virtex 5 FPGA device. In [6], Hirak M. et al. proposed an FPGA implementation of an image watermarking algorithm using Reversible Contrast Mapping (RCM) in the spatial domain. The implemented algorithm and the resulting architecture were relatively simple. In [20], Karthigai kumara P. et al. put forward an FPGA implementation of an image watermarking system using the XSG tool. Their suggested system consists in embedding a binary watermark in the discrete wavelet domain of a host image. The main disadvantage of the proposed system is that the corresponding hardware design consumed a lot of hardware resources despite that the system used only the DWT tool.

After this review of the existing work that addresses the hardware implementation of watermarking systems, we can note that the majority of their present inefficiency is in terms of hardware performances or in terms of robustness of the hardware design against attacks. Many of them are applied in the spatial domain with, some time, very simple techniques to be implemented as well as a lack of hardware speed efficiency. However, hiding confidential data in the spatial domain is generally vulnerable against hackers. In this work, we suggest a novel and efficient hardware implementation of a watermarking system based on Haar DWT. We aim at developing a watermarking system that ensures high performance in terms of hardware efficiency with high imperceptibility (PSNR) and robustness (Normalized Cross-Correlation, NC). The system is designed using the XSG tool and synthesized for Xilinx Virtex-5 FPGA of the ML507 platform. A comparison with existing watermarking systems will be undergone to show the effectiveness of the proposed module in terms of hardware performances with the high imperceptibility and robustness against several attacks.

The rest of the paper is organized as follows: In Section 2, a description of the different steps of the adopted watermarking algorithm is given. In Section 3, we describe the hardware

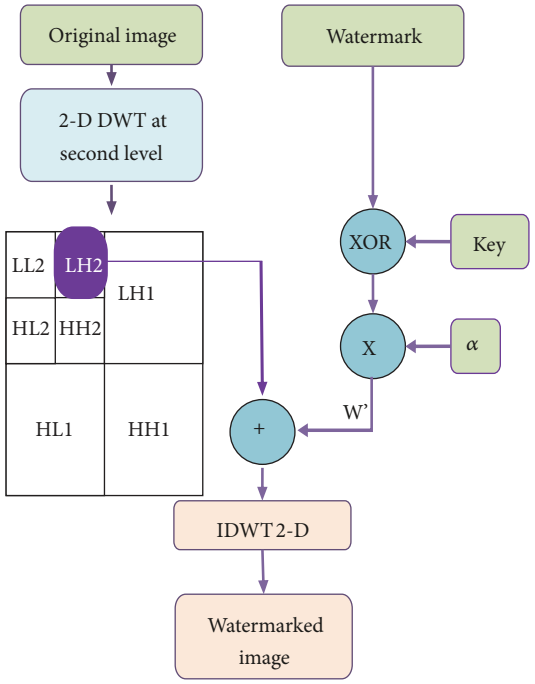



Figure 1: The insertion step.

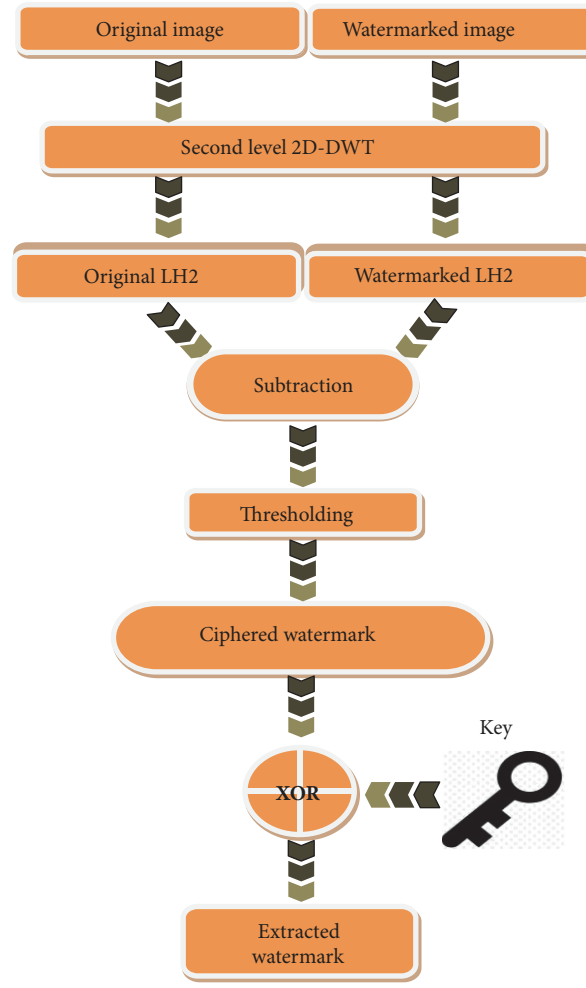



Figure 2: The extraction step.

design of the watermarking system. The implementation results and the performance evaluation of the developed watermarking system are presented in Section 4.

## 2. Description of Watermarking Algorithm

Watermarking systems of digital images are composed of two main parts: insertion and detection [21]. The diffusion process includes the attacks applied to watermarked images.

*2.1. Insertion Step.* As illustrated in Figure 1, the proposed system is an additive scheme. The watermark insertion is expressed by

$$\begin{aligned} P_i &= P_{ori} + \left(q_i\left(C\right) \oplus W_i\right) \times \alpha \\ with\ q &= \text{key generator} \\ C &= \text{Binary random sequence} \\ i &= i^{th}\ iteration \\ \alpha &= Visibility\ factor \\ P_i &= i^{th}\ \text{watermarked coefficient} \\ P_{ori} &= i^{th}\ \text{original coefficient} \\ W_i &= i^{th}\ \text{bit of the watermark} \end{aligned} \tag{1}$$

In the insertion phase, our system requires four data inputs:

(i) The original image (I) that will contain the data to be preserved and protected.

(ii) The watermark (W), which represents the information to be inserted (a binary information).

(iii) The key (C), which is a binary sequence to be mixed with the watermark for its protection.

(iv) The visibility factor ($\alpha$), which is the marking strength in the image. This coefficient must be adequately chosen to maintain a best compromise between robustness and imperceptibility factors of the scheme.

After the second level of decomposition using the 2D Haar DWT, we obtain four subbands of 1/8 of the input image size (Figure 1): approximation (LL2 band: low frequencies) and details (horizontal (LH2), vertical (HL2), and diagonal (HH2)). In our adopted method, we opt for inserting the watermark in the LH2 subband, which includes the medium frequencies. In the end of this phase, 2D IDWT is applied to construct the watermarked image.

*2.2. Extraction Step.* As depicted in Figure 2, the extraction step consists in following the same steps as in the insertion phase. The 2D Haar DWT is applied at the second level of

the decomposition. After that, the watermark is recovered by using the following equation:

$$W'(i) = \left[\frac{[LHt(i) - LHo(i)]}{\alpha} \oplus C(i)\right]$$
$$With\ LHt: Watermarked\ sub-band \quad (2)$$
$$LHo: Original\ sub-band$$

The watermarked image may be subject to alterations caused by attacks. Indeed, a thresholding phase is necessary for the proper extraction of the watermark. Equation (3) is applied to set the value of the watermark.

$$if \begin{cases} P_{al}(i) \geq S & \Longrightarrow W(i) = 1 \\ else & \Longrightarrow W(i) = 0 \end{cases} \quad (3)$$

with $P_{al}(i)$ : $i^{th}$ is the difference between watermarked and the original coefficient

S: Threshold value, determined empirically

## 3. Hardware Design of the Watermarking System

Xilinx Company proposes an Integrator Design Environment (IDE) for FPGA under the Matlab tool. This IDE is aiming to increase the abstraction level of the hardware design and to minimize the manual intervention of the HDL code generation [22]. This tool is named XSG; it is a high-level design tool that allows using the MathWorks Simulink environment in the design of digital circuits dedicated to Xilinx FPGAs [23]. It is used for hardware system generation, simulation, and validation throughout the hardware cosimulation technique.

The structure of a system is created in the Simulink modeling environment using a specific library offered by Xilinx. All the designing steps for the implementation on FPGA, including synthesis, placement, and routing, are automatically performed to generate an FPGA programming file.

The designer starts with creating the system model in Simulink. Next, "Sysgen" automatically generates the bitstream to program the FPGA. Intermediate steps, which are synthesis, placement, and routing, are performed by intermediate tools. Figure 3 describes the XSG based design flow.

In our design, the acquisition and display of input and output images are performed using the Matlab tool. At this phase, data are presented in a double-precision floating number. The processing algorithm is implemented by using XSG blocks. In the XSG design, boolean and fixed-point formats are used for data representation. To adapt the representation differences between the XSG design and the Matlab software part, Xilinx offers a simple interfacing utilizing predefined "Gateway-In" and "Gateway-Out" blocks provided in the Xilinx Blockset Library. The global design of the watermarking system is divided into two principal modules: insertion and extraction.

*3.1. Insertion Module.* As shown in Figure 4, the global design of the insertion module is composed of two main blocks. The first one corresponds to the decomposition and reconstruction of the DWT at the second level. The second one corresponds to the insertion step.

*3.1.1. 2D Haar DWT*

*(a) Decomposition Step of the 2D DWT of Haar*. The one-dimensional decomposition is obtained by applying the equations of the decomposition "A" for approximations and "M" for details.

$$A = \frac{X_{(2n)} + X_{(2n+1)}}{2}: \quad X: \text{Input signal}$$
$$M = \frac{Y_{(2n)} - Y_{(2n+1)}}{2}: \quad Y: \text{Output signal} \quad (4)$$

As shown in Figure 5, the Haar wavelet decomposition in two dimensions is mainly performed in two stages. The first stage consists in applying (1) and (2) along lines. This allows obtaining two subbands, generally denoted as L and H. Then a transposition is made in order to reach the second stage, which consists in applying the same equations on columns. So, the four subbands named LL, LH, HL, and HH will be obtained.

Figure 6 gives the various parts of the 2D Haar DWT global design.

*(i) Preprocessing Subsystem*. The preprocessing subsystem allows the preparation of the input data for accelerating the wavelet computing. The idea consists in decomposing the entire image into four components, so, separating the even and odd pixels from each even and odd image line. This process allows performing the wavelet steps in one go. The design is presented in Figure 7.

*(ii) Calculation of the Subsystem*. This subsystem computes the coefficient of wavelet field. Thus, it receives and processes the outputs of the preprocessing subsystem in order to produce four outputs, which are LL, LH, HL, and HH coefficients. Obviously, as shown in Figure 8, the calculation

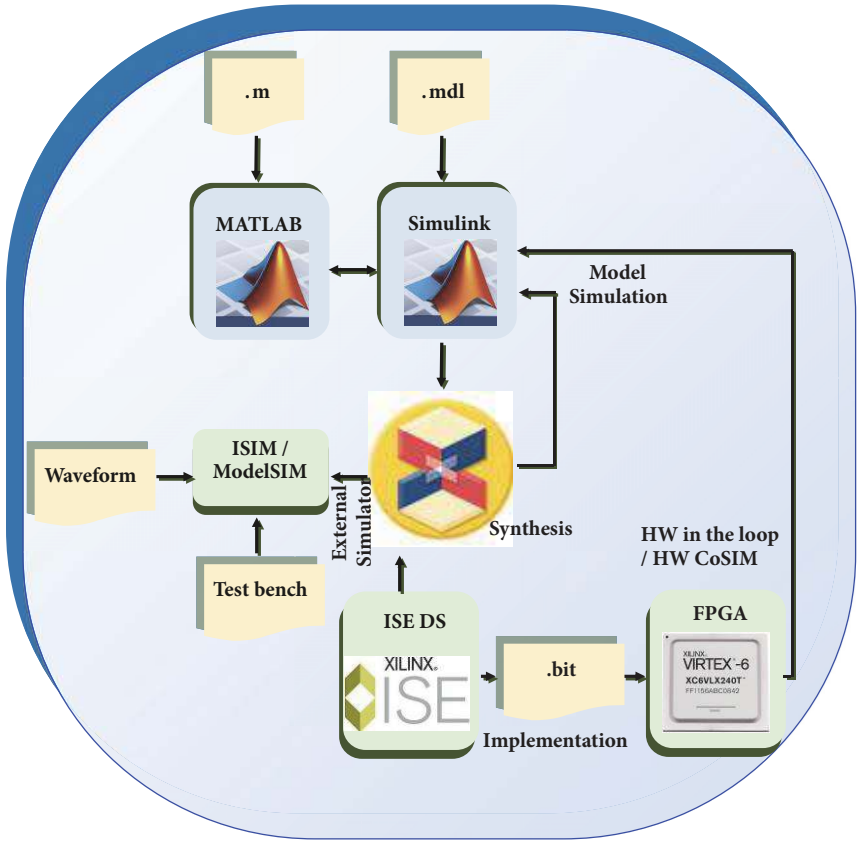



Figure 3: XSG based design flow.

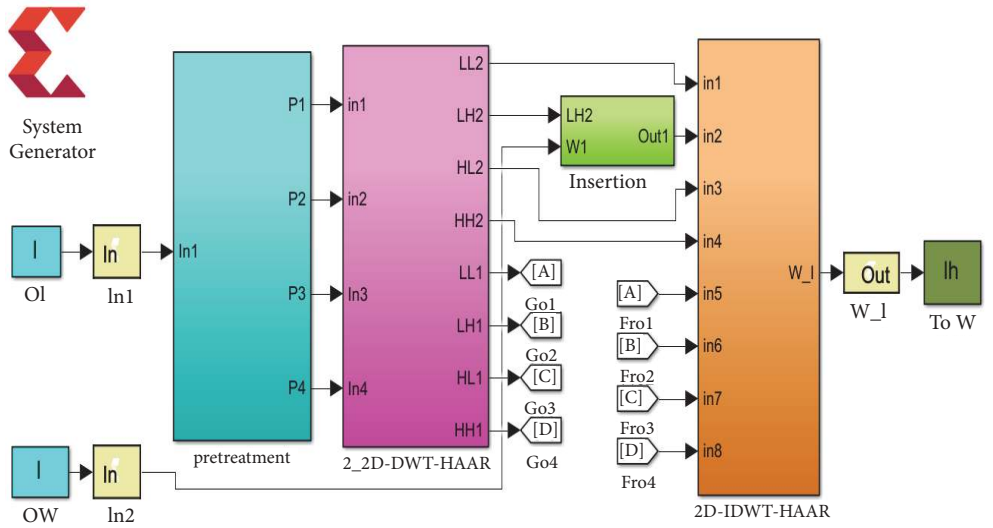



Figure 4: Part 1: XSG blocks for the insertion phase.

is done by the addition, subtraction, and multiplication blocks.

*(iii) Storage Subsystem*. After wavelet computing, a storage stage is required; hence we present the objective of "Storage" subsystem. However, to accelerate the write/read of data, we have opted for using internal RAM blocks. The Storage subsystem design is presented in Figure 9.

*(b) Inverse Transformation of 2D DWT of Haar*. The principle of calculating the coefficients of the original image is depicted in Figure 10. From four subbands (LL, LH, HH, and HL), the

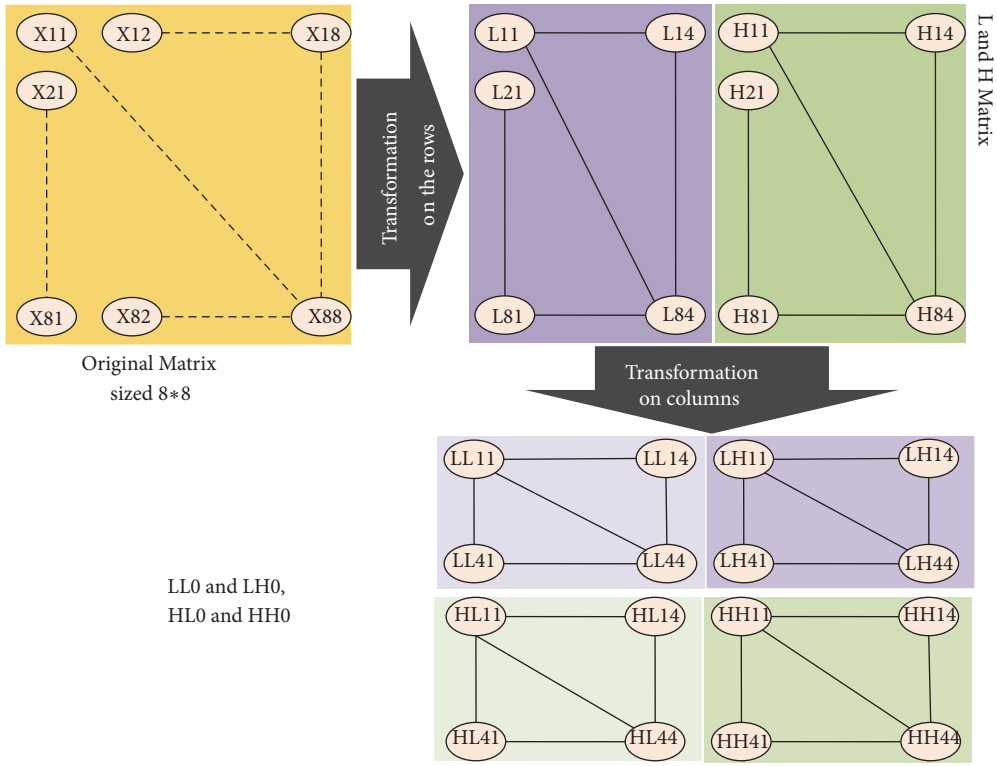



Figure 5: Principle of the 2D Haar DWT.

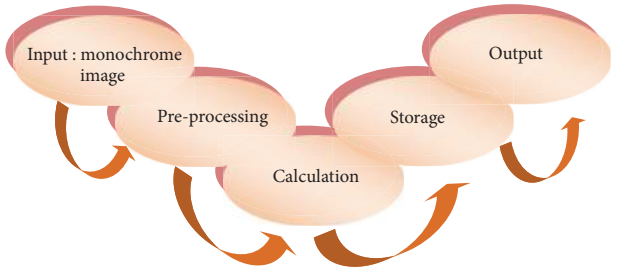



Figure 6: Different blocks of the proposed architectures of the 2D Haar DWT.

first step is to calculate L and H. Then in the second step, the original pixels are calculated.

The process of calculation is as follows:

(i) We begin with the computation of the L subband coefficients. This is done by browsing, at the same time, the two LL and LH subbands along the columns using the following equations:

$$L(2\times i) = \frac{LL(j) + LH(j)}{2}$$
$$L(2\times i - 1) = \frac{LL(j) - LH(j)}{2} \quad (5)$$
$$i, j = 1 \longrightarrow \frac{N}{2};\ N = \text{number of pixels in a column}$$

(ii) Thereafter, with HL and HH, we calculate the coefficients of band H. This is achieved by the following equations:

$$H(2\times i) = \frac{HL(j) + HH(j)}{2}$$
$$H(2\times i - 1) = \frac{HL(j) - HH(j)}{2} \quad (6)$$
$$i, j = 1 \longrightarrow \frac{N}{2};\ N = \text{nombre de pixels d'un colonnes.}$$

(iii) At this stage, the original pixels are calculated, browsing at the same time the two bands L and H along lines using the following equations:

$$P\_ori(2\times i - 1) = \frac{L(i) - H(i)}{2}$$
$$P\_ori(2\times i) = \frac{L(i) + H(i)}{2} \quad (7)$$

(iv) After this last step, we have the original pixels. Finally the pixels are organized to reform the input image.

For the implementation of the IDWT of Haar with XSG tools, we propose the subsystem shown in Figure 10. Thus, the subsystem processes the coefficients of wavelet field in order to acquire the original data. Hence, the computing of the original data is done with addition and subtraction blocks. Also, we use other logic blocks for data control and shaping.

*3.1.2. Hiding Watermark on the Host Image.* As presented in Figure 11, the second step is about the insertion system. At

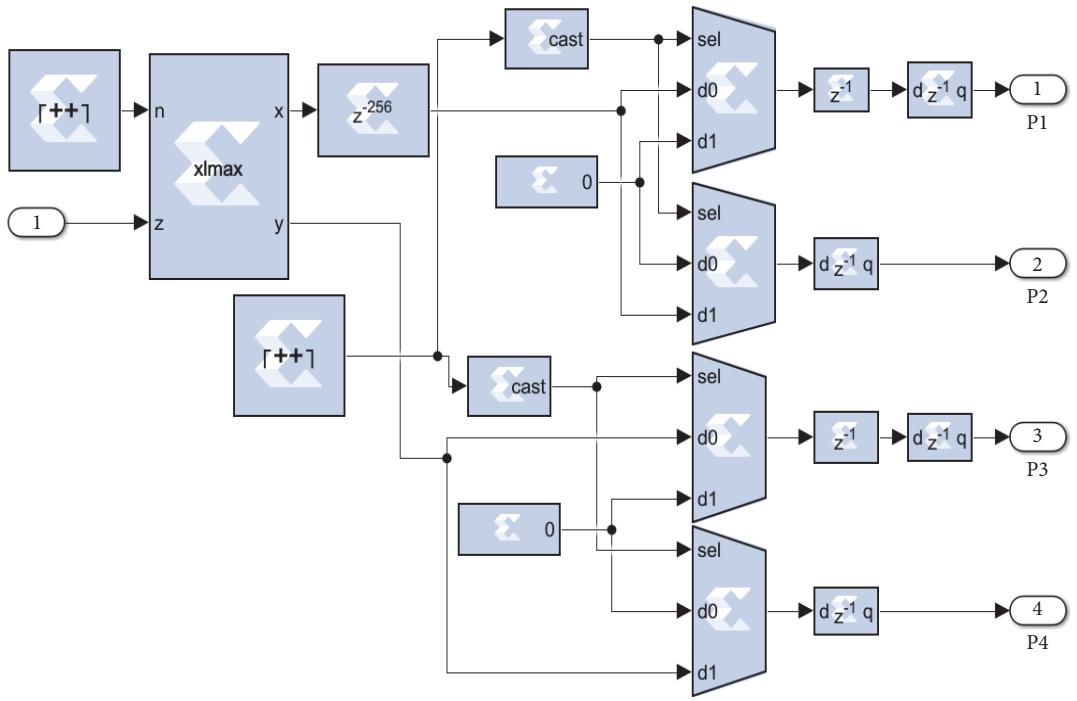



Figure 7: Block diagram of the "pre-processing" subsystem.

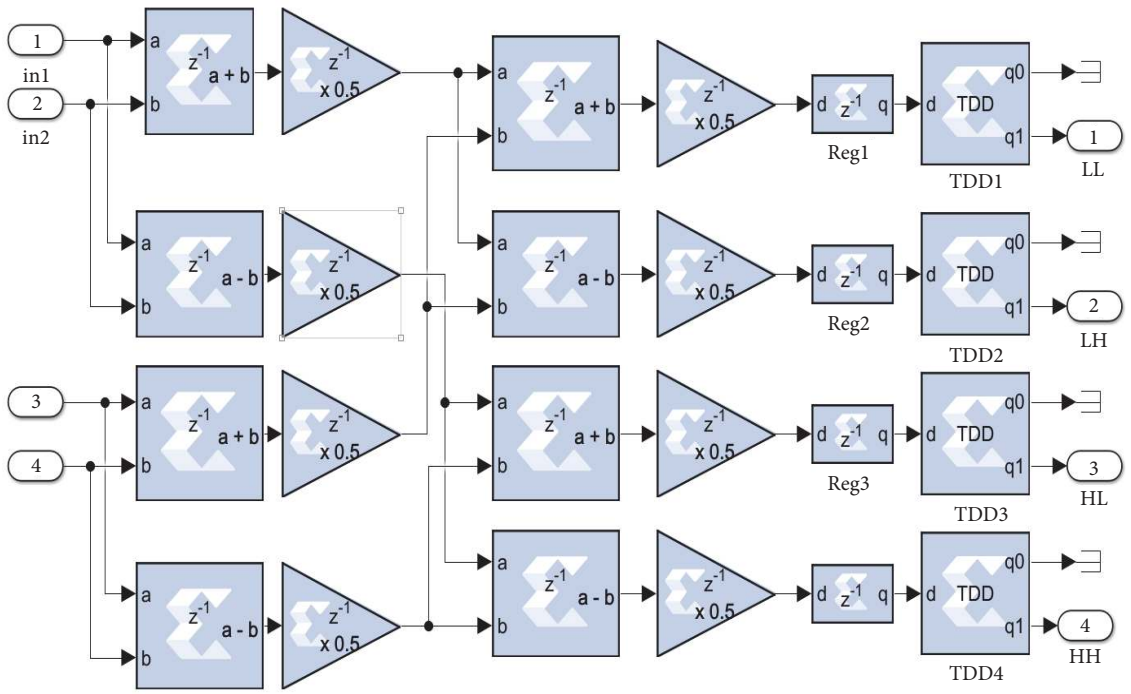



Figure 8: Block diagram of calculation sub-system.

this step, the totality of the watermark is embedded in LH2 (second horizontal subband). The watermark is scrambled by a secret key generated by the "LFSR" block. Afterward, the "DSP48 macro" block is used to carry out the addition of the scrambled watermark multiplied by the "$\alpha$" visibility factor.

The inputs of the "DSP48 macro" block are, respectively, LH2, alpha, and the scrambled watermark. Its output is the watermarked LH2.

*3.2. Extraction Module.* The extraction step is the last phase of the watermarking system, which aims to extract the

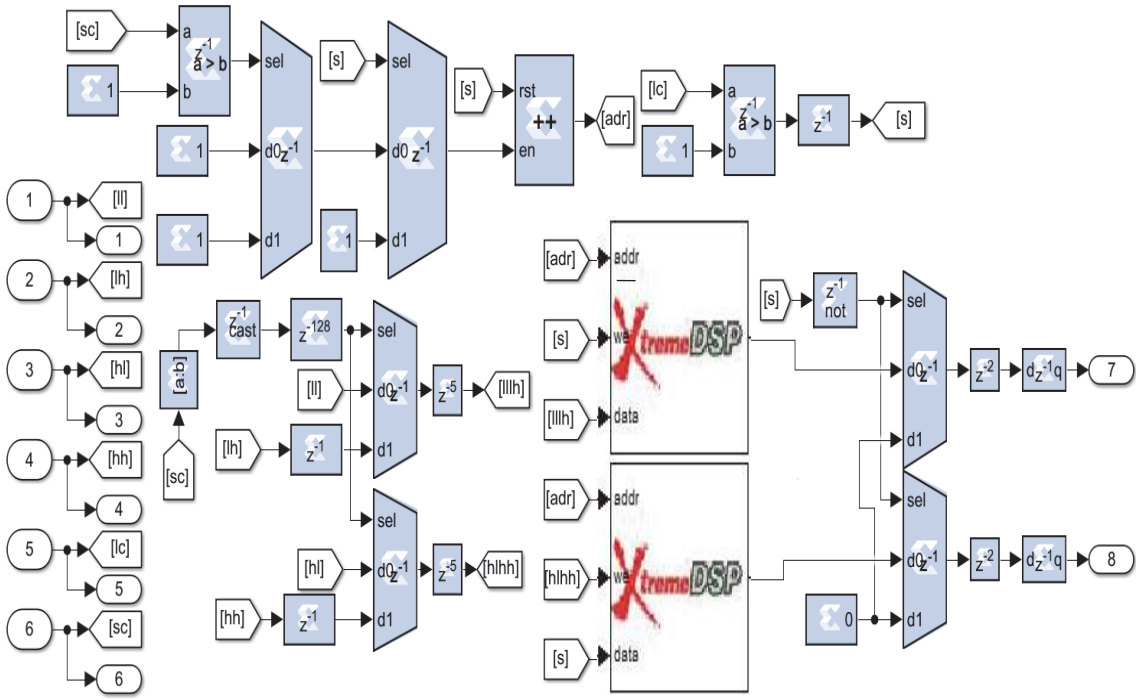



Figure 9: Block diagram of storage subsystem.

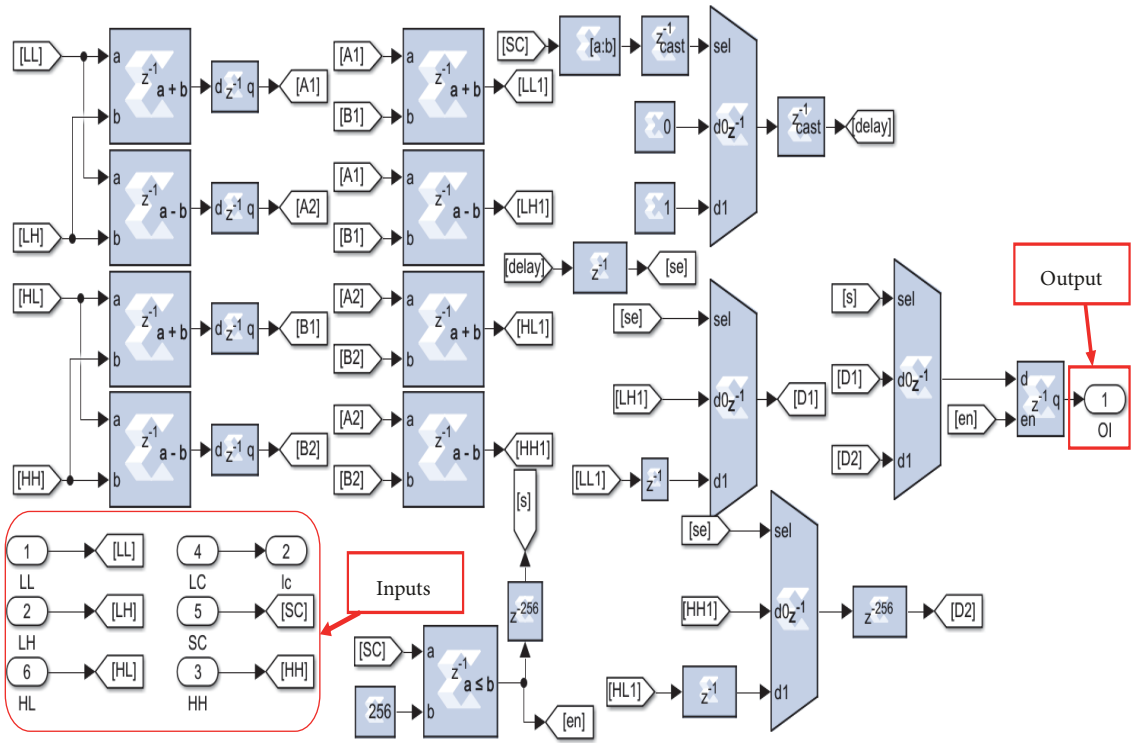



Figure 10: Block diagram of different subsystems of inverse Haar DWT.

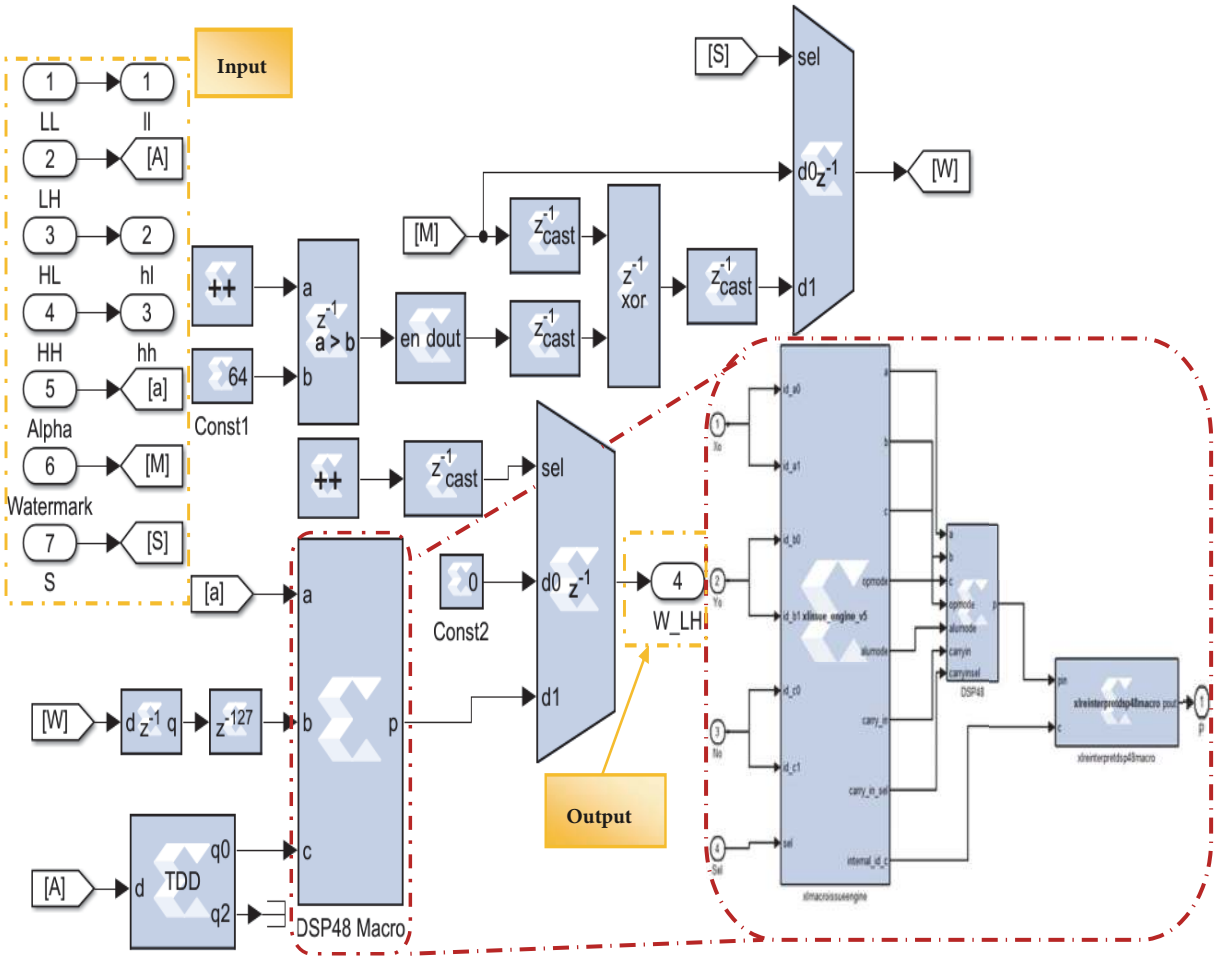



Figure 11: Watermark insertion model.

inserted data. Figure 12 represents the global design of the extraction system. At this step, the same procedure is reversely used.

The main difference, relative to the insertion step, is the extraction block. Figure 13 presents the design of the extraction block. We obtain the original and watermarked subbands. After that, a subtraction is applied to extract the modified watermark, named W'. The latter is stocked in FIFO. Finally, by using the thresholding, the final watermark is extracted.

## 4. Implementation Results and Performance Evaluation

In this section, we start by presenting the hardware implementation results of the adopted system. Some examples of the cosimulation results of the generated hardware block will be present. The efficiency of the proposed system is then discussed according to the PSNR value, between the original and the watermarked image, and the NC value between the original and the detected watermark against several attacks. A comparison with some existing works will be described in the following.

*4.1. Cosimulation Results.* After the validation of the adopted algorithm by the software simulation, we proceed to the implementation on a Xilinx platform. The configuration file is obtained automatically by following the necessary steps to convert the design into an FPGA synthesizable module (Figure 14). The target device selected for this work is Virtex-5 FPGA on the ML507 platform.

The hardware implementation of the insertion and extraction steps, on the ML 507 target, generates the results of the FPGA resource consumer in Table 1. The Register Transfer Level (RTL) diagrams of the insertion and extraction systems are presented in Figure 15.

For the validation of our study, we considered an ordinary image base known as the image "Cameraman," "Lena," "Barbara," etc. In Figure 16, we present some implementation results of the adopted watermarking system, on the "Cameraman" image with a variation of the value $\alpha$ (equal to 3, 6, 10, and 20). However, we notice that the increase in the visibility factor leads to the loss of the psychovisual quality of the watermarked image. It should be noted that, in the absence of attacks, the watermark is well extracted, from which we can conclude that the implemented system gives results similar to those obtained by software implementation.

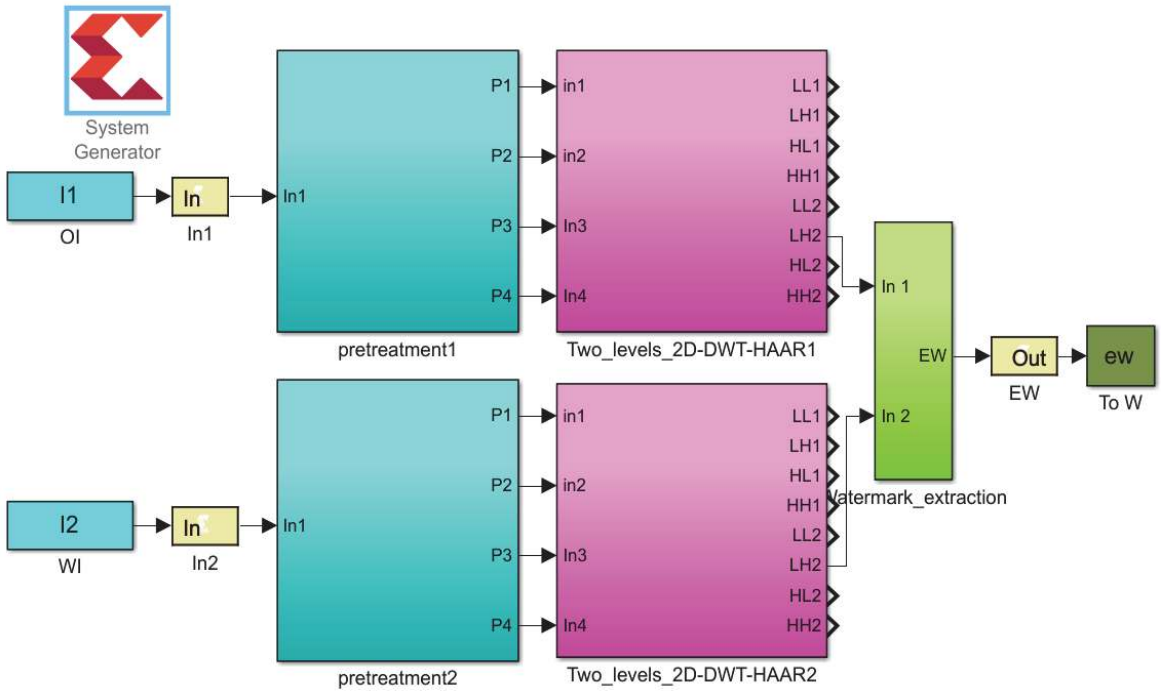



Figure 12: Part 2: XSG blocks for extraction phase.

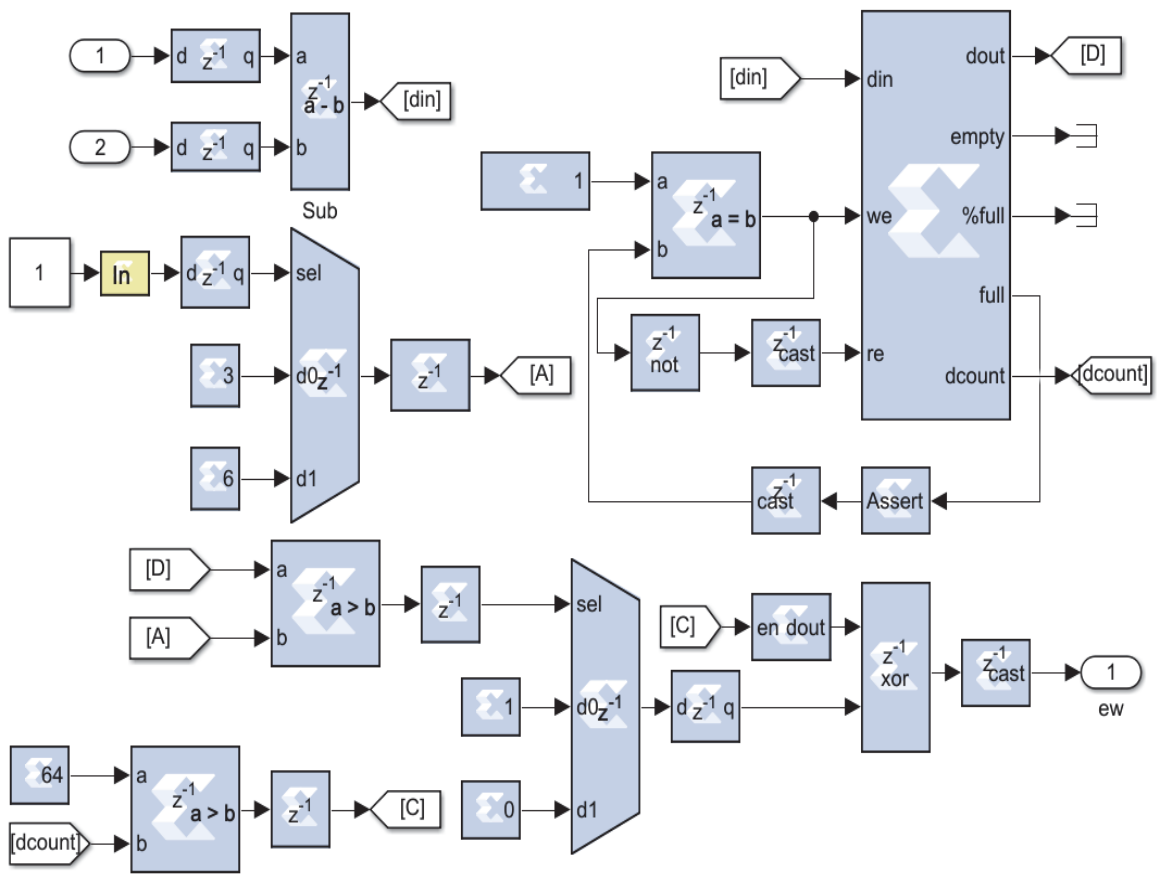
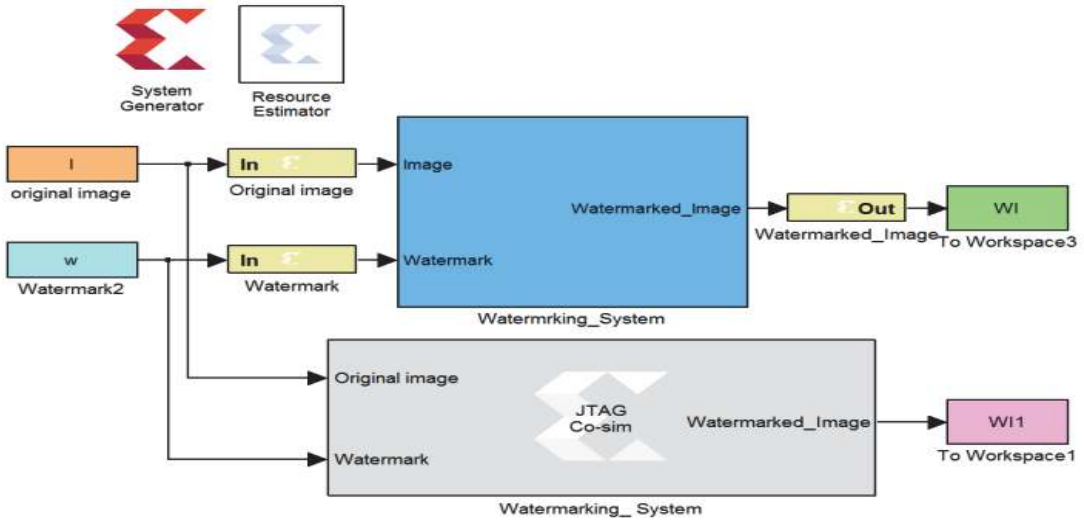
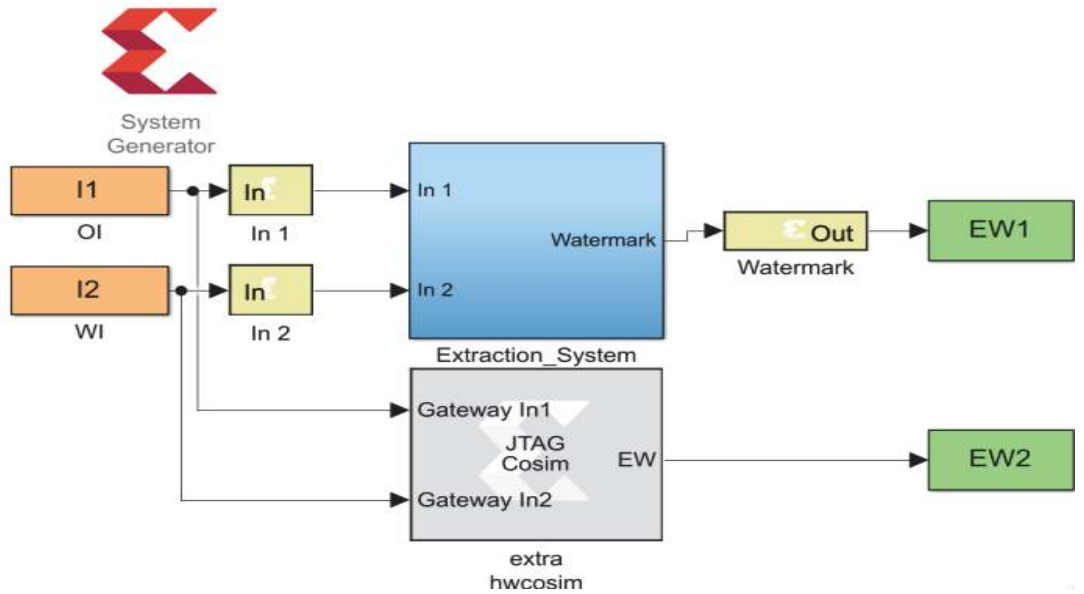



Figure 13: Extraction Block.




(a)

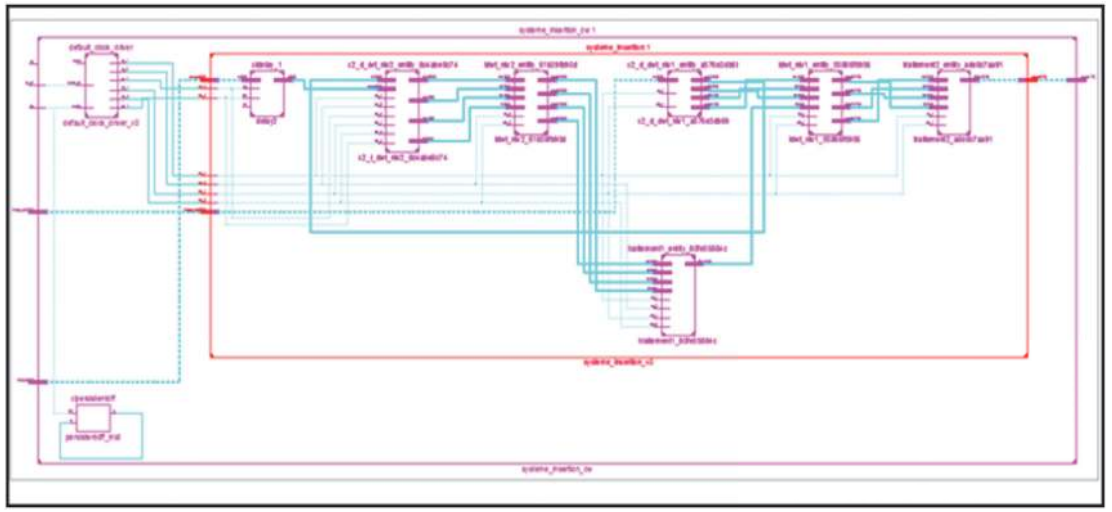



(b)

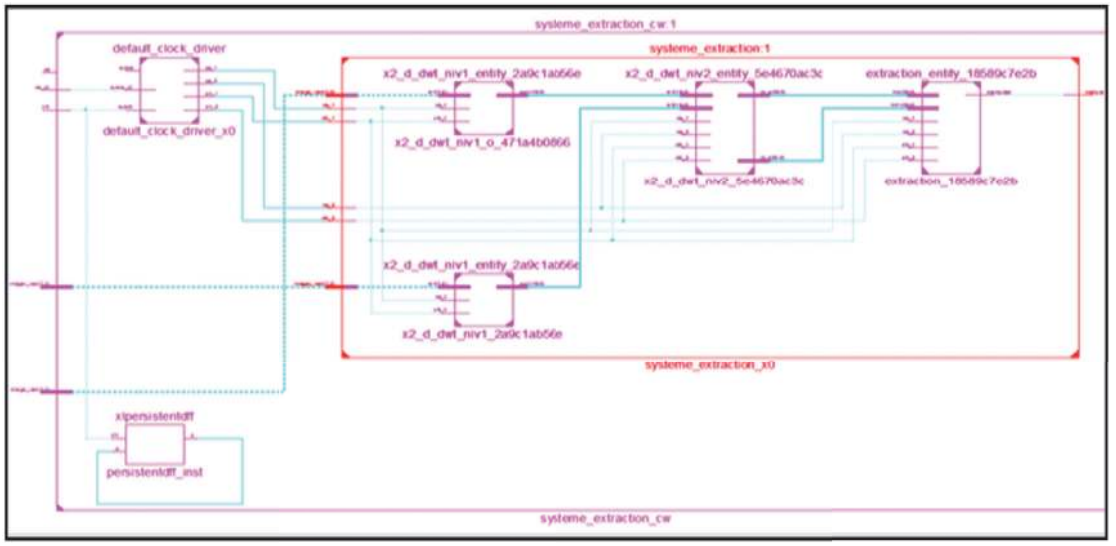
Figure 14: Hardware block generation of insertion and extraction steps.

Table 1: Consumed hardware resources in insertion and extraction steps resources.

| Resources | Insertion Step | | | Extraction Step | | |
|---|---|---|---|---|---|---|
| | Used | Available | Percentage | Used | Available | Percentage |
| **Number of register slices** | 1,536 | 32,640 | 4% | 619 | 32,640 | 2% |
| **Number of slice LUT** | 2,092 | 32,640 | 6% | 1,002 | 32,640 | 3% |
| **Number of used logic blocks** | 2,494 | 32,640 | 8% | 532 | 32,640 | 1% |
| **Number of DSP48** | 1 | 48 | 2% | 1 | 48 | 2% |
| **Number of BRAM** | 9 | 148 | 6% | 12 | 148 | 8% |
| | **Maximum frequency =224 MHz** | | | **Maximum frequency =232 MHz** | | |



(a)




(b)

Figure 15: RTL schematic of insertion and extraction steps.

*4.2. Performance Evaluation against Several Attacks of Implemented System.* Following the literature, the main constraints of the watermarked scheme are imperceptibility and robustness factors. The first one is named PSNR and presented in (8). PSNR is accepted if its value is greater than 30 dB [24]. The second one, named NC, is presented in (9). The NC value is accepted if its value is greater than 0.7 [25].

$$(PSNR)_{dB} = 10\log_{10}\left[\frac{A \times B \times \max\left(I_{(i,j)}\right)^2}{\sum_{i=1}^{A}\sum_{j=1}^{B}\left(I_{(i,j)} - I'_{(i,j)}\right)^2}\right] \quad (8)$$

$I_{(i,j)}, I'_{(i,j)}$ : pixel values of the pixel $(i, j)$ in the host and watermarked image.

A, B: Width and Height of the host image.

$$NC = \frac{\sum_m \sum_n \left(X_{(m,n)} - \overline{X}\right) \times \left(Y_{(m,n)} - \overline{Y}\right)}{\sqrt{\left(\sum_m \sum_n \left(X_{(m,n)} - \overline{X}\right)^2\right) \times \left(\sum_m \sum_n \left(Y_{(m,n)} - \overline{Y}\right)^2\right)}} \quad (9)$$

$\overline{X}$ and $\overline{Y}$: Mean values of the coefficients of two matrices

After several empirical tests, applied to ordinary images with and without attacks, we have found that, for $\alpha$ equal to 3, the adopted algorithm ensures a maximum compromise between robustness and imperceptibility factors.

Figure 17 shows the results of the hardware cosimulation of ordinary images in the absence of attacks. It can be concluded that the values of PSNR are well acceptable with respect to the previous work and with respect to results obtained for the software implementation.

Our algorithm, implemented on the hardware, is more robust against other types of attacks. After applying several

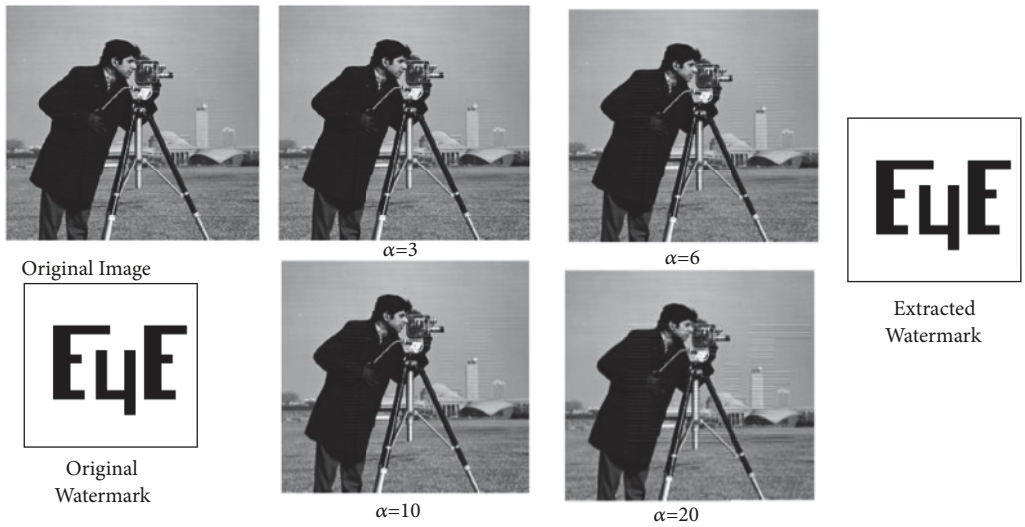



Figure 16: Hardware co-simulation results of insertion system.

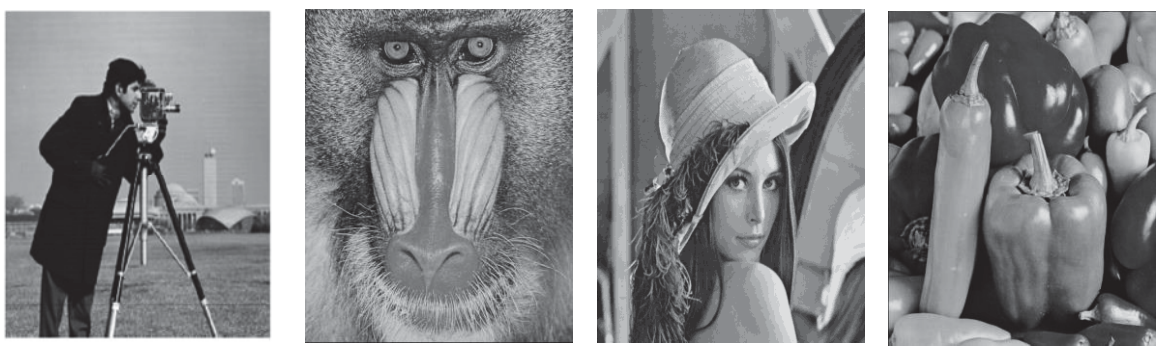



Figure 17: Hardware co-simulation results of watermarked ordinary images without attacks.

attacks, we extract the watermark and we compare it to the original one. The main goal is to ensure that the extracted watermark is not modified by attacks. It is important to get an NC value close to 1 and a good PSNR value. The robustness against diverse types of attacks such as JPEG 2000 attacks, impulsive noise, median filter, cropping, flipping, and stretching is among the important watermarking constraints.

After attacking the 6 types of ordinary images, we attempt to extract our watermark and calculate the NC value. Our aim is to conclude on the degree of robustness of our scheme against diverse attacks. Tables 2, 3, 4, and 5 and Figure 18 show the experimental results relative to the NC and PSNR values between the host and extracted watermark after applying attacks.

*4.3. Discussion of the Proposed Scheme.* In this section, we compare the obtained results of the suggested system with results relative to the systems cited in the related work section. For this comparison, we consider the most typical and recent related papers [6, 18–20]. The latter represent almost the most important works addressing watermarking systems hardware design with interesting results. First, for psychovisual quality of the original and watermarked images, our hardware implementation provides very good results compared to the software implementation ones.

As provided in Table 6, in the absence of any type of attack, PSNR, for the image "Lena" is equal to 48.4715, which represents a better result than those aforementioned algorithms. Among the most serious attacks, we apply the JPEG attack. The obtained result shows that the proposed scheme is very effective against this kind of attacks. In fact, the results presented in Table 2 show that, from a compression rate equal to 50%, the NC value is greater than 0.7. Compared with previous work (Table 6), we note that our implemented system gives better results.

The evaluation of our implemented method against impulsive noises shows very promising results as presented in Table 3. In fact, the recovery of the watermark is greater than 0.7 for a density equal to 0.01. Beyond this value, the recovery of the watermark is not acceptable. Indeed, our implemented approach has proven its robustness against this type of attack, and, compared to other works, our implemented system gives better results.

Also, we test our system against the median filter. The test is evaluated with various sized windows (from [3×3] until [9×9]) (Table 4). The detection by correlation between the

Table 2: NC and PSNR values for watermarked and attacked images by JPEG-2000 compression.

| Ratio (%) | PSNR | | | | NC | | | |
|---|---|---|---|---|---|---|---|---|
| | Cameraman | Mandrill | Lena | Pepper | Cameraman | Mandrill | Lena | Pepper |
| **10** | 44.05 | 42.10 | 44.05 | 40.15 | 1 | 1 | 1 | 1 |
| **20** | 39.09 | 38.47 | 39.09 | 38.19 | 1 | 1 | 1 | 1 |
| **30** | 37.36 | 36.61 | 37.36 | 34.55 | 1 | 1 | 1 | 1 |
| **40** | 37.04 | 35.23 | 37.04 | 32.76 | 0.957 | 0.925 | 0.962 | 0.941 |
| **50** | 36.12 | 34.19 | 36.12 | 30.17 | 0.872 | 0.870 | 0.771 | 0.868 |
| **60** | 35.68 | 33.12 | 35.68 | 28.9 | 0.764 | 0.753 | 0.694 | 0.772 |
| **70** | 35.24 | 31.82 | 35.03 | 25.01 | 0.744 | 0.679 | 0.690 | 0.679 |
| **80** | 34.73 | 29.95 | 34.21 | 23.66 | 0.730 | 0.520 | 0.654 | 0.453 |
| **90** | 31.34 | 26.78 | 32.91 | 22.12 | 0.665 | 0.489 | 0.618 | 0.432 |

Table 3: NC and PSNR values for watermarked, decrypted and attacked images by impulsive noise.

| Density | PSNR | | | | NC | | | |
|---|---|---|---|---|---|---|---|---|
| | Cameraman | Mandrill | Lena | Pepper | Cameraman | Mandrill | Lena | Pepper |
| **0.0001** | 41.71 | 41.94 | 42.48 | 42.09 | 1 | 1 | 1 | 1 |
| **0.0005** | 37.59 | 36.80 | 36.87 | 37.10 | 1 | 1 | 1 | 1 |
| **0.0009** | 35.65 | 35.94 | 35.14 | 35.28 | 1 | 1 | 1 | 1 |
| **0.001** | 34.74 | 35.05 | 34.16 | 34.89 | 0.980 | 0.968 | 0.986 | 0.984 |
| **0.005** | 27.76 | 28.25 | 28.23 | 28.39 | 0.945 | 0.911 | 0.969 | 0.948 |
| **0.009** | 25.53 | 26.14 | 25.84 | 25.79 | 0.894 | 0.867 | 0.879 | 0.829 |
| **0.01** | 25.00 | 25.54 | 25.38 | 24.99 | 0.780 | 0.691 | 0.678 | 0.714 |
| **0.05** | 18.06 | 18.57 | 18.49 | 18.33 | 0.677 | 0.578 | 0.577 | 0.609 |
| **0.09** | 15.50 | 16.04 | 15.84 | 15.74 | 0.603 | 0.502 | 0.497 | 0.532 |
| **0.01** | 15.10 | 15.56 | 15.45 | 15.27 | 0.585 | 0.492 | 0.471 | 0.510 |

Table 4: NC and PSNR values for watermarked, decrypted and attacked images by median filter.

| Window size | PSNR | | | | NC | | | |
|---|---|---|---|---|---|---|---|---|
| | Cameraman | Mandrill | Lena | Pepper | Cameraman | Mandrill | Lena | Pepper |
| **[2×2]** | 29.64 | 26.80 | 29.55 | 29.56 | 1 | 1 | 1 | 1 |
| **[3×3]** | 36.52 | 29.80 | 35.30 | 35.12 | 1 | 1 | 1 | 1 |
| **[4×4]** | 28.81 | 25.05 | 28.89 | 28.45 | 0.874 | 0.812 | 0.898 | 0.856 |
| **[5×5]** | 31.07 | 24.61 | 31.23 | 31.02 | 0.771 | 0.754 | 0.780 | 0.756 |
| **[6×6]** | 27.36 | 22.92 | 27.77 | 27.48 | 0.529 | 0.524 | 0.551 | 0.531 |
| **[7×7]** | 27.10 | 22.61 | 29.03 | 28.45 | 0.517 | 0.512 | 0.535 | 0.530 |
| **[8×8]** | 25.28 | 21.88 | 26.78 | 25.07 | 0.371 | 0.331 | 0.348 | 0.312 |
| **[9×9]** | 25.31 | 21.82 | 27.66 | 24.69 | 0.114 | 0.103 | 0.135 | 0.175 |

Table 5: NC and PSNR values for watermarked, encrypted and attacked images by cropping.

| Window size | PSNR | | | | NC | | | |
|---|---|---|---|---|---|---|---|---|
| | Cameraman | Mandrill | Lena | Pepper | Cameraman | Mandrill | Lena | Pepper |
| **[8×8]** | 38.88 | 42.05 | 38.73 | 40.96 | 0.988 | 0.952 | 0.980 | 0.746 |
| **[16×16]** | 33.92 | 38.01 | 33.84 | 36.82 | 0.793 | 0.812 | 0.898 | 0.736 |
| **[32×32]** | 28.17 | 31.51 | 28.16 | 28.77 | 0.759 | 0.780 | 0.766 | 0.698 |
| **[64×64]** | 22.10 | 23.93 | 22.30 | 22.85 | 0.657 | 0.587 | 0.570 | 0.605 |
| **[128×128]** | 15.70 | 17.94 | 18.09 | 18.09 | 0.452 | 0.406 | 0.388 | 0.410 |

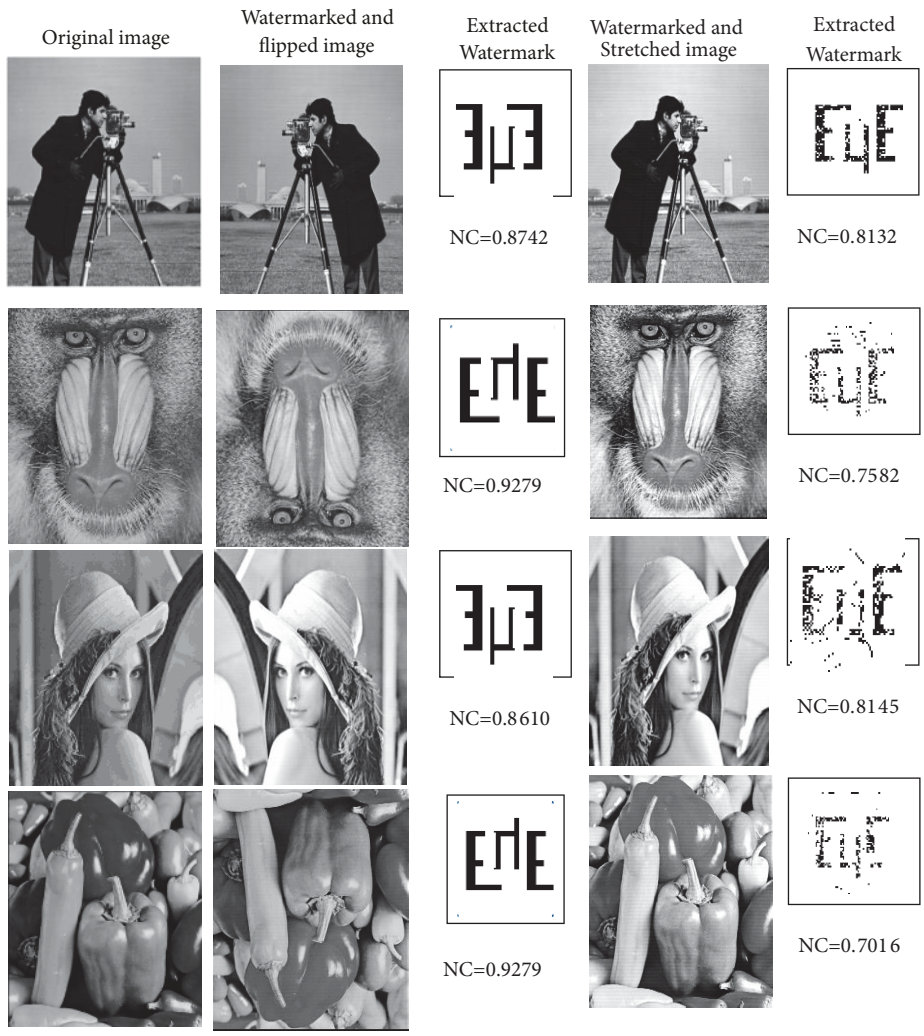



Figure 18: NC value for watermarked, encrypted and attacked images by flipping and stretching.

Table 6: NC and PSNR comparison against other works.

| | Attacks | Method | PSNR | NC |
|---|---|---|---|---|
| **In presence of attacks** | JPEG-2000 Q=80 | Ref [18] | 32.1072 | 0.989 |
| | | Ref [19] | 44.06 | 0.994 |
| | | **Proposed algorithm** | **39.09** | **1** |
| | Salt & pepper 0.05 | Ref [18] | 23.1951 | 0.918 |
| | | Ref [19] | 26.33 | 0.847 |
| | | **Proposed algorithm** | **28.23** | **0.969** |
| | Cropping 25% | Ref [18] | --- | --- |
| | | Ref [19] | 29.07 | 0.847 |
| | | **Proposed algorithm** | **18.09** | **0.410** |

extracted and inserted watermarks has shown that our implemented system is robust against median-filter attacks (NC is greater than 0.7 for a window size coefficient less than or equal to [5x5]) and keeps the visual appearance of the image after watermarking. As illustrated in Table 6, in general, the proposed architecture gives relatively good results.

The proposed architecture gives also acceptable results (NC greater than 0.7) against geometric attacks such as the flapping and stretching of the watermarked images. The last attack applied on the proposed system is the so-called "cropping" attack. Note that for a window lower than or equal to 25% of the size of the watermarked image, the NC value is

Table 7: Hardware performance comparison.

| Method | Device | Number of Slice LUT | Number of Slice Register | Max Frequency (MHz) |
|---|---|---|---|---|
| **Proposed algorithm** | **Xilinx Virtex-5** | **2092** | **1536** | **224** |
| **Ref [18]** | Altera Flex 10K | 1477 | 45 | 58.48 |
| **Ref [19]** | Xilinx Virtex-5 | 103 | 203 | 183.8 |
| **Ref [6]** | Xilinx Spartan-3E | --- | 9881 | 98.7 |
| **Ref [20]** | Xilinx Virtex-6 | 4708 | 3922 | 344.34 |

less than 0.7. Compared to previous works we note that this is also an acceptable result for our architecture.

The hardware performances of the proposed system have been evaluated relatively to the operating frequency the FPGA resources occupancy rate. According to Table 7, broadly the proposed architecture gives better results. The highest operating frequency reported in previous work is 183.8 MHz [19]. However, for our algorithm, the maximum operating frequency is 224 MHz. Compared to [20], we noted that even if the proposed architecture is slower, it presents a better hardware resources occupation rate.

## 5. Conclusion

In this paper a novel and efficient hardware implementation of an image watermarking system based on the Haar Discrete Wavelet Transform has been developed. The performance of the proposed hardware implementation in terms of processing latency has been evaluated and compared to other previous work. The XSG tool has been used for system development. The utilization of this tool has a big benefit in terms of conception time, since the same design has been firstly used for the software validation and then for hardware system generation. A hardware cosimulation strategy using the XSG was applied to prove the validity of the proposed implementation. The hardware cosimulation results showed the effectiveness of the developed architecture in terms of visibility and robustness against several attacks.

## Data Availability

The obtained results used to support the findings of this study are included within the article.

## Conflicts of Interest

The authors declare no conflicts of interest.

## Authors' Contributions

All authors helped in conceiving the experiments. Mohamed Ali Hajjaji designed and performed the experiments. At the same time, Mohamed Ali Hajjaji and Mohamed Gafsi wrote the main part of the paper. Abdellatif Mtibaa and Abdessalem Ben Abdelali contributed to interpreting the results and revising and writing of the paper.

## References

[1] M. A. Hajjaji, E.-B. Bourennane, A. Ben Abdelali, and A. Mtibaa, "Combining Haar wavelet and Karhunen Loeve transforms for medical images watermarking," *BioMed Research International*, vol. 2014, Article ID 313078, 15 pages, 2014.

[2] I. J. Cox, J. Kilian, F. T. Leighton, and T. Shamoon, "Secure spread spectrum watermarking for multimedia," *IEEE Transactions on Image Processing*, vol. 6, no. 12, pp. 1673–1687, 1997.

[3] Y. Zhang, "Blind watermark algorithm based on HVS and RBF neural network in DWT domain," *WSEAS Transactions on Computers*, vol. 8, no. 1, pp. 174–183, 2009.

[4] R. O. Preda and D. N. Vizireanu, "A robust digital watermarking scheme for video copyright protection in the wavelet domain," *Measurement*, vol. 43, no. 10, pp. 1720–1726, 2010.

[5] W.-B. Lee and T.-H. Chen, "A public verifiable copy protection technique for still images," *The Journal of Systems and Software*, vol. 62, no. 3, pp. 195–204, 2002.

[6] H. K. Maity and S. P. Maity, "FPGA implementation of reversible watermarking in digital images using reversible contrast mapping," *The Journal of Systems and Software*, vol. 96, pp. 93–104, 2014.

[7] S. P. Mohanty, N. Ranganathan, and R. K. Namballa, "VLSI implementation of invisible digital watermarking algorithms towards the development of a secure JPEG encoder," in *Proceedings of the 2003 IEEE Workshop on Signal Processing Systems, SIPS 2003*, pp. 183–188, August 2003.

[8] N. J. Mathai, D. Kundur, and A. Sheikholeslami, "Hardware implementation perspectives of digital video watermarking algorithms," *IEEE Transactions on Signal Processing*, vol. 51, no. 4, pp. 925–938, 2003.

[9] A. Basu, T. S. Das, S. K. Sarkar et al., "FPGA prototype of visual information hiding," in *Proceedings of the 2010 Annual IEEE India Conference: Green Energy, Computing and Communication, INDICON 2010*, pp. 1–4, 2011.

[10] A. Garimella, M. V. V. Satyanarayan, R. S. Kumar, P. S. Murugesh, and U. C. Niranjan, "VLSI Impementation of Online Digital Watermarking Techniques With Difference Encoding for the 8-bit Gray Scale Images," in *Proceedings of the International Conference on VLSI Design*, pp. 792–796, 2003.

[11] S. P. Mohanty, N. Ranganathan, and R. K. Namballa, "A VLSI architecture for visible watermarking in a secure still digital camera (S 2DC) design," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 13, no. 8, pp. 1002–1011, 2005.

[12] S. P. Mohanty, N. Ranganathan, and K. Balakrishnan, "A dual voltage-frequency VLSI chip for image watermarking in DCT domain," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 53, no. 5, pp. 394–398, 2006.

[13] S. P. Mohanty, O. B. Adamo, and E. Kougianos, "VLSI Architecture of an Invisible Watermarking Unit for a Biometric-Based Security System in a Digital Camera," in *Proceedings of the 2007 Digest of Technical Papers International Conference on Consumer Electronics*, pp. 1-2, Las Vegas, NV, USA, 2007.

[14] M. Santi, A. Banerjee, A. Abhijit, and K. Malay, "VLSI Design of Spread Spectrum Image Watermarking," in *Proceedings of the 13th National Conference on Communication NCC, 2007*, IIT Kanpur, India, 2007.

[15] K. Rajitha, U. R. Nelakuditi, V. N. Mandhala, and T.-H. Kim, "FPGA implementation of watermarking scheme using XSG," *International Journal of Security and Its Applications*, vol. 9, no. 1, pp. 89–96, 2015.

[16] R. M. Khoshki, "Hardware Based Implementation of an Image Watermarking System," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 3, no. 5, 2014.

[17] B. Rahate Kunal, A. S. Bhalchandra, and S. S. Agrawal, "VLSI Implementation of Digital Image Watermarking," *International Journal of Engineering Research & Technology (IJERT)*, vol. 2, no. 6, 2013.

[18] S. M. Sakthivel and A. Ravi Sankar, "A real time watermarking of grayscale images without altering it's content," in *Proceedings of the 2015 International Conference on VLSI Systems, Architecture, Technology and Applications, VLSI-SATA 2015*, IEEE, January 2015.

[19] M. R. Nayak, J. Bag, S. Sarkar, and S. K. Sarkar, "Hardware implementation of a novel water marking algorithm based on phase congruency and singular value decomposition technique," *International Journal of Electronics and Communications*, 2017.

[20] P. Karthigai kumara and K. B. Anumolb, "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, Procedia Engineering*, Elsevier, 2011.

[21] M. A. Hajjaji, A. Ben Abdellali, N. Farhani, M. Gafsi, and A. Mtibaa, "Real time implementation of numerical watermarking system using Xilinx system generator," in *Proceedings of the 16th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering, STA 2015*, pp. 404–409, 2016.

[22] R. Hmida, A. Ben Abdelali, and A. Mtibaa, "Hardware implementation and validation of a traffic road sign detection and identification system," *Journal of Real-Time Image Processing*, pp. 1–18, 2016.

[23] M. Gafsi, S. Ajili, M. A. Hajjaji, and A. Mtibaa, "XSG for hardware implementation of a robust watermarking system," in *Proceedings of the 17th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering, STA 2016*, pp. 117–122, December 2016.

[24] P. M. Naini, *Digital Watermarking Using MATLAB*, INTECH Open Access Publisher, pp. 465-480, 2011.

[25] H. Guan, Z. Zeng, J. Liu, and S. Zhang, "A novel robust digital image watermarking algorithm based on two-level DCT," in *Proceedings of the 2014 International Conference on Information Science, Electronics and Electrical Engineering, ISEEE 2014*, pp. 1804–1809, April 2014.