DESIGN OF MEDICAL IMAGE WATERMARKING SYSTEM USING VLSI BASED ARITHMETIC OPERATIONS

[1]GAJULA MANISHA, [2]Dr.M.V SHRUTHI
[1]M.Tech , [2]Associate Professor
Department of ECE
Dr.K.V.Subba Reddy Institute of Technology, Kurnool

Abstract

One of the ways to protect the intellectual property rights of the digital media is digital watermarking. With rapid increase in use of internet and digital media, transmission and reproduction of digital products has become very convenient but it has some drawbacks also. The digital revolution provides tools to unlimited copying without loss in fidelity. The people can easily steal the digital work of others like image, videos, and audio clips and claim their rights on the stolen things. This situation creates the need of copyright protection of digital media. Digital watermarking can be used for content authentication, detection of illegal duplication and alteration, secret communication as watermarking is very helpful in hiding the secret messages or information in the digital media. Using watermarking the people can keep their work copyrighted. The watermarking algorithm incorporates the watermark in the object, whereas the verification algorithm authenticates the object by determining the presence of the watermark and its actual data bits.

The VLSI implementation of digital watermarking system must meet the low area (look up tables, slices, registers, memory elements), delays (path, logical) delays. However, the conventional wavelet transform based VLSI watermarking systems suffers with higher consumption of these attributes due to down sampler. So, this work is focused on implementation of hybrid stationary wavelet transform based watermarking system using VLSI fundamentals.

I.INTRODUCTION

1.1 OVERVIEW

A VLSI (Very Large Scale Integration) based image watermarking system is a system that is designed to embed a watermark into a digital image at the VLSI chip level. This system is used to protect the copyright of the original digital image and to authenticate the image.

The VLSI-based implementation of the watermarking system offers several advantages over software-based approaches. It can provide faster processing speeds and lower power consumption, making it more suitable for real-time applications. The VLSI chip can be designed to perform specific tasks related to watermarking, making it more efficient than a general-purpose processor. In this technique, the SWT is used to decompose the image into multi-resolution sub-bands. The high-frequency sub-bands are suitable for embedding the watermark as they are less sensitive to human perception. A pseudo-random noise sequence or a binary image is used as the watermark, which is then manipulated and embedded into the selected sub-bands.

The VLSI architecture of the system includes hardware modules for image pre-processing, SWT transform, watermark extraction, and embedding. The image pre-processing module receives the input image and performs necessary operations such as cropping, resizing, and color space conversion. The SWT module decomposes the pre-processed image into sub-bands and modifies the coefficients according to the watermark signal. The watermark extraction module extracts the original watermark from the watermarked image. The embedding module combines the original image and the generated watermark to create the final watermarked image. So, the VLSI based image watermarking system using SWT technology is a reliable and efficient method for embedding hidden messages or watermarks into digital images. It provides a high level of security and copyright protection for digital media.

1.2 MOTIVATION

Motivation for developing a VLSI-based image watermarking system using SWT technology:

1. **Security:** With the increasing use of digital media, there is a growing concern about the unauthorized use and distribution of copyrighted materials. Image watermarking can help to protect against these types of abuses by embedding an invisible signature into the image that can be used to verify its authenticity.
2. **Efficiency:** VLSI-based image watermarking systems offer significant advantages over software-based approaches in terms of processing speed and power consumption. This makes them ideal for real-time applications such as video streaming or live broadcasting.
3. **Robustness:** DWT-based watermarking algorithms are known for their robustness against common image processing operations such as compression, cropping, and scaling. This means that the watermark remains intact even if the image undergoes some modifications.
4. **Flexibility:** DWT-based watermarking can be customized to meet different security requirements, depending on the type of information being protected

and the level of risk involved. For example, high-security applications may require more complex algorithms that involve multiple levels of encryption and authentication.

Overall, developing a VLSI-based image watermarking system using DWT technology can provide a powerful tool for protecting digital content and ensuring its authenticity in a wide range of applications.

II. LITERATURE SURVEY

[1] Kumar, M. Pradeep, et al. "VLSI implementation of Digital Watermarking Technique for security and authentication of Digital Data." 2021 International Conference on Smart Generation Computing, Communication and Networking (SMART GENCON). IEEE, 2021.

Kumar, M. Pradeep (2021) [1], demonstrated various methods of digital watermarking techniques. In this approach ato watermark the image in the digital camera we can integrate the watermarking chip within framework, to demonstrate that this chip can embedded the watermark directly at the stage of capturing process itself which acts as a real time insertion and reduces the time consuming process of externally adding the watermark to the original image or video after capturing it. Illegal recovering and transmission of multimedia data can be protected via watermark or digital signature or copyright label that authenticates the data. This approach extended the opportunity of protecting the content after the release of content to the open environment.

[2] Jayanthi, V. E., Senthil Pitchai, and M. Smitha. "Design a hybrid FPGA architecture for visible digital image watermarking in spatial and frequency domain." Journal of Circuits, Systems and Computers 31.01 (2022): 2250020.

Jayanthi, V. E., Senthil Pitchai, and M. Smitha (2022) [2], proposed an approach is mainly designed for watermarking the images taken from digital cameras of various sizes. The padding technique is used for unequal sizes of the watermark image and original host image. The architecture data path consists of eight and six stages of pipeline capable of watermarking on the pixel-based operation and vector-based operation, respectively. The dual image watermarking architecture data path consists of a 13-stage pipeline. Pipeline and parallelism mechanisms are used to improve throughput. To improve the performance in discrete cosine transform operations at the frequency domain, the shift-add technique replaces the conventional multipliers. The clock gating technique is employed to reduce the power by preventing unnecessary switching in a path.

III. PROBLEM STATEMENT

The problem statement is to design a VLSI-based image watermarking system that uses discrete wavelet transform (DWT) technology to embed and extract a watermark from an image efficiently. The system should be able to handle different types of image formats, such as JPEG, PNG, and BMP, and also be robust against various attacks such as noise addition, cropping, and scaling.

Image watermarking is a technique used to protect the ownership and authenticity of digital images. Digital watermarking is a secure way to embed additional data in the original image without affecting its quality. In recent years, with the increase in copyright infringements, theft, and illegal distribution of digital images, image watermarking has become an essential technology for image authentication.

Additionally, the system should be low power, cost-effective, and have a small footprint. It should also be scalable and able to handle large-sized images without compromising the processing speed and accuracy. These challenges make the design of a high-performance VLSI-based image watermarking system using SWT technology an interesting and complex problem that requires interdisciplinary research expertise.

IV.EXISTING METHOD

4.1 WAVELET TRANSFORM

A signal analysis method similar to image pyramids is the discrete wavelet transform. The main difference is that while image pyramids lead to an over complete set of transform coefficients, the wavelet transform results in a nonredundant image representation. The discrete 2-dim wavelet transform is computed by the recursive application of lowpass and high pass filters in each direction of the input image (i.e. rows and columns) followed by sub sampling. One major drawback of the wavelet transform when applied to image fusion is itswell known shift dependency, i.e. a simple shift of the input signal may lead to complete different transform coefficients. This results in inconsistent fused images when invoked in image sequence fusion. To overcome the shift dependency of the wavelet fusion scheme, the input images must be decomposed into a shift invariant representation. There are several ways to achieve this: The straightforward way is to compute the wavelet transform for all possible circular shifts of the input signal. In this case, not all shifts are necessary and it is possible to develop an efficient computation scheme for the resulting wavelet representation. Another simple approach is to drop the subsampling in the decomposition process and instead modify the filters at each decomposition level, resulting in a highly redundant signal representation.

The actual fusion process can be described by a DWT multiresolution fusion scheme which is applicable both to image pyramids and the wavelet approach. The basic idea of the generic multiresolution fusion scheme is motivated by the fact that the human visual system is primary sensitive to local contrast changes, i.e. edges. Motivated from this insight, and in mind that both image pyramids and the wavelet transform result in an multiresolution edge representation, it is straightforward to build the fused image as a fused multiscale edge representation. The fusion process is summarized in the following: In the first step the input images are decomposed into their multiscale edge representation, using either any image pyramid or any wavelet transforms. The actual fusion process takes place in the difference resp. wavelet domain, where the fused multiscale representation is built by a pixel-by-pixel selection of the coefficients with maximum magnitude.

Daubechies wavelets

The most commonly used set of discrete wavelet transforms was formulated by the Belgian mathematician Ingrid Daubechies in 1988. This formulation is based on the use of recurrence relations to generate progressively finer discrete samplings of an implicit mother wavelet function; each resolution is twice that of the previous scale. In her seminal paper, Daubechies derives a family of wavelets, the first of which is the Haar wavelet. Interest in this field has exploded since then, and many variations of Daubechies' original wavelets were developed.

Calculating wavelet coefficients at every possible scale is a fair amount of work, and it generates an awful lot of data. What if we choose only a subset of scales and positions at which to make our calculations?

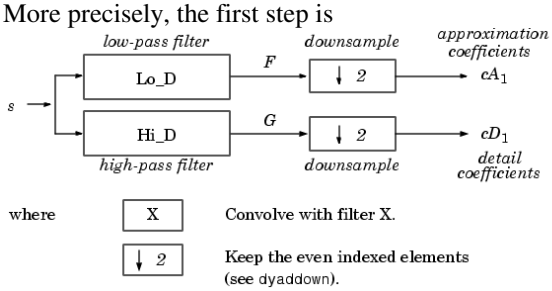
It turns out, rather remarkably, that if we choose scales and positions based on powers of two -- so-called dyadic scales and positions -- then our analysis will be much more efficient and just as accurate. We obtain such an analysis from the discrete wavelet transform (DWT).

An efficient way to implement this scheme using filters was developed in 1988 by Mallat. The Mallat algorithm is in fact a classical scheme known in the signal processing community as a two-channel sub band coder. This very practical filtering algorithm yields a fast wavelet transform -- a box into which a signal passes, and out of which wavelet coefficients quickly emerge.

Given a signal s of length N, the DWT consists of log2N stages at most. Starting from s, the first step produces two sets of coefficients: approximation coefficients cA1, and detail coefficients cD1.

These vectors are obtained by convolving s with the low-pass filter Lo_ D for approximation, and with the high-pass filter Hi_ D for detail, followed by dyadic decimation.

More precisely, the first step is



The length of each filter is equal to 2N. If n = length (s), the signals F and G are of length

n + 2N - 1, and then the coefficients cA1 and cD1 are of length

$$\text{floor}\left(\frac{n-1}{2}\right)+N$$

The next step splits the approximation coefficients cA1 in two parts using the same scheme, replacing s by cA1 and producing cA2 and cD2, and so on.

V.PROPOSED METHOD

Grayscale image watermarking is a technique used to embed information into digital images for various applications such as copyright protection, authentication, and tamper detection. VLSI (Very Large Scale Integration) based grayscale image watermarking systems are designed to perform this embedding process efficiently and securely. The VLSI-based grayscale image watermarking system consists of three main components: the watermark encoder, the watermark embedding module, and the watermark decoder. The watermark encoder is responsible for generating a watermark based on the input data, which can be text, an image, or any other type of digital information. The encoder then converts this watermark into a binary format suitable for embedding in the grayscale image.

The watermark embedding module is the core of the system, which embeds the binary watermark into the host image. The watermark is inserted into the image by modifying the pixel values of the image, such that the watermark is imperceptible to the human eye but can be extracted later using a decoding algorithm. The watermark decoder is responsible for extracting the watermark from the watermarked image. This involves analyzing the differences between the original image and the watermarked image, and then applying a decoding algorithm to extract the embedded information.

The VLSI-based grayscale image watermarking system requires careful consideration of several

factors, such as the size of the watermark, the robustness of the watermark, and the imperceptibility of the watermark. These factors must be optimized to ensure that the watermarking system is both effective and efficient. Overall, the VLSI-based grayscale image watermarking system provides a reliable and efficient way to embed information into digital images. It has numerous applications in various fields, including security, copyright protection, and authentication.

4.1 Proposed Model

For decades, many conventional signal processing methods have been applied in the image fusion field to extract image features, such as discrete wavelet transform (DWT), contourlet transform, shift-invariant shearlet transform and quaternion wavelet transform etc. For the infrared and visible image fusion task. But these methods may introduce artifacts into the fused image. To overcome these problems optimization based fusion schemes are proposed.

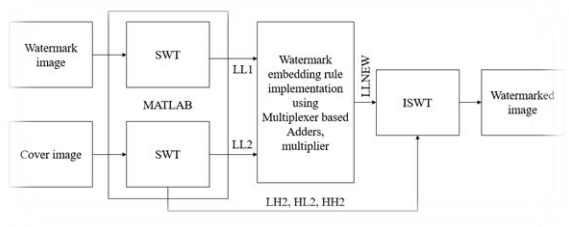


Figure 4.1. Proposed block diagram.

Grayscale image watermark embedding and extraction using SWT (Stationary Wavelet Transform) is a popular technique for watermarking digital images. It involves decomposing the image into different frequency bands using the SWT algorithm, and then embedding the watermark into one or more of these frequency bands. The embedding process involves the following steps:

Step 1: Decomposition: The grayscale image is decomposed into several frequency bands using the SWT algorithm. This results in a set of coefficients representing the image in each frequency band.

Step 2: Selection of coefficients: A subset of coefficients is selected from one or more of the frequency bands for embedding the watermark. The selection of coefficients is based on their sensitivity to changes in the pixel values.

Step 3: Embedding: The selected coefficients are modified to embed the watermark. The modification is performed using a pseudo-random sequence, which is generated using a secret key. The sequence determines the locations and values of the watermark bits that will be embedded in the selected coefficients. Here, the embedding formula is

$$LLnew = LL1 + alpha.* \ LL2$$

Here, addition, and multiplication is implemented using VLSI based adders and subtractors.

Step 4: Inverse SWT: The modified coefficients are used to reconstruct the watermarked image using the inverse SWT algorithm.

The extraction process involves the following steps:

Step 5: Decomposition: The watermarked image is decomposed into several frequency bands using the SWT algorithm. This results in a set of coefficients representing the image in each frequency band.

Step 6: Selection of coefficients: The same subset of coefficients that were used for embedding the watermark is selected from one or more of the frequency bands.

Step 7: Extraction: The selected coefficients are analyzed to extract the watermark. The extraction process involves applying a correlation operation between the coefficients and the pseudo-random sequence that was used for embedding the watermark. The correlation operation reveals the locations and values of the embedded watermark bits.

Step 8: Verification: The extracted watermark is compared to the original watermark to verify the integrity of the watermark. This is done by computing the correlation coefficient between the extracted and original watermarks. If the correlation coefficient is above a certain threshold, the watermark is authentic.

4.2 SWT

The stationary wavelet transform (SWT) is a signal processing technique that decomposes a signal into different frequency bands using wavelets. It is similar to the discrete wavelet transform (DWT), but unlike the DWT, the SWT does not downsample the signal at each level of decomposition, which leads to a shift-invariant representation of the signal. Here's a detailed algorithmic analysis of the stationary wavelet transform:

1. Input: A signal x(n) of length N, and a set of wavelet filters h(n) and g(n), where h(n) is the low-pass filter and g(n) is the high-pass filter.

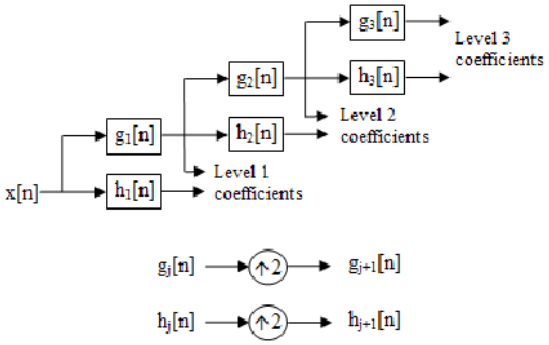


Figure. SWT

2. Initialize a vector of approximation coefficients V0 = x(n).
3. For each level of decomposition i = 1 to J, do the following:

a. Compute the wavelet coefficients Wj,i(n) and the approximation coefficients Vj,i(n) using the following equations:

$$Wj,i(n) = (Vj-1,i * g(n)) * 2\hat{}(i/2)$$
$$Vj,i(n) = (Vj-1,i * h(n)) * 2\hat{}(i/2)$$

where Vj-1,i is the approximation coefficients from the previous level of decomposition.

b. Store the wavelet coefficients Wj,i(n) and the approximation coefficients Vj,i(n) in two separate matrices, W and V, respectively.

4. Output: The wavelet coefficients W and the final approximation coefficients VJ.

Let's break down each step in detail:

Step 1: Input The input to the SWT algorithm is a signal x(n) of length N, and a set of wavelet filters h(n) and g(n), where h(n) is the low-pass filter and g(n) is the high-pass filter.

Step 2: Initialize We start by initializing a vector of approximation coefficients V0, which is just the original signal x(n).

Step 3: Decomposition For each level of decomposition i = 1 to J, we compute the wavelet coefficients and approximation coefficients as follows:

a. Compute the wavelet coefficients Wj,i(n) and the approximation coefficients Vj,i(n) using the following equations:

$$Wj,i(n) = (Vj-1,i * g(n)) * 2\hat{}(i/2)$$
$$Vj,i(n) = (Vj-1,i * h(n)) * 2\hat{}(i/2)$$

4.3 Watermark embedding rule

The watermark embedding rule is

$$LLnew = LL1 + alpha.* LL2$$

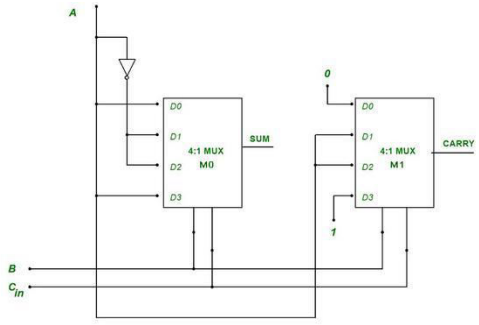


Figure . Full adder.

It is also possible to implement a full adder using only four-to-one muxes. One way to do this is as follows:

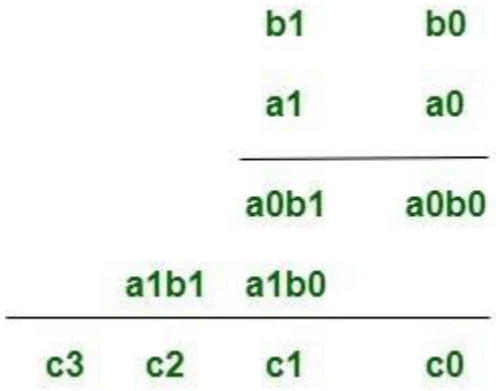
1. Use two 4-to-1 muxes to generate the sum output (S):
   - Connect A and B to the inputs of one of the muxes.
   - Connect Cin and the output of the other mux to the select inputs of the muxes.
   - Connect the outputs of the two muxes to the inputs of a third 4-to-1 mux.
   - Connect the select inputs of the third mux to the outputs of the first two muxes.
   - The output of the third mux is the sum output (S).
2. Use another 4-to-1 mux to generate the carry output (Cout):
   - Connect the carry input (Cin) and the output of the third mux to two of the inputs of the mux.
   - Connect the sum output (S) and the select input of the mux to the other two inputs of the mux.
   - The output of the mux is the carry output (Cout).

.

Hybrid Multiplier

An array multiplier is a digital combinational circuit used for multiplying two binary numbers by employing an array of full adders and half adders. This array is used for the nearly simultaneous addition of the various product terms involved. To form the various product terms, an array of AND gates is used before the Adder array.

For implementation of array multiplier with a combinational circuit, consider the multiplication of two 2-bit numbers as shown in figure. The multiplicand bits are b1 and b0, the multiplier bits are a1 and a0, and the product is c3c2c1c0



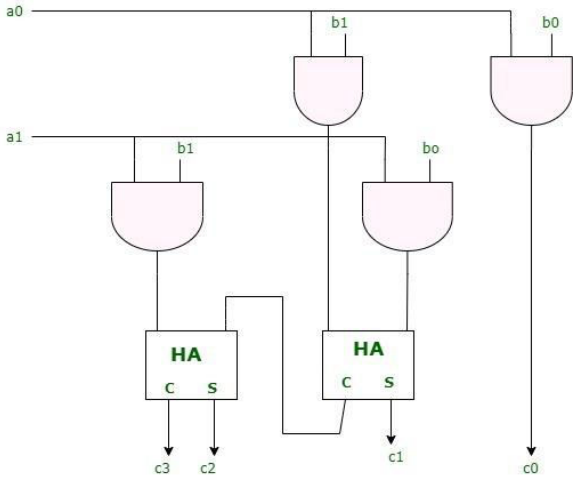
Assuming A = a1a0 and B= b1b0, the various bits of the final product term P can be written as:-

1. P(0)= a0b0
2. P(1)=a1b0 + b1a0
3. P(2) = a1b1 + c1 where c1 is the carry generated during the addition for the P(1) term.

4. P(3) = c2 where c2 is the carry generated during the addition for the P(2) term.

For the above multiplication, an array of four AND gates is required to form the various product terms like a0b0 etc. and then an adder array is required to calculate the sums involving the various product terms and carry combinations mentioned in the above equations in order to get the final Product bits.

1. The first partial product is formed by multiplying a0 by b1, b0. The multiplication of two bits such as a0 and b0 produces a 1 if both bits are 1; otherwise, it produces 0. This is identical to an AND operation and can be implemented with an AND gate.
2. The first partial product is formed by means of two AND gates.
3. The second partial product is formed by multiplying a1 by b1b0 and is shifted one position to the left.
4. The above two partial products are added with two half-adder(HA) circuits. Usually there are more bits in the partial products and it will be necessary to use full-adders to produce the sum.
5. Note that the least significant bit of the product does not have to go through an adder since it is formed by the output of the first AND gate.



A combinational circuit binary multiplier with more bits can be constructed in similar fashion. A bit of the multiplier is ANDed with each bit of the multiplicand in as many levels as there are bits in the multiplier. The binary output in each level of AND gates is added in parallel with the partial product of the previous level to form a new partial product. The last level produces the product. For j multiplier bits and k multiplicand we need j*k AND gates and (j-1) k-bit adders to produce a product of j+k bits

Array multiplier is the simplest structure of parallel multiplier. This multiplier using the standard add and shift operation based on 'add and shift' algorithms to perform a multiplication operation. The structure of 4-bit array multiplier is presented in fig. 3. The partial products generator consists of n number of 'AND' gates to multiply the multiplicand with each bit of the multiplier and then these partial products are shifted depending on their order and this summation operation can be performed by using full adder and a half adder. In 4x4 array multiplier, 4x4 AND gates used to generate partial products and 4x (4-2) full adders and 4 half adders used to generate.

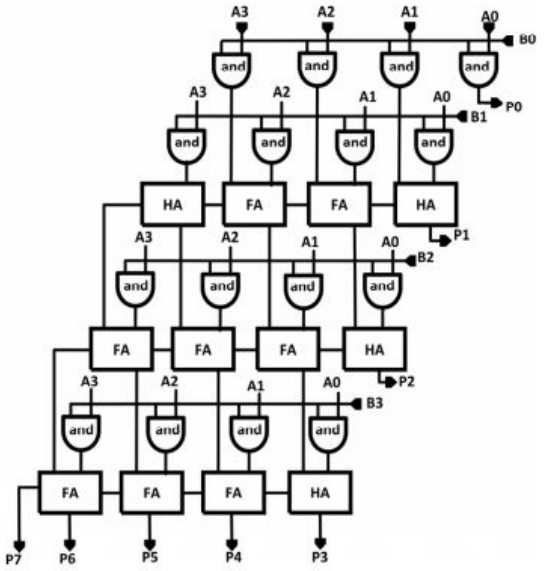


Fig. 3. 4x4 Array Multiplier

V.RESULTS AND DISCUSSIONS

The Xilinx, MATLAB software's was used throughout the whole process of programming and designing any and all of the suggested ideas. Figure 5 displays the simulation output of a system or circuit. It appears that this system takes several inputs, including LL1, LL2, and alpha, which are data inputs. Additionally, it takes input from three other sources: input file 1, input file 2, and num values, which are obtained from the Matlab environment. The simulation likely processes these inputs based on some defined algorithm or operation, and as a result, it generates an output referred to as LLNEW. This LLNEW output is then stored into an output file using Matlab.

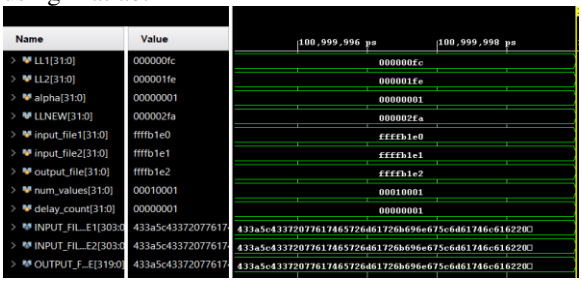


Figure 5. Simulation output

In Figure 6, the design summary is provided, which includes information about the device utilization in a digital circuit or system. Specifically, it mentions that 1518 look-up tables (LUTs) are used out of the available 41000. This information indicates that the design consumes relatively little area on the programmable logic device. LUTs are fundamental building blocks in FPGA logic. They can be configured to implement various combinational logic functions. Each LUT has a certain number of inputs and produces a corresponding output based on the configured logic function.

Resource	Utilization	Available	Utilization %
LUT	1518	41000	3.70
IO	160	300	53.33

Figure 6. Design summary

In Figure 7, the time summary of the proposed system is presented, providing important information about the timing characteristics of the system's operation. The average total delay represents the time it takes for a signal or data to propagate through the entire system. In this case, the average total delay is reported as 2.5 nanoseconds (ns). This portion of the total delay, 1.5 ns, is attributed to the time it takes for logic gates and components to process and compute data within the system. Logic delay typically includes the time spent within combinational logic elements like AND gates, OR gates, multiplexers, and other digital logic components. The remaining 1 ns of the total delay is associated with the net delay. Net delay refers to the time it takes for signals to travel along wires or interconnections between different components or elements in the system. It includes factors like signal propagation delay through wires and routing delays. Timing information is critical in digital systems design because it ensures that signals are stable and valid when they are sampled or used by other components. Meeting timing requirements is essential for the proper operation of digital systems, as it helps prevent issues like data corruption, glitches, or setup/hold time violations.

Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay
Path 1	∞	3	2	3	LL1[2]	out[2]	2.499	1.513	0.986
Path 2	∞	3	2	3	LL1[10]	out[10]	2.506	1.525	0.981
Path 3	∞	3	2	3	LL1[12]	out[12]	2.515	1.529	0.986
Path 4	∞	3	2	2	LL1[7]	out[7]	2.519	1.543	0.975
Path 5	∞	3	2	2	LL1[13]	out[13]	2.524	1.523	1.001
Path 6	∞	3	2	3	LL1[6]	out[6]	2.540	1.502	1.038
Path 7	∞	3	2	3	LL1[4]	out[5]	2.560	1.515	1.044
Path 8	∞	3	2	3	LL1[2]	out[3]	2.568	1.523	1.044
Path 9	∞	3	2	3	LL1[0]	out[0]	2.574	1.505	1.068
Path 10	∞	3	2	3	LL1[4]	out[4]	2.579	1.528	1.051

Figure 7. Time summary

In Figure 8, the power consumption of the proposed system is presented, providing detailed information about different components of power consumption. Dynamic power consumption refers to the power used by the digital circuitry during transitions, when gates switch states and signals change. It is a significant component of the total power consumption and can vary depending on the activity within the circuit. In this case, the dynamic power is reported as 118.22 micro watts (µW). Device static power, sometimes referred to as leakage power, is the power consumed by the hardware even when no computation or signal transitions are occurring. It's the power that "leaks" through transistors due to the properties of semiconductor materials. The reported device static power is 1.029 µW. Signal power is 29.025µW, which is associated with the energy required to transmit signals through interconnections and wires within the system. It includes the power needed to drive signals across the wires or to overcome the resistance in the wires. Logic power is 32.703 µW, which represents the power consumed by the digital logic elements, such as gates and flip-flops, when they are actively processing data or performing logic operations. In-out power is 58.498 µW, which is typically refers to the power associated with the inputs and outputs of the system. This includes the power required to drive signals in and out of the system, often through input and output pins or ports.

Figure 9 shows the watermarking process outcomes. Initially, the cover and watermark images are considered. After successful watermarking process, watermarked image generated, which looks like cover image. After successful watermark extraction, the extracted image looks similar to watermarked image.

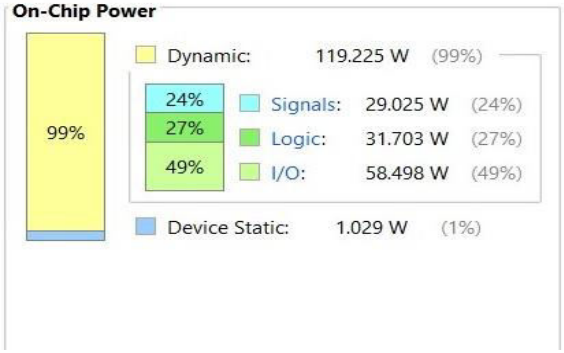


Figure 8. Power summary.

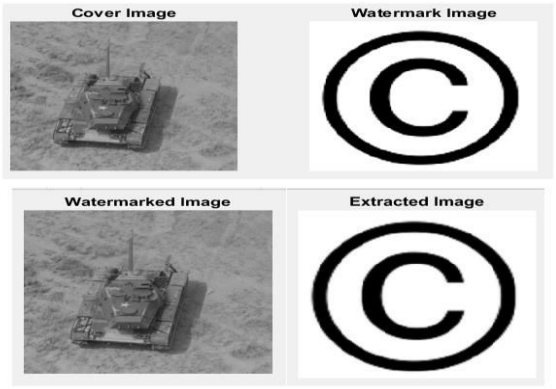


Figure 9. Watermarking process outcome.

Table 1 compares the hardware resource utilization performance. The proposed system demonstrates substantial improvements compared to the existing methods, QCA and DWT-SVD, across multiple metrics. The proposed system reduces the number of LUTs by approximately 17.68% compared to QCA and about 8.01% compared to DWT-SVD, indicating more efficient resource utilization. The proposed system significantly reduces time delay by approximately 37.34% compared to QCA and approximately 30.01% compared to DWT-SVD, demonstrating faster signal processing. In terms of logic delay, the proposed system achieves a substantial improvement of approximately 58.27% compared to QCA and about 34.10% compared to DWT-SVD, highlighting more efficient logical operations. The proposed system reduces net delay by about 53.27% compared to QCA and approximately 25.37% compared to DWT-SVD, resulting in faster data transmission. The proposed system exhibits a significant reduction in dynamic power consumption, with approximately 44.14% improvement compared to QCA and approximately 16.38% improvement compared to DWT-SVD, indicating energy-efficient operation.

The proposed system achieves a notable reduction in device static power, with approximately 71.32% improvement compared to QCA and about 58.74% improvement compared to DWT-SVD, demonstrating efficient utilization of static power resources. The proposed system reduces signal power by approximately 10.57% compared to QCA and about 5.55% compared to DWT-SVD, indicating more efficient signal management. In terms of logic power, the proposed system shows an improvement of approximately 10.93% compared to QCA and approximately 6.43% compared to DWT-SVD, reflecting more efficient logical operations. The proposed system achieves a reduction in in-out power by approximately 8.05% compared to QCA and about 3.13% compared to DWT-SVD, showcasing more efficient input and output power utilization.

Table. 1. Hardware performance Comparison Table.

Metric	**QCA [21]**	**DWT-SVD [23]**	**Proposed System**
LUTs	1844	1649	1518
Time delay (ns)	3.992	3.573	2.5
Logic delay (ns)	3.59	2.29	1.5
Net delay (ns)	2.14	1.34	1.0
dynamic power (µW)	211.453	141.384	118.20
device static power (µW)	3.59	2.49	1.029
Signal power(µW)	32.49	30.59	29.025
Logic power (µW)	36.69	34.96	32.703
In-out power(µW)	63.59	60.48	58.498

Table 2 compares the watermarking process performance comparison. The proposed system demonstrates a substantial reduction in mean square error (MSE), decreasing it from 0.00184 to an impressively low value of 0.0001, indicating a significant improvement in image quality and reconstruction accuracy. The proposed system exhibits a noteworthy improvement in PSNR, increasing it from 33.992 dB (QCA) and 35.573 dB (DWT-SVD) to a remarkable 37.71 dB, highlighting a substantial enhancement in image fidelity and quality. The proposed system achieves a significant increase in SSIM, elevating it from 0.79 (QCA) and 0.89 (DWT-SVD) to a high value of 0.98, indicating a substantial enhancement in structural similarity and image preservation. The proposed system attains a notably high Normalized Cross-Correlation (NCC) score of 0.999, representing a substantial improvement compared to QCA (0.64) and DWT-SVD (0.84), indicating highly accurate image reconstruction and alignment.

Table. 2. Watermarking performance Comparison Table.

Metric	**QCA [21]**	**DWT-SVD [23]**	**Proposed System**
MSE	0.00184	0.00164	0.0001
PSNR (dB)	33.992	35.573	37.71
SSIM	0.79	0.89	0.98
NCC	0.64	0.84	0.999

VI.CONCLUSION

This work implemented the SWT based image watermarking method. Grayscale image watermark embedding and extraction using SWT

(Stationary Wavelet Transform) is a popular technique for watermarking digital images. It involves decomposing the image into different frequency bands using the SWT algorithm, and then embedding the watermark into one or more of these frequency bands. The grayscale image is decomposed into several frequency bands using the SWT algorithm. This results in a set of coefficients representing the image in each frequency band. A subset of coefficients is selected from one or more of the frequency bands for embedding the watermark. The selection of coefficients is based on their sensitivity to changes in the pixel values. The selected coefficients are modified to embed the watermark. The modification is performed using a pseudo-random sequence, which is generated using a secret key. The sequence determines the locations and values of the watermark bits that will be embedded in the selected coefficients. Here, addition, and multiplication is implemented using VLSI based adders and subtractors. The modified coefficients are used to reconstruct the watermarked image using the inverse SWT algorithm.

REFERENCES

[1] Kumar, M. Pradeep, et al. "VLSI implementation of Digital Watermarking Technique for security and authentication of Digital Data." 2021 International Conference on Smart Generation Computing, Communication and Networking (SMART GENCON). IEEE, 2021.

[2] Jayanthi, V. E., Senthil Pitchai, and M. Smitha. "Design a hybrid FPGA architecture for visible digital image watermarking in spatial and frequency domain." Journal of Circuits, Systems and Computers 31.01 (2022): 2250020.

[3] Sakthivel, S. M., and A. Ravi Sankar. "Computation-efficient image watermarking architecture with improved performance." Computers & Electrical Engineering 84 (2020): 106649.

[4] Bhat, Anil K. "VLSI Implementation and Software Co-Simulation of Digital Image Watermarking with Increased Security." International Journal of Sensors Wireless Communications and Control 10.4 (2020): 634-638.

[5] Kaibou, Redouane, et al. "Real-time FPGA implementation of a secure chaos-based digital crypto-watermarking system in the DWT domain using co-design approach." Journal of Real-Time Image Processing 18.6 (2021): 2009-2025.

[6] Thakur, S., et al. "Improved DWT-SVD-based medical image watermarking through hamming code and chaotic encryption." Advances in VLSI, Communication, and Signal Processing: Select Proceedings of VCAS 2018. Springer Singapore, 2020.

[7] Nagaraj, P., et al. "VLSI implementation of image compression using TSA optimized discrete wavelet transform techniques." 2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT). IEEE, 2020.

[8] Joshi, Amit M., Samar Ansari, and M. Ravikumar. "VLSI architecture of integer DCT based watermarking with error correction capability." TENCON 2017-2017 IEEE Region 10 Conference. IEEE, 2017.

[9] Das, Subhajit, Reshmi Maity, and N. P. Maity. "VLSI-based pipeline architecture for reversible image watermarking by difference expansion with high-level synthesis approach." Circuits, Systems, and Signal processing 37 (2018): 1575-1593.

[10] Joshi, Amit M. "VLSI implementation of video watermarking for secure HEVC coding standard." Cryptographic and Information Security. CRC Press, 2018. 353-377.

[11] Jana, Poulami, and Amit Phadikar. "Low-power FPGA-based hardware implementation of reversible watermarking scheme for medical image." Computational Advancement in Communication Circuits and Systems, 2019.

[12] Maity, Goutam Kumar, et al. "Power-aware VLSI design of reversible watermarking for access control." Microsystem Technologies, 2019.

[13] Charles, Subodha, Vincent Bindschaedler, and Prabhat Mishra. "Digital watermarking for detecting malicious intellectual property cores in noc architectures, 2022.

[14] Dwivedi, Ranjana, and Vinay Kumar Srivastava. "Geometrically Robust Digital Image Watermarking Based on Zernike Moments and FAST Technique." Advances in VLSI, Communication, and Signal Processing, (2022).

[15] Prabhu, V., et al. "A discrete wavelet transform-based audio watermarking technique for digital security, 2022.

[16] Baaouni, J., Choura, H., Chaabane, F., Frikha, T., & Baklouti, M. (2022, July). “Design of Multiprocessor Architecture for Watermarking and Tracing Images Using QR Code”. In Intelligent Decision Technologies: Proceedings of the 14th KES-IDT 2022 Conference (pp. 109-122). Singapore: Springer Nature Singapore.

[17] Sinhal, Rishi, Deepak Kumar Jain, and Irshad Ahmad Ansari. "Machine learning based blind color image watermarking scheme for copyright protection, 2021.

[18] Kavitha, R. S., U. Eranna, and M. N. Giriprasad. "DCT-DWT based digital watermarking and extraction using neural networks, 2021.

[19] Nanammal, V., B. M. Abirami, and J. Venugopalakrishnan. "VLSI based desgin of an efficient hybrid water marking scheme for

multimedia content protection." Indian Journal of Science and Technology 8.19 (2015): 1-8.