ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 2, Issue 4, July 2013

Invisible Watermark Algorithm Embedded in VLSI Chip for Authenticating Digital Image

Mustafa Osman Ali [(1)], Isam Abdelnabi Osman [(2)], Rameshwar Rao[(3)]
[(1)(3)]Electronics and Communication Engineering Department, [(2)] Electronics Engineering School
[(1)(3)]Osmania University, [(2)] Sudan University of Science and technology
[(1)(3)]Hyderabad - India, [(2)] Khartoum - Sudan

***Abstract*— Authenticating transaction of the digital media becomes an active research field now a day. It is emerging due to the illegal use of data resources and hacking by unauthorized users. In this article, we implemented new watermarking system to be modeled as a System-on-Chip (SoC).The implemented system is designed according to the embedding algorithm which we had proposed in [1]. The implemented system is designed to embed invisible watermark in spatial domain of digital images. A secret key is used to embed and extract the watermark as well. The watermark, according to the secret key, is located in a random manor vertically or horizontally across the base-image. The chip interface and localization are shown in this article. The system is modeled by using Xilinx ISE 9.2i and coded by Verilog HDL. We believe that our system is useful and has a high level of reliability to authenticate digital images.**

Index Terms—**Base-Image, Mark-Image, Marked-Image, Embedding, Extracting.**

I. INTRODUCTION

More information is transmitted in a digital format now than ever, such as digital images, digital audio, and digital videos. Digital information is susceptible to having copies made at the same quality as the original. A digital watermark is carried out to provide copyright protection for intellectual property that is in digital format. Digital watermarking is a technique providing embedded copyright information in images. The watermark can be either visible or invisible. Visible watermark is well known because every user can observe it and may be has some knowledge about the purpose of its use; such as watermark in the banknotes, formal documents – e.g., academic certificates, TV. Channels, etc...

Invisible watermark gives the watermarking technology the similarity if get compared with steganography; both of them hide a secret message inside an image. But steganography hide a sensitive secret that to be delivered safely to the other side without notice of unauthorized user, where watermarking hide information in an image that can be used later to verify rights of an author and his ownership of the carrier image. The invisible watermark may be embedded as fragile or robust. The invisible fragile watermark is embedded on the primary image in such a way that any manipulation or modification of the image would alter or destroy the watermark. The invisible robust watermark is embedded on the primary image in such a way that an alteration made to the pixel value is perceptually not noticeable and it can be recovered only with an appropriate decoding mechanism [2].

II. WATERMARKING SYSTEM: HARDWARE VERSUS SOFTWARE

A watermarking system can be implemented with either software or hardware. The software implementation of the watermarking algorithms is significantly large, whereas the hardware implementation of the algorithms is lacking [3]. In a software implementation, the algorithm's operations are performed as code running on a microprocessor [4]. This code should be stored in a memory e.g., RAM and require a dedicated processor that occupies more area, consumes significantly more power, and may still not perform adequately fast.

Although it might be faster to implement an algorithm in software, there are a few compelling reasons for a move toward hardware implementation. In a hardware implementation the algorithm's operations are fully implemented in custom-designed circuitry. This investigates great advantages such as reduce hardware scheme area, decrease power consumption and increase speed of performance [3],[4],[5],[6]. Therefore a hardware watermarking solution are often more economical. Generally, the hardware watermarking scheme can be done by using each of the domains (spatial or frequency). But due to the simplicity of spatial domain computational overhead and its easiness for its application if compared to the frequency domain, the spatial domain is usually preferred for hardware implementation [3], [5],[6],[7].

III. EMBEDDING ALGORITHM

This algorithm was illustrated in details in [1]. Simply this algorithm embeds a watermark in specific area in a base-image. The watermark is going to embed a cross the base-image horizontally through a group of eight blocks, or it can be embedding vertically. The selection group is chosen according to the parameters of the secret key. The watermark – mark-image – is going to spread over the eight blocks bit by bit. So, each bit from the mark-image will embed in a corresponding byte in the base-image. The embedding algorithm will resize the mark-image to be equal to the single block size in the base-image. The size of the base-image will be set to [256 × 256]; so, the size of each block is [32 × 32] similar to the mark-image new size. With reference to [1], the embedding mechanism of the algorithm is designed according to the issue and sequence of (1), (2), and (3). These equations run sequentially in a *nesting for loop* consists of three *for loops*, (Pixel Loop, Row Loop, and Block Loop).

$$I_B = \sum_{i=m}^{M} \sum_{j=n}^{N} (I_{B_{ij}} \wedge \ 254) \qquad \dots \ (1)$$

$$Mask_{Emb} = \sum_{p=m'}^{M/8} \sum_{q=n'}^{N/8} \left(\frac{\left(I_{W_{pq}} \wedge \ 2^{(K-1)} \right)}{\left(2^{(K-1)} \right)} \right) \dots \ (2)$$

$$I_H = \sum_{i=m}^{M} \sum_{j=n}^{N} (I_B \vee Mask_{Emb}) \qquad \dots \ (3)$$

Where: I_B, I_W and I_H represent the base-image, mark-image and marked-image respectively. M and N are the dimensions of the base-image. Parameters m and n are the values of the first pixel dimensions of the selected group; where m' and n' indicate the values of the first pixels of a certain block in the selected group. K is the counter of the block sequence loop counting up to eight. Lastly, $Mask_{Emb}$ is a byte value which is used to embed a bit from I_W into I_B to get I_H. The $Mask_{Emb}$ is carried out from the Mark-Mask byte by shifting right operation to adjust the extracted bit into LSB position. Fig. 1, explains how the nesting loop counters trace the base-image pixels; this nesting will embed mark-image horizontally in the first group of rows.

The main target of this embedding mechanism is to embed a single bit taken from the mark-image into a single specified byte in the base-image. It aims to hide the mark-image inside the base-image; therefore, the expected result is an invisible watermark resides in a certain image. The embedding algorithm is going to embed a watermark randomly automatically to achieve more complexity and security as well as using a secret key. The algorithm is capable to randomize the location of the watermark in different base-images. On the other hand, an extraction algorithm is prepared and tested successfully. Both algorithms are coded in Matlab 7.9.0 (R2010b) [1].

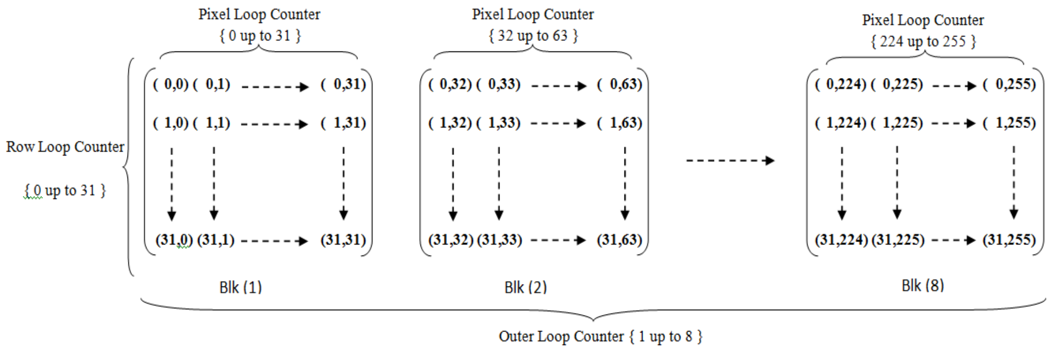


Fig. 1, Embedding Algorithm Nesting Loop Counters Rotation

IV. The Proposed Embedding System Design

The most important part of the embedding algorithm is the “Embedding Mechanism” stage. This stage has three inputs, base-image pixel, mark-image pixel, and secret key parameters. The product output of these three inputs is a marked-image pixel. The 1st and 2nd inputs are in fact 8-bits e.g., byte and are the base elements of the embedding mechanism. The third input is used for adding more security features to the algorithm and it consists of many parameters [1]. From the “Embedding Mechanism” stage the hardware implementation design can be carried out manually. To get started, basically the design will omit the third input; because it represents a set of conditions that will be used later to extract watermark and the incentive in this unit is about embedding watermark. The primary system is carried out of the proposed embedding system is shown in Fig. 2.

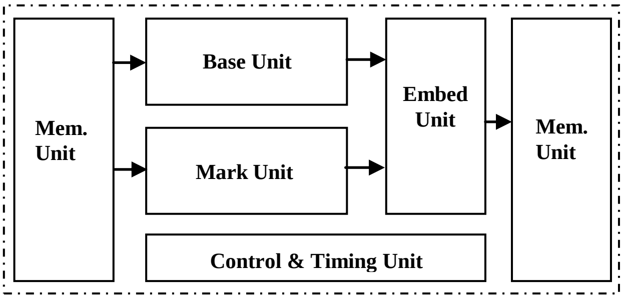
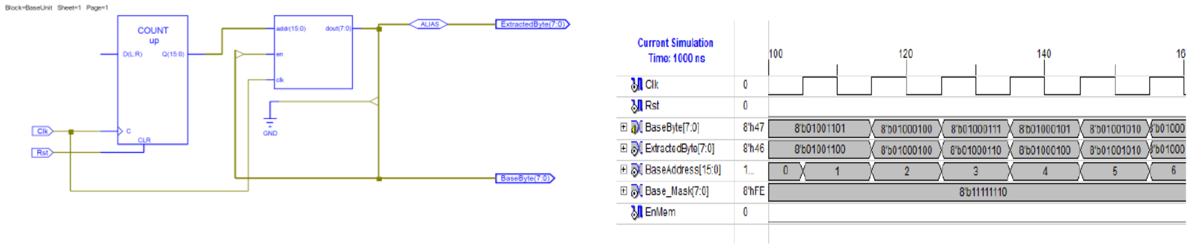


Fig. 2, The Proposed Embedding System Structure

The Verilog HDL is used to generate the proposed system into four integral codes, ‘Base Unit’, ‘Mark Unit’, ‘IP-Core Memory’, and Watermarking System. Both ‘Base Unit’ and ‘Mark unit’ are using ‘IP-Core Memory’ in their structures. The following sections will offer the details of these codes.

A. The ‘Base Unit’ Structure

The main task of this unit is providing the base-image bytes byte by byte after extracting LSB. But it doesn’t need to extract the LSB from the whole bytes of base-image! Except the bytes over the position where watermark will embed in; therefore a control signal ‘Embed’ is used. This control signal controls the extraction of the LSBs from the base-image. Two more control signals are needed, Clk and Rst signals. The unit has two outputs: ‘Extracted Byte’ and ‘BaseByte’, where the first one is the base-image pixel after its LSB is extracted, and the second one is the original base-image pixel without extracting its LSB. One of these outputs will be released out according to the ‘Embed’ signal state. Fig. 3, shows the RTL schematic of the ‘Base Unit’ and its generated waveform time diagram.



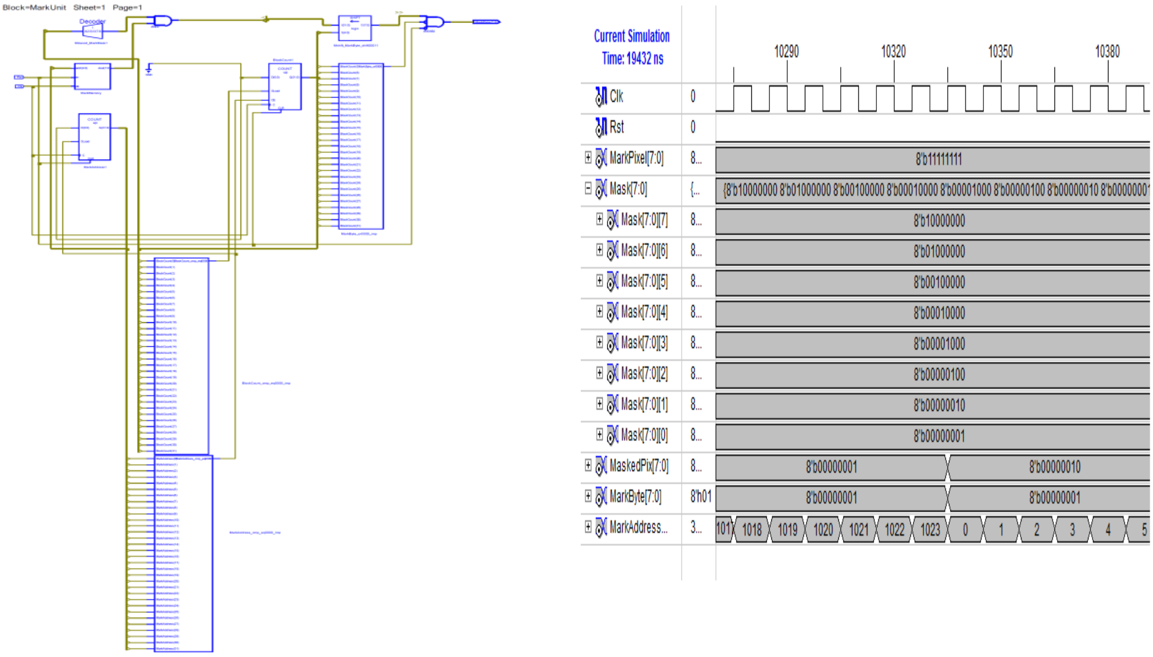
(a) **(b)**

Fig. 3, The ‘Base Unit’ HDL Model.
(a) ‘Base Unit’ RTL Structure. (b) ‘Base Unit’ Waveform Time Diagram

B. The ‘Mark Unit’ Structure

The ‘Mark Unit’ is using two control signals: ‘Clk’ and ‘Rst’ and has only one output. The input of this unit is a byte –‘MarkByte’- that comes from ‘MarkMemory’. Two logical operations summarize the unit task. The first operation is ‘AND’ between ‘MarkByte’ and ‘MarkMask’. The ‘MarkByte’ will be extracted bit by bit so as to be embedded into eight different base-image pixels. The extraction operation executed using ‘MarkMask’ which is a byte of Zeros with unique One. This unique One allows extracting mark-image pixel bits bit by bit in separate intervals by shifting left eight times. The ‘MarkMask’ is generated internally in the ‘Mark Unit’ according to the sequence of the operations controlled by ‘Clk’ and ‘Rst’.

The second operation is shift right operation. This operation is applied to the output byte of the extraction operation – eight dual and-gates output byte. Since the extracted bit is going to embed in the LSB of the base-image pixel; shifting operation is executed to match the order of extracted bit in the output byte to the LSB of the base-image extracted pixel. The number of shifting right times is varying into (0 to 7), and it is setup internally according to the logical conditions in the unit controlled by ‘Clk’ and ‘Rst’. Fig. 4, shows the RTL schematic of the ‘Mark Unit’ and its generated waveform time diagram.



(a) **(b)**

Fig. 4, The ‘Mark Unit’ HDL Model.
(a) ‘Mark Unit’ RTL Structure. (b) ‘Mark Unit’ Waveform Time Diagram

C. The ‘IP-Core Memory’ Structure

Two single port memory units are needed in the designs of ‘Base Unit’ and ‘Mark Unit’, base-image [256 × 256] Memory (width: 8 depth: 65536), and mark-image [32 × 32] Memory (width: 8 depth: 1024). To construct these memories in Verilog, HDL *Core Generator* is used. The *Core Generator* is generally used to model HDL units which use *index pointer* (IP). It enables a programmer/designer to combine pieces of IP-core in order to form a custom SoC within an FPGA such as memories, stacks, queue, …, etc.

Here is an explanation of how a single port block memory can be modeled and then used as a ‘Base Memory’, but for further details, a reader is advised to refer to *XILINX* user manual. Before starting the memory modeling, an initial data must be stored basically in a coefficient-file (.coe file). The coefficient-file is a file that can be loaded into the *XILINX* core generator. The base-image can be converted into the corresponding .coe file using a suitable converting tool such as *VIM converter* or writing a Matlab m-function file. By using Matlab m-function or VIM tool a .coe file can be generated and saved as ‘filename.coe’ e.g., ‘base.coe’. The content of this file will be 65536 hexadecimal byte values for base-image [256 × 256].

In Verilog HDL environment a new source –IP (Coregen & Architecture Wizard)- should be created inside the model of the ‘Base unit’. This new source should have a name such as: ‘BaseMemory’ and then proceed in its modeling to assign the features of the memory and loading .coe file in it. Same process will be done to generate ‘MarkMemory’ with depth of 1024 locations e.g., [32 × 32].

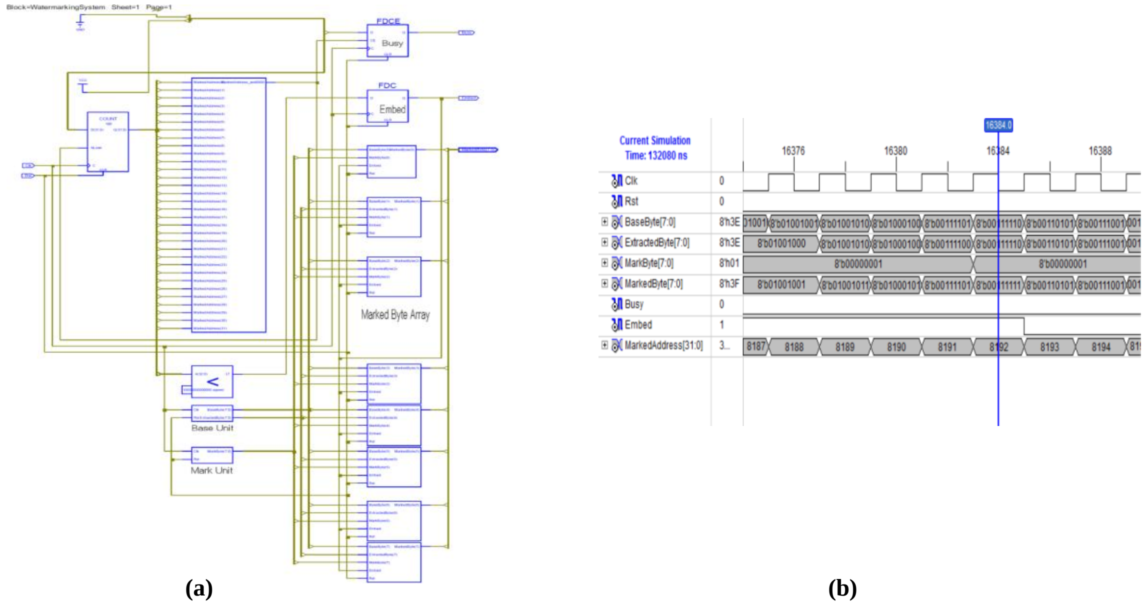
D. The Top Model ‘Watermarking System’ Structure

With reference to the Fig. 2, three units had been modeled, only two units are bending for modeling: ‘Embed Unit’ and ‘Control & Timing Unit’. The remaining units are going to be coded into one HDL structure called

‘Watermarking System’. This structure has two inputs: ‘Clk’ and ‘Rst’, and it has three outputs: ‘MarkedByte’, ‘Busy’, and ‘Embed’. There are more two inputs and assumed to be used as internal inputs, ‘BaseByte’ and ‘MarkByte’. The ‘BaseByte’ is provided by the ‘Base Unit’ and the ‘MarkByte’ is generated by ‘Mark Unit’ as illustrated before.

As it’s declared before, the ‘Base Unit’ provides two outputs: ‘ExtractedByte’ and ‘BaseByte’. The ‘Watermarking System’ is going to select either of these outputs by using the ‘Embed’ signal as internal control signal beside its usage as an output indicator. The ‘Embed’ signal enables the extraction of LSBs from the ‘BaseByte’ to become ‘ExtractedByte’ when it’s ‘High’ (e.g., Embed = 1), otherwise the ‘BaseByte’ is going to be assigned as an output for the ‘Base Unit’ directly. One more output signal is ‘Busy’. It indicates that the “watermarking System’ is currently generating marked-image. This signal is useful when this system interfaces with another system.

The main output of the ‘Watermark system’ is a byte named as: ‘MarkedByte’. The expected number of output bytes is equal to the base-image bytes number. So, a memory is needed in this structure, but it is omitted here for a later handshaking interface. Fig. 5, shows the RTL schematic of the ‘Watermarking System’ and its generated waveform time diagram.

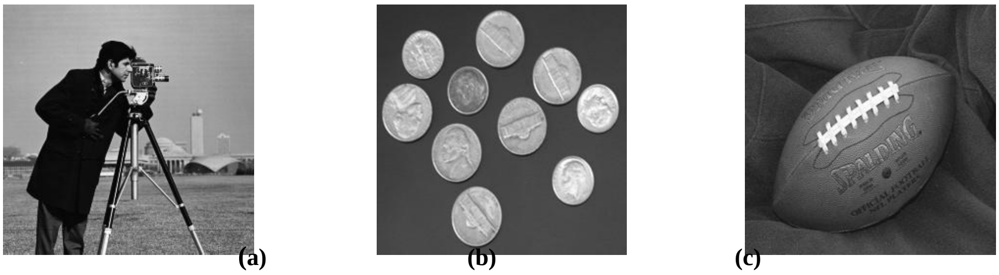


(a) **(b)**

Fig. 5, The ‘Watermarking System’ HDL Model.
(a) RTL Structure (b) Waveform Time Diagram.

V. Results and Discussion

The proposed model for watermarking system had been implemented out of the issues of our proposed algorithm in [1]. The results show that the proposed system has the ability to insert an invisible watermark into a spatial domain of a base-image. Tens of digital images with various types have been used to examine the embedding and extracting algorithms’ performance. Most of these images are available in Matlab library, but also other special images have been used. The types of the tested images are ‘.jpg’, ‘.png’, and ‘.tif’. Fig. 6, shows the achieved marked-images produced by the implemented system. The logo of ***Osmania University, Hyderabad, India*** is used as a mark-image in the shown marked-image; and it is embedded randomly as invisible watermark correctly.



(a) **(b)** **(c)**

Fig. 6, The Reconstructed Marked-images Produced by The Watermarking System.
(a) Cameraman.tif,(b) Coins.png, and (c) Football.jpg.

The mean squared error (*MES*) and peak signal-to-noise ratio (PSNR) are computed for the hardware resulted images. The obtained values were listed in table I. Generally, these values are in the acceptable range and match the standard imperceptibility evaluation; since they are reside in the range between 50 and 30. But compared to the values that gained in [1], the latest values have less quality. Simply, this evaluation is decided according to the fact of the lesser MSE means excellent quality and vice versa for PSNR [3],[5]. According to that, the implemented system yields marked-images with high imperceptibility and robustness quality. The system is able to randomize the location of the watermark in different base-images.

The watermark - e.g., Osmania University logo- is extracted correctly by using an extracting algorithm proposed in [1]. The mean of MES for the extracting watermarks is '1.0125', and the mean of PENR is '48.0770'. These calculated values support our system's capability and reliability.

Table. (I). MSE and PSNR for the Tested Images

Image	**MSE**	**PSNR**
Cameraman.tif	*1.0047*	*48.1105(db)*
Mandi.tif	*1.0060*	*48.1050 (db)*
Coins.png	*1.0041*	*48.1131 (db)*
Pears.png	*1.3516*	*46.8224 (db)*
Football.jpg	*1.0555*	*47.8960 (db)*
Office_5.jpg	*1.1897*	*47.3766 (db)*

VI. THE PROPOSED CHIP AND APPLICATIONS

A. Chip Interface

The modeled system which had been shown in Fig. 5, has only twelve interface pins; Clk pin, Rst pin, Busy pin, Embed pin, and eight pin for output byte. Actually, this number of pins is not satisfying to hand shake and operate the system in the real application. In the hardware model, there was no need for an image input because images are already stored in the memories which are built inside the design. But in the real application indeed the system is in need.

The proposed interface pin layout is shown in Fig. 7-a. Eight pins are associated for input byte of the base-image. Three control pins are added to the proposed system, 'Enable', 'Load', 'Key_Increment'. Enable pin is used to enable the chip and it is usually set as active 'Low'.

Load pin is added to allow system to load the mark-image (watermark) in the 'Mark Memory' which is built inside the system. The mark-image is going to be loaded into the internal memory through the same input of the base-image byte, e.g., 'Data_In'. The sharing of the input pins by the base-image and mark-image will not affect the performance of the system; this is because the images are not going to load into the system in the same time. Usually, mark-image is loaded in advance and at varying times according to the user's desire.

Key_Increment pin is used for incrementing the image counter; this counter exists in the proposed system and is used to count base-images that have been treated by the system. This counter should be similar to the counter of the digital system device – e.g., camera- which the proposed system will associate in. The 'Key_Increment' signal is the same control signal that controls the counter of the digital system device. The importance of this pin is in enabling the counter to provide the initial key, which will be used to set the secret key and determine the embedding location (*for further explanation see [1]*).

The output pin 'Embed' has no need in the proposed chip interface; it is just added in the watermarking system prototype to monitor the limitation of the embedding operation (starting and ending). But 'Busy' output is useful because it indicates the host system that the base-image is under process and also acknowledges that the base-image had been watermarked when it becomes 'High'.

The proposed chip interface offers a chip in a standard number of pins since it has 22 pins in addition to two more for biasing voltage and ground. So, the total number of pins is 24 which is standard pins number in VLSI technology.

B. Chip Localization

The main target for designing the hardware watermarking system is to enhance digital image system security. The system will be able to secure image in the origin time of capturing it. This means that the proposed chip must be located inside the structure of the digital image system. To illustrate this idea, a digital camera will be considered as a digital image device. Fig. 7-b, shows a typical digital camera system architecture.

The digital camera system simply consists of five basic units in addition to input and output units. The input unit normally is the physical lens, and output is the display unit *(LCD).* The light rays from the sight are charged through the lens into a *Charged Coupled Device (CCD) sensor*, which is sensing for the luminous of the basic three colors, *Red, Green, and Blue (RGB).* The output of the sensor is converted to digital pulses using *(ADC).*

The image frame and format will be recognized in the *Digital Signal Processor (DSP)* unit. And then the image is going to be stored in a memory in the same time while it is displaying in the output unit. All processes and sequencing activities are controlled and timed by the system controller.

From this brief explanation, it becomes clear that the suitable stage to locate the proposed watermarking chip is between DSP unit and memory. This is because the output of the DSP unit is a ready digital image. Since the image is going to be stored in the memory byte by byte, the proposed chip will process these bytes sequentially and then release them so as to be stored in the memory.

The hand shaking between the digital camera system and the proposed watermarking chip is done as in the following by tied *DSP* output to chip ‘*Data_In*’, Chip ‘*Data_out’* tied to the digital camera memory, *Clk, Rst, Enable, Load,* ‘*Key_Increment*’ tied to digital camera controller, and ‘*Busy*’ tied to digital camera controller.

Note: ‘*Data_in*’ should have a connection with the main system data path for interfacing with external memory, flash, or PC in order to load mark-image.

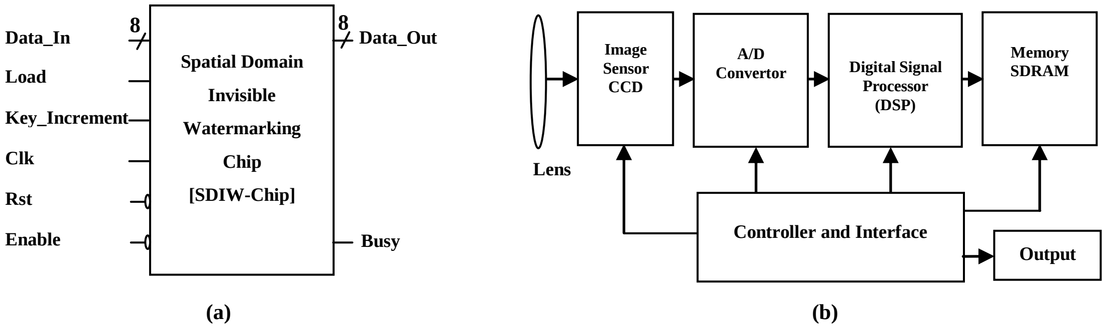


Fig. 7, The Proposed Chip Application.
(a) Chip Lay out Pin Diagram. (b) Digital Camera System Architecture.

VII. Conclusion

A primary system design is carried out manually to realize the embedding algorithm that had been proposed in [1]. The proposed system hardware consists of five units: ‘Base Unit’, ‘Mark Unit’, ‘Embed Unit’, ‘Memory Unit’, and ‘Control & Timing Unit’. From the proposed system and by using Verilog HDL a complete system for watermarking has been modeled. The ‘Watermarking System’ scheme is built out of three integrated HDL models: ‘Base Unit’ model with build-on ‘Base Memory’, ‘Mark Unit’ model with build-on ‘Mark Memory’, and the top model ‘Watermarking System’ which contains and maintains all the stated units. The proposed watermarking system has been capsulated in a chip format and the pin layout is shown. The modeled HDL system is perfectly run and successfully performed all the tasks leading to achieve good results.

REFERENCES

[1] Mustafa Osman Ali, Elamir Abu Abaida Ali Osman, and Rameshwar Row. “Invisible Digital Image Watermarking in Spatial Domain with Random Localization”. In International Journal of Engineering and Innovative Technology (IJEIT), 2(5): pp. 227-231, November 2012.

[2] Rajesh Kumar, Cisco Security Bible: 1st Eddtion, IDG Books India, 2002.

[3] 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 Proc. of the IEEE Workshop on Signal Processing Systems, pp. 183-188, 2003.

[4] N. J. Mathai, D. Kundur, and A. Sheikholeslami. Hardware Implementation Perspectives of Digital Video Watermarking Algorithms. In MATHAI et al.: Hardware Implementation Perspectives of Digital Video Watermarking Algorithms. Proc of IEEE Transaction on Signal Processing, 51(4): pp. 925-938, 2003.

[5] A. Basu, T. S. Das, S. Maiti, N. Islam, and S. K. Sarkar. FPGA Based Implementation of Robust Spatial Domain Image Watermarking Algorithm. Proc. in International Conference on Computers and Devices for Communication. 2009.

[6] A. Basu, T. Das, S. Sarkar, A. Roy and N. Islam. FPGA Prototype of Visual Information Hiding. IEEE. 2010.

[7] D. Samanta, A. Basu, T. S. Das, V. H. Mankar, A. Ghosh, M. Das, and S. K Sarkar. SET Based Logic Realization of a Robust Spatial Domain Image Watermarking. In IEEE (ICECE). Proc. of 5th International Conference on Electrical and Computer Engineering. Dhaka, Bangladesh, pp. 986-993, 2008.

AUTHOR BIOGRAPHY

The first author did his B.Sc. and M.Sc. degrees in Computer Engineering at Sudan University for Science and Technology (SUST), Khartoum, Sudan in 2006. Currently, he is pursuing his Ph.D. degree in VLSI at Osmania University, Hyderabad, India. His research interests include Image Processing, Computer Interfacing, and Digital Communications. Mr. Mustafa Osman Ali, is a senior Lecturer in Nile Valley University, Eng. College, Atbara, Sudan since March 2003 – he was a former head of Electrical & Electronic Eng. Dept. for three years (2007 – 2010). Also he is assistant professor in SUST University, Elshikh Abdallah Elbadri Technical College, and open education in his country. Also he is a member in Sudanese Engineering Sociaty, And Sudanese Engineering Council, Khartoum, Sudan.

The second author has received his B.Sc. degree in Computer Engineering at Sudan University for Science and Technology (SUST), Khartoum, Sudan, PG Diploma degree in Advanced Information Technology at International Institute of Information Technology (I^2IT) Pune, India in 2003, and M.Sc. degrees in Information Technology (Computer Network Systems) at (SUST), Khartoum, Sudan in 2005. Currently, he is working towards the Ph.D. degree in Computer Engineering at SUST. His research interests include RCS reduction, Stealth technology, Computer Interfacing, and Digital Communications. Mr. Isam Abdelnabi, is a senior lecturer in SUST, since July 2001 – he was a head of Scheduling and Examination. Dept. for five years (2007 – 2012).

The third author has obtained his Bachelor of Engineering in Electronics and Communication Engineering from University College of Engineering, Osmania University, Hyderabad. He obtained both his M.Tech in Communication Engineering and Ph.D from IIT, Bombay. His interests encompass: Digital Communication, Digital Design, Computer Networks, VLSI Design, and Mobile Cellular Communication

Prof. Rameshwar Rao, had been the *Dean* of the University College of Engineering, Osmania University, Hyderabad for three years; and right now he is the *Vice Chancellor* of JNTU, Hyderabad. He guided more than 20 Ph.D students so far.