IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.Doi Number

On-The-Fly Parallel Processing IP-Core for Image Blur Detection, Compression, and Chaotic Encryption Based on FPGA

Ahmed A. Rezk[1], Ahmed H. Madian[3,4], (Senior member, IEEE), Ahmed G. Radwan[2,3], (Senior member, IEEE), and Ahmed M. Soliman[5], (Life senior member, IEEE).

[1]University of Science and Technology, Zewail City, Giza 12578, Egypt
[2]Engineering Mathematics and Physics Department, Faculty of Engineering, Cairo University, Giza 12613, Egypt
[3]Nanoelectronics Integrated Systems Center (NISC), Nile University, Giza 12588, Egypt
[4]Radiation Engineering Department, NCRRT, Egyptian Atomic Energy Authority, Cairo 13759, Egypt
[5]Electronics and communications Engineering Department, Faculty of Engineering, Cairo University, Giza 12613, Egypt

Corresponding author: Ahmed A. Rezk (e-mail: ahrezk@ zewailcity.edu.eg).

ABSTRACT This paper presents a 3 in 1 standalone FPGA system which can perform color image blur detection in parallel with compression and encryption. Both blur detection and compression are based on the 3-level Haar wavelet transform, which is used as a common building block to save the resources. The compression is based on performing the hard thresholding scheme followed by the Run Length Encoding (RLE) technique. The encryption is based on the 128-bit Advanced Encryption Standard (AES), which is considered one of the most secure algorithms. Moreover, the modified Lorenz chaotic system is combined with the AES to perform the Cipher Block Chaining (CBC) mode. The proposed system is realized using HDL and implemented using Xilinx on XC5VLX50T FPGA. The system has utilized only 25% of the available slices. Furthermore, the system can achieve a throughput of 3.458 Gbps, which is suitable for real-time applications. To validate the compression performance, the system has been tested with all the standard 256×256 images. It is shown that depending on the amount of details in the image, the system can achieve 30dB PSNR at compression ratios in the range of (0.08-0.38). The proposed system can be integrated with digital cameras to process the captured images on-the-fly prior to transmission or storage. Based on the application, the blurred images can be either marked for future enhancement or simply filtered out.

INDEX TERMS AES, Blur, Chaos, Compression, DWT, Encryption, FPGA, Haar, HDL, RLE.

I. INTRODUCTION

Nowadays, the digital images are used as a source of information in many fields, such as the social media, education, research, medical examinations, and surveillance systems. Accordingly, extensive research efforts have been conducted throughout the past years to facilitate the use of the digital images in a both efficient and secure way. The three main research fields are: image compression, encryption, and image processing. The image blur detection is one of the important applications in the field of image processing, and it will be presented along with the compression and encryption throughout this paper.

A. Image compression

The image compression is one of the most popular applications that is mainly used for saving both the storage space and the network's bandwidth. The general structure of the lossy image compression is shown in Fig. 1 [1]. First, the source encoder is used to transform the input image into a sparse representation. The source encoder can be constructed using a variety of transforms, such as the Discrete Wavelet Transform (DWT), the Discrete Cosine Transform (DCT), or the Discrete Fourier Transform (DFT) [2]. The DWT is a computationally efficient algorithm, and it has been adopted in the JPEG-2000 standard [1]. Despite the efficient compression ratios that can be achieved by the DWT, there are some limitations, such as the oscillations, the shift variance, the aliasing, and the lack of directionality [3]. The complex

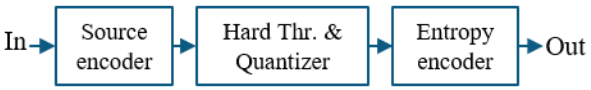


FIGURE 1. The general structure of the lossy image compression.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3086044, IEEE Access

wavelet transform can overcome these limitations, but at the expense of the computational cost [3]. Moreover, the fractional wavelet packet transform can be used with signals that contain high fractional frequency components, but its computational cost is also higher than the DWT [4]. Therefore, the DWT will be more suitable for real-time applications that require an efficient implementation area and high operating speed.

After the source encoder, the hard thresholding is performed to reduce the number of non-zero coefficients in the transformed image. Then the quantization is used to decrease the number of bits for each non-zero coefficient. Finally, the entropy encoder is used to perform a lossless compression technique, such as the Huffman encoder, the arithmetic encoder, or the Run Length Encoder (RLE). The Huffman encoder is used in the JPEG standard and it can provide good compression results, however the RLE is more suitable for real-time applications [1].

B. Image encryption

In addition to the compression process, the images must be also encrypted before sending them over insecure communication channels. Since the encryption process destroys the correlation between the pixels, it is usually done after the compression. A lot of encryption standards are available, and there is always a tradeoff between the encryption strength and the computational cost.

The Advanced Encryption Standard (AES) is an ISO/IEC 18033-3:2010 standard for symmetric block cipher. The AES is one of the most powerful encryption standards, and it has been used in a lot of protocols, such as the IEEE802.11 wireless Local Area Network (LAN) and the IEEE802.15.4 wireless sensor networks [5]. The AES encryption has been frequently used in the Cipher Block Chaining (CBC) mode, which needs a Pseudo Random Number Generator (PRNG) to generate the Initialization Vector (IV).

The PRNGs are based on deterministic functions that are implemented in the digital domain using the computers or the Field Programmable Gate Arrays (FPGAs). In the previous years, the chaos theory has played an important role in the design of the PRNGs, image hashing [60], and modulation schemes [61]. The Rössler [6] and Lorenz [7] are two of the most famous chaotic systems, and they have been utilized in a lot of encryption applications [8,9]. However, these two systems are based on multiplication operations that affect the hardware performance, regarding the area and speed. Hence, modified versions, called the modified Rössler and the modified Lorenz, were presented in [10,11]. These modified versions do not utilize any hardware expensive multiplier.

In [12], six different multiplier-less chaotic PRNGs were implemented using FPGA, and their performance was compared using the NIST suite. It has been found in [12] that the modified Lorenz can pass all the NIST tests and can provide the best hardware performance as well. Hence, it will be suitable for low area and high-speed applications.

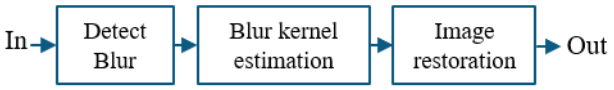


FIGURE 2. The general structure of image restoration.

The modified Lorenz is modelled using (1), where a, b, and c are the system's parameters while X, Y, and Z are the system's variables. The modified Lorenz hardware implementation is presented in detail in [12].

$$\dot{X} = a.(Y - X) \quad (1a)$$
$$\dot{Y} = -sgn(X).(Z - b) \quad (1b)$$
$$\dot{Z} = sgn(X).X - c.Z \quad (1c)$$

C. Image blur detection

The image blur detection is one of the fundamental applications in the field of image processing, and it is usually performed at the start of any image restoration algorithm as shown in Fig. 2 [13]. Similar to the image compression, the blur detection requires image transformation at the beginning.

A lot of no-reference image blur detection techniques have been proposed in the literature [14]. The blur metrics of these techniques are based on various approaches, such as the Haar DWT [15], the Sobel edge detection [16-19], the image power spectrum [20,21], the DCT [22,23], and a hybrid of curvelet, wavelet, and cosine transforms [24]. The simplest of these approaches is the Haar DWT [15], and it can be implemented on the hardware level without using a lot of resources.

Tong et al.'s [15] Haar-based blur detection algorithm can discriminate blurred images with high accuracy. This algorithm was tested on a database that include 2355 images, and the reported accuracy was 98.6% [15]. Furthermore, this algorithm can be combined with neural networks to assess the overall image quality [25-27]. In [27], the output of Tong et al.'s algorithm was considered the best performing feature that can be used for image quality assessment using neural networks. Also, Tong et al.'s algorithm can be used to filter out blurred images in many applications, for example the lifelogging wearable cameras [28].

D. The FPGA-based implementation

Image processing techniques in addition to the compression and encryption are usually implemented using software solutions that run on general-purpose processors. However, due to the sequential nature of these processors, the execution time will not be suitable for real-time applications, especially with high resolution images. This problem can be solved by using the Hardware (HW) solutions, such as the FPGAs, which can perform a lot of operations in parallel. The FPGAs are used in many applications, such as the PRNG [12], Neural Networks [29], encryption [30], edge detection [31], and compression [32].

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3086044, IEEE Access

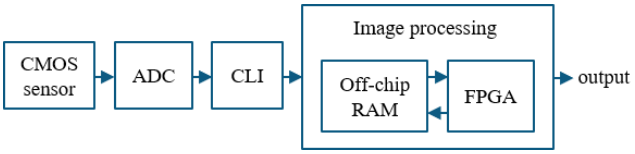


FIGURE 3. The general structure of the FPGA-based image processing systems.

In the past decade, the FPGAs have been used in the field of image processing due to their powerful parallel processing capabilities [33]. They can exploit the temporal and spatial parallelism of many image processing algorithms. The spatial parallelism refers to the utilization of multiple copies of the hardware to perform multiple tasks in parallel. On the other hand, the temporal parallelism refers to the utilization of a pipelined hardware unit that divides the processing task into several cascaded stages.

The FPGAs can be integrated with the digital cameras as shown in Fig. 3 [34, 35]. First, the CMOS image sensor is used to capture the image. Next, the Analog to Digital Converter (ADC) is used to convert the analog values of the image's pixels into digital values. Then the Camera Link Interface (CLI) is used to send the digital image to the FPGA. Finally, the FPGA is used to perform all the necessary processing tasks on-the-fly. The word on-the-fly means as soon as the data is received as explained in [36].

Since the FPGA's internal memory will not be sufficient to store the full uncompressed image, an external off-chip memory will be utilized as shown in Fig. 3. To emphasize this point, consider one of the most powerful FPGAs, the Xilinx Virtex 7 XC7V2000T, which includes only 46512kb Block Random Access-Memory (BRAM) [29]. This powerful FPGA can store only one 1408×1408 color image with 24-bits per pixel. Hence, the off-chip memory will be needed to store high resolution images, which are in the range of tens of Megapixels.

E. Motivations and objectives

Over the past few years, the wearable lifelogging cameras have been used in a lot of applications. They are used to track our daily activities by automatically capturing hundreds of images throughout the day. This vast number of images need to be stored instantly in the cloud. Hence, the compression and encryption must be done on-the-fly before transmitting the images to the cloud. Also, since a lot of the captured images may suffer from blur, most probably they will be deleted later on by the life-loggers [28]. Instead of deleting the blurred images at a later stage, a better option is to filter out the images directly after they are captured by the camera. In this way, only the unblurred images will be sent to the cloud. Accordingly, the network's bandwidth and the cloud's storage space will be saved. So, performing the blur detection on-the-fly and in parallel with the compression and encryption will be very beneficial to the lifelogging cameras.

There is no solution presented in the literature, software, or hardware, that can perform the image blur detection in parallel with compression and encryption. In most systems, the blur detection is usually performed at a later stage after transmitting the compressed images to the cloud [28].

Most of the recent compression-encryption systems are implemented using software, and they suffer from long computational time [37]-[44]. Therefore, they are not suitable for real-time applications. Only few old compression-encryption systems are implemented using FPGAs, but they do not utilize a strong encryption scheme [45]-[47]. The fastest FPGA implementation is based on the DCT compression and the stream cipher [47]. The stream cipher is not as strong as the block cipher technique, for example the AES-CBC.

The main contributions of this paper are based on the following two aspects:

(1) To the best of the authors' knowledge, this paper presents the first HW implementation of Tong et al.'s [15] Haar-based blur detection algorithm.

(2) The blur detection is integrated with image compression and encryption as a 3 in 1 parallel HW solution, which is suitable for real-time applications. Depending on the application, the blurred images can be either marked for future enhancement or simply filtered out. The compression is based on both the RLE and the Haar DWT. The encryption is implemented using one of the most secure algorithms: the 128-bit AES, which is combined with the modified Lorenz chaotic PRNG to perform the CBC mode. The proposed system can be integrated with digital cameras to process the captured images prior to transmission or storage. Once the *N* pixels image is fully delivered to the frame buffer, the proposed system will just read the image in only *N* clock cycles, and all the tasks will be performed on-the-fly in a pipelined manner.

The paper is organized as follows. Section I presents the introduction. Section II presents the HW implementation of the blur detection algorithm. Section III presents the HW implementation of the full system. Section IV presents the results. Section V presents the conclusion.

II. The HW implementation of the image blur detection

A. The Haar DWT

The 1-level 2d Haar transform is illustrated as shown in Fig. 4(a) [48]. Every four neighboring pixels in the original image, such as *A*, *B*, *C*, and *D*, are transformed into *X*, *Y*, *Z*, and *W*. X is called the average coefficient while *Y*, *Z*, and *W* are called the vertical, horizontal, and diagonal detail coefficients, respectively. The transformed matrix will be divided into four bands: Low-Low (LL), High-Low (HL), Low-High (LH), and High-High (HH). The LL band will contain the average coefficients while the other bands will contain the detail coefficients. The 3 decomposition levels are obtained by performing the 1-level 2d transform recursively on the LL band as shown in Fig. 4(b).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3086044, IEEE Access

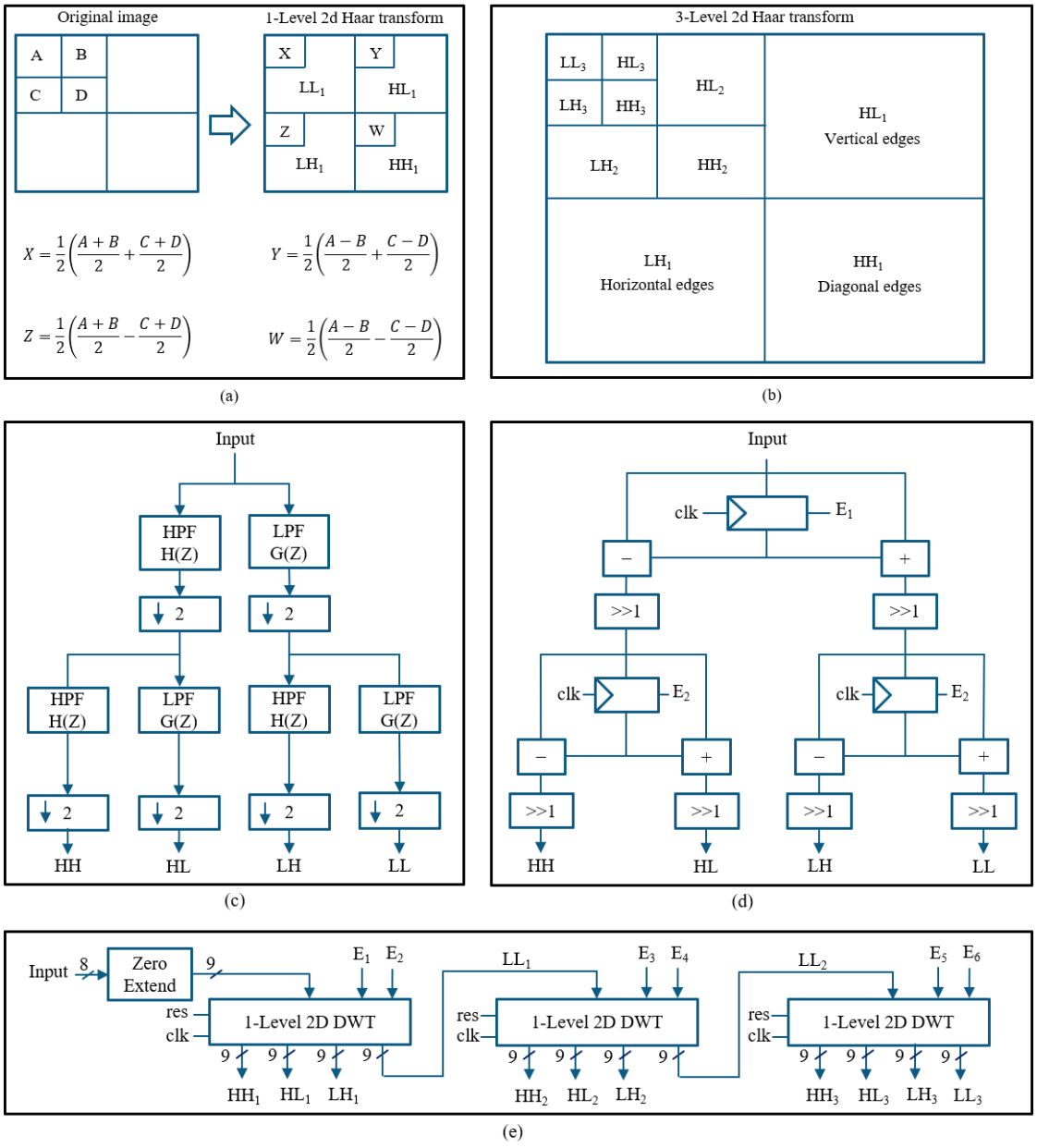


FIGURE 4. **a) The 1-level 2d DWT, b) The 3-level 2d DWT (pyramid structure), c) The implementation of the 1-level 2d DWT using the convolutional method, d) The HW implementation of the 1-level 2d DWT, and e) The HW implementation of the 3-level 2d DWT.**

The 1-level 2d DWT is built using the conventional convolutional method, which consists of a High Pass Filter (HPF) and a Low Pass Filter (LPF) followed by down samplers, as shown in Fig. 4(c) [49]. The Haar transform's LPF and HPF are based on computing the sum and difference between two consecutive inputs as shown in (2) [48].

$$G[Z] = \frac{1}{2}(1 + Z^{-1}) \qquad (2a)$$

$$H[Z] = \frac{1}{2}(1 - Z^{-1}) \qquad (2b)$$

The hardware implementation of the 1-level 2d Haar transform is presented in Fig. 4(d). The registers are used to provide the delay elements in (2), the hardwired shifters are used to perform the multiplication by ½, and the enable signals, E_1 and E_2, are used to control the down sampling. The fractional bit that arises after the multiplication by ½ will be truncated to maintain the same number of bits throughout the design.

The 3-level 2d DWT is implemented using cascaded 1-level 2d DWT units as shown in Fig. 4(e). The enable signals, E_1 to E_6, are used to control the down sampling of the three levels. E_1 is always enabled while E_2, E_3, E_4, E_5, and E_6 are enabled once every 2, 4, 8, 16, and 32 clock cycles, respectively. To use these cascaded units properly, the image must be scanned

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3086044, IEEE Access

in a certain order, which is different from the conventional row by row technique.

Once the N pixel image is available in the frame buffer, it will be scanned as shown in the example in Fig. 5. In this example, a 4×4 image is stored row by row as indicated by the memory location written in each block. The image scanning is performed as follows. First, the image is divided into four quadrants as depicted by the dashed squares. Then, each quadrant is divided again into four quadrants. This process is performed until the smallest quadrant, which contains only 1 pixel, is reached. Finally, every four equal sized quadrants are scanned in this order: upper left, upper right, lower left, then lower right. For the example in Fig. 5, the memory addresses must be generated in this order: 0, 1, 4, 5, 2, 3, 6, 7, 8, 9, 12, 13, 10, 11, 14, 15.

Accordingly, a 4-bit counter will be used to generate the 16 different addresses, but the counter's bits will be rearranged in a different order. Suppose that the counter's bits are normally arranged as $A_3A_2A_1A_0$, then to generate the required sequence, the bits must be reordered as $A_3A_1A_2A_0$. This means that starting from the Least Significant Bit (LSB), the even bits will be inserted before the odd bits. This concept can be applied to any n-bit counter. The proposed scanning technique can be applied on square images with any size (N pixels) to perform the 3-level DWT in only N clock cycles. For the case of non-square images, the zero-padding method can be used to change the input image to a square.

B. The image blur detection

Tong et al.'s blur detection algorithm [15] is based on classifying the image edges into four types: Dirac, A-step, G-step, and Roof. The classification is based on the variation in the pixels' intensity as shown in Fig. 6(a) [15]. Blurred images usually do not have Dirac or A-step edges. Instead, they have a lot of G-step and Roof edges. Accordingly, the algorithm can detect blurred images by identifying the type of edges in the image.

The first step of the algorithm is to perform the 3-level 2d Haar wavelet transform. The second step is to construct an edge map for each level using (3a) for i = 1,2,3. To improve the hardware performance, the edge maps can be approximated using (3b).

$$Emap_i = \sqrt{LH_i^2 + HL_i^2 + HH_i^2} \quad (3a)$$

$$Emap_i \approx |LH_i| + |HL_i| + |HH_i| \quad (3b)$$

The third step of the algorithm is to divide the edge maps using partitioning windows. Since the size of the edge map is scaled down by a factor of 4 after every decomposition, the partitioning windows must be scaled down by the same factor. Therefore, the sizes of the partitioning windows used for level 1, 2, and 3 are 8×8, 4×4, and 2×2, respectively. In this way, the three edge maps will have the same number of partitions.

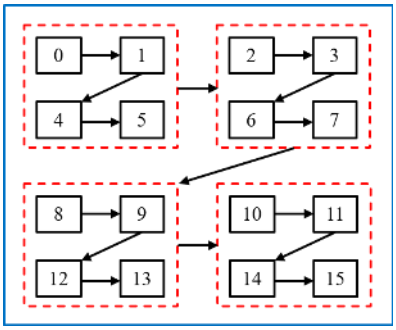
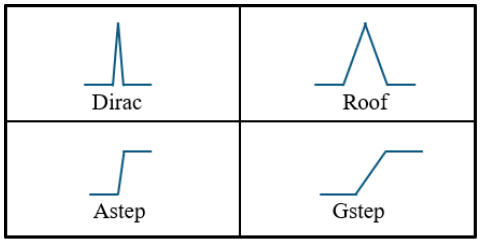


FIGURE 5. A demonstration of the proposed scanning technique on a 4×4 image, which is stored row by row as indicated by the index of each pixel.



(a)

	$Edge_{Max1}^{j}$	$Edge_{Max2}^{j}$	$Edge_{Max3}^{j}$
Dirac	Highest	Middle	Lowest
Astep	Highest	Middle	Lowest
Gstep	Lowest	Middle	Highest
Roof	Lowest	Middle	Highest
	Lowest	Highest	Middle

(b)

FIGURE 6. a) The graphical description of different edge types. b) The rules for discriminating the edges based on the maximum edge values at the 3 decomposition levels.

The fourth step is to find the maximum value, $Edge_{max\,i}{}^{j}$, in each partitioning window where j is the index of the partition and i is the decomposition level. If $Edge_{max1}{}^{j}$ or $Edge_{max2}{}^{j}$ or $Edge_{max3}{}^{j}$ is higher than a certain threshold (Th), then the partition with index j will have an edge point. The type of the edge can be identified using the rules presented in Fig. 6(b). Furthermore, in case the edge is identified as a G-step or Roof, but $Edge_{max1}$ is lower than Th, then it will be considered as an unsharp edge point. The final step is to divide the total number of Dirac and A-step edges (N_{DA}) by the total number of edges (N_{edges}). If this ratio is lower than a very small threshold (e.g., 0.05), then the image will be considered blurred. The blur extent is calculated by dividing the total number of unsharp G-step and Roof edges (N_{BRG}) by the total number of G-step and Roof edges (N_{RG}).

The hardware implementation of the blur detector unit is presented in Fig. 7. The blur detector unit will receive the detail coefficients of levels 1, 2, and 3 from the DWT unit every 4, 16, and 64 clock cycles, respectively.

First, the sum of the absolute values of HH, HL, and LH is computed for each level to generate $Edge_1$, $Edge_2$, and $Edge_3$.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3086044, IEEE Access

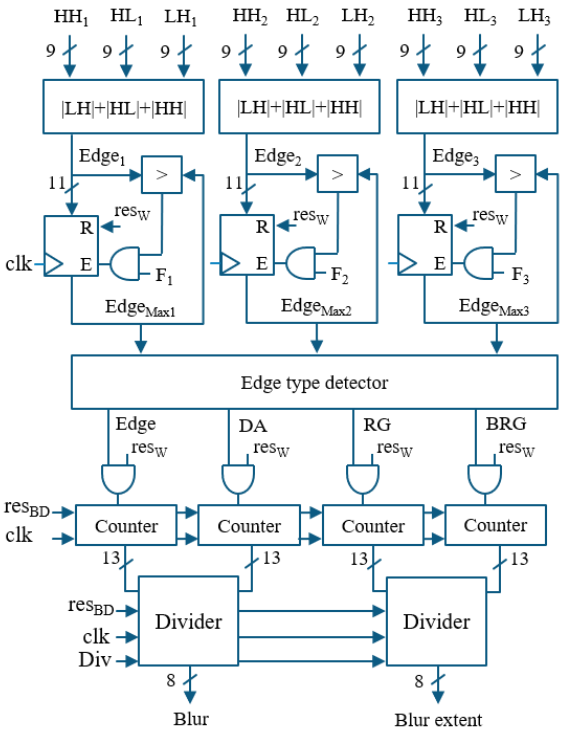


FIGURE 7. The HW implementation of the blur detection algorithm.

The computation of the Edge value, ($|LH|+|HL|+|HH|$), is implemented using 3 cascaded adders/subtractors as ($0\pm LH\pm HL\pm HH$) where the adder/subtractor's control signal is driven by the sign bit of the detail coefficient.

Next, the registers connected to $Edge_1$, $Edge_2$, and $Edge_3$ will be enabled only if the generated edge value is larger than the stored value. In this way, the registers will store the maximum value they receive. Furthermore, to allow the registers to be updated only when the correct edge value is generated, the comparators of levels 1, 2, and 3 will be ANDed with the input flags F_1, F_2, and F_3, which are enabled once every 4, 16, and 64 clock cycles, respectively.

The proposed image scanning technique allows the system to scan a new 16×16 block from the input image every 256 clock cycles. Hence, the maximum values of the current edge map partition, $EdgeMax_1$, $EdgeMax_2$, and $EdgeMax_3$, will be available in the registers at the end of every 256 clock cycles. After finding these maximum values, the registers will be reset using res_W before proceeding to the next partition. The edge type detector will receive the maximum values and will update the counters according to the rules in Fig. 6(b). Finally, two dividers are used to compute the required ratios, one to check whether the image is blurred or not, and the other to calculate the blur extent. The res_{BD} signal is used to reset the counters and the dividers before processing a new image.

III. The proposed system

The HW architecture of the proposed system is presented in Fig. 8. The proposed architecture performs the color image blur detection in parallel with the compression and encryption in a pipelined fashion. The image is sent to the system pixel by pixel where each color pixel *RGB* consists of 24 bits.

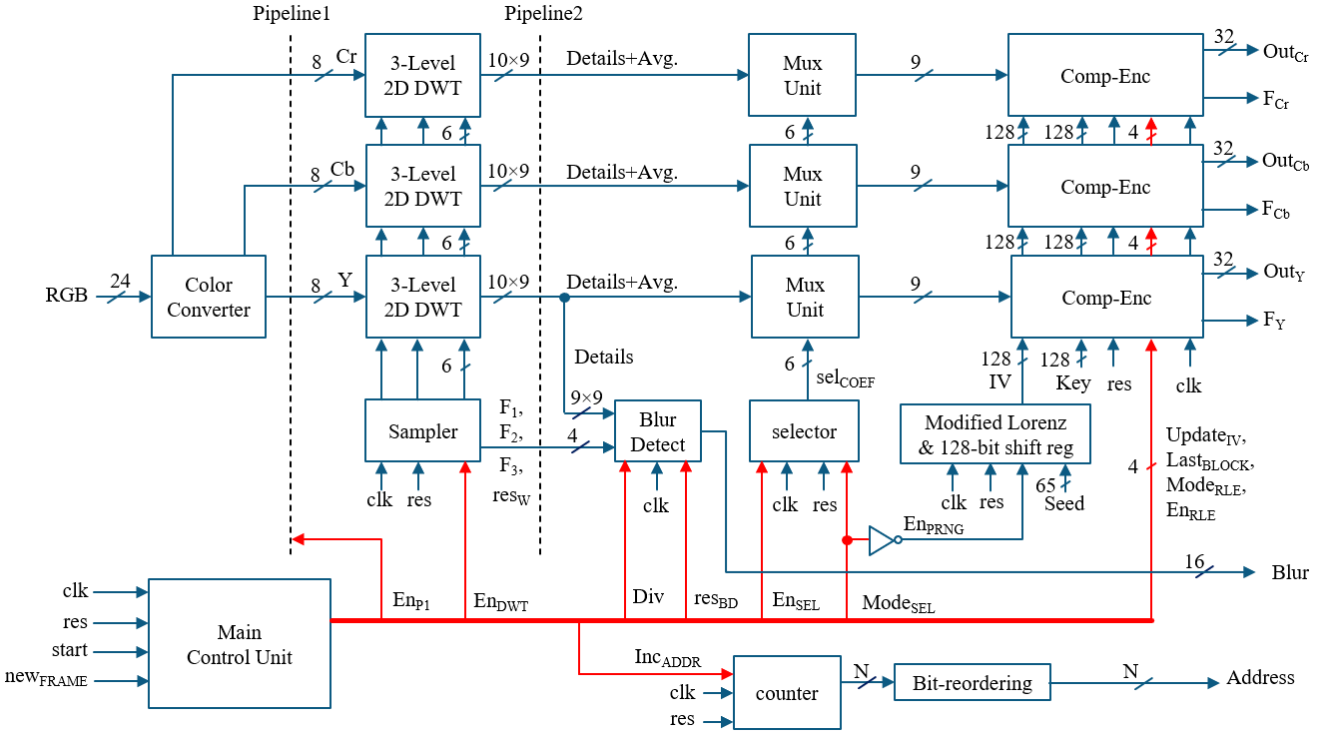


FIGURE 8. The block diagram of the proposed system.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3086044, IEEE Access

The main data path of the system is explained as follows. First, the input color pixel is transformed from RGB into YCbCr components. Then each color component will pass through a separate 3-level DWT unit. Next, the blur detection unit will process the detail coefficients of the Y color component only. Finally, three compression-encryption units are used to encode the 3 color components in parallel. Each compression-encryption unit will use the 10 output channels of the corresponding 3-level DWT unit in a sequential manner. Therefore, a multiplexing unit is needed to interface the 10 output channels of the 3-level DWT unit with the compression-encryption unit. The encrypted outputs are provided in words of 32-bits. Also, flag signals are generated to indicate that a new encrypted data is available at the outputs.

The *IV* is generated from the modified Lorenz PRNG which uses a 65-bit seed value. The modified Lorenz is implemented using the design in [12], which can past all the NIST tests [50]. This PRNG generates 24-bits per clock cycle; therefore, a 128-bit shift register is needed to receive the generated bits sequentially, and then provide all the 128-bits in parallel.

Finally, the main control unit is used to synchronize and control all the blocks in addition to incrementing the address generator. The *address* is used to read the image from the external frame buffer. Moreover, the control unit uses other internal signals as inputs, such as, *address*, F_1, F_2, F_3, res_W, sel_{COEF}, F_Y, F_{Cb}, and F_{Cr}; however, they are not shown in Fig.8 just for the sake of simplicity. The control unit will be explained in detail in a subsequent section.

A. The sampler

The sampler circuit is used to generate the enable signals of the 3-level 2d DWT unit in addition to the flag signals of the blur detector. The sampler circuit consists of an 8-bit counter and a set of comparators as shown in Fig. 9. The flag signals, F_1 and F_2, are the same as the sampler signals, E_3 and E_5, respectively.

B. The color converter

The color converter is used to change the input RGB components to YCbCr components where Y is the luminance while Cb and Cr are the chrominance. The luminance Y contains the grayscale pixel while the chrominance Cb and Cr contain the change in blue and red color, respectively. This conversion improves the compression performance as will be clarified in the results section. Also, the blur detection algorithm must be applied on grayscale images only; therefore, obtaining the Y component is a prerequisite for the blur detection.

The conversion is performed as shown in (4) [51]. The conversion requires 3 multipliers and 3 adders for each component to compute Y, C_b, and C_r in parallel. Thus, the converter utilizes a total of nine 8-bit multipliers and nine 8-bit adders. The 8-bit multipliers will multiply the input component (R, G, or B) with the corresponding numerator in (4), (e.g., R×65). Accordingly, the output of the multiplier will

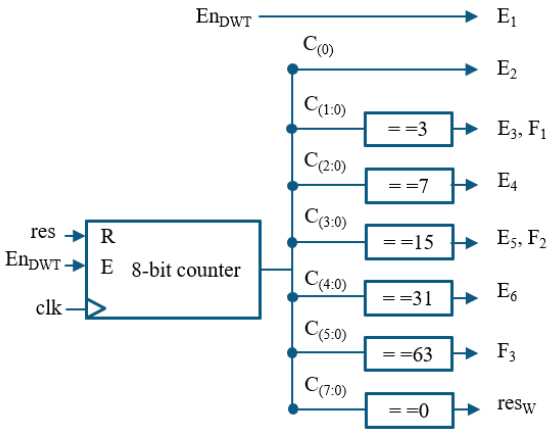


FIGURE 9. **The HW implementation of the sampler.**

have 16 bits. Only the 8 Most Significant Bits (MSBs) of the multiplier will be taken, which is equivalent to dividing by 256 and then truncating the fraction part. Furthermore, the adders are connected in a tree structure to reduce the critical path.

$$Y = 16 + \frac{65}{256}R + \frac{129}{256}G + \frac{25}{256}B \qquad (4a)$$

$$Cb = 128 - \frac{38}{256}R - \frac{74}{256}G + \frac{112}{256}B \qquad (4b)$$

$$Cr = 128 + \frac{112}{256}R - \frac{94}{256}G - \frac{18}{256}B \qquad (4c)$$

C. The multiplexing unit

The MUX unit and the selector are shown in Fig. 10. This unit consists of three cascaded 4-input multiplexers, which are used to select between the coefficients of the 3 levels. The select signals of the multiplexers are generated from the 2-bit counters. The En_{sel} is used to enable or disable the counters.

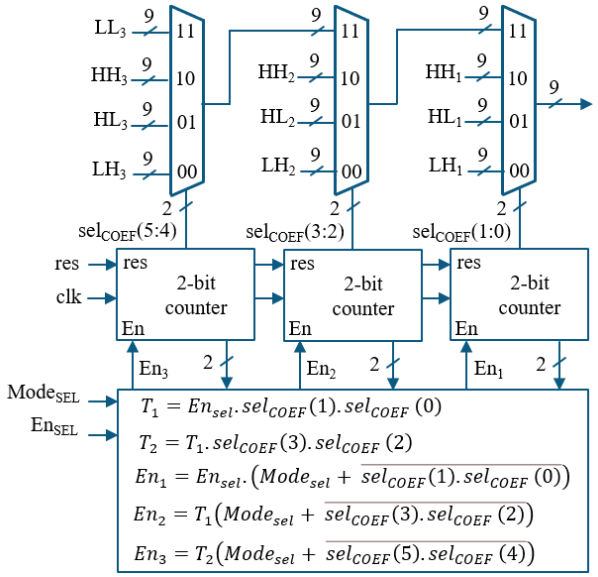


FIGURE 10. **The HW implementation of the multiplexing unit.**

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3086044, IEEE Access

The $Mode_{sel}$ is used to choose the counting mode. If $Mode_{sel}$ is set high, the counters will operate normally from 0_{10} to 63_{10}. Otherwise, the sel_{COEF} will change, such that the last 10 coefficients will pass in this order: LH_1, HL_1, HH_1, LH_2, HL_2, HH_2, $LH3$, HL_3, HH_3, LL_3.

D. The compression-encryption unit

The HW implementation of the compression-encryption unit is presented in Fig. 11. First, Mux_1 is used to perform the hard thresholding by selecting between *In* and 0_{10}. The selection is based on whether *In* is greater than the compression threshold ($comp_{TH}$) or not. Next, an 8-bit counter will count the number of zero bytes that pass through MUX_1. Accordingly, Z_{COUNT} and the output of Mux_1 will form the RLE 16-bit word. For example, $\{00_{16},00_{16},00_{16},00_{16},0F_{16},\}$ will be encoded as $\{04_{16},0F_{16}\}$. The RLE word will be inserted into a 128-bit shift register (Reg_1), which is enabled using the RLE_{FLAG}. The main problem of the RLE can be explained using the following example. Suppose that $comp_{TH}$ is set to 3_{10}, and the input sequence has 14 consecutive coefficients greater than or equal to $comp_{TH}$ (e.g., In=$\{03_{16},04_{16},05_{16}\ldots0F_{16},10_{16}\}$). Accordingly, this sequence will be encoded as $\{00_{16},03_{16},00_{16},04_{16}\ldots00_{16},10_{16}\}$. This implies that the 14 input bytes will be encoded in 28 bytes, which has a negative effect on the compression performance. To solve this problem, the *In* bytes will be inserted directly into another 112-bit shift register (Reg_2), which is enabled by En_{RLE}.

In addition, 2 zero bytes will be concatenated with Reg_2 to complete the 128-bits and indicate that they are not run length encoded. Accordingly, the 14 input bytes will be encoded in 16 bytes as $\{03_{16},04_{16}\ldots.0F_{16},10_{16},00_{16},00_{16}\}$, which is in this case better than the RLE. After filling Reg_1 and Reg_2, MUX_2 will select between these 2 registers as follows. If Reg_1 is filled before Reg_2, then MUX_2 will select Reg_2 as explained in the previous example. Otherwise, the RLE will be the better option, and MUX_2 will select Reg_1. To control MUX_2, 2 flip flops are used to hold the present state of Reg_1 and Reg_2 (P_1,P_2). The 3-bit counter is used to count the number of RLE words that enter Reg_1. Similarly, the 4-bit counter is used to count the number of *In* bytes that enter Reg_2. Reg_1 will be filled after receiving 8 RLE words while Reg_2 will be filled after receiving 14 *In* bytes. Once the register is filled, its present state will be set to logic 1. Since the *In* bytes are received every clock cycle, the AES unit will have at least 14 clock cycles to encrypt the input block. The AES will encrypt the 128-bit block in only 11 clock cycles as will be explained in the next subsection. Finally, a logic unit is used to control all the blocks. The En_{RLE} is used to enable the compression-encryption unit. The $Mode_{RLE}$ is used as an indicator for the last *In* byte. The $Last_{BLOCK}$ is used to start the encryption of the last 128-bit block even if Reg_1 and Reg_2 are not yet filled .

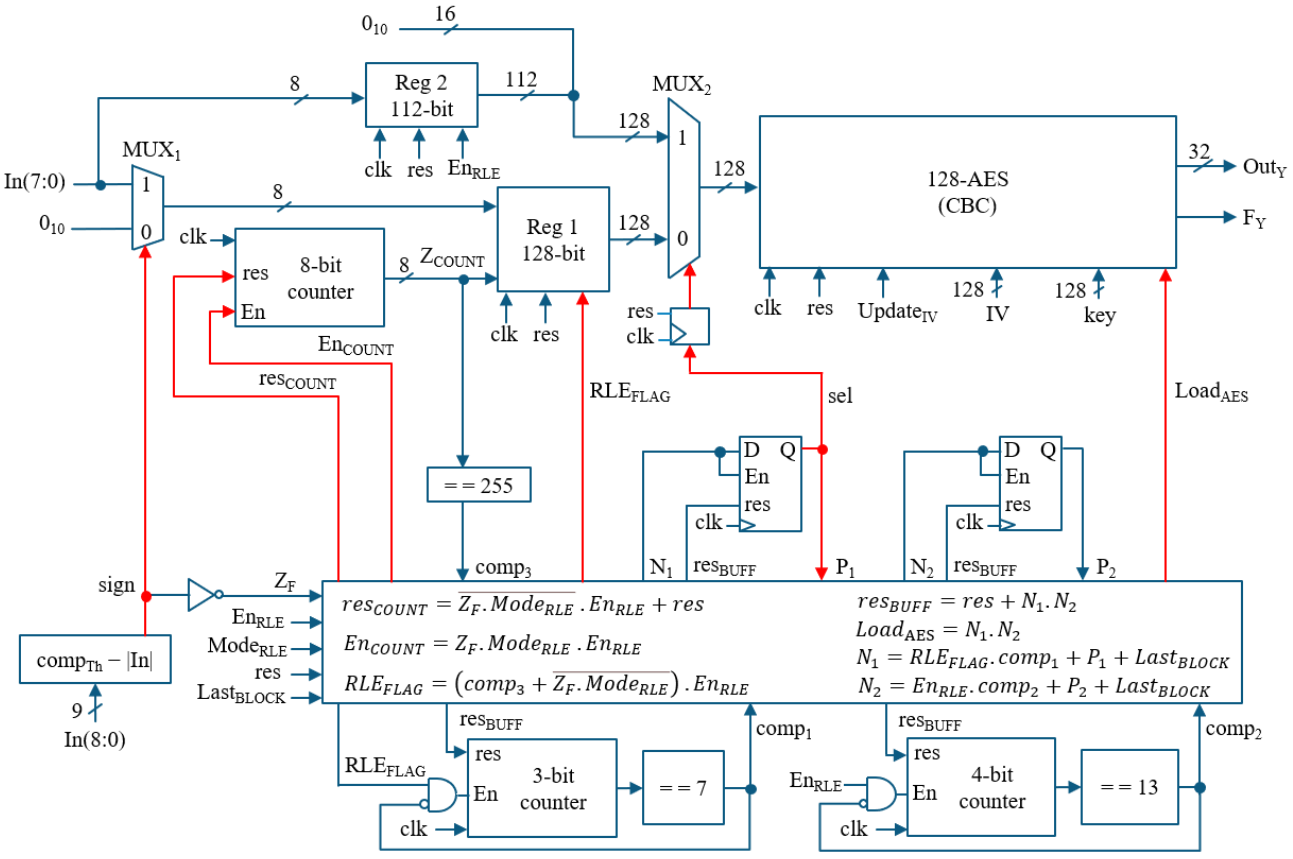


FIGURE 11. The HW implementation of the compression-encryption unit.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3086044, IEEE Access

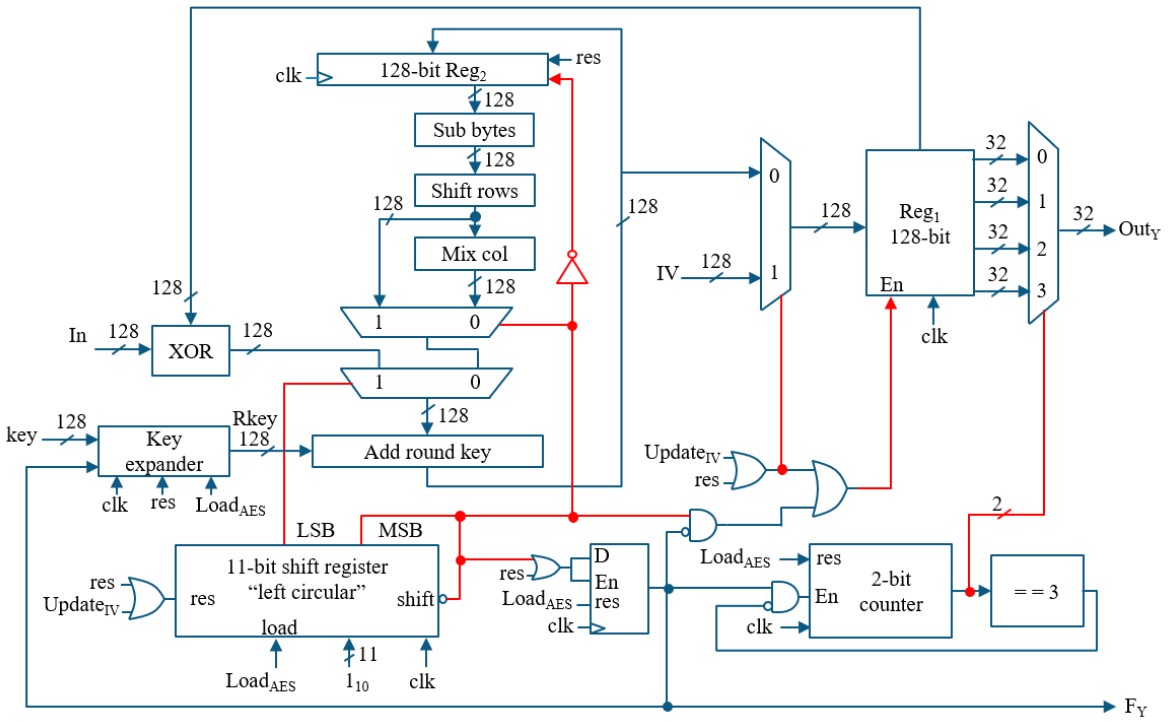


FIGURE 12. The HW implementation of the 128-bit AES-CBC.

E. The AES

The HW implementation of the 128-AES-CBC is shown in Fig. 12. All the details of the AES are available in [52]. The encryption is performed as follows. The initialization vector (*IV*) will be updated and loaded into Reg_1 at the start of every new image. Then the data in Reg_1 will be used as a feedback and XORed with the input to perform the CBC mode. After feeding back the content of Reg_1, the AES will start the encryption process, which consists of 11 rounds. Each round will have a unique Round Key (*Rkey*), which will be generated using the key expander.

In the proposed system, each round will be performed in 1 clock cycle. In the first clock cycle, the *Rkey* will be added to the input, and the result will be stored in Reg2. In the next 9 clock cycles, the data stored in Reg_2 will be updated after applying a series of operations: the sub bytes substitution using the S-box, then the shift rows, followed by the Mix columns, and finally adding the *Rkey*. In the 11^{th} clock cycle, the Mix columns operation will be skipped, and the final result will be stored in the output register Reg_1. The output flag F_Y is used to indicate that the encrypted block is now available in Reg1. The encrypted block will be provided through Out_Y in 32-bit words using a 4-input multiplexer, which is controlled by a 2-bit counter.

As explained above, the operations performed in the initial round is different from the middle 9 rounds and the last round as well. Hence, multiplexers are used to select the required operations in each round. The multiplexers will be controlled using the LSB and the MSB of an 11-bit left circular shift register. Initially, the shift register will be loaded with 1_{10}; thus, the LSB will be high while the MSB will be low. Accordingly, the data coming from the input side will be selected and only the Add round key operation will be performed. In the next 9 cycles, the bits will be moving in the shift register towards the left, and both LSB and MSB will be at logic 0. Hence, all the AES operations will be performed. In the 11^{th} cycle, the MSB will be high while the LSB will be low; therefore, the Mix columns will be skipped.

The Add round key block is implemented using a group of XOR gates. The sub bytes block is implemented using 16 parallel S-boxes, which are implemented using Look Up Tables (LUTs). The shift rows block is implemented using hardwired shifters; hence, no computational resources will be utilized. Finally, the Mix columns block is implemented using a series of XOR operations as explained in [52].

The key expander is implemented as shown in Fig. 13. The key expansion process is performed as follows. First, the input key will be loaded in a 128-bit register, which will provide the *Rkey*. Next, the 32 LSBs of the 128-bit register will pass through a series of operations before they are combined back with all the 128-bits. The 32 LSBs will first undergo a hardwired byte cyclic rotation. Then 4 parallel S-boxes will be used to substitute the rotated bytes. Accordingly, the 4 S-boxes will provide a total of 32 bits, only the 8 MSBs will be XORed with the *Rcon* signal, and then they will be concatenated back with the remaining 24 LSBs.

To generate the *Rcon* signal, first a value of 1_{10} will be loaded in an 8-bit register. Next, the register will be updated by shifting the stored bits to the left and then adding the signal *con*. The signal *con* is constructed using *Rcon* as follows {0, 0, 0, *Rcon*(7), *Rcon*(7), 0, *Rcon*(7), *Rcon*(7)}.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3086044, IEEE Access

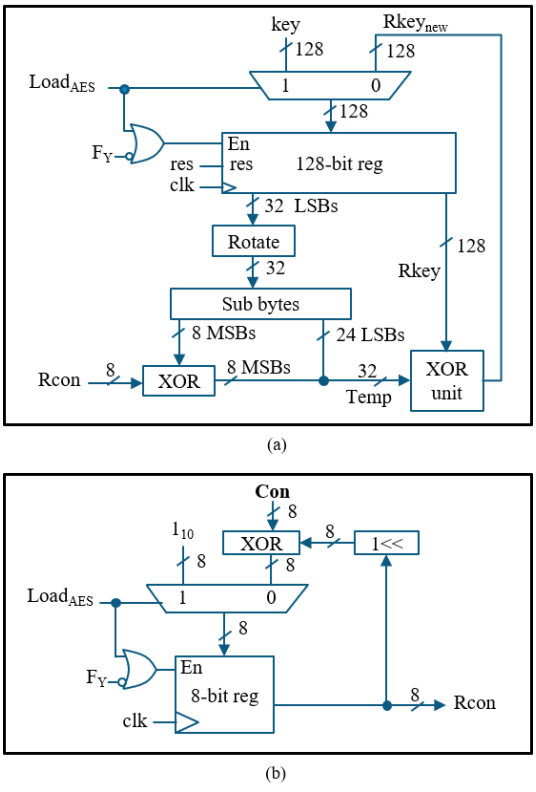


FIGURE 13. **The HW implementation of the AES key expander. a) The Key expander block diagram. b) The Rcon function.**

F. The main control unit

The main control unit is implemented using a Finite State Machine (FSM) as shown in Fig. 14. To simplify the state diagram, the outputs written in each state are the ones that are set to logic 1. As illustrated in Fig. 14, The FSM will jump from the initial state, St_0, to St_1 in case the input *start* is set to logic 1. In St_1, the FSM will start to increment the address of the frame buffer to read the image. The FSM will spend 1 clock cycle in St_1 until the first pixel is read from the frame buffer. The next state, St_2, is used to add another 1 clock cycle delay due to the first pipeline stage in the system, see $pipeline_1$ in Fig. 8. In case the system is not pipelined, the FSM will jump directly from St_1 to St_3.

In St_3, the FSM will enable the selector and the sampler to start the image processing. The FSM will remain in St_3 for 4 clock cycles until the first set of coefficients are generated from $level_1$ as indicated by the input flag F_1.

Next, the FSM will jump to St_4 to enable the RLE. Since the coefficients of $level_2$ are not generated yet, the FSM will enable the RLE for 3 times every 4 clock cycles to encode the 3 detail coefficients of $level_1$ only. The FSM will remain in St_4 until the first set of coefficients are generated from $level_2$, as indicated by the input flag F_2. Then the FSM will jump to St_5. Similarly, the FSM will remain in St_5 until the first set of coefficients are generated from $level_3$, and then will jump to St_6.

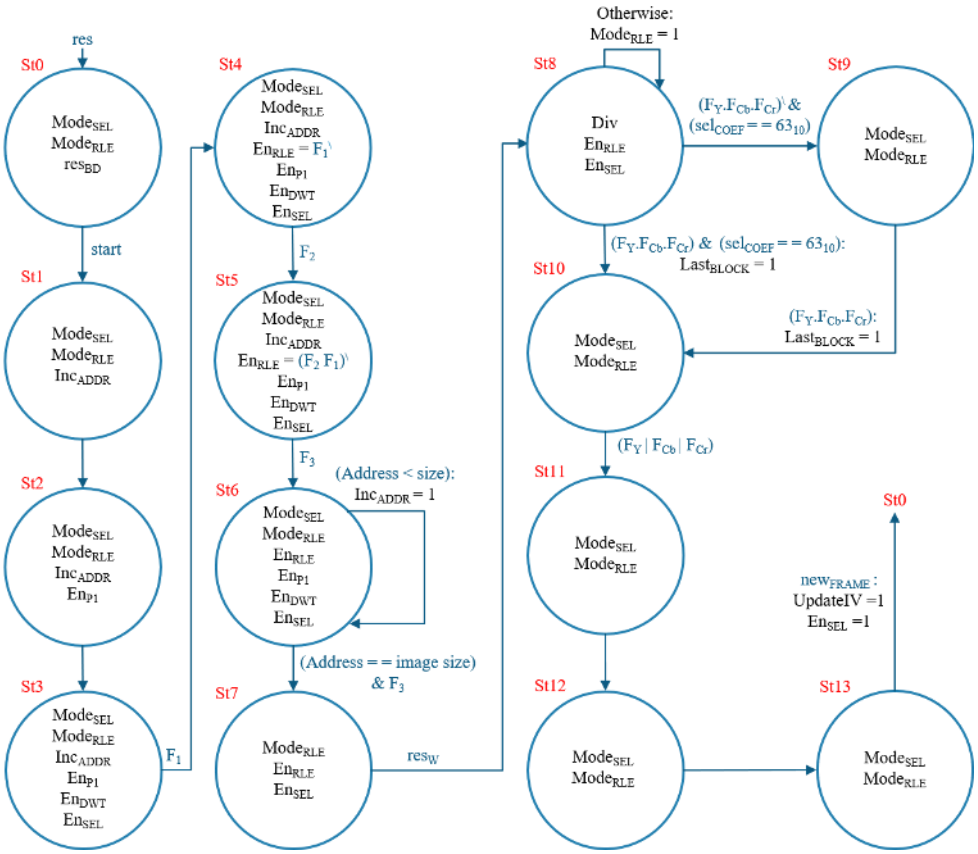


FIGURE 14. **The FSM of the control unit. The inputs and outputs are written in blue and black colors, respectively.**

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3086044, IEEE Access

In St_6, the RLE will be enabled all the time to encode the coefficients of all the 3-levels sequentially. The FSM will remain in St_6 until the image is fully read from the frame buffer and then will jump to St_7. The FSM will wait for 1 clock cycle in St_7 until the counters of the blur detector are updated, and then will jump to St_8 to start the division process of the blur detector. Also, in St_7 and St_8, the last 10 coefficients will be multiplexed to the compression-encryption unit in the order explained in the previous subsection. Accordingly, the $Mode_{sel}$ will be set to logic 0. The FSM will remain in St_8 until the last coefficient from the transformed image is encoded as indicated by ($sel_{COEF} = = 63_{10}$).

Next, the FSM will jump to St_9 or St_{10} according to whether the AES units are ready to process the last block or not. Suppose that one of the AES units has started to encrypt a given block and there are still 2 non-zero coefficients left in the transformed image. In this case the AES buffer will be ready after 2 clock cycles only while the AES unit will still need another 9 clock cycles to finish the encryption process. Accordingly, the FSM will move to St_9 and wait until all the AES units are done. In St_{10}, the system will encrypt the last block in the image. St_{11} and St_{12} are delay states to ensure that the 128-bit encrypted block is fully transferred through the 32-bit output port of the AES unit. Finally, the FSM will wait in St_{13} until a new image is received, and then will repeat the same process again.

IV. Results

The proposed system is realized using HDL and implemented on XC5VLX50T FPGA using Xilinx. The simulation process is performed in three steps. First, the input image is written in a Memory Initialization File (MIF) using MATLAB. Next, the HDL design is tested with the MIF, and then the encrypted image is written in a text file using Xilinx simulator. Finally, the generated text file is decrypted and decompressed using MATLAB to verify the compression-encryption process. The standard test images are summarized in Fig. 15. The proposed system is evaluated using both color and grayscale images to ensure a fair comparison with most of the recent compression-encryption systems. In case of grayscale images, the YCbCr conversion will be skipped.

A. The HDL simulation results

The HDL simulation results are illustrated in 3 parts. The first part will verify the YCbCr conversion, the 3-level DWT, and the blur detection. The second part will verify the AES. The last part will verify the compression scheme. The simulation is performed on Lena 256×256 color image.

The first part of the simulation is presented in Fig. 16. The Y component of the color converter, which is the grayscale image, is presented in Fig. 16(a). To see how the edge maps are generated, the simulation will focus on a block of 16×16 pixels near Lena's hat as depicted in Fig. 16(a). This block of pixels is magnified in Fig. 16(b) just for the sake of illustration. Figures 16(c) to 16(e) show the generated edge maps for the 3 levels. In case the edge threshold is set to 10, the position of the edge points will follow the edges in Lena's hat. The Xilinx simulation results for the edge map of level 2 is presented in Fig. 16(f)

The second part will verify the AES using the example in [52]. The HDL simulation results, shown in Fig. 17, match exactly the results in [52].

The last part of the simulation, which verifies the compression operation, is shown in Fig. 18 where the $comp_{TH}$ is set to 3_{10}. The input sequence consists of 7 consecutive bytes with magnitude less than or equal to the $comp_{TH}$ followed by the non-zero coefficient 4_{10}. Accordingly, the output RLE word will be 0704_{16}.

FIGURE 15. The standard test images.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3086044, IEEE Access

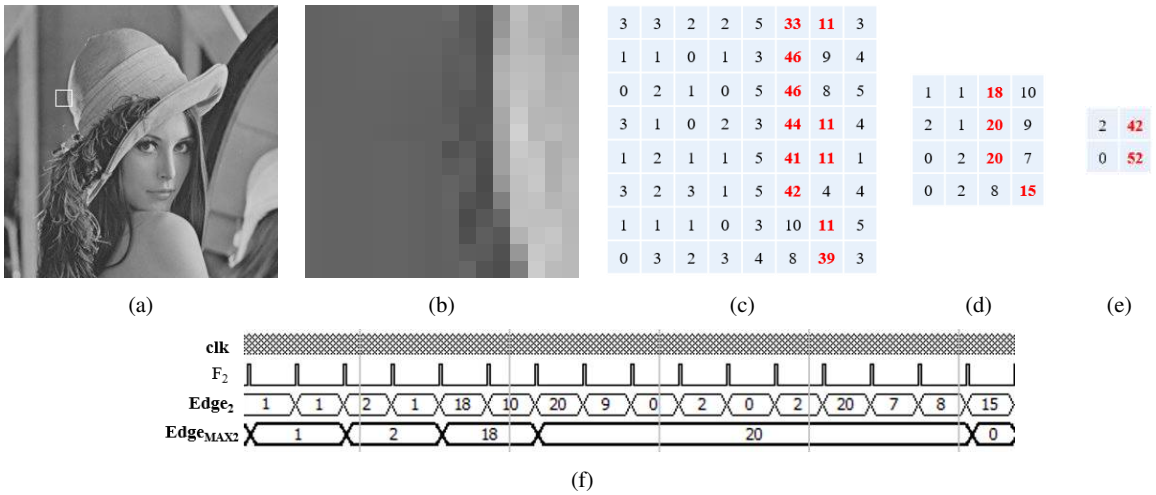


FIGURE 16. The HDL simulation results of the blur detector. a) The test image, b) The cropped part, c) Edge map 1, d) Edge map 2, e) Edge map 3, and f) The output of Xilinx's simulator for Edge map 2.

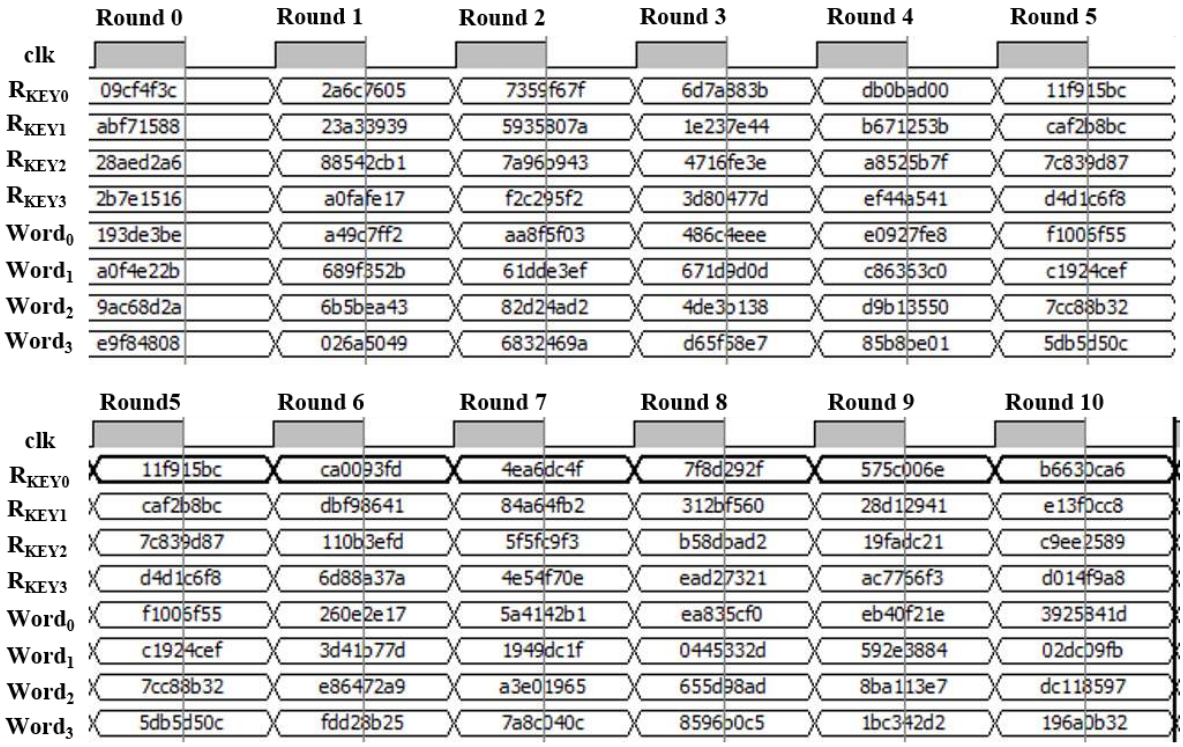


FIGURE 17. The AES HDL simulation results for the testcase in [52], which shows the encrypted words after each round.

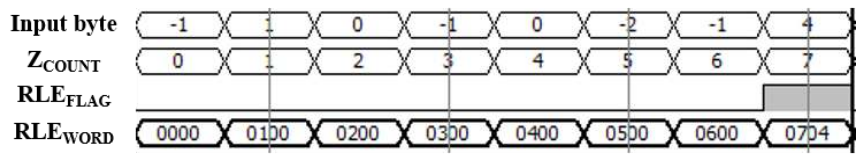


FIGURE 18. The HDL simulation of the compression process with a threshold value equal to 3.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3086044, IEEE Access

TABLE 1
THE POST PLACEMENT AND ROUTING RESULTS.

Logic utilization	Used	Available	Utilization
Slices	1866	7200	25%
Registers	2955	28800	10%
LUTs	5736	28800	19%
Frequency	144.11 MHz		
Throughput	3.458 Gbps		

B. The HW performance

The proposed system is fully implemented using slice registers and LUTs. No DSP48Es or block RAMs are utilized. The post placement and routing results are summarized in Table 1. The proposed system has utilized 25% of the available slices. This is a very efficient utilization relative to the functionalities offered by the system: color image blur detection, compression, and AES-CBC encryption. Furthermore, the timing analysis shows that the maximum clock frequency is 144.11 MHz. Since the system processes a 24-bit color pixel per clock cycle, the throughput will be 3.458Gbps. This throughput allows the system to process 1-MP image in only 7.276ms. Therefore, the rate will be 137 Frames Per Second (FPS) for 1-MP or 30 FPS for 4.5-MP. Accordingly, the proposed system is well suited for real-time applications.

The HW performance of different FPGA image compression-encryption systems are summarized in Table 2. The proposed system is the only one capable of integrating the blur detection with image compression-encryption. Furthermore, the proposed system can process color images, and not just grayscale images as the other systems. Moreover, the proposed system has the highest throughput, which is even twice as fast as a simple DCT compression and stream cipher system [47]. Also, the throughput of the proposed system is 62% higher than the FPGA-based system presented in [62], which accommodates a single AES-128 core with Keccak-f[400]. Regarding the HW implementation area, the proposed system has the largest slice utilization. This is due to the 128-AES-CBC, which improves the security level at the expense of utilizing more resources. The AES-CBC is more secure than the TEA, the stream cipher, and the AES-ECB. Also, the implementation area has increased due to the YCbCr conversion and the blur detection, which are not available in the other systems.

Most of the image compression-encryption systems are implemented using software solutions. Accordingly, the proposed FPGA system is also compared with other CPU based systems, which use new and non-standardized encryption methods, as shown in Table 3.

TABLE 2
THE DEVICE LOGIC UTILIZATION AND THROUGHPUT FOR DIFFERENT FPGA-BASED COMPRESSION-ENCRYPTION SYSTEMS.

System	Image	Detect Blur	Compression	Cipher	Device	Slice LUT	Slice Reg	Slices	DSP48E	Clock (MHz)	Throughput (Mbps)
[45]	Gray	✗	Haar DWT SLCCA	128-AES (ECB)	Altera EPF10K250A	NA	NA	NA	NA	40	330
[46]	Gray	✗	JPEG	TEA	Virtex-6	2130	1456	NA	9	87.290	698.32
[47]	Gray	✗	DCT	Stream cipher	XC5VLX330	2058	1536	NA	0	NA	1648
[62]	NA	✗	✗	Keccak-f[400] 128-AES	Virtex-6	NA	NA	809	NA	332.98	2131.07
This work	**Color**	**✓**	**Haar DWT RLE**	**128-AES (CBC)**	**XC5VLX50T**	**5736**	**2955**	**1866**	**0**	**144.11**	**3458**

TABLE 3
THE EXECUTION TIME (ET) COMPARISON WITH DIFFERENT CPU -BASED IMAGE COMPRESSION-ENCRYPTION SYSTEMS.

System	Device Specs			Image type	Detect Blur	Compression	256×256 ET (s)	512×512 ET (s)	1024×1024 ET (s)
	Processor	RAM (GB)	Clock (GHz)						
[37]	Core i3- 4005U	2	1.7	Color	✗	DWT	12.18	47.42	NA
[38]	Core i7-6500U	4	3.1	Gray	✗	Comp. sensing	7.07	NA	NA
[39]	Core i7	8	3.4	Gray	✗	Comp. sensing	6.2	NA	NA
[40]	NA	4	3.3	Color	✗	Comp. sensing	1.1168	3.9593	NA
[41]	Core i7-4770K	16	3.5	Gray	✗	JPEG	0.98	2.64	8.33
[42]	Core i5	4	3.1	Gray	✗	DWT, SPIHT	0.773	3.56	NA
[43]	Core i7-3667U	8	2.5	Gray	✗	Comp. sensing	0.4769	1.4712	NA
[44]	Core i7-6700	8	2.9	Gray	✗	Comp. sensing	0.2387	0.731	NA
[53]	Core i5-6500	8	3.2	Gray	✗	Comp. sensing	0.03178	0.10335	NA
[54]	Core i3	4	2.13	Gray	✗	Comp. sensing	0.01983	0.054	NA
[55]	Core2 Duo	2	2.4	Gray	✗	Chinese Remainder	0.018	0.05	0.268
[56]	Core i3–2120	2	3.3	Gray	✗	Comp. sensing	NA	0.0278	NA
[57]	Core 2 Duo	4	1.86	Gray	✗	DCT, Huffman	NA	0.02632	NA
This work	**FPGA XC5VLX50T**		**0.144**	**Color**	**✓**	**DWT, RLE**	**0.00046**	**0.00182**	**0.007276**

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3086044, IEEE Access

Like the previous discussion, the proposed system is superior in terms of speed. Only [37] and [40] work on color images, however their Execution Time (ET) is not suitable for real-time applications. According to the results in Table 3, the proposed system is 13.9 times as fast as the best CPU based implementation [57].

Furthermore, the ET of the proposed FPGA system is compared with the software implementation of image blur detection. In [58], Tong et al.'s blur detection algorithm was implemented on Jetson TK1 using different approaches: the single-core Sequential approach [15], the multi-core parallel approach [59], and the GPU parallel approach [58]. The comparison is presented in Fig. 19, which shows that the FPGA implementation is 1.773 times as fast as the GPU-based implementation.

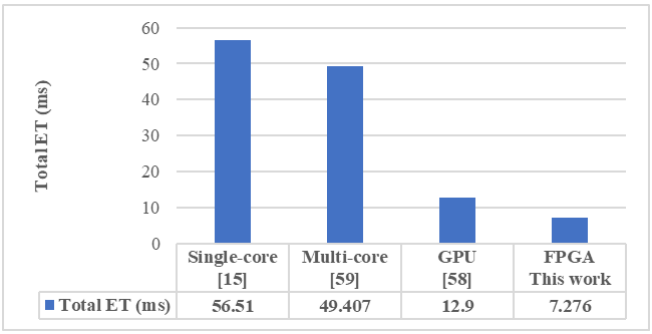


FIGURE 19. The blur detection ET for 1024×1024 color images using different implementations of Tong et. al.'s algorithm. The SW solutions were implemented on Jetson Tk1.

C. The blur detection results

The blur detection feature of the proposed system is tested on 256×256 images with different blur levels. The blur threshold of the algorithm is set to 10. The input image is considered blurred if the ratio of N_{DA} to N_{edges} is less than 5%, otherwise, the image is considered unblurred. The simulation results are shown in Fig. 20 where the images on the left are classified by the system as unblurred while the other images are classified as blurred. Furthermore, by examining the same image at different blur levels, it is clear that the estimated blur extent can be used to rank the images according to the amount of blur.

D. The compression-encryption results

The compression performance of the system is assessed by calculating the Peak Signal-to-Noise Ratio (PSNR) versus the Compression Ratio (CR). The CR is calculated as shown in (5). The PSNR is calculated as shown in (6a), (6b), and (6c) where *MSE* is the Mean Square Error, $M \times N$ is the dimensions of the image, $f_1(x,y)$ is the plain image, and $f_2(x,y)$ is the reconstructed image. In color images, the PSNR is calculated for each color component, (Red, Green, and Blue), and then the average value is calculated as shown in (6c). In lossy image compression, the PSNR is inversely proportional to the CR. It is important to keep the PSNR above certain limit to achieve good image quality. The reconstructed image quality is considered good if the PSNR values are between 30dB and 50dB. The compression is unacceptable if the PSNR is below 20dB [37].

$$CR = \frac{size\ of\ cipher\ image}{size\ of\ plain\ image} \tag{5}$$

$$MSE = \frac{1}{M \times N} \sum_{y=1}^{M} \sum_{x=1}^{N} \left(f_1(x,y) - f_2(x,y)\right)^2 \tag{6a}$$

$$PSNR = 10 \times log_{10}\left(\frac{255^2}{MSE}\right) \tag{6b}$$

$$PSNR_{RGB} = \frac{PSNR_{Red} + PSNR_{Green} + PSNR_{Blue}}{3} \tag{6c}$$

The simulation results of the compression-encryption process are shown in Fig. 21. The CR ratio is adjusted to approximately 20% for the three 256×256 test images, "Baboon", "Barbara", and "Lena". The images are successfully ciphered by the AES as shown in the figure. Regarding the quality of the restored images, the "Baboon" has the lowest PSNR while "Lena" has the highest PSNR. This is due to the amount of details in the image. Regarding the "Baboon" image, which has a lot of details, the compression threshold must be set high to reach the 20% CR at the expense of lowering the PSNR.

For further investigation, the PSNR vs the CR is evaluated for all the standard test images. For each image, the compression threshold is increased until the PSNR drops to 30dB. Since the compression threshold is restricted to integer values, it is hard to get exactly the 30dB point. Accordingly, the point closest to the 30dB will be marked for each image.

The PSNR vs the CR for all the color and gray images are presented in Fig. 22 and Fig. 23, respectively. Each image is tested with two resolutions, 256×256 and 512×512 as shown by the red and blue curves, respectively. By inspecting the 30dB point for each test image, it is found that the CR is greatly affected by the amount of details in the image. For example, in the 256×256 images, the "House" has the best CR, which is 0.08 and 0.11 for color and grayscale, respectively. On the other hand, the "Baboon" has the worst CR, which is 0.38 and 0.75 for color and grayscale, respectively.

Moreover, the results show that the CR is better in color images compared to the grayscale images. This is due to the YCbCr conversion, which puts most of the details in the *Y* component as the human visual system is more sensitive to the luminance than chrominance. By inspecting the 30dB point in the 256×256 "House" and "Baboon", it is found that the CR of the color image is better than the grayscale image by 27% and 49%, respectively.

Furthermore, the results show that the compression performance is improved at higher resolutions. For example, by examining the 30dB point of the "Plane" color image, the CR at resolution 512×512 is better than 256×256 by 43.75%.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3086044, IEEE Access

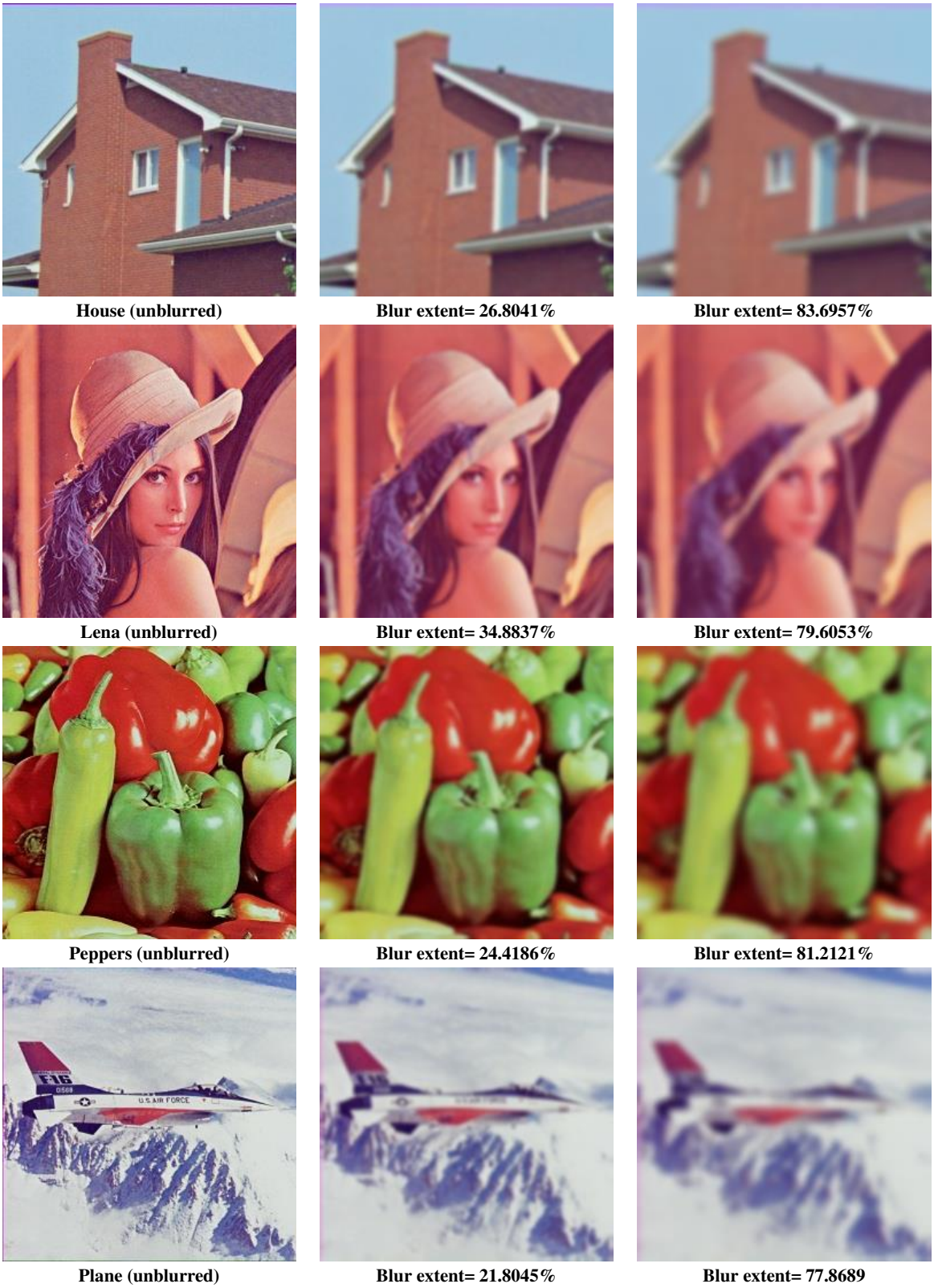


FIGURE 20. **The Simulation results of the blur detection for different 256×256 test images. The calculated blur extent is written for each image.**

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3086044, IEEE Access

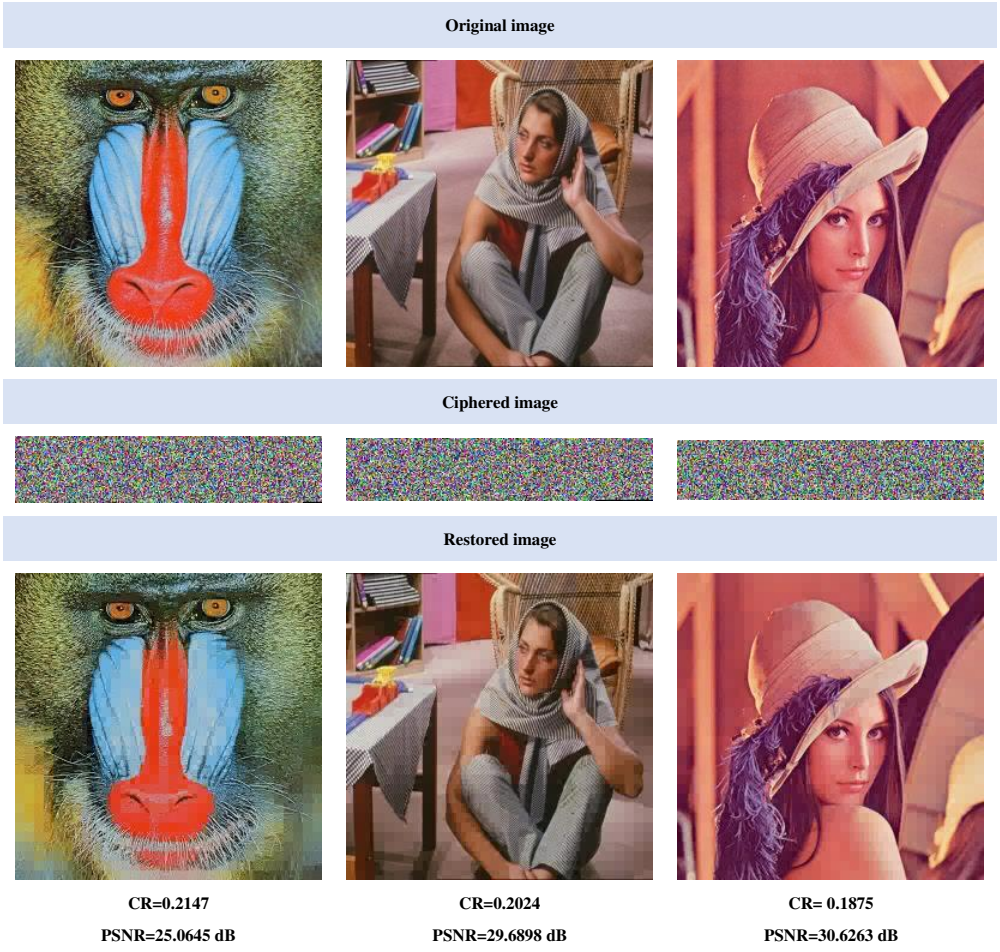


FIGURE 21. The simulation results for the compression encryption process.

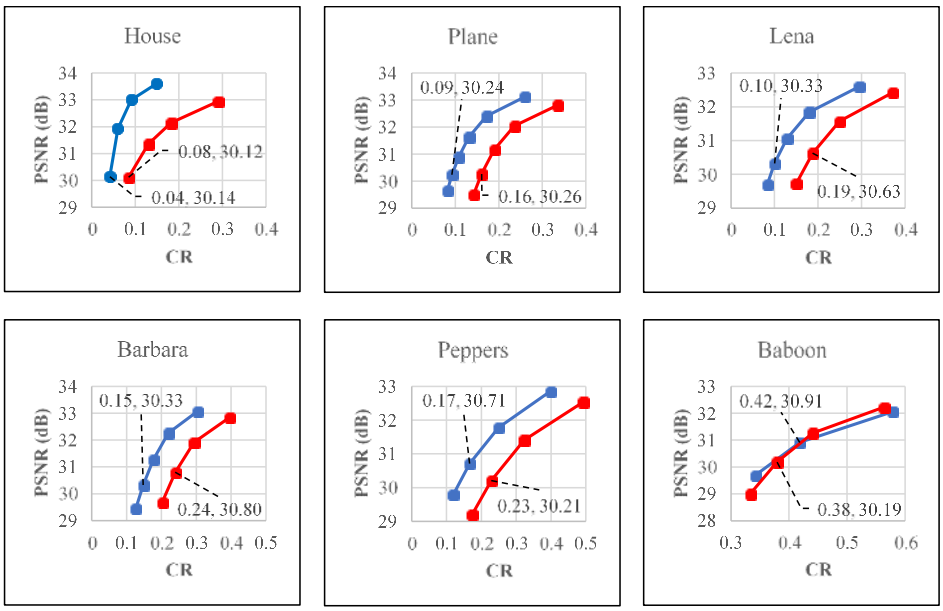


FIGURE 22. The PSNR vs CR of the proposed system for color images. The red and blue curves represent the 256×256 and 512×512 images.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3086044, IEEE Access

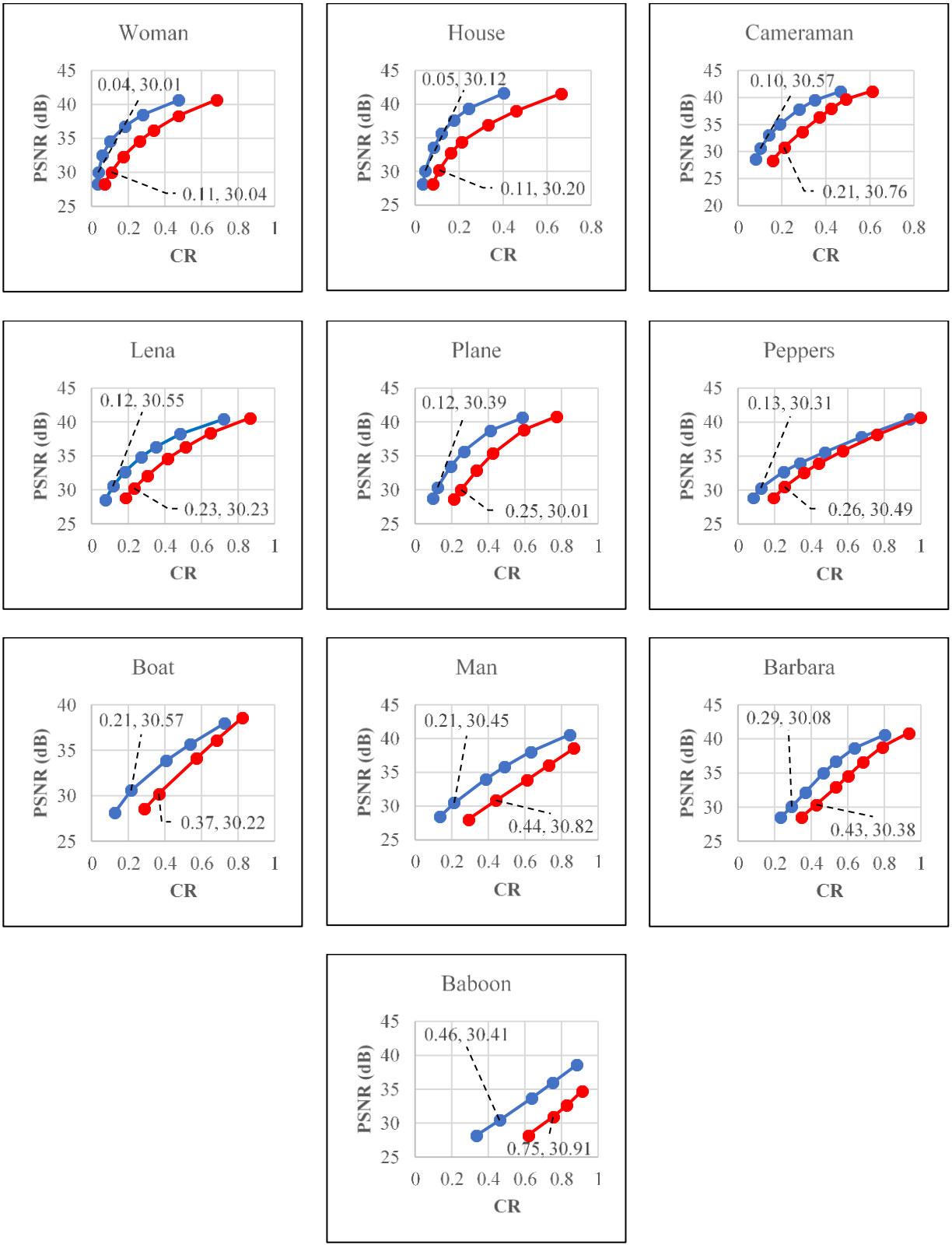


FIGURE 23. **The PSNR vs CR of the proposed system for several gray images. The red and blue curves represent the 256×256 and 512×512 images, respectively.**

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3086044, IEEE Access

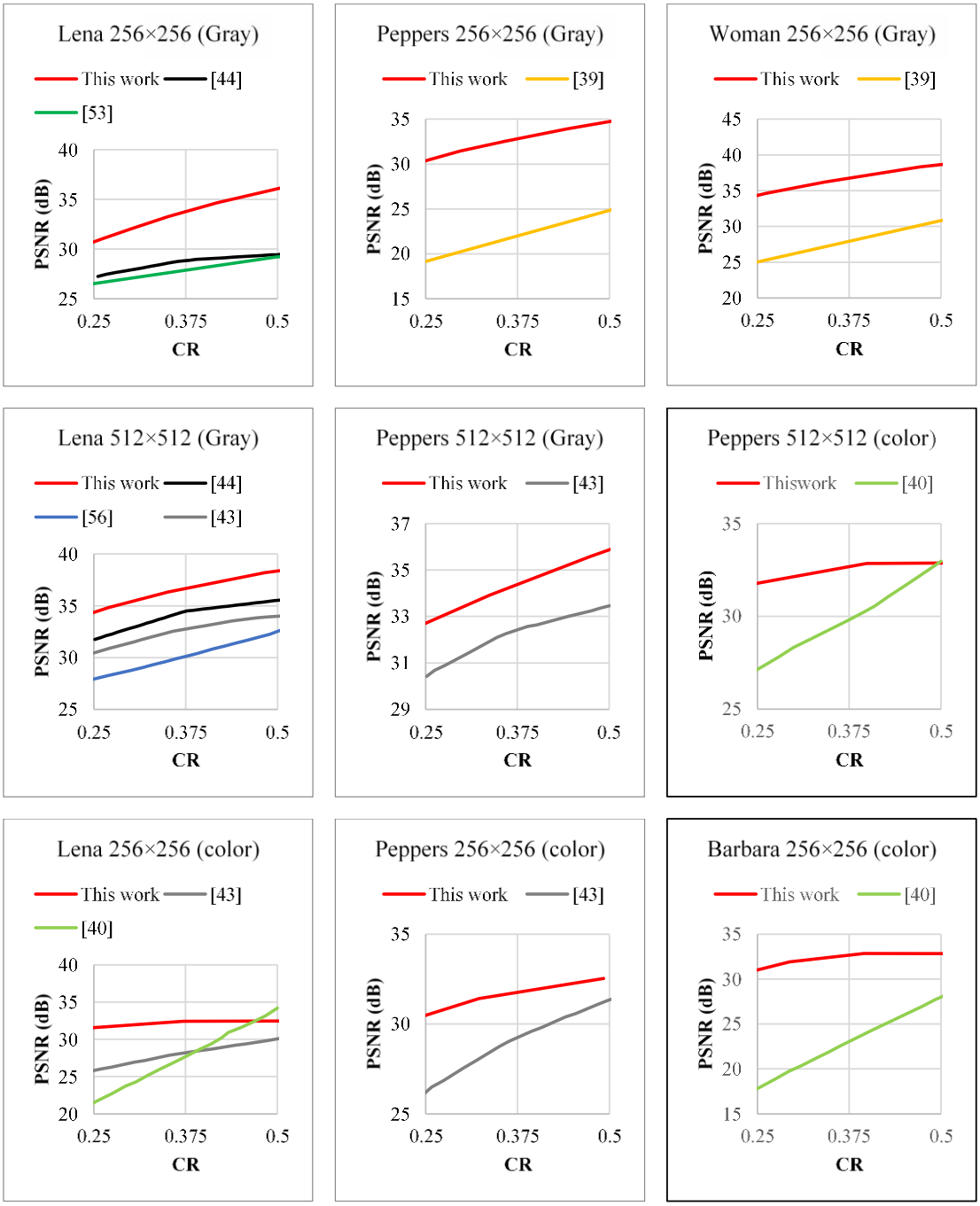


FIGURE 24. The compression performance comparison for different compression-encryption systems.

The PSNR vs CR of the proposed system is compared with other systems as shown in Fig. 24. The results show that the proposed system can achieve high PSNR at low CR. By inspecting the lowest CR in the figure, which is the 25%, it is found that the PSNR of the proposed system is better than the other systems by more than 8%. However, the compression scheme in [40] can provide better PSNR at CRs higher than 50%. This is due to the truncation rounding scheme of the YCbCr converter, which limits the maximum PSNR of the proposed system to 33dB.

In addition to the PSNR, the Mean Structural Similarity (MSSIM) index is also used to measure the similarity between the original and the restored images from three aspects: brightness, contrast, and structure. The MSSIM can be calculated using the equations presented in [38]. Table 4 compares the MSSIM of the proposed system with one of the most recent compression-encryption systems [38] at 25% CR.

TABLE 4
THE MSSIM INDEX OF THE PROPOSED SYSTEM AT 25% CR

Image	This Work	[38]
Lena (256×256)	0.5710	0.6211
Pepper (256×256)	0.5733	0.6354
Man (256×256)	0.5695	0.5375

Three different 256×256 grayscale images are used in table 4. The results show that the MSSIM of the proposed work is close to the system presented in [38]. In the case of "Lena" and "Peppers", the MSSIM is 9% lower than the system in [38]. On the other hand, in the case of the "Man" image, the MSSIM of the proposed system is 6% higher than the system in [38].

V. Conclusion

In this paper, the color image blur detection has been integrated with compression and encryption as a 3 in 1 parallel HW architecture. The 128-bit AES-CBC has been combined with the modified Lorenz chaotic PRNG to provide a highly secure encryption scheme. To reduce the resources, the Haar DWT has been used as a common building block for both blur detection and compression. Furthermore, to achieve high speed, the entropy encoder has been implemented using the RLE technique. The proposed system has been implemented on FPGA (XC5VLX50T) using only 25% of the available slices. The system can process 4.5MP images at a rate of 30 FPS, which is more than 100% faster than all the available FPGA-based image compression-encryption systems. In addition, the system has been tested with all the standard 256×256 images. It is shown that depending on the amount of details in the image, the system can achieve 30dB PSNR at CRs in the range of (0.08-0.38). Furthermore, it is shown that the PSNR of the proposed system at 25% CR is higher than most of the newly published compression-encryption systems by more than 8%.

REFERENCES

[1] K. H. Talukder and K. Harada, "Haar Wavelet Based Approach for Image Compression and Quality Assessment of Compressed Image", *IAENG int. J. Applied Math*., vol. 36, 2007.

[2] A. J. Hussain, A. Al-Fayadh, and N. Radi, "Image compression techniques: A survey in lossless and lossy algorithms," *Neurocomputing*, vol. 300, Jul. 2018.

[3] I. W. Selesnick, R. G. Baraniuk and N. C. Kingsbury, "The dual-tree complex wavelet transform," *IEEE Signal Process. Magazine*, vol. 22, no. 6, pp. 123-151, Nov. 2005.

[4] J. Shi, X. Liu, W. Xiang, M. Han and Q. Zhang, "Novel Fractional Wavelet Packet Transform: Theory, Implementation, and Applications," *IEEE Trans. Signal Process.*, vol. 68, pp. 4041-4054, 2020.

[5] R. Ueno *et al*., "High Throughput/Gate AES Hardware Architectures Based on Datapath Compression," *IEEE Trans. Computers*, vol. 69, no. 4, pp. 534-548, 1 April 2020.

[6] O. E. Rössler, "An equation for continuous chaos,". *Phys. Lett. A*, vol. 57, no. 5, pp. 397–398, July 1976.

[7] E. Lorenz, "Deterministic nonperiodic flow,". *Atmos Sci.*, vol. 20, no. 2, pp. 130-141, Mar 1963.

[8] J. Schmitz and Lei Zhang, "Rössler-based chaotic communication system implemented on FPGA," in *Proc. IEEE Canadian Conf. Elect. Comp. Eng. (CCECE)*, 2017, pp. 1-4.

[9] L. Zhang, "System generator model-based FPGA design optimization and hardware co-simulation for Lorenz chaotic generator," in *Proc. IEEE Asia Pacific Conf. Intel. Robot Syst. (ACIRS)*, 2017, pp. 170-174.

[10] A. S. Elwakil, and M. P. Kennedy, "Chaotic oscillator configuration using a frequency dependent negative resistor," *J. Circ. Syst. Comp.*, vol 9, no 3, pp. 229-242, 1999.

[11] A. S. Elwakil and M. P. Kennedy, "Construction of classes of circuit-independent chaotic oscillators using passive-only nonlinear devices," *IEEE Trans. Circ. Sys. I: Fundamental Theory and Applications*, vol. 48, no. 3, pp. 289-307, March 2001.

[12] A. A. Rezk, A. H. Madian, A. G. Radwan and A. M. Soliman, "Multiplierless chaotic pseudo random number generators," *AEU-Int. J. Electron. Commun.*, vol. 113, Jan. 2020.

[13] B. T. Koik and H. Ibrahim, "A literature survey on blur detection algorithms for digital imaging, " in *Proc. IEEE 1st Int. Conf. Artif. Intell. Modelling Simulation*, 2013, pp. 272-277.

[14] V. Kamble and K. M. Bhurchandi, "No-reference image quality assessment algorithms: A survey," *Optik*, vol. 126, no. 11, pp. 1090-1097, Jun. 2015.

[15] H. Tong, M. Li, H. Zhang and C. Zhang, "Blur detection for digital images using wavelet transform," in *Proc. IEEE Int. Conf. Multimedia Expo*, 2004, pp. 17-20.

[16] P. Marziliano, F. Dufaux, S. Winkler and T. Ebrahimi, "Perceptual blur and ringing metrics: Application to JPEG2000," *Signal Process. Image Commun.*, vol. 19, no. 2, pp. 163-172, Feb. 2004.

[17] R. Ferzli and L. J. Karam, "A no-reference objective image sharpness metric based on the notion of just noticeable blur (JNB)," *IEEE Trans. Image Process.*, vol. 18, no. 4, pp. 717-728, Apr. 2009.

[18] S. Wu, W. Lin, Z. Lu, E. P. Ong, and S. Yao, "Blind blur assessment for vision-based applications," *J. Vis. Commun. Image Represent.*, vol. 20, no. 4, pp. 231-241, May 2009.

[19] N. D. Narvekar and L. J. Karam, "A no-reference image blur metric based on the cumulative probability of blur detection (CPBD)," *IEEE Trans. Image Process.*, vol. 20, no. 9, pp. 2678-2683, Sep. 2011.

[20] I. Marais and W. H. Steyn, "Robust defocus blur identification in the context of blind image quality assessment," *Signal Process. Image Commun.*, vol. 22, no. 2, pp. 833-844, 2007.

[21] E. Cohen and Y. Yitzhaky, "No-reference assessment of blur and noise impacts on image quality," *Signal Image Video Process.*, vol. 4, no. 3, pp. 289-302, 2010.

[22] J. Zhang, S. H. Ong and T. M. Le, "Kurtosis-based no-reference quality assessment of JPEG2000 images," *Signal Process. Image Commun.*, vol. 26, no. 1, pp. 13-23, Jan. 2011.

[23] X. Marichal, W.-Y. Ma and H. Zhang, "Blur determination in the compressed domain using DCT information,"in *Proc. IEEE Int. Conf. Image Process.*, 1999, pp. 386-390.

[24] J. Shen, Q. Li and G. Erlebacher, "Hybrid no-reference natural image quality assessment of noisy blurry JPEG2000 and JPEG images," *IEEE Trans. Image Process.*, vol. 20, no. 8, pp. 2089-2098, Aug. 2011.

[25] S. Yousaf and S. Qin, "Closed-Loop Restoration Approach to Blurry Images Based on Machine Learning and Feedback Optimization," *IEEE Trans. Image Process.*, vol. 24, no. 12, pp. 5928-5941, Dec. 2015.

[26] Y. Wu, X. Shen, T. Mei, X. Tian, N. Yu, and Y. Rui, "Monet: A System for Reliving Your Memories by Theme-Based Photo Storytelling," *IEEE Trans. Multimedia*, vol. 18, no. 11, pp. 2206-2216, Nov. 2016.

[27] M. Kucer, A. C. Loui and D. W. Messinger, "Leveraging Expert Feature Knowledge for Predicting Image Aesthetics," *IEEE Trans. Image Process.*, vol. 27, no. 10, pp. 5100-5112, Oct. 2018.

[28] S. Chowdhury, M. S. Ferdous and J. M. Jose, "A user-study examining visualization of lifelogs," in *Proc IEEE Int. Workshop on Content-Based Multimedia Indexing (CBMI)*, Bucharest, Romania, 2016.

[29] L.-W. Kim, "DeepX: Deep learning accelerator for restricted boltzmann machine artificial neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 5, pp. 1441-1453, May 2018.

[30] A. A. Rezk, A. H. Madian, A. G. Radwan and A. M. Soliman, "Reconfigurable chaotic pseudo random number generator based on FPGA," *AEU Int. J. Electron. Commun.*, vol. 98, pp. 174-180, Jan. 2019.

[31] J. Lee, H. Tang and J. Park, "Energy efficient canny edge detector for advanced mobile vision applications," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 4, pp. 1037-1046, Apr. 2018.

[32] S. H. Farghaly and S. M. Ismail, "Floating-point discrete wavelet transform-based image compression on FPGA," *AEU-Int. J. Electron. Commun.*, vol. 124, 2020.

[33] J. Wang, B. Li and K. Xing, "A New Real-Time Lucky Imaging Algorithm and its Implementation Techniques," *IEEE Access*, vol. 8, pp. 52192-52208, 2020.

[34] S. M. Hussain, F. U. D. Farrukh, S. Su, Z. Wang and H. Chen, "CMOS Image Sensor Design and Image Processing Algorithm Implementation for Total Hip Arthroplasty Surgery," *IEEE Trans. Bio. Circ. Sys.*, vol. 13, no. 6, pp. 1383-1392, Dec. 2019.

[35] J. Dubois, D. Ginhac, M. Paindavoine and B. Heyrman, "A 10 000 fps CMOS Sensor With Massively Parallel Image Processing," *IEEE J. Solid-State Circ.*, vol. 43, no. 3, pp. 706-717, March 2008.

[36] C. Krintz and S. Sucu, "Adaptive on-the-fly compression," *IEEE Trans. Par. Dis. Sys*, vol. 17, no. 1, pp. 15-24, Jan. 2006.

[37] E. Setyaningsih, R. Wardoyo and A. K. Sari, "Securing color image transmission using compression-encryption model with dynamic key generator and efficient symmetric key distribution," *Digit. Commun. Netw.*, vol. 6, no. 4, pp. 486-503, Nov. 2020.

[38] H. Hu, Y. Cao, J. Xu, C. Ma, and H. Yan, "An Image Compression and Encryption Algorithm Based on the Fractional-Order Simplest Chaotic Circuit," *IEEE Access*, vol. 9, pp. 22141-22155, 2021.

[39] L. Gong, K. Qiu, C. Deng, and N. Zhou, "An image compression and encryption algorithm based on chaotic system and compressive sensing," *Opt. Laser Technol.*, vol. 115, pp. 257-267, Jul. 2019.

[40] X. Chai, J. Bi, Z. Gan, X. Liu, Y. Zhang, and Y. Chen, "Color image compression and encryption scheme based on compressive sensing and double random encryption strategy," *Signal Process.*, vol. 176, Nov. 2020.

[41] P. Li and K. Lo, "A Content-Adaptive Joint Image Compression and Encryption Scheme," *IEEE Transactions on Multimedia*, vol. 20, no. 8, pp. 1960-1972, Aug. 2018.

[42] M. Hamdi, R. Rhouma and S. Belghith, "A selective compression-encryption of images based on SPIHT coding and chirikov standard map," *Signal Process.*, vol. 131, pp. 514-526, Feb. 2017.

[43] Z. Zhu, Y. Song, W. Zhang, H. Yu, and Y. Zhao, "A novel compressive sensing-based framework for image compression-encryption with S-box," *Multimedia Tools Applicat.*, vol. 79, no. 1, pp. 25497–25533, 2020.

[44] L. Zhu, H. Song, X. Zhang, M. Yan, L. Zhang, and T. Yan, "A Novel Image Encryption Scheme Based on Nonuniform Sampling in Block Compressive Sensing," *IEEE Access*, vol. 7, pp. 22161-22174, 2019.

[45] S. Ou, H. Chung, and W. Sung, "Improving the compression and encryption of images using FPGA-based cryptosystems," *Multimedia Tools Applicat.*, vol. 28, no. 1, pp. 5-22, Jan. 2006.

[46] A. Ç. Bağbaba and B. Örs, "Hardware implementation of novel image compression-encryption system on a FPGA," in *Proc. IEEE Int. Conf. Electrical and Electronics Engineering (ELECO)*, Bursa, Turkey, 2015, pp. 1159-1163.

[47] M. Jridi and A. AlFalou, "A VLSI implementation of a new simultaneous images compression and encryption method," in *Proc. IEEE Int. Conf. Imaging Syst. and Tech.*, Thessaloniki, Greece, 2010.

[48] L. T. Ko, J. E. Chen, and H.C. Hsin, "Haar-Wavelet-Based Just Noticeable Distortion Model for Transparent Watermark," *Math. Prob. Eng. Arcticle*, 2012.

[49] M.E. Angelopoulou, K. Masselos, P.Y.K. Cheung and Y. Andreopoulos, "Implementation and Comparison of the 5/3 Lifting 2D Discrete Wavelet Transform Computation Schedules on FPGAs," *J. VLSI Signal Processing*, pp. 3-21, 2007.

[50] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, and E. Barker, "A statistical test suite for random and pseudorandom number generators for cryptographic applications," technical report, DTIC Document; 2001.

[51] Microsemi, "UG0639 User Guide Color Space Conversion," datasheet, [Revised Sept. 2021].

[52] M. Dworkin, E. Barker, J. Nechvatal, J. Foti, L. Bassham, E. Roback, and J. Dray, "Advanced Encryption Standard (AES)," NIST FIPS, [Online]. Available: https://www.nist.gov/publications/advanced-encryption-standard-aes.

[53] Q. Xu, K. Sun, C. Cao, and C. Zhu, "A fast image encryption algorithm based on compressive sensing and hyperchaotic map", *Opt. Lasers Eng.*, vol. 121, pp. 203-214, Oct. 2019.

[54] A. H. Brahim, A. A. Pacha, and N. H. Said, "Image encryption based on compressive sensing and chaos systems," *Opt. Laser Technol.*, vol. 132, 2020.

[55] H. Zhu, C. Zhao, and X. Zhang, "A novel image encryption–compression scheme using hyper-chaos and Chinese remainder theorem," *Signal Process. Image Commun.*, vol. 28, no. 6, pp. 670-680, Jul. 2013.

[56] T. Chen, M. Zhang, J. Wu, C. Yuen, and Y. Tong, "Image encryption and compression based on Kronecker compressed sensing and elementary cellular automata scrambling," *Opt. Laser Technol.*, vol. 84, pp. 118-133, Oct. 2016.

[57] C.-H. Yuen and K.-W. Wong, "A chaos-based joint image compression and encryption scheme using DCT and SHA-1," *Appl. Soft Comput.*, vol. 11, no. 8, pp. 5092-5098, 2011.

[58] G. S. Tran, T. P. Nghiem, and J-C. Burie, "Fast parallel blur detection on GPU," *J. Real Time Image Process.,* vol. 17, pp. 903–913, 2018.

[59] G. S. Tran, T. P. Nghiem, N. Q. Doan, A. Drogoul, L. C. Mai, "Fast parallel blur detection of digital images," in *Proc. IEEE Int. Conf. Comp. Comm. Tech., Research, Innovation, and Vision for the Future (RIVF*), 2016, pp. 147–152.

[60] M. F. Khan, S. M. Monir, and I. Naseem, "Robust image hashing based on structural and perceptual features for authentication of color images," *Turkish J. Elect. Eng. Comp. Sci.*, vol 29, pp.648-662, 2021.

[61] K. Altun and E. Günay, "A novel chaos-based modulation scheme: adaptive threshold level chaotic on–off keying for increased BER performance," *Turkish J. Elect. Eng. Comp. Sci.*, vol 28, pp.606-620, 2020.

[62] D. -e. -S. Kundi, A. Khalid, A. Aziz, C. Wang, M. O'Neill and W. Liu, "Resource-Shared Crypto-Coprocessor of AES Enc/Dec With SHA-3," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 67, no. 12, pp. 4869-4882, Dec. 2020.

AHMED A. REZK received the B.Sc. degree in Electronics from the German University in Cairo, Egypt, in 2012 and the M.Sc. degree in Electronics and Communications from the American University in Cairo, Egypt, in 2015. He is currently pursuing the Ph.D. degree in Electronics and Communications at Cairo University, Egypt.

From 2012 to 2015, he was a Research Assistant at the Centre of Nano Electronics and Devices, Zewail City, Egypt. From 2016 to 2019, he was a Teaching Assistant at the Electronics and Computer Engineering department, Nile University, Egypt. Since 2019, he has been a Teaching Assistant at the Communications and Information Engineering department, University of Science and Technology, Zewail city, Egypt. His research interests include digital design and implementation of encryption systems, image processing techniques, and Photovoltaic maximum power point tracking algorithms.

AHMED H. MADIAN (SM'12) received his Ph.D. and M.Sc. degrees from Cairo University, Egypt, in 2007 and 2002, respectively. He is currently Professor at the Department of Electronics and computer Engineering, Faculty of Engineering and applied science, NILE University, Giza, Egypt. He is the director of Microelectronics System Design Master Program since sept. 2015. Also, He is the director of Nanoelectronics Integrated System Design Research center (NISC) since 2016. He has published more than 150 papers in international conferences and journals. His H-index is currently 20. Also, he served in the many technical and organizing committee of many international conferences. He received many research grants as Principle Investigator (PI), CO-PI, or Consultant from different national/international organizations. He won the best researcher award (Dr. Hazem Ezzat award 2017) for his outstanding research profile. His research interests are in circuit theory; low-voltage analog CMOS circuit design, current-mode analog signal processing, Memristors, Fractional systems, VLSI, Encryption systems and mixed/ digital applications on field programmable gate arrays. Also, he is member of the national radio of science committee (NRSC) since 2018. Dr. Madian is actively serving as a reviewer in several journal and conference publications including IEEE conferences and journals. He served as guest associate editor for many international journals. He is the founder of IEEE Circuits and systems (CASS) Egypt technical chapter and co-founder of the IEEE Robotics and automations (RAS) Egypt technical chapter. He is currently the IEEE Egypt Section Secretary and member of Ex-COM.

AHMED G. RADWAN Ahmed G. Radwan (Senior Member, IEEE) was the Former Director of the Nanoelectronics Integrated Systems Center (NISC), Nile University, Egypt, and the Technical Center for Carrier Development (TCCD), Cairo University, Egypt. He was a Visiting Professor with the Computational Electromagnetic Laboratory (CEL), Electrical and Computer Engineering Department (ECE), McMaster University, Canada, from 2008 to 2009, then he was selected to be a part of the first foundation research teams to join the King Abdullah University of Science and Technology (KAUST), from 2009 to 2011. He is currently the Vice President of Research with Nile University and a Professor with the Engineering Mathematics and Physics Department, Cairo University. During the last ten years, he has many academic visits as a Session Chair/Organizer, an Invited Speaker, and attending international conferences in several countries, such as Malaysia, Czech Republic, Brazil, France, Tunis, UAE, Sweden, China, Germany, KSA, Lebanon, Portugal, Canada, Spain, Hong Kong, Austria, and Italy. He has more than 340 articles, H-index 43, and more than 5600 citations based on Scopus database, seven international books, and 18 book chapters in the highly ranked publishers, such as Elsevier and Springer. He has Six U.S. patents in several interdisciplinary topics. He is the Founder of the patent "3D-Fractional Order Smith Chart." Moreover, he is the Founder of the undergraduate student research activity "Undergraduate Research Forum" https://nu.edu.eg/research-forum/. He was awarded the Best Master Thesis Award, the Best Thesis Supervisor Award for four Ph.D. theses, and the three M.Sc. theses from different universities on several research tracks. He received many research grants as a Principle Investigator (PI), a CO-PI, or a Consultant from different national/international organizations, such as ASRT, STDF, Cairo University, and the Newton-Mosharafa Funding Agency. His research interests include interdisciplinary concepts between mathematics and engineering applications, such as fractional-order systems, bifurcation, chaos, memristor, and encryption, Dr. Radwan is a member of the National Committee of Mathematics and the Applied Science Research Council, specialized scientific councils, the Academy of Scientific Research and Technology (ASRT), Egypt. Moreover, he was a Former Member of the Egyptian Young Academy of Science (EYAS), ASRT, Egypt. He is a fellow of the African Academy of Sciences. Based on Scival database, he is on the top authors worldwide for the two research tracks "Capacitors|Networks(circuits)|Fractional-order capacitor (T.21555)" and "Chaos theory|Chaotic systems|Multi-scroll chaotic (T.8806)." He was currently selected as a MC Observer to COST Action CA15225 http://www.cost.eu/COST_Actions/ca/CA15225. He received the Scopus Award in engineering and technology, in 2019, from Elsevier as the Top Researcher in Egypt, from 2014 to 2018, (based on H-index, field-weighted citation impact, and number of publications), the State First Class Medal of Science and Arts, the State Excellence Award for advanced technological sciences, in 2018, the Cairo University Excellence Award for research in the engineering sciences, in 2016, the Abdul Hameed Shoman Award for Arab researchers in basic sciences (Information and Data Security), in 2015, the State Achievements Award for research in mathematical sciences, in 2012, Prof. Mohamed Amin Lotfy Award from ASRT in the mathematical sciences, in 2016, the Cairo University Achievements Award for research in the engineering sciences, in 2013, and the Best Researcher Awards from Nile University, in 2015 and 2016. He awarded the best paper/poster awards in several international conferences, such as Modern Circuits and Systems Technologies (MOCAST) 2017-Greece, ICECS2015-Egypt, ECTI-CON2016-Thailand, International Conference of Microelectronics 2013-Lebanon, and the International Conference of Microelectronics 2011-Tunis. He won the International Publications Award endowed by Cairo University for the top researchers in all fields through the years 2011, 2012, 2013, 2014, 2015, 2016, 2017, and 2018, individually. He was Honored from Cairo University President in the 13th and 15th Cairo University Science Festival Days, in December 2015 and 2017. He has also been invited by ASRT to attend the Presidential Science Festival, in 2014 and 2017. He is the Founder and a General Co-Chair of the First and Second Novel Intelligent and Leading Emerging Sciences Conference NILES2019–NILES2020, Egypt. He was the Technical Program Co-Chair from the 28th IEEE International Conference in Microelectronics (ICM2016), Cairo. He organized several special sessions: PIER2011 (China), PIER2012 (Malaysia), NOLTA2015 (Hong Kong), MOCAST2017, and MOCAST2019 (Greece). He selected to be a Counselor of the IEEE Nile University Student Branch (NUSB), from

October 2014 to 2016. He is selected on the Editorial Board of Journal of Engineering and Applied Science. He lead as a Guest Editor of different Special Issues, such as Journal of Circuits, Systems and Signal Processing in 2015 (1.922), Mathematical Problems in Engineering in 2017 (1.179), Complexity in 2017 (4.621), International Journal of Electronics and Communications (AEU) in 2018 (2.853), Microelectronics Journal in 2019 (1.284), International Journal of Electronics and Communications (AEU) in 2020 (2.853), and Journal of Advanced Research in 2020 (5.054).

AHMED M. SOLIMAN was born in Cairo Egypt, on November 22, 1943. He received the B.Sc. degree with honors from Cairo University, Cairo, Egypt, in 1964,the M.S. and Ph.D. degrees from the University of Pittsburgh, Pittsburgh, PA., U.S.A., in 1967 and 1970, respectively, all in Electrical Engineering. He is currently Professor Emeritus; Electronics and Communications Engineering Department, Cairo University, Egypt. From September 1997-September 2003, Dr Soliman served as Professor and Chairman Electronics and Communications Engineering Department, Cairo University, Egypt. From 1985-1987, Dr. Soliman served as Professor and Chairman of the Electrical Engineering Department, United Arab Emirates University, and from 1987-1991 he was the Associate Dean of Engineering at the same University. He has held visiting academic appointments at San Francisco State University, Florida Atlantic University and the American University in Cairo. He was a visiting scholar at Bochum University, Germany (Summer 1985) and with the Technical University of Wien, Austria (Summer 1987).In 1977, Dr. Soliman was decorated with the First Class Science Medal, from the President of Egypt, for his services to the field of Engineering and Engineering Education. In 2008, Dr Soliman received the State Engineering Science Excellency Prize Award from the Academy of Scientific Research Egypt. In 2010, Dr Soliman received the State Engineering Science Appreciation Prize Award from the Academy of Scientific Research Egypt. In 2013, Dr. Soliman was decorated with the First Class Science Medal, from the President of Egypt, for his services to Egypt. Dr. Soliman was a Member of the Editorial Board of the IET Proceedings Circuits Devices and Systems and is associate editor now. Dr. Soliman served as a Member of the Editorial Board of Electrical and Computer Engineering (Hindawi). Dr. Soliman is a Member of the Editorial Board of Analog Integrated Circuits and Signal Processing. Dr. Soliman is also a Member of the Editorial Board of Scientific Research and Essays. Dr. Soliman served as Associate Editor of the IEEE Transactions on Circuits and Systems I (Analog Circuits and Filters) from December 2001 to December 2003 and is Associate Editor of the Journal of Circuits, Systems and Signal Processing from January 2004-Now.Dr. Soliman is Associate Editor of the Journal of Advanced Research (JAR) Cairo University. Dr. Soliman is the inventor (with Dr. Inas Awad) of the pathological Voltage Mirror and the pathological Current Mirror. Dr. Soliman is the inventor (with Dr. Inas Awad) of the family of the Inverting Current Conveyors which completes the set of CCII invented by Dr Sedra and Dr Smith. Dr Soliman received the Excellency Award Five Times from Center of advancement of Post Graduate Studies and Researches in Engineering Sciences, Faculty of Engineering, Cairo University. On 16th October 2020 a report published by Stanford University, showing the "World's Top 2% Scientists List" based on Scopus Database for all fields. In the "Electrical & Electronic Engineering" field, Prof. Ahmed Soliman ranked as 36 worldwide and the first Egyptian on the list. Also in the Table of the Top 2% in all areas from Cairo University; Prof Ahmed Soliman is ranked as the first in the list.