A Reconfigurable Architecture for Secure Multimedia Delivery

Amit Pande and Joseph Zambreno
Department of Electrical and Computer Engineering
Iowa State University, Ames, IA 50011
Email: {amit, zambreno}@iastate.edu

***Abstract*—This paper introduces a reconfigurable architecture for ensuring secure and real-time video delivery through a novel parameterized construction of the Discrete Wavelet Transform (DWT). This parameterized construction promises multimedia encryption and is also well-suited to a hardware implementation due to our derivation of rational filter coefficients. We achieve an efficient and high-throughput reconfigurable hardware implementation through the use of LUT-based constant multipliers enabling run-time reconfiguration of encryption key. We also compare our prototype (using a Xilinx Virtex 4 FPGA) to several existing implementations in the research literature and show that we achieve superior performance as compared to both traditional CPU-based and custom VLSI approaches while adding features for secure multimedia delivery.**

I. Introduction

Security is a major concern in an increasingly multimedia-defined world. The recent emergence of embedded multimedia applications such as mobile-TV, video messaging, and telemedicine have increased the impact of multimedia and its security in our personal lives. For example, a significant increase in the application of distributed video surveillance technology to monitor traffic and public places has raised concerns regarding the privacy and security of the targeted subjects.

The large data volumes, interactive operations, real-time responses, and scalability features inherent to real-time multimedia delivery limit the practical application of traditional private and public-key cryptographic schemes. Furthermore, embedded multimedia systems have constraints on power consumption, available computational resources, and performance; restricting the operations of full-encryption schemes or selective-encryption schemes [1], [2].

The authors in [3] present a joint signal processing and cryptographic approach to multimedia encryption. They use index mapping and constrained shuffling to achieve confidentiality protection. This ensures that the encrypted bitstream still complies with state-of-the-art multimedia coding techniques. The scheme gives good results, however, it requires several extra computations (and hence extra hardware resources) to implementation. [4] presents a multimedia encryption scheme based on wavelet coefficients confusion. A scheme based on wavelet coefficient permutations alone is bound to be separable and weak against any cryptanalysis. In this work, we do use a wavelet coefficient permutation referred to as 'subband reorientation' which is optimized for implementation without any computational overhead. However, our cryptographic approach has more parameters that build the key space which prevents an adversary from easily cracking our scheme by parallel brute force trials in the individual sub bands.

The initial results regarding a parameterized construction of the Discrete Wavelet Transform (DWT) were presented in [5]. In this paper, we use the parameterized DWT along with the subband re-orientation property to build a large and secure multimedia encryption keyspace. Furthermore, we map this encryption architecture to reconfigurable hardware, and perform additional optimizations to obtain a zero-overhead scheme. Reconfigurable Constant Multipliers (RCM) were used in the design to obtain a LUT-based abstraction and LUT-based hardware acceleration.

The main contributions of this work can be summarized as follows:

1) This paper presents a reconfigurable hardware architecture for multimedia encryption.
2) RCMs are used to accelerate DWT operations and provide LUT-based hardware abstraction. The use of an RCM typically boosts the clock frequency by mapping complex multiplications into consecutive Look-up and addition operations. A multiplier-less hardware implementation is presented in the paper.
3) Initial investigation regarding the security features of proposed scheme has been provided in this paper.
4) Performance comparison with existing CPU-based software implementations and custom hardware implementations indicates favorably toward design of such reconfigurable architectures for secure multimedia delivery.

Our DWT-based architecture serves as a compression-cum-encryption system that provides dual features of high throughput multimedia compression and embedded multimedia security. Section II gives a brief introduction to the underlying mathematics behind the DWT operation. We derive an expression for the parameterized DWT. Section III discusses the security performance of our scheme while Section IV gives details of hardware architectures and optimizations. Section V presents the synthesis results for the DWT architecture on a Xilinx Virtex-4 FPGA, its performance comparison with traditional architectures and the speedup over a CPU-based implementation. Section VI gives a short summary of our work.

II. Parameterization of the Discrete Wavelet Transform

A. Preliminaries

The two most common DWT filters used in image compression are the Le Gall's 5/3 filter and the Daubechies 9/7 filter [6], accepted in the JPEG2000 standard. The Le Gall's filter has rational coefficients and its hardware implementation requires less resources. The Daubechies 9/7 filter has better compression performance, however, it has irrational coefficients and leads to lossy compression.

B. Parameterized DWT derivation

The Bi-orthogonal Wavelet Filter Banks are used in image compression because of their excellent image compression properties. They must satisfy the perfect reconstruction (PR) condition and are expected to have a large number of vanishing moments (VMs) for good approximation property [7]. Daubechies 9/7 filter has 4 VMs each for the analysis and synthesis low pass filters. The irrational coefficients in Daubechies' 9/7 filter limit its precision of implementation to fixed point hardware such as FPGAs and ASICs.

Liu et al. [8] discusses the derivation of the rational coefficients filter for DWT with an arbitrary number of taps. The parameterization is achieved by reducing two VMs in the filter expression to introduce a free parameter *a* in the design.

Let $H_1(z)$ and $H_2(z)$ denote the analysis and synthesis low pass filter coefficients. On introducing a free parameter α in the equations for $H_1(z)$, the corresponding value of $H_2(z)$ is obtained by solving for conditions for linear phase, PR and low pass filter [8].

$$\begin{aligned} H_1(z) &= \left(z^{-\frac{1}{2}} + z^{\frac{1}{2}}\right)^4 \times \left(\alpha + (1-\alpha)\left(z^{\frac{-1}{2}} + z^{\frac{1}{2}}\right)^2\right) \\ H_2(z) &= \left(z^{-\frac{1}{2}} + z^{\frac{1}{2}}\right)^2 \times Q(z) \end{aligned}$$

where

$$Q(z) = \sum_{n=0}^{3} q_n \times \left(z^{\frac{-1}{2}} + z^{\frac{1}{2}}\right)^{2n}$$

The values q_n are calculated by the following expression:

$$q_n = \sum_{k=0}^{n} \left(\begin{matrix} L+n-k-1 \\ L-1 \end{matrix} [2(1-\alpha)]^k \right), n = 0, \ldots, L-1$$

and

$$q_L = \frac{1}{2\alpha} \left\{ \begin{matrix} 2L-k-1 \\ L-1 \end{matrix} [2(1-\alpha)]^k + (1-2\alpha) \sum_{n=0}^{L-1} q_n \right\}$$

with $L = l_1 + l_2$.

An approximate expression for the rational representation of 9/7 filter coefficients (q_n) is obtained by simplifying the Taylor series expansion for the above expression. We get $q_0 = 1; q_1 = 5 - 2 \times \alpha; q_2 = 4 \times \alpha^2 - 14 \times \alpha + 16$, and $q_3 = 36 \times \alpha - 8 \times \alpha^2 - 60 + 32/\alpha$.

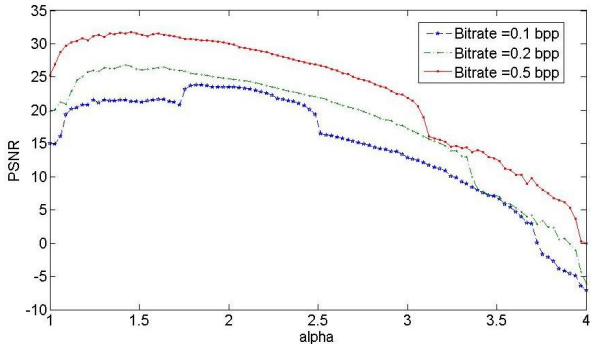

Fig. 1. Variation in PSNR with parameter α

Simplifying these expressions, we get the following expression for $H_1(z)$ and $H_2(z)$.

$$\begin{aligned} H_1(z) = (-9\alpha/64 + \alpha^2/32 + 15/64 - 1/(8\alpha))(z^4 + 1/z^4) \\ +(-\alpha^2/16 + 11\alpha/32 - 11/16 + 1/(2\alpha))(z^3 + 1/z^3) \\ +(1/8 - 1/(2\alpha))(z^2 + 1/z^2) \\ +(-11\alpha/32 + \alpha^2/16 + 15/16 - 1/(2\alpha))(z + 1/z) \\ +(9\alpha/32 - \alpha^2/16 - 7/32 + 5/(4\alpha)) \\ H_2(z) = (1/32 - \alpha/32)(z^3 + 1/z^3) + (1/8 - \alpha/16)(z^2 + \\ 1/z^2) + (7/32 + \alpha/32)(z + 1/z) + (1/4 + \alpha/8) \end{aligned}$$

The rational terms in the expressions for these filters can be implemented in hardware using shifts and adds instead of multiplication operations. This is a big saving over the original Daubechies filter in terms of hardware requirements. However, we need to perform multiplication with the free parameter α and its exponents. This filter is implemented in our DWT architecture and is explained in the following sections.

III. Multimedia Security

In this section, we give a brief summary of the security aspect of our design. One level of decomposition using DWT requires two sets of low and high pass filters (one each along the rows and columns). For a High Definition Video (with resolution 1024×768 pixels and above), we can have up to nine levels of wavelet decomposition. Thus, DWT parameterization gives us a wide range of α to choose from.

Moreover, we can implement simple transformations such as transposing the matrix and reverse-ordering of the subbands along the rows and columns of individual subbands resulting from the DWT operations. Thus, we use DWT parameterization and subband re-orientation operations to build a key space for multimedia encryption. These simple transformations can be implemented without any computational overhead by changing the memory access pattern, but such shuffling of subband information leads to significant degradations in images reconstructed with an incorrect key.

A. Building the Keyspace

Figure 1 shows the variation in PSNR of the reconstructed image at a bitrate of 0.2 bpp using the set partitioning in

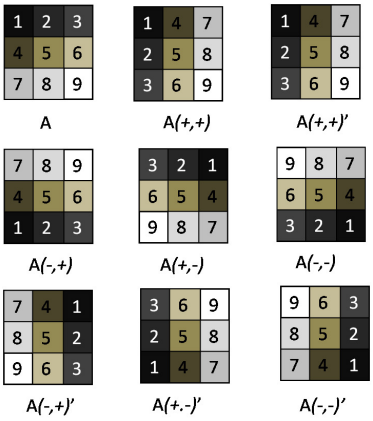

Fig. 2. Possible transpose relationships for sub bands. A is the original matrix. The eight permutations are achieved using transpose relationship ('), and reverse-ordering of the subbands ($-$ for reverse, $+$ for forward read access) along both rows and columns

hierarchical trees (SPIHT) coder with the parameter α. The variation of α beyond the range of 1 to 4 yields a poor PSNR value reflecting the poor compression property of the resultant DWT coefficients. Thus, we can divide the interval from 1 to 4 into subintervals of 0.0117 so that it takes 8 bits to encode one α parameter. For an image of 512x512 pixels, 6 levels of DWT operations are applied for the transform operation, each requiring two filters: one along the rows and the other along the columns. Thus, we have 12 DWT kernels each requiring 8 bits to represent its parameter α, giving us a 96 bit long keyspace for the DWT operation.

The parent-child coding gain in the DWT-based coders was quantified by Marcellin et al. [9]. These dependencies are generally credited for the excellent mean square error (MSE) performances of zero-tree-based compression algorithms such as SPIHT and others. The subbands were rotated by 90° with respect to the previous scale prior to zero-tree coding. The experiments indicate that, for natural images, the coding gain due to these dependencies is not considerable (typically around 0.40 dB for SPIHT-NC and 0.25 dB for SPIHT-AC).

The novel parameterization of DWT was first presented in [5]. The above discussion on rotation of subbands allows us to improve the performance of our parameterized DWT scheme by using simple transformations such as transposing the matrix, reverse-ordering of the subbands along the rows and columns. This allows us to build a larger key space without any computational overhead or any significant loss in compression.

In Figure 2, we illustrate how we can represent the same subband in eight different orientations: we have four orientations of the subband decided by the forward or reverse ordering of the matrix along rows or columns. We get four more orientations by transposing the above four, summing up to eight possible transformations for each subband. We need a 3 bit value to represent this transformation. We have 19 subbands in a 6 level DWT decomposition, each one of which can be independently transformed giving us a 57 bit key space.

TABLE I
PSNR VALUES FOR IMAGE RECONSTRUCTION WITH VARIOUS RANDOM KEYS (ENCODED WITH KEY0)

	Key0	Key1	Key2	Key3	Key4
Aerial	∞	12.36	11.17	11.67	11.77
San Francisco	∞	18.40	17.34	18.21	18.46
Brick Wall	∞	14.75	13.39	14.34	13.58
Airplane	∞	13.19	11.26	11.63	12.43

TABLE II
PSNR VALUES FOR IMAGE RECONSTRUCTION WITH SOME BITS IN KEY FLIPPED

No. bits flipped:	0	1	4	6	8
Aerial	∞	50.3	23	16.04	13.18
San Francisco	∞	36.27	30.98	22.61	21.09
Brick Wal	∞	50.27	37.5895	25.9	23.2
Airplane	∞	44.28	21.64	21.43	16.16

Thus we have a total key space of $12 \times 8 + 19 \times 3 = 153$ bits.

B. Multimedia Security

We can further increase the security feature of this scheme by introducing more parameterizations and by non-linear transformations on input key [11].

The main advantage of our lightweight encryption scheme is that, while maintaining competitive compression performance and providing security, it comes at an extremely low computational overhead. [12] uses a wavelet filter parameterization scheme to provide key dependency to a blind watermarking algorithm. Similarly, the 153 bits keyspace can be used for real-time encryption of the input frames of a video. Figure 3 shows the results of encrypting our image with a random 153 bit key. Fig 3(i) gives the original image. Fig 3(ii) gives the reconstructed image when same key was used for reconstruction. Fig 3(iii-vi) give the results for the case of reconstruction with randomly generated keys. The numerical values of PSNR values between original image and the reconstructed images are given in Table I. The low PSNR values with wrong random keys indicate poor reconstruction with random keys.

Next, we perform experiments to evaluate the performance of our encryption scheme by flipping a certain number of bits in the key randomly. We assume that the adversary only has the information of the output images (something similar to cipher-text only attacks). Any attempts to recover the correct key information from incorrect trials would use correlation between different key-trials.

Shannon's 1949 paper [13], which serves as the foundational treatment of modern cryptography calls this property as the 'confusion' property. Ideally, change in one bit of the key should change the cipher text completely.

We performed 1000 simulations where we flipped k bits of the key and performed the inverse transform with the new modified key. Results are given in Table II. It can be observed that $k > 4$ gives a low PSNR value which shows the low correlation of the reconstructed image with the input image. We can argue that change in few bits of key ($k > 4$) leads to

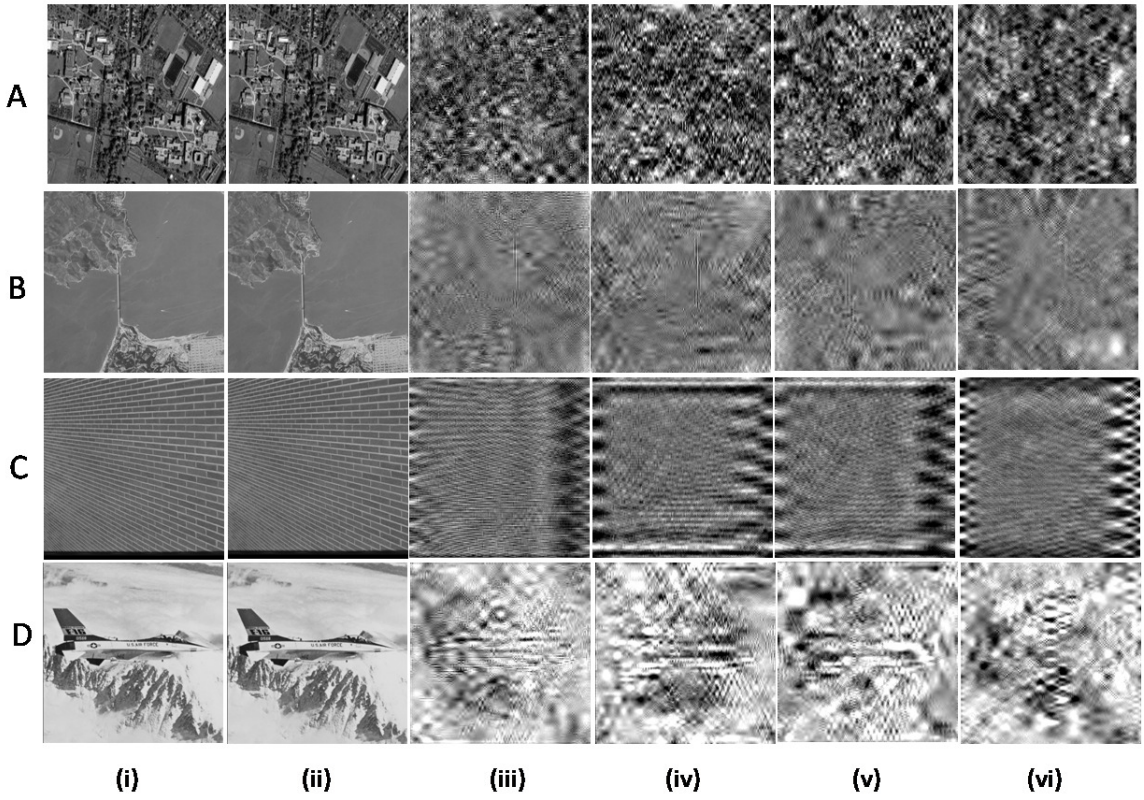

Fig. 3. Image reconstruction using DWT-based multimedia encryption scheme. A: aerial map image, B: San Francisco Golden gate aerial image, C: brick wall (texture) image and D: airplane image. (i)-Original image encrypted with key0, (ii)- Image decrypted with same key, (iii)-(vi) Image decrypted with randomly generated keys. Note: Images (B-D) are taken from the USC-SIPI Image Database [10].

poor correlation between input and output images. This level of security can suffice for the soft encryption requirements of mobile multimedia applications [14] and surveillance applications as previously mentioned.

IV. DWT Architecture and Design

Figure 4 provides an overview of our parameterized DWT architecture. The input data (one pixel input per cycle) x is pipelined for eight cycles. In this block, we perform shift and add operations to implement fractional multiplications for the DWT filter. The high and low pass filter coefficients are the final outputs of the DWT filter.

We performed several optimization steps to reduce the cost of the underlying hardware, as summarized below:

1) Division by binary coefficients (e.g. $1/64, 1/16, 1/4$) was performed using arithmetic shift operations. This eliminates the requirement for multipliers in the circuit and reduces the number of multipliers from 69 to 23.
2) Eight out of the nine inputs are passed through four adders to reduce the number of variables to five. These values (labeled as w_0, w_1, w_2, w_3 and w_4) are multiplied with a, a^2 and a^{-1} to get the necessary intermediate values which are input to shift and add logic. This optimization gives a tremendous savings in hardware. It reduces the number of adders in the design from 70 to 41 and the number of multipliers from 23 to 13.
3) The input stream was pipelined. As shown in Figure 4, our architecture takes one pixel (or channel input) as the input and outputs the low and high pass signal coefficients with a finite latency. Increasing the system latency allows us to achieve a higher clock speed (and hence higher throughput). A direct implementation of this architecture using a Virtex-XC4VLX40 FPGA resulted in a clock frequency of 60 MHz which was improved by pipelining the critical path [5].

A. Reconfigurable Constant Multiplier

The parameterized low and high pass filters were implemented using architecture given in Figure 4 using several multiplier units. The w_i, $i \in \{0, 1, 2, 3, 4\}$ values are obtained by summing the inputs for symmetric taps in the DWT implementation as shown in Figure 4. w_i is calculated as follows (where x(i-j) is the input x pipelined by j cycles):

$$w_i(k) = x(k+i) + x(k-i), i \in \{0, 1, 2, 3, 4\}$$

Then, we can represent the filter expressions as:

$$H_1(k) = \sum_{i=0}^{4} K_i(\alpha) \times w_i(k)$$

and

$$H_0(k) = \sum_{i=0}^{3} \hat{K}_i(\alpha) \times w_i(k)$$

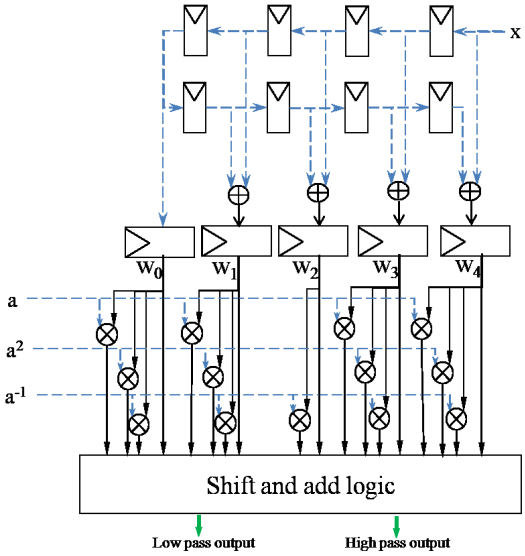

Fig. 4. Hardware architecture for the parameterized DWT

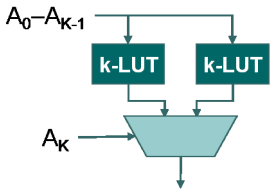

Fig. 5. Building a $(K+1)$-LUT from K-LUT

```
          110110010001   (Operand 1)
        x      AAAAAAAA   (Operand 2)
SSSSSSSSSSSSSSSSSSSS   (Product  )
```

(a)

```
          110110010001   (Operand 1)
        x      AAAAaaaa   (Operand 2)
      XXXXXXXXXXXXXXXX   (aaaa * 0001)
+YYYYYYYYYYYYYYYY        (AAAA * 1001)
SSSSSSSSSSSSSSSSSSSS   (Product  )
```

(b)

Fig. 6. Illustration of 12-bit constant multiplication with a 8-bit input.(a) The individual bits of product are obtained as output of a 8-LUT. (b) 4-LUTs are used in the implementation with the input A divided into 2 4-bit values.

Here $K_i(\alpha)$ and $\hat{K}_i(\alpha)$ are the functions of the variable α, and w_i are obtained from the pipelined input. The values of functions $K_i(\alpha)$ and $\hat{K}_i(\alpha)$ remain the same as long as we have the same α parameter. This implies that the values of these functions behave as constants and change only when we change the encryption key (and the associated parameter α). This value can thus be computed and hard-coded into the circuit. This constant multiplication can easily be mapped to a reconfigurable hardware with programmable LUTs. If the input is represented by B_1 bits and constant is represented by B_2 bits, we can use $(B_1 + B_2)$ B_2-input LUTs to get the output values of $H_1(k)$ and $H_2(k)$. Alternatively we can break down a $(B_1 \times B_2)$ bit multiplication into smaller input LUTs. Thus, the LUTs based multiplication can be reconfigured to incorporate any changes in encryption key.

We discuss the implementation of a 4×4 bit multiplier to explain the LUT mappings.

1) 4×4 Bits Multiplier using LUTs: Arbitrary hardware multipliers can be implemented using Propagate and Generate algorithm[15]. We make some interesting observations to build a direct LUT-based multiplier.

Let A and B be the two operands, both being 4 bits long. We define $P_i = A_i \oplus B_i$, and $G_i = A_i B_i$. The output bit and the sum at each stage can be represented as:

$$S_i = P_i \oplus C_i \; C_{i+1} = G_i + P_i C_i$$

On simplification, we get

$$C_1 = G_0 = A_0 B_0$$

$$C_2 = G_1 + P_1 C_1 = A_1 B_1 + (A_1 \oplus B_1) A_0 B_0 \ldots$$

$$S_1 = A_1 \oplus B_1 \oplus C_1 = A_1 \oplus B_1 \oplus A_0 B_0$$

$$S_2 = A_2 \oplus B_2 \oplus C_2 = A_2 \oplus B_2 \oplus (A_1 B_1 + (A_1 \oplus B_1) A_0 B_0) \ldots$$

We can observe that S_i is a function of inputs and is characterized uniquely by a logical expression. If one of the inputs (say B) is a constant, S_i can be represented as a logic function of bit values of A.

$$S_i = f_i(A_3, A_2, A_1, A_0)$$

The truth table of these functions $f_i(...)$ can be evaluated either by logical simplification or by exhaustive search over the input values. Thus, we can implement a 4×4 bit constant multiplication using 8 4-input LUTs or more generically, we can implement a $M \times K$ bit constant multiplication using $(M + K)$ K-input LUTs.

It has been discovered that the LUT size of 4 to 6 provides the best area-delay product for an FPGA [16]. Most commercial reconfigurable devices such as FPGAs have 4-input LUTs. We therefore discuss the mapping of an $M \times K$ bit constant multiplier into 4-LUTs in the next subsection.

2) Mapping a generic RCM into LUTs: The multiplication of two inputs A and B (M-bit variable input A, K-bit reconfigurable constant B) can be mapped to LUTs similar to 4×4 bits multiplier by obtaining a generic expression for $S_1, S_2 \ldots S_{M+K-1}$. S_i values can be represented as $f(A_{M-1}, A_{M-2}, \ldots A_1)$ and can be therefore mapped into an M-input LUT. We have $(M + K - 1)$ S_i values, requiring $(M + K - 1)$ M-input LUTs to multiply A and B.

A $(K+1)$-input LUT can be built from 2 K-input LUTs (as shown in the Figure 5). For example, we can build a 8-LUT from 2 7-LUTs which can be synthesized from $2 \times 2 = 4$ 6-LUTs. Thus, one 8-LUT can be made from $2^4 = 16$ 4-LUTs and an arbitrary M-LUT from 2^{M-4} 4-LUTs.

Figure 6 gives an example of multiplication of 8-bit number with 12-bit constant ($M = 8$, $K = 12$). Figure 6(a) depicts

TABLE III
HARDWARE UTILIZATION OF DWT ARCHITECTURE ON XILINX VIRTEX-XC4VLX40 FPGA

	Daub. 9/7	[17]	[18]	[19]	δ	θ
4-LUTs	210	-	-	-	649	5781
Multipliers	16	12	36	12	13	0
Adders	15	16	36	16	41	11
Registers	144	-	-	-	92	112
Critical Path	T_m+ $4T_a$	T_m+ $2T_a$	T_m+ $4T_a$	$4T_m$ $+8T_a$	T_m+ $5T_a$	$4T_a$ $+T_l$
Clock Freq. (MHz)	107	-	-	-	60	103
Encryption	No	No	No	No	Yes	Yes

δ- Direct mapping into Virtex-4 FPGA
θ- RCM based implementation
Note: T_m, T_a and T_l represent the time delay in multiplier, adder and look up operation respectively.

TABLE IV
HARDWARE ACCELERATION ON VIRTEX IV FPGA

Image Size	Software	Hardware	Speedup
256×256	32400μs	456 μs	71.05
512×512	91500μs	1890 μs	48.41

implementation using 8-LUTs. 20 8-LUTs or equivalently 128 4-LUTs are used in the design.

Figure 6(b) provides an alternative implementation of the same multiplication by breaking the input number into multiples of 4-bit values. 4-input LUTs are used to obtain the X and Y values which are then added together using an adder. This implementation requires 32 4-LUTs and a 20 bit adder. This design requires less LUTs but the presence of 20-bit adder may slow down the clock speed of such a design.

V. HARDWARE PERFORMANCE

We estimated the hardware performance of our architecture by synthesizing the design on a Xilinx Virtex-4 XC4VLX40 FPGA, using ModelSim SE 6.4 for simulation and Xilinx ISE 10.1 for synthesis. This design just serves as the proof of concept for our architecture. An ASIC implementation with fixed interconnects for LUTs can achieve significant improvements in clock speed and throughputs.

We achieved a clock frequency of 103 MHz for our reconfigurable architecture. The hardware requirements of our implementation are summarized and compared with other implementations in Table III. The critical path of the design is dependent on the mapping of LUTs. Thus, a hard-wired LUT implementation will give considerable improvements in clock frequency over the present value of 103 MHz. It is noteworthy that together with [5], this is the first known hardware implementation of the parameterized DWT filter (to the best of knowledge of the authors). Thus, comparison with reported architecture only indicates the trade offs involved in building a secure DWT scheme. A large number of 4-LUTs (over 5000) were used in the design.

Table IV gives speed up of our hardware implementation over Virtex IV FPGA over an optimized software implementation. The software implementation was done using Matlab R2008a on a dual-core Intel 2.0GHz processor with 2 GB RAM. The exact speed-up is highly platform dependent, but we still get an order of magnitude speed-up by FPGA implementation.

VI. CONCLUSIONS

The proposed multimedia encryption scheme gives promising results for image and video encryption while the underlying hardware architecture is developed using LUTs allowing reconfiguration and providing high throughput. The key information is embedded into the configuration bit stream of reconfigurable hardware. To the best of our knowledge, this is the first such scheme, optimized to provide high throughput multimedia delivery alongside with multimedia encryption using parameterization of compression blocks.

REFERENCES

[1] A. Pommer and A. Uhl, "Selective encryption of wavelet-packet encoded image data: efficiency and security," *ACM / Springer Multimedia Systems*, vol. 9, no. 3, pp. 279–287, 2003.

[2] W. Zeng and S. Lei, "Efficient frequency domain selective scrambling of digital video," *IEEE Trans. Multimedia*, vol. 5, no. 1, pp. 118–129, March 2003.

[3] Y. Mao and M. Wu, "A joint signal processing and cryptographic approach to multimedia encryption," *IEEE Trans. Image Processing*, vol. 15, no. 7, pp. 2061–2075, July 2006.

[4] S. Lian and Z. Wang, "Comparison of several wavelet coefficient confusion methods applied in multimedia encryption," in *Intl. Conf. Computer Networks and Mobile Computing*, Oct. 2003, pp. 372–376.

[5] A. Pande and J. Zambreno, "An Efficient Hardware Architecture for Multimedia Encryption and Authentication using Discrete Wavelet Transform," in *IEEE CS Intl. Symp. VLSI*, 2009, pp. 85–90.

[6] C. Christopoulos, A. Skodras, and T. Ebrahimi, "The JPEG2000 still image coding system: an overview," *IEEE Trans. Consumer Electronics*, vol. 46, no. 4, pp. 1103–1127, Nov 2000.

[7] M. Vetterli and J. Kovačevic, *Wavelets and subband coding*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1995.

[8] Z. Liu and N. Zheng, "Parametrization construction of biorthogonal wavelet filter banks for image coding," *Springer Signal, Image and Video Processing*, vol. 1, no. 1, pp. 63–76, 2007.

[9] M. Marcellin and A. Bilgin, "Quantifying the parent-child coding gain in zero-tree-based coders ," *IEEE Signal Processing Letters*, vol. 8, no. 3, pp. 67–69, Mar 2001.

[10] "USC SIPI Image database," *http://sipi.usc.edu/database/*.

[11] H. Kim, J. Wen, and J. Villasenor, "Secure Arithmetic Coding," *IEEE Trans. Signal Processing*, vol. 55, no. 5, pp. 2263–2272, May 2007.

[12] M. Brachtl, A. Uhl, and W. Dietl, "Key-dependency for a wavelet-based blind watermarking algorithm," in *Proc. ACM work. Multimedia and security (MM&Sec) 2004*. New York, NY, USA: ACM, 2004, pp. 175–179.

[13] C. E. Shannon, "Communication theory of secrecy systems," *Bell Systems Technical Journal*, vol. 28, pp. 656–715, 1949.

[14] D. Engel and A. Uhl, "Parameterized biorthogonal wavelet lifting for lightweight JPEG 2000 transparent encryption," in *Proc. ACM work. Multimedia and security (MM&Sec) 2005*. ACM, 2005, pp. 63–70.

[15] M. M. Mano and M. D. Ciletti, *Digital Design (4th Edition)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2006.

[16] E. Ahmed and J. Rose, "The effect of LUT and cluster size on deep-submicron FPGA performance and density," *IEEE Trans. VLSI Syst.*, vol. 12, no. 3, pp. 288–298, 2004.

[17] J. M. Jou, Y.-H. Shiau, and C.-C. Liu, "Efficient VLSI architectures for the biorthogonal wavelet transform by filter bank and lifting scheme," *IEEE Intl. Symp. Circuits and Systems, (ISCAS) 2001.* , vol. 2, pp. 529–532 vol. 2, May 2001.

[18] M. Vishwanath, R. Owens, and M. Irwin, "VLSI architectures for the Discrete Wavelet Transform," *IEEE Trans. Circuits and Systems II: Analog and Digital Signal Processing*, vol. 42, no. 5, pp. 305–316, May 1995.

[19] C. Huang, P. Tseng, and L. Chen, "Flipping structure: an efficient VLSI architecture for lifting-based discrete wavelet transform," *IEEE Trans. Signal Processing*, vol. 52, no. 4, pp. 1080–1089, Apr. 2004.