A Flexible Architecture for the Computation of Direct and Inverse Transforms in H.264/AVC Video Codecs

T. Dias, S. López, *Member*, IEEE, N. Roma, *Member*, IEEE, and L. Sousa, *Senior Member*, IEEE

Abstract — *A new high throughput and scalable architecture for unified transform coding in H.264/AVC is proposed in this paper. Such flexible structure is capable of computing all the 4×4 and 2×2 transforms for Ultra High Definition Video (UHDV) applications (4320×7680@30fps) in real-time and with low hardware cost. These significantly high performance levels were proven with the implementation of several different configurations of the proposed structure using both FPGA and ASIC 90 nm technologies. In addition, such experimental evaluation also demonstrated the high area efficiency of the proposed architecture, which in terms of Data Throughput per Unit of Area (DTUA) is at least 1.5 times more efficient than its more prominent related designs1.*

Index Terms — Video coding, H.264/AVC, Unified transform kernel, Scalable architecture.

I. INTRODUCTION

In the last few years, the electronic consumer market has experienced profound changes owing not only to the expansion of the Internet to other domains than the traditional computer area, but also to an ever increasing user demand for more, innovative, and better quality interactive multimedia services. Applications based on digital video, such as 3GPP mobile multimedia telephony, video telephony, Internet video streaming, or video surveillance, are integrated in this class of multimedia services, which nowadays must be supported up to some extent by the most recent consumer electronic products with multimedia capabilities, in order to guarantee its success. Consequently, video encoders and decoders (codecs) have become one of the fundamental building blocks of these devices. In fact, depending on the type of application, one or more codecs can even be made available by product developers in a single equipment, so that multiple standards are supported. Nonetheless, the majority of such multimedia devices typically implement a single video codec, which nowadays should be conformant to the H.264/AVC standard due to its widespread usage.

H.264/AVC [1] is currently considered to be the *de-facto* standard for modern multimedia applications based on digital video. Such achievement results not only from its high compression efficiency levels, but also from an extraordinarily high flexibility that allows it to be efficiently used in several different and distinct classes of applications [2]. However, most H.264/AVC coding tools impose a very significant cost in terms of computational complexity for both video encoding and decoding operations, which is mostly owed to its more elaborate and compute intensive algorithms. As a result, in H.264/AVC several combinations of coding profiles and levels have been defined, in order to constrain the computational complexity requirements for any given decoder. In addition, this feature also provides the flexibility required to efficiently implement H.264/AVC in distinct application domains (e.g., low bit-rate Internet streaming applications, HDTV broadcasting, and Digital Cinema). Nonetheless, there are still some fundamental operations that are common to all profiles and levels, like transform coding.

Just like in the previous video standards, transform coding is still one key component in H.264/AVC, since it provides one of the fundamental mechanisms for compressing the video data by reducing the spatial redundancies. However, to guarantee the desired coding performance levels the residual transform coding block of a H.264/AVC codec is much more complex than in former standards. In fact, such transform unit, which has to be present in the processing loops of both the encoder and the decoder, as well as in some modern still image encoders that only apply the H.264/AVC intra coding modes [3] (used mainly in digital photographic cameras and video surveillance equipment), must support several different forward and inverse transform functions and also residual blocks with multiple and smaller sizes. This poses several challenges in real-time applications, and even greater ones for multimedia systems implementing the newest extensions to the standard (e.g., the H.264/AVC SVC amendment [4], which most likely will be integrating most future real-time video codecs), due to the involved computational complexity requirements.

From the previous discussion, it can be easily concluded that efficient, scalable and unified transform coding

[1] This work was supported by the Portuguese Foundation for Science and for Technology (INESC-ID multiannual funding) through the PIDDAC Program funds and under project *HELIX: Heterogeneous Multi-Core Architecture for Biological Sequence Analysis* (PTDC/EEA-ELC/113999/2009), by the PROTEC program funds under the research grant SFRH/PROTEC/50152/2009, by the Spanish Government under project *DR.SIMON: Dynamic Reconfigurability for Scalability in Multimedia Oriented Networks* (TEC2008-06846-C02-02), and by the 7th Framework Program of the HiPEAC European Network of Excellence.

T. Dias is with the Department of Electronics, Telecommunications and Computers Engineering, ISEL/IP Lisbon, and with INESC-ID Lisbon, Rua Alves Redol 9, 1000-029 Lisbon, Portugal (e-mail: tiago.dias@inesc-id.pt).

S. López is with the Institute for Applied Microelectronics (IUMA) at the University of Las Palmas de Gran Canaria (ULPGC), Campus Universitario de Tafira, E-35017, Spain (e-mail: seblopez@iuma.ulpgc.es)

N. Roma is with the Department of Computer Science and Engineering, IST/TU Lisbon, and with INESC-ID Lisbon, Rua Alves Redol 9, 1000-029 Lisbon, Portugal (e-mail: nuno.roma@inesc-id.pt).

L. Sousa is with the Electrical and Computer Engineering Department, IST/TU Lisbon, and with INESC-ID Lisbon, Rua Alves Redol 9, 1000-029 Lisbon, Portugal (e-mail: leonel.sousa@inesc-id.pt).

architectures are required for the development of modern video codecs to be included in future electronic consumer multimedia products. On the one hand, to guarantee the realization of all transform operations in an efficient manner, not only in terms of processing speed but also in what concerns to the hardware efficiency. On the other hand, to achieve scalable and adaptable systems that can be easily configured to better comply with the performance and the hardware cost requirements of any given video application. In accordance, this paper presents a high throughput and scalable processor for unified transform coding in H.264/AVC that is able to fulfill all of the above requirements.

The rest of this paper is organized as follows. In section II, the transform coding procedure for the most commonly adopted H.264/AVC profiles is first reviewed. Then, the most relevant related works are introduced and discussed. Section III presents the proposed scalable architecture for unified transform coding in H.264/AVC, revealing its components, operation principles, and scalability properties. Experimental results considering the implementation of such processing structure using both FPGA and ASIC technologies are presented and discussed in section IV. Finally, section V concludes the paper.

II. BACKGROUND

A. Transform coding in H.264/AVC

In H.264/AVC, six different two-dimensional (2-D) transform functions are used to encode each block of residual data, namely, the 8×8 and the 4×4 forward and inverse integer Discrete Cosine Transforms (DCTs), the 4×4 and the 2×2 Hadamard transforms. The usage of all the 4×4 and 2×2 transforms is mandatory for all the H.264/AVC profile/level combinations, while the 8×8 transforms are only used in the High Profile levels. To avoid inverse transform mismatch problems, all the transform functions operate with integer resolution. Such important characteristic allows them to be implemented using only addition and shift operations, which significantly reduces the computational complexity requirements.

The 4×4 forward integer DCT is applied to all 4×4 blocks of residual data resulting from both the motion-compensated prediction and the intra-prediction stages. Likewise the conventional DCT, this integer transform is also orthogonal, separable, and presents a strong decorrelating performance. In practice, it consists of a simplified 4×4 DCT, where the scaling factor component has been transferred to the quantization stage of the encoding algorithm [1]. Hence, the forward transform contains only four different coefficient values (i.e., 1, -1, 2 and -2), in order to minimize its implementation requirements. Moreover, due to its separable nature, it can also be implemented using the usual row-column decomposition process. The same considerations also apply to the corresponding inverse transform, owing to its orthogonality property.

For macroblocks being encoded using the 16×16 intra-prediction mode, which may be employed to better exploit the redundancies in smoother areas of a picture, a 4×4 Hadamard transform is further used to transform the sixteen DC coefficients of all the luminance blocks within a macroblock. Once more, this transform can be regarded as a simplified version of the forward integer DCT, where the coefficients 2 and -2 are replaced by the coefficients 1 and -1, respectively. Hence, the 4×4 Hadamard transform retains all the properties of the 4×4 integer DCT, but it further reduces the complexity requirements of the corresponding implementation algorithm. Namely, the forward and inverse transform functions are exactly the same due to its symmetric nature, and can be implemented using only integer additions and subtractions.

Likewise the encoding of the luminance DC coefficients, a Hadamard transform is also used to encode the four DC coefficients of each of the two 2×2 chroma blocks of a macroblock. Consequently, this simpler 2×2 transform also shares the same properties offered by the 4×4 Hadamard transform.

The transform kernels for the forward (C_f) and inverse (C_i) 4×4 integer DCTs, the 4×4 Hadamard transform ($H_{4\times4}$) and the 2×2 Hadamard transform ($H_{2\times2}$) are depicted in equations (1) to (4), respectively.

$$C_f = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} \quad (1) \qquad C_i = \begin{bmatrix} 1 & 1 & 1 & \frac{1}{2} \\ 1 & \frac{1}{2} & -1 & -1 \\ 1 & -\frac{1}{2} & -1 & 1 \\ 1 & -1 & 1 & -\frac{1}{2} \end{bmatrix} \quad (2)$$

$$H_{4\times4} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \quad (3) \qquad H_{2\times2} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \quad (4)$$

B. Related Work

In the last few years, several different specialized and dedicated architectures for transform coding in H.264/AVC have been proposed in the literature [5]-[15], [19]-[20]. Such proposals mostly consist of efficient VLSI designs implementing fast and optimized algorithms for transform coding, in order to mitigate the involved computational complexity. They can be classified either as *dedicated transform kernels* or *unified transform kernels*.

Dedicated transform kernels implement a single forward or inverse transform function. Typically, fast direct 2-D transform architectures are used when high performance implementations are being considered [5]. This type of structures usually involves huge amounts of hardware resources, which are required to assure the parallel processing of all the transform coefficients. Consequently, they are often characterized by having significantly high hardware costs and power consumption values. Moreover, the typical cascaded processing design of these architectures (with long depth processing data paths), further characterizes them as being

high latency architectures. This poses significant constraints when real-time operation is required.

To overcome such constraints, some designs that have been proposed actually follow a row-column decomposition approach [6]-[8]. In this approach, a single one-dimensional (1-D) transform architecture with either a transpose memory or a transposition switch is used to compute the considered 2-D transform function. Such structures greatly improve the hardware efficiency, since the same 1-D transform processing circuit is used twice in the computation of the 2-D transform function. In addition, they also diminish the power consumption requirements and the hardware cost of this class of architectures. Nonetheless, in most cases such gains are not as high as it would be expected, owing to the usage of a transposition memory. Moreover, this circuit also introduces some delay in the processing of the 2-D transform coefficients that, again, can be critical to achieving real-time operation. On the other hand, the highly optimized implementations of both the 2-D and the 1-D transform kernels supporting this class of architectures prevents any modifications to its base structure, thus restricting its usage in the design of future video codecs.

Unified transform kernels are another class of architectures for transform coding that are capable of implementing multiple transform functions [9]-[11]. The supported transform functions can be computed either in parallel or at distinct time instants, and using transform engines with both direct 2-D architectures and 1-D structures based on the row-column approach. Recently, this class of designs has had its functionality further extended with the integration of the quantization and rescaling operations into the transform coding design [12]-[13]. For both cases, a couple of reconfigurable solutions have been recently proposed in the literature [14]-[15]. Nevertheless, not so many *unified* transform architectures have been proposed, and in fact most of these designs present significant limitations, such as: *i)* high throughput solutions targeting high resolution image formats are typically based on direct 2-D transform circuits that present huge hardware cost and power consumption requirements, thus reducing its attractiveness for low power applications (e.g., portable handheld and other battery supplied devices); *ii)* poor hardware efficiency rates of most unified transform designs, resulting in significant hardware overheads and increased power consumption values, which also reduces its attractiveness for low cost products; *iii)* none of such structures can be effectively *scaled* and very few can be *reconfigured* to meet the requirements of the target video coding application, i.e., performance, power consumption and hardware cost.

To simultaneously comply with all these requirements, a different category of architectures is now proposed based on a systolic array structure. This type of architectures has proved to provide rather efficient solutions in terms of performance, hardware efficiency, scalability and power consumption [16]. Moreover, it is also especially adequate to implement regular algorithms with high data throughput and computational rates. However, few systolic designs have been proposed to implement transform kernels for video coding, and most of them concern the 8×8 DCT adopted in previous ISO MPEG-x and ITU-T H.26x standards [17]. In fact, to the best knowledge of the authors, the proposed architecture is the first *scalable* systolic array structure for *unified* transform coding in H.264/AVC that has been proposed.

III. SCALABLE UNIFIED TRANSFORM KERNEL

Although some programmable processors might be able to fulfill real-time and High Definition (HD) requisites of some video codecs (e.g., Application Specific Instruction Set Processors (ASIPs) or current single and multi-core general purpose processors with Single Instruction Multiple Data (SIMD) extensions), there are several factors that greatly compromise its usage in the most common multimedia consumer electronic products. For portable and battery supplied devices, the hardware efficiency and power consumption requirements of such processors are considered prohibitive in most cases. For other classes of applications not imposing such constraints, the system memory bandwidth requirements and the lack of good parallel algorithm implementations are two factors that also usually prevent the usage of these processors in many products. In addition, its static architectural nature also does not allow to scale the processor performance with the application requirements, and thus to dynamically adapt to the specific characteristics of a given application or operation scenario. In this scope, systolic array processors have proved to provide rather convenient solutions to overcome all of the above mentioned problems.

The development of systolic array structures for any given algorithm can be accomplished using a simple, effective and well established methodology [16]. Such methodology defines several different techniques (e.g., projection, algorithm decomposition into subparts and scheduling), which can be differently combined to achieve viable hardware implementations providing different trade-offs between throughput, latency, hardware cost, scalability, etc. For the case of the H.264/AVC 2-D transform algorithms, which are defined over a four-dimensional index space, there are several different architectures that can be derived using this methodology, such as serial implementations based on a single Processing Element (PE), 1-D linear processing structures for iterative computations, or even fully parallel 2-D transform arrays structures. However, not so many of such structures are able to simultaneously comply with the high performance, scalability and reduced hardware cost requirements that are aimed for the proposed unified transform coding architecture.

To achieve such goals, three different techniques were considered in the design of the processing structure that is now proposed. Firstly, the base 2-D transform algorithm was *decomposed in two sub-parts*, each implementing a 1-D transform function. This allows achieving a hardware implementation based on the row-column decomposition, which significantly reduces the hardware cost of the

architecture and significantly increases the utilization rate of the involved PEs. Then, *projection* and *scheduling* techniques were applied, in order to obtain a 4×4 systolic array structure for the implementation of the 1-D transform functions. Such array significantly accelerates the realization of the transform operations by allowing the parallel and simultaneous computation of 16 different operations. Finally, the architecture of the PEs that integrate the 2-D array was designed to allow the computation of all the 4×4 and 2×2 H.264/AVC transform operations. This feature not only further improves the hardware efficiency rate of the proposed architecture, but also provides it with a unified transform capability.

A. Architecture

The architecture of the unified transform kernel that is herein proposed, and whose block diagram is depicted in Fig. 1, is composed by a control unit and three functional modules, i.e., an *array of PEs*, a *row-column transposition switch* and an *input buffer*.

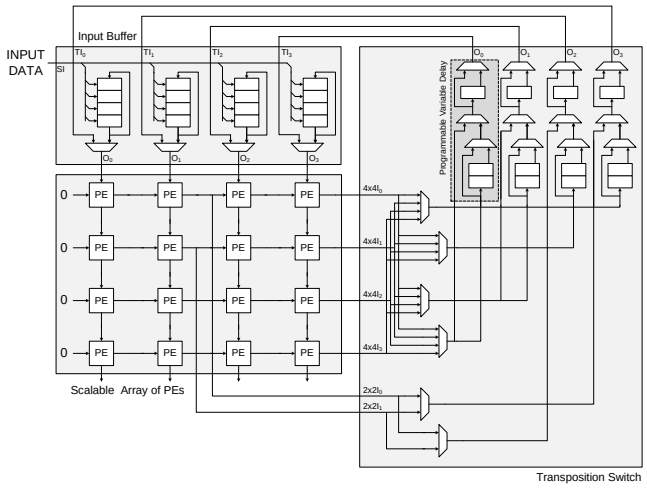


Fig. 1. Block diagram of the proposed unified transform architecture.

The PE array is used to compute the 1-D transform operations corresponding to the row-column decomposition of the transform algorithms. In its base configuration, illustrated in Fig. 1, it is composed by 16 PEs in order to allow a straightforward computation of all the 4×4 transforms using a quite simple control unit. However, it is worth noting that this 4×4 configuration of PEs also allows the simultaneous computation of two 2×2 transforms, at the cost of a small hardware overhead (the two 2×1 multiplexers depicted in the lower end of the transposition switch shown in Fig. 1) and a minimum increase of the control circuit complexity. Within the PE array, all the 16 PEs compute the same operations and share an identical architecture. They communicate with each other by using a simple and reduced signal interface, as it can be seen in Fig. 2. Moreover, to minimize the delays resulting from control operations (and thus to maximize the data processing rate), all the control logic required for the correct circuit operation is distributed and contained within each PE.

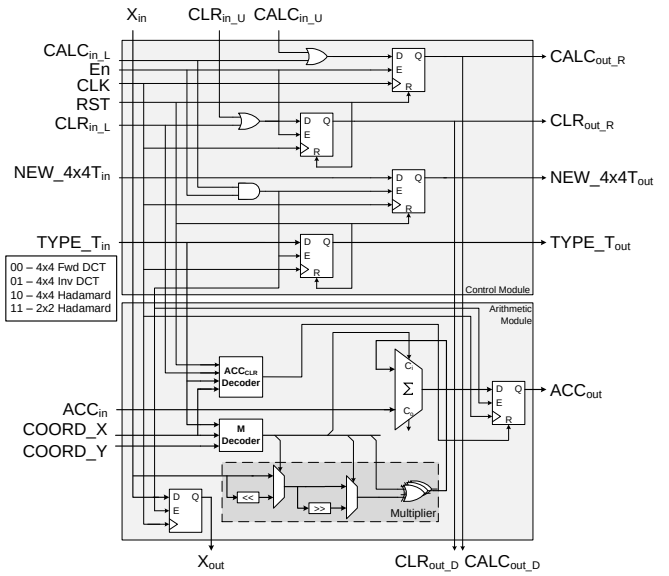


Fig. 2. Architecture of the PEs composing the array.

From Fig. 2 it can also be seen that the proposed PE is composed by two main blocks: the *arithmetic module* and the *control module*. The transform operations are computed in the *arithmetic module* using an accumulator and a specialized multiplier, which has its multiplier programmed according to the type of transform being implemented. In this scope, the data values to be processed (X_{in}) are first loaded into an internal standing-data register. Moreover, the partial value of the transform operation being computed, which is calculated by a different PE in a previous clock cycle, is placed at one of the inputs of the PE's accumulator (ACC_{in}). Then, depending on the transform to be implemented (specified by the TYPE_T signal) and on the coordinates of the coefficient under processing (identified by the COORD_X and COORD_Y signals), such partial value is updated with the result of the multiplication of the residue value (or the intermediate transform coefficient value) by a specific multiplier value (i.e., 1, -1, 2, -2, 1/2 or -1/2). This value is also stored in another internal standing-data register before being propagated to the next PE, in order to shorten the critical path of the circuit. On the other hand, the *control module* is responsible for generating the control signals required to command the operation of the PE, as well as to guarantee the correct flow of such signals to the row-column transposition switch. More specifically, such control signals allow to enable/disable the multiply and accumulate operations and to clear the accumulated values within the PEs on a clock cycle basis.

The *transposition switch* in Fig. 1 is used to implement a hardwired row-column transposition of the data processed by the 1-D transform kernel. However, unlike most of the existing transposition units, it does not include any memory circuits. In fact, thanks to an efficient data arrangement and manipulation scheme that is now proposed, it mostly consists of a set of multiplexers that allow a direct row-column transposition not only for 4×4 blocks of data (using the four 4×1 multiplexers), but also for two simultaneous 2×2 blocks of data (the top two

4×1 multiplexers and the lower two 2×1 multiplexers). This innovative design provides a significant saving of hardware resources for the implementation of the transposition switch, since it avoids the usage of the typical memory companion cells. Moreover, when compared to the data processing circuits of the proposed PEs there isn't any penalty introduced in the performance of the whole circuit, owing both to the pipelined processing nature of the PE array and the much lower propagation time of the transposition switch.

Finally, the *input buffer* in Fig. 1 is used to feed the PEs of the systolic array either with the residue data from the intra- and inter-prediction loops of a video encoder, or with the transform coefficient values in a video decoder. This unit is highly required not only to minimize the delays when accessing the external data memories where such data is stored, but also to guarantee a regular dataflow within the systolic array. Consequently, the input buffer was designed to work concurrently with the transform computation circuits, in order to optimize the overall data processing rate. Moreover, to better exploit the cache access patterns when accessing data located in the external memory, it also allows the parallel loading of a full line of residue values (or transform coefficients) from this memory. In addition, this unit is also capable of feeding the transform processing array with the previously processed row-column transposed data. Similarly to what happens with the residue values (and the transform coefficients), these data elements are sent in a serial fashion to the several columns of the array to comply with its pipelined dataflow.

B. Dataflow

The dataflow of the proposed architecture was designed to optimize the data processing rate within the PE array, using a quite simple control scheme that provides the maximum data throughput rate with four data values per clock cycle. Fig. 3 illustrates this dataflow for the processing of a complete 1-D 4×4 transform, where the three represented data-sets correspond to the processing of two consecutive 4×4 blocks: two data-sets of residue values (or transform coefficients), depicted using a solid-line, and one data-set of intermediate values for the row-column decomposition algorithm, represented using a dashed-line.

As it can be seen from Fig. 3, the transform coefficients (or pixel values) are computed in a wavefront manner within the PE array. The data to be processed is fed into the PEs in the top row of the array through the input buffers, and are then propagated in the vertical direction to the remaining PEs of the array. Conversely, the control signals for all the PEs enter the array through the PE on its top-left corner, which then propagates them to the remaining PEs of the array in both the horizontal and vertical directions. Such signals are also propagated into the transposition switch, where they are used by the control logic to select the proper data to be bypassed to each column of the PE array.

Altogether, the proposed processing scheme makes it possible to start the computation of a different transform coefficient in a new row of the array on each clock cycle, provided that the input buffers are not empty. In addition, it also allows executing another iteration of the considered 1-D transform algorithm for all the coefficients that are being processed in previous rows of PEs. Consequently, this approach offers a maximization of the data processing rate within the array, since the only PEs that will be stalled at any given time instant are the ones lacking the data to be processed. In such conditions, which correspond to a full pipelined processing, the proposed structure is able to compute a 2-D 4×4 transform in 14 clock cycles.

C. Scalability

In what concerns the offered scalability, the proposed unified transform architecture presents increased advantages when compared to other existing processing structures with similar functionality. Such property strictly concerns the array of PEs, which can be configured to support the following three setups: a 4×4 PE array, a 2×4 PE array and a 1×4 PE array.

The 4×4 PEs' setup, which consists of the base configuration described in section III.A, offers the highest performance levels. However, it also imposes the highest hardware cost. Nevertheless, it is the most suitable configuration for applications that mainly focus on throughput and performance, such as real-time systems or high resolution video codecs.

Conversely, the 1×4 PEs' setup offers the greatest savings in terms of hardware resources, due to the usage of only a single line with four columns of PEs, as it can be seen in Fig. 4 a). However, this comes at the cost of a significant reduction in terms of computation performance, since for this setup four (two) times more clock cycles are required to process a 4×4 (2×2) H.264/AVC transform function. As a result, this setup is more suitable for applications that do not require very high performance levels.

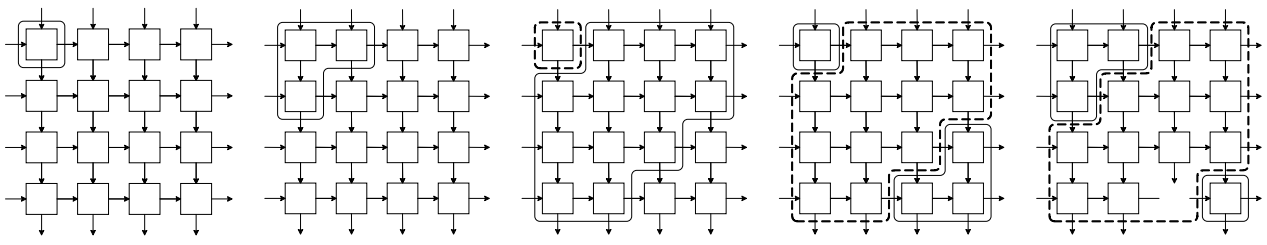
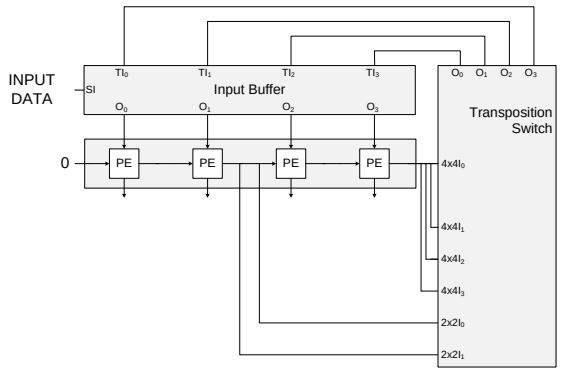
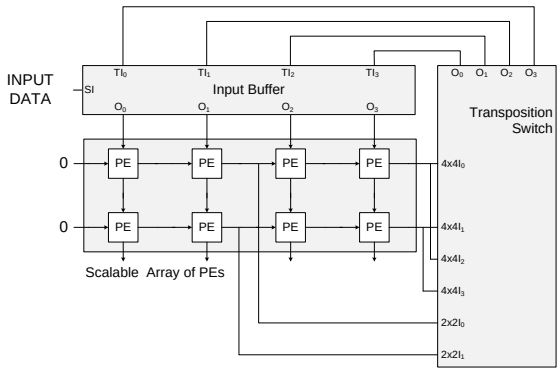


Fig. 3. Data flow in the PE array for one 4×4 data block.



a) Setup with 1×4 PEs.



b) Setup with 2×4 PEs.

Fig. 4. Alternative setups for the proposed unified transform architecture.

Finally, the 2×4 PEs' setup presented in Fig. 4 b) comes as a compromise solution between performance and hardware cost. When compared to the base setup, it offers a reduction of the required hardware resources, by only using two lines with four columns of PEs. On the other hand, it only imposes a moderate penalty in the architecture throughput. In fact, the performance is only affected when 4×4 transform functions are computed, which require twice as much clock cycles as in the base setup. The performance trade-offs provided by the three setups are summarized in Table 1, where the architecture throughput is assessed in terms of the number of samples (S), i.e., the residue or the transform coefficient values for the forward/inverse transform computations, respectively, that are processed in each clock cycle (CC).

TABLE 1
PERFORMANCE LEVELS PROVIDED BY THE DIFFERENT SETUPS OF THE PROPOSED ARCHITECTURE.

Configuration	Latency	Throughput
4×4 PEs	8 CC	4 S/CC
2×4 PEs	16 CC	2 S/CC
1×4 PEs	32 CC	1 S/CC

It is important to note that independently of the adopted setup for the array of PEs, the hardware structure of all the remaining functional modules of the proposed architecture remains unchanged. Extending the scalability property to the remaining modules of the architecture does not introduce any significantly advantage: it not only seriously increases the complexity of the control module, but it also involves the reconfiguration of very fine grained hardware structures (e.g., multiplexers and registers). Such procedure either results in a quite inefficient task when dynamic reconfiguration scenarios are considered [18], or it does not bring much advantages for implementations based on static hardware implementations (i.e., ASIC). Hence, to comply with the static nature of the internal structures of the input buffer and transposition switch modules, some programmable interconnection structures and auxiliary processing elements were embedded in the proposed architecture. The output interface of the PE array was carefully designed in order to always provide the same interface signals, through which the correct data for each of the three possible setups of the PE array is exchanged with the row-column transposition switch. Furthermore, programmable variable delay elements were also introduced in the row-column transposition switch, to allow the realization of the transposition operation for all the three PE array setups.

The need for these delay elements results from the fact that with the elimination of two or more rows of PEs it becomes impossible to do the transposition operation without additional registers, which are used to temporarily store the intermediate results that are being computed within the PE array. Fig. 1 highlights these delay elements, which mostly consist of a set of standing data registers and programmable bypass multiplexers. As a result, the performance of the architecture is not influenced by the programmable variable delay elements, since they merely extend the PE array pipeline into the transposition switch.

Lastly, it should be noted that for dynamic reconfigurable implementations of the proposed architecture, the PE array reconfiguration to implement any of the three presented setups is commanded by the main control module of the reconfigurable system. Such command can be triggered under several different scenarios, such as: *i)* a change of the video resolution or frame rate, *ii)* the energy supply capabilities of the system's power supply module or *iii)* on demand by the application itself to adjust the allocated hardware resources to the requirements of the whole system.

IV. IMPLEMENTATION AND RESULTS

The proposed unified transform architecture was implemented using FPGA and ASIC technologies to evaluate its performance gains. Such implementations were based on parameterizable IEEE-VHDL descriptions of the proposed structure, which follow a strict modular approach using independent and self-contained functional blocks to comply with the desired scalability requirements.

A. FPGA Implementation

The three configurations of the proposed processing structure (described in section III.C) were synthesized in a general purpose 90 nm FPGA device, in order to assess their individual performance levels and hardware cost. Table 2 presents the obtained results for the considered

implementations, which demonstrate the several advantages offered by the proposed multi-transform architecture.

TABLE 2
IMPLEMENTATION RESULTS OF THE PROPOSED SCALABLE ARCHITECTURE IN A GENERAL PURPOSE FPGA DEVICE.

Configuration	Slices	LUTs	Max. Freq.	GOPS
4×4 PEs	1029	1723	248 MHz	3.96
2×4 PEs	737	1258	249 MHz	1.99
1×4 PEs	455	784	247 MHz	0.99

In terms of implementation resources, the results presented in Table 2 express the very low hardware cost of the proposed architecture. Moreover, such data also demonstrate that the required hardware resources scale well with the PE array configuration. This is a very important factor for dynamically reconfigurable implementations, since despite the low hardware cost of the proposed PEs, the systolic array requires between 50% and 80% of the total hardware resources of the architecture. Consequently, the ability to choose different setups of the proposed structure enables its implementation in FPGA devices with both reduced and high amounts of hardware resources. On the other hand, the presented results also expose the very high hardware efficiency of the proposed architecture. In fact, it even requires fewer hardware resources to compute all the H.264/AVC 4×4 and 2×2 transforms than most of the existing designs that only compute a single transform function, as it is discussed in section IV.C.

Regarding to the obtained performance levels, the presented maximum allowed clock frequencies evidence the high processing rate (up to 3.96 GOPS) that is offered by the proposed architecture. Furthermore, by taking into account these results and the data presented in Table 1, it can be demonstrated that this processing structure allows to compute, in real-time, the whole set of the H.264/AVC forward and inverse 4×4 transforms for ultra high-definition video (UHDV) sequences (4320×7680@30fps).

B. ASIC Implementation

The same VHDL description of the proposed architecture was also used to implement, in ASIC, the three setups described in section III.C. Such implementations were carried out using a 90 nm standard cell CMOS technology and a synthesis procedure with distinct optimization goals from their FPGA counterparts, owing to the intrinsic static nature of ASIC implementations. As a result, no boundary block constraints were imposed to the synthesis tool, allowing it to fully and jointly optimize the design from its top level. Table 3 presents the obtained synthesis results, for which the typical operating conditions (Vdd=1.2V, T=25ºC) and the *G10k* wire load model were considered.

These results demonstrate that all the considered ASIC implementations provide significantly higher processing rates (about 1.5 times higher) than their corresponding FPGA counterparts: 6.11 GOPS, 3.04 GOPS, and 1.52 GOPS for the 4×4, 2×4, and 1×4 setups, respectively. Such performance gains, which mainly result from the higher clock frequencies used in the ASIC implementations, allow to compute, in real-time, the H.264/AVC 4×4 transforms for video sequences up to the UHDV format using the 2×4 setup. In addition, Table 3 also shows that the gate counts of the considered architecture implementations are rather reduced, and thus that the three configurations occupy small silicon implementation areas. Hence, it can be concluded that the hardware efficiency of the proposed architecture is quite high.

TABLE 3
IMPLEMENTATION RESULTS OF THE PROPOSED SCALABLE ARCHITECTURE USING A 90nm TECHNOLOGY.

Configuration	Area	Gate Count	Max. Freq.	GOPS
4×4 PEs	37.70 mm^2	4391	381 MHz	6.11
2×4 PEs	23.84 mm^2	2521	380 MHz	3.04
1×4 PEs	12.06 mm^2	1213	380 MHz	1.52

C. Discussion

In order to evaluate the performance and hardware cost benefits of the proposed architecture when compared with other alternative solutions, several different designs described in the literature were reviewed. In this subsection it is presented a comparative analysis of the proposed architecture with the most related and prominent designs [7]-[9], [12]-[13], [19]-[20]. Moreover, due to the diverse set of technologies that were considered, the distinct considerations that might have been adopted, and the different functionalities presented by each implementation, extra effort was put on this comparative analysis in order to achieve a comparison as fair as possible.

Firstly, a more comprehensive figure of merit was adopted: the Data Throughput per Unit of Area (DTUA), which is defined as the ratio of the data throughput rate (in samples per second) over the hardware cost (in terms of unit of area). In ASIC implementations the number of gates was chosen to represent the unit of area, while in FPGA implementations it was considered a slice as a unit of area. Then, since almost all of the reviewed designs implemented in ASIC adopted a 0.18 μm technology, the same CMOS technology was also used to implement the proposed architecture. However, when compared with the results obtained using the 90 nm technology, the maximum allowed clock frequencies are somewhat lower and the silicon areas are slightly higher. Conversely, to compare the different FPGA implementations there was no need to re-synthesize the design for a different device, because all the evaluated alternative structures were also implemented in FPGAs with an identical slice structure (two 4-input LUTs and two flip-flops per slice) and Configurable Logic Block (CLB) organization.

Table 4 and Table 5 present the results of this comparative analysis for the subset of designs with FPGA and ASIC implementations that were reviewed, respectively. Such data concerns only to the implementation of the transform computation module of all the architectures, excluding the 8×8 transform operations.

TABLE 4
COMPARISON WITH OTHER ARCHITECTURES BASED ON FPGA IMPLEMENTATIONS.

Design	Transform Function	Slices ($\times 10^3$)	Max. Freq. (MHz)	Latency (CC)	Throughput (S/CC)	DTUA (S/s $\times 10^3$)	Applications
[19] *	4×4 Forward	0.6	107	1	16	2867.4	UHDV
[20]	4×4 Inverse	4.2	132	64	1	31.4	HD1080p
[9]	4×4 Unified	2.1	167	8	8	636.2	UHDV
[13]	4×4 Forward	1.6	225	8	8	562.5	Digital cinema
Proposed 4×4 PEs	4×4 Unified	1.0	248	8	4	962.3	UHDV
Proposed 2×4 PEs	4×4 Unified	0.7	249	16	2	674.7	Digital cinema
Proposed 1×4 PEs	4×4 Unified	0.5	247	32	1	542.7	Digital cinema

* The reported data concerns only to the transform kernel module.

TABLE 5
COMPARISON WITH OTHER ARCHITECTURES BASED ON ASIC IMPLEMENTATIONS.

Design	Transform Function	Technology	Gates ($\times 10^3$)	Max. Freq. (MHz)	Latency (CC)	Throughput (S/CC)	DTUA (S/s $\times 10^3$)	Applications
[12]	4×4 Forward	0.35 µm	20.21	98	2	8	38.8	Digital cinema
[7]	4×4 Unified	0.18 µm	6.48	100	2	8	123.5	Digital cinema
[8]	4×4 Unified	0.18 µm	9.50	125	19	3.4	44.7	Digital cinema
[9]	4×4 Inverse	0.18 µm	33.3	231	8	8	55.5	UHDV
Proposed 4×4 PEs	4×4 Unified	0.18 µm	4.48	120	8	4	107.6	Digital cinema
Proposed 2×4 PEs	4×4 Unified	0.18 µm	2.52	122	16	2	96.7	Digital cinema
Proposed 1×4 PEs	4×4 Unified	0.18 µm	1.33	123	32	1	93.2	HD1080p

A straightforward analysis of the data presented in Table 4 and in Table 5 clearly shows that the proposed architecture for unified transform coding in H.264/AVC presents one of the highest computation rates for both FPGA and ASIC implementations. In fact, the proposed structure outperforms all the other designs, except for the structures presented in [7] and [19] that seem to achieve higher DTUAs.

Regarding to the structure proposed in [19], it is important to observe that it only implements the forward 4×4 integer-DCT and that the reported data only concerns the transform kernel module itself. Hence, it can be expected that the real DTUA of this architecture shall be much lower. In fact, by simply taking into account that this architecture requires 32 16-bit adders and 32 16-bit subtractors to implement a direct 2-D 4×4 forward integer-DCT kernel, while the one that is herein proposed requires only 16 32-bit adders to compute the same 2-D transform function, it can be claimed that the proposed architecture is much more efficient in terms of DTUA.

In what concerns the structure introduced in [7], the data presented in Table 5 reveals that it provides a slightly higher DTUA value (1.1 times higher) than the proposed architecture. Nevertheless, such structure consists of a highly optimized and dedicated direct 2-D transform kernel, which means that its higher clock frequency and throughput values result from several algorithmic simplifications and optimizations. However, contrasting to the proposed architecture, such optimizations also prevent future changes and adaptations to the architecture (e.g., extend its functionality to support the 8×8 transforms), as well as inhibit it from being scaled.

The previous observations concerning the superior efficiency of the proposed architecture in terms of DTUA are still valid when such structure is compared with other unified transform coding architectures For example, among the considered FPGA implementations, the DTUA of the proposed 2×2 setup is at least 1.1 times higher than that of [9], which is one of the most efficient unified transform architectures among the ones that were reviewed. Such results are mainly owed both to the very low hardware cost and the high operating frequency of the PEs that compose the array, leading to very high throughput rates.

Finally, the data presented in Table 4 and Table 5 also emphasizes the advantages of the proposed architecture in terms of scalability. As it can be observed, the scaling of the PE array allows to efficiently trade-off the achieved performance (in terms of throughput) for hardware savings. More specifically, in FPGA implementations the 2×4 and the 1×4 setups allow to reduce the hardware cost in about 28% and 56%, respectively, when compared with the base 4×4 PEs setup. For the ASIC implementations, the obtained savings are much higher (44% for the 2×4 setup and 70% for the 1×4 setup), due to the distinct optimization goals that are adopted in the synthesis procedure. Even though, owing to the significantly higher operating frequency values and to the highly efficient pipeline structure of the PE array, which offers the maximum achievable throughput rate (without any need for data stalls), both the 2×4 and 1×4 setups still allow real-time operation for digital cinema (4096×2048@30fps) or at least HDTV (1920×1080@30fps) applications.

V. CONCLUSIONS

In this paper, a high performance and hardware efficient architecture with multi-transform coding capabilities for the H.264/AVC standard was proposed. Such processing structure presents a highly flexible and scalable structure, capable of efficiently sustaining the computation of all the 4×4 and 2×2 forward and inverse transforms using a memory-free row-column decomposition approach. These characteristics,

combined with the usage of a novel high-throughput processing element for the computation of all the transform operations, allow to scale the design in terms of performance and hardware cost according to the requirements of the target video applications. The experimental assessment of the proposed architecture was realized by synthesizing three different setups for both a FPGA device and a standard cell library based on a 90 nm technology. The obtained performance results demonstrated that the proposed architecture can achieve real-time multi-transform processing of UHDV (4320×7680@30fps). Such results also demonstrated its superior hardware efficiency levels, leading to throughput per unit of area ratios at least 1.5 times higher than those presented by most similar existing designs.

REFERENCES

[1] T. Wiegand, G. Sullivan, G. Bjntegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 560-576, Jul. 2003.

[2] J. Ostermann, J. Bormans, P. List, D. Marpe, M. Narroschke, F. Pereira, T. Stockhammer, and T. Wedi, "Video coding with H.264/AVC: tools, performance, and complexity," *IEEE Circuits and Syst. Magazine*, vol. 4, no. 1, pp. 7-28, Jan. 2004.

[3] D. Marpe, V. George, H. L. Cyconb, and K. U. Barthel, "Performance evaluation of Motion-JPEG2000 in comparison with H.264/AVC operated in pure intra coding mode," *Proc. SPIE*, vol. 5266, pp. 129-137, Feb. 2004.

[4] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable video coding extension of the H.264/AVC standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 9, pp. 1103-1120, Sep. 2007.

[5] J. Li and M. Ahamdi, "Realizing high throughput transforms of H.264/AVC," *Proc. IEEE Int. Symp. Circuits and Syst.*, pp. 840-843, May 2008.

[6] Z. Liu, D. Wang, and T. Ikenaga, "Hardware optimizations of variable block size Hadamard transform for H.264/AVC FREXt," *Proc. 16th IEEE Int. Conf. on Image Processing*, pp. 2701-2704, Nov. 2009.

[7] K.-H. Chen, J.-I. Guo, and J.-S. Wang, "A high-performance direct 2-D transform coding IP design for MPEG-4AVC/H.264," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, no. 4, pp. 472-483, April 2006.

[8] Y-C. Chao, H-H. Tsai, Y-H. Lin, J-F. Yang, and B-D. Liu, "A Novel Design for Computation of all Transforms in H.264/AVC Decoders", *Proc. 2007 IEEE Int. Conf. Multimedia Expo*, pp.1914-1917, Jul. 2007.

[9] T.T.T. Do and T.M. Le, "High throughput area-efficient SoC-based forward/inverse integer transforms for H.264/AVC," *Proc. 2010 IEEE Int. Symp. on Circuits and Syst.*, pp. 4113-4116, June 2010.

[10] M. Nadeem, S. Wong, and G. Kuzmanov, "An efficient realization of forward integer transform in H.264/AVC intra-frame encoder," *Proc. Int. Conf. on Embedded Computer Systems: Architectures, Modeling and Simulation (SAMOS X)*, July 2010.

[11] Y.-K. Lai and Y.-F. Lai, "A reconfigurable IDCT architecture for universal video decoders," *IEEE Trans. Consumer Electro.*, vol. 56, no. 3, pp. 1872-1879, Aug. 2010.

[12] G. Pastuszak, "Transforms and Quantization in the High-Throughput H.264/AVC Encoder Based on Advanced Mode Selection," *Proc. IEEE Int. Sym. on VLSI*, pp. 203-208, Apr. 2008.

[13] R. Husemann, M. Majolo, A. Susin, V. Roesler, and J. Valdeni de Lima, "Highly efficient transforms module solution for a H.264/SVC encoder," *Proc. 2010 IEEE Computer Society Annual Symp. on VLSI*, pp. 86-91, July 2010.

[14] C. Wei, H. Hui, L. Jinmei, T. Jiarong, and M. Hao, "A high-performance reconfigurable 2-D transform architecture for H.264," *Proc. 15th IEEE Int. Conf. on Electronics, Circuits and Syst.*, pp. 606-609, 2008.

[15] C.-C. Lo, S.-T. Tsai, and M.-D. Shieh, "Reconfigurable architecture for entropy decoding and inverse transform in H.264," *IEEE Trans. Consumer Electron.*, vol. 56, no. 3, pp. 1670-1676, Aug. 2010.

[16] S. Y. Kung, *VLSI Array Processors*, Prentice Hall, 1988.

[17] C. Cheng and K.K. Parhi, "A novel systolic array structure for DCT," *IEEE Trans. on Circuits and Systems II: Express Briefs*, vol. 52, no. 7, pp. 366-369, 2005.

[18] T. Dias, S. López, N. Roma, and L. Sousa, "High throughput and scalable architecture for unified transform coding in embedded H.264/AVC video coding systems," in *Proc. Int. Conf. Embedded Computer Systems: Architectures, Modeling and Simulation (SAMOS XI)*, Jul. 2011.

[19] R. Kordasiewicz and S. Shirani, "Hardware implementation of the optimized transform and quantization blocks of H.264," *Proc. 2004 Canadian Conf. on Electrical and Computer Engineering*, vol. 2, pp. 943-946, May 2004.

[20] L. Agostini, M. Porto, J. Guntzel, R. Porto, and S. Bampi, "High throughput FPGA based architecture for H.264/AVC inverse transforms and quantization," *Proc. 49th IEEE Int. Midwest Symp. Circuits Syst.*, vol. 1, pp. 281-285, Aug. 2006.

BIOGRAPHIES

Tiago Dias received the B.Sc. and M.Sc. degrees on Electrical and Computer Engineering in 2004 and 2006 from the Technical University of Lisbon, where he is now also pursuing his PhD degree. His research activities are being performed in the Signal Processing Systems Group (SiPS) of Instituto de Engenharia de Sistemas e Computadores - R&D in Lisbon (INESC-ID), where he has been since 2004. He is also an assistant lecturer at the Electronic and Telecom. and Computer Engineering Department of the High Institute of Engineering of Lisbon (ISEL), Polytechnic Institute of Lisbon (IPL), where he lectures courses on embedded systems and computer architectures. His current research interests are specialized and reconfigurable architectures, as well as the design of multi-core embedded systems for video coding.

Sebastián López received the Electronic Engineer degree by the University of La Laguna in 2001, obtaining regional and national awards for his CV during his degree. He got his PhD degree by the University of Las Palmas de Gran Canaria in 2006, where he is actually an Assistant Professor, developing his research activities at the Integrated Systems Design Division of Institute for Applied Microelectronics (IUMA). Since 2008 he is a member of the IEEE Consumer Electronics Society as well as a member of technical program committee of the IEEE International Conference on Consumer Electronics. His research interests include video coding standards and reconfigurable architectures.

Nuno Roma received the Ph.D. degree in Electrical and Computer Engineering from Instituto Superior Técnico (IST), Technical University of Lisbon, Portugal, in 2008. He is currently an Assistant Professor with the Department of Computer Science and Engineering at IST, and a Senior Researcher of the SiPS Group of INESC-ID. His research interests include specialized computer architectures for digital signal processing (including biological sequences processing and image and video coding/transcoding), embedded systems design and compressed-domain video processing algorithms. Dr. Roma is a member of the IEEE Circuits and Systems Society and a member of ACM.

Leonel Sousa (M'01-SM'03) Leonel Sousa received the PhD degree in Electrical and Computer Engineering from Instituto Superior Técnico (IST), Technical University of Lisbon, Portugal, in 1996. He is currently a Full Professor of the Electrical and Computer Engineering Department at IST and a senior researcher at INESC-ID. He has contributed to more than 200 papers in journals and international conferences. He is an associate editor of the Eurasip Journal on Embedded Systems and served in the program committee of several conferences, for example as General Chair of ISPDC'09, Topic Co-Chair of SAMOS'10 and Topic Chair of Euro-Par'11. He is a senior member of both the IEEE and the ACM. His research interests include VLSI architectures, parallel and distributed computing and multimedia systems.