LIGHTWEIGHT HARDWARE IMPLEMENTATION OF VVC TRANSFORM BLOCK FOR ASIC DECODER

I. Farhat, W. Hamidouche, A Grill, Daniel Menard, O. Deforges

LIGHTWEIGHT HARDWARE IMPLEMENTATION OF VVC TRANSFORM BLOCK FOR ASIC DECODER

I. Farhat[†⋆], W. Hamidouche[†], A. Grill[⋆], D. Menard[†] and O. Déforges[†]

[†] Univ Rennes, INSA Rennes, CNRS, IETR - UMR 6164, Rennes, France.
[⋆] VITEC, 99 rue Pierre Semard, 92320 Chatillon, France.
E-mail: ibrahim.farhat@vitec.com, adrien.grill@vitec.com, whamidou@insa-rennes.fr

ABSTRACT

Versatile Video Coding (VVC) is the next generation video coding standard expected by the end of 2020. Compared to its predecessor, VVC introduces new coding tools to make compression more efficient at the expense of higher computational complexity. This rises a need to design an efficient and optimised implementation especially for embedded platforms with limited memory and logic resources. One of the newly introduced tools in VVC is the Multiple Transform Selection (MTS). This latter involves three Discrete Cosine Transform (DCT)/Discrete Sine Transform (DST) types with larger and rectangular transform blocks. In this paper, an efficient hardware implementation of all DCT/DST transform types and sizes is proposed. The proposed design uses 32 multipliers in a pipelined architecture which targets an ASIC platform. It consists in a multi-standard architecture that supports the transform block of recent MPEG standards including AVC, HEVC and VVC. The architecture is optimized and removes unnecessary complexities found in other proposed architectures by using regular multipliers instead of multiple constant multipliers. The synthesized results show that the proposed method which sustain a constant throughput of two pixels/cycle and constant latency for all block sizes can reach an operational frequency of 600 Mhz enabling to decode in real-time 4K videos at 48 fps.

Index Terms— VVC, Multiple Transform Selection, Hardware implementation, ASIC, cross-standard implementation.

1. INTRODUCTION

The next generation video coding standard named Versatile Video Coding (VVC) is under development by the Joint Video Experts Team (JVET), established by Motion Picture Experts Group (MPEG) and Video Coding Experts Group (VCEG). The VVC standard, expected by the end of 2020, introduces several new coding tools enabling up to 40% of coding gain beyond High Efficient Video Coding (HEVC) standard [1, 2]. One of the newly introduced tools is the Multiple Transform Selection (MTS) which involves three transform types including Discrete Cosine Transform (DCT) type II (DCT-II), DCT type VIII (DCT-VIII) and Discrete Sine Transform (DST) type VII (DST-VII) with block sizes that can reach 64 $\times$ 64 for DCT-II and 32 $\times$ 32 for DCT-VIII/DST-VII. The use of DCT/DST families gives the ability to apply separable transforms, the transformation of a block can be applied separately in horizontal and vertical directions. If the DST-VII horizontal direction is selected, VVC enables the use of DCT-VIII or DST-VII for the vertical direction, and vice versa, unlike DCT-II which is applied for both directions when it is selected by the encoder.

In this paper, two hardware implementations with two different architectures of the MTS transforms are proposed. The first uses Hcub Multiplierless Multiple Constant Multiplier (MCM) algorithm [3] and the second relies on Regular Multiplier (RM) to compute the transform. These implementations support 1-D Inverse DCT-II (IDCT-II) of orders from 4 to 64 and IDST-VII/IDCT-VIII of orders from 4 to 32. The 2-D transform can be performed using two 1-D transforms by adding an intermediate transpose memory in a folded architecture. Both modules can support the transform of the recent MPEG video coding standards including AVC, HEVC and VVC. The main consideration of these implementations is to conserve a fixed throughput of 2 pixels/cycle and a fixed system latency for all transform sizes and types. This enables accurate prediction of the process performance while facilitating chaining between transform blocks. To the best of our knowledge, this is the first implementation that includes all VVC-MTS transforms with size up to 64 for the DCT-II. Synthesis results using Synopsys Design Compiler (DC) tool, show that the RM architecture consumes 63% less gates than the MCM one. For our multi-standard decoder, we adopt the RM architecture due to the significant surface gain.

The rest of this paper is organized as follows. Section 2 presents the background of the MTS kernels and the existing works related its hardware implementations. In Section 3, the detailed hardware implementations of the MCM and RM architectures are presented. Experimental setup, test conditions and results are described in Section 4, along with a comparison with the state of art works. Finally, Section 5 concludes this paper.

2. RELATED WORKS

2.1. Background of the MTS

The HEVC standard is based on the DCT type II as the main transform function and the DST of type VII applied only for Intra blocks of size 4 $\times$ 4. In VVC, the MTS scheme can be used for coding both Intra and Inter blocks.

The basis functions of the three transforms considered in the MTS module are expressed by Equations (1), (2) and (3) for DCT-II C_2, DST-VII S_7 and DCT-VIII C_8, respectively. MTS extends the use of the DST-VII/DCT-VIII for blocks of sizes 8×8, 16×16 and 32×32 including all possible asymmetric blocks, and also considers the 64×64 block size for the DCT-II.

$$C_{2\,i,j} = w_i \sqrt{\frac{2}{N}} \cos\left(\pi i \frac{(2j+1)}{2N}\right), \tag{1}$$

$$\text{with } w_i = \begin{cases} \sqrt{\frac{1}{2}} & i = 0 \\ 1 & i \in \{1, \ldots, N-1\} \end{cases}.$$

$$S_{7\,i,j} = \sqrt{\frac{4}{2N+1}} \sin\left(\pi \frac{(2i+1)(j+1)}{2N+1}\right). \tag{2}$$

$$C_{8\,i,j} = \sqrt{\frac{4}{2N+1}} \cos\left(\pi \frac{(2i+1)(2j+1)}{4N+2}\right). \quad (3)$$

The 2D forward transform of an input block X of size $M \times N$ is computed by

$$Y = B_V \cdot X \cdot B_H^T \quad (4)$$

where B_H^T is a matrix of size $N \times N$ of horizontal transform coefficients, B_V is a matrix of size $M \times M$ of vertical transform coefficients. The inverse 2D transform is computed by Equation (5)

$$\hat{X} = B_V^T \cdot \tilde{Y} \cdot B_H \quad (5)$$

Finally, to further decrease the computational complexity of the transform module, high frequency transform coefficients are zeroed out for the transform blocks of sizes equal to 64 for DCT-II and 32 for transform types DST-VII/DCT-VIII. Therefore, only lower-frequency coefficients are retained.

2.2. Existing Hardware Implementations

Several DCT-II hardware implementations have been proposed in the literature, as it is the main transform used in the previous video coding standards. Shen *et al.* [4] have presented a unified 4/8/16 and 32-point DCT-II targeting ASIC platform. They used MCM for sizes 4×4 and 8×8 and RM for 16×16 and 32×32 transform blocks. Moreover, the proposed architecture is a multi-standard supporting five video standards including AVC, HEVC, AVS, VC1 and MPEG-2/4 with a fixed throughput of 4 pixels/cycle.

Recently, several works on hardware implementation of the VVC transform have been published. Kammoun *et al.* [5] proposed a 1-D hardware implementation of the Adaptive Multiple Transform (AMT) including five transform types DCT-II, DST-I, DST-V, DST-VII and DCT-VIII for only size 4×4 using adders and shifts instead of regular multipliers. Although this work presents a hardware implementation for all transform types (including the MTS types), it only supports 4×4 block sizes. Mert *et al.* [6] proposed another hardware implementation supporting also all transform types for sizes 4×4 and 8×8. This solution investigated two hardware methods with a fixed 8 pixels/cycle throughput. The first one uses separate data paths and the second one considers re-configurable data paths for all 1-D transforms. This solution is limited to block size of 8×8, knowing that the transform of large block size (16×16, 32×32 and 64×64) is much more complex and requires more resources.

Garrido *et al.* [7] have proposed a pipelined 1-D hardware implementation of the AMT of size 4×4, 8×8 ,16×16 and 32×32. It includes the five transform types within the JEM software [8]. The proposed design has a fixed throughput of 4 pixels/cycle on a fully pipelined architecture. The design relies on two 1D implementations connected with a transpose memory. Garrido *et al.* in [9] extended their work to support a 2D transform. The synthesis results targeting multiple state of the art Field-Programmable Gate Array (FPGAs) platforms show that the design can support in the best scenario 4K video resolution at 23 frames per second. Although the work in [7] supports all VVC transform types, it does not support blocks of size 64×64 which significantly increases the complexity. Kammoun *et al.* [10] proposed another hardware implementation for the same transform types and sizes. It implements the transformation using IP cores multiplier taking advantage of DSPs of the FPGA device. The work in [10] proposes a 2-D hardware implementation. Although it includes 2-D implementation, blocks of sizes 64×64 are also not supported. Authors in [11] proposed a 2D hardware architecture for DCT-VIII and DST-VII of VVC. They implement all DST-VII DCT-VIII transform sizes using adders and shifts. Although it is the first architecture that includes asymmetric blocks, it does not include the DCT-II.

This paper proposes a comparison of two 1-D hardware architectures for 4/8/16/32/64 1-D MTS transform cores. The first architecture uses adders and shifts for all types and sizes to perform a 1-D transform operation, while the second one relies on 32 regular multipliers to perform the required multiplications. Read-Only Memory (ROM) is used to store coefficients for the RM architecture. Up the best of our knowledge, this is the first implementation of a VVC MTS architecture supporting all transform types and sizes.

3. PROPOSED HARDWARE IMPLEMENTATIONS

In this section, we present a brief description of the two proposed hardware designs for the 1-D MTS transforms. For both architectures, the constraint is to sustain a fixed throughput of 2 pixels/cycle and a fixed latency regardless of the block size. This choice enables the transformation to process a 1-D horizontal or vertical 4, 8, 16, 32 and 64-point sizes in 8, 32, 128, 512 and 2048 clock cycles, respectively. Moreover, we introduced a delay line at the output to provide a fully pipelined structure. The size of the delay line has been determined according to the throughput and the largest transform block latency. This static latency enables a smooth chaining between processing engines of the decoder.

The MTS processor interfaces for both designs are summarized in table 1. A positive pulse in $input\_enable$ launches the transform process, with the size, type and transform direction are defined in the tr_size, tr_type and tr_dir input signals, respectively. Following the $input\_enable$ the $data\_in$ signal will carry two N_{bi} coefficients at the next clock cycle. After the computation, outputs can be read from the $data\_out\_inter$ or $data\_out\_final$ signals, depending on the transform direction, at a speed of two N_{bi} pixels each clock cycle. For both implementations that integrate data loading and pipeline stages, the design starts generating the result of 1-D row/columns DCT/DST after a fixed system latency. It then continues generating the outputs every clock cycle without any stalls.

Table 1. MTS design interface

Signal	I/O	#BITS	Description
clk	I	1	System Clock
rst_n	I	1	System reset, active low
$input\_enable$	I	1	Activation pulse to start
avc_vvc	I	1	Video standard: 0, avc; 1, hevc or vvc
tr_type	I	2	Transform type: 0, DCT-II; 1, DCT-VIII; 2,DST-VII
tr_size	I	3	Transform size: 0:4-pnt; 1:8-pnt; 2:16-pnt; 3:32-pnt;4:64-pnt
tr_dir	I	1	Transform direction : 0:Horizontal; 1: Vertical
$data\_in$	I	$2\times N_{bi}$	Input data
$data\_enable$	O	1	Activation pulse to indicate end of N-point
$data\_out\_inter$	O	$2\times N_{bi}$	Intermediate output data
$data\_out\_fin$	O	$2\times N_{bo}$	Final result

3.1. Multiplierless Architecture

The MCM architecture is a design in which multipliers are replaced by adders and shifts. It is widely used and for small block sizes, it has been proven to be more efficient than a regular multiplier architecture. However, VVC introduces new transform types with larger 64×64 block size. These new transforms increase the number of coefficients and the number of possible multiplications. For that RM architecture is also reconsidered in this paper.

The MCM architecture contains five separable modules, one module for the N-point IDCT-II in a unified architecture and four independent modules for 4-point, 8-point, 16-point and 32-point IDCT-VIII/IDST-VII. We take advantage of the butterfly decomposition to regroup all IDCT-II sizes (4-point, 8-point, 16-point, 32-point, 64-point) into one unified architecture. For the IDCT-VIII and IDST-VII which do not have any specific decomposition, we created one module for each block size including 4-point, 8-point, 16-point and 32-point. The relationship between IDCT-VIII and IDST-VII enables to use only the IDST-VII kernel to compute both transformations by adding two stages for preprocessing and postprocessing as shown in Figure 1. Equation (6) computes the IDCT-VIII C_8^T from the IDST-VII S_7^T using pre-processing Λ and post-processing Γ matrices.

$$C_8^T = \Lambda \cdot S_7^T \cdot \Gamma, \tag{6}$$

where Λ and Γ are permutation and sign changes matrices computed by Equations (7) and (8), respectively.

$$\Lambda_{i,j} = \begin{cases} 1, & if\, j = N-1-i, \\ 0, & \text{otherwise}, \end{cases} \tag{7}$$

$$\Gamma_{i,j} = \begin{cases} (-1)^i, & if\, j = i, \\ 0, & \text{otherwise}, \end{cases} \tag{8}$$

with $i, j \in \{0, 1, \ldots, N-1\}$ and $N \in \{4, 8, 16, 32\}$.

Therefore, the IDCT-VIII is computed only by inverting the input order using a pre-processing stage and assigning the appropriate outputs signs using a post-processing stage.

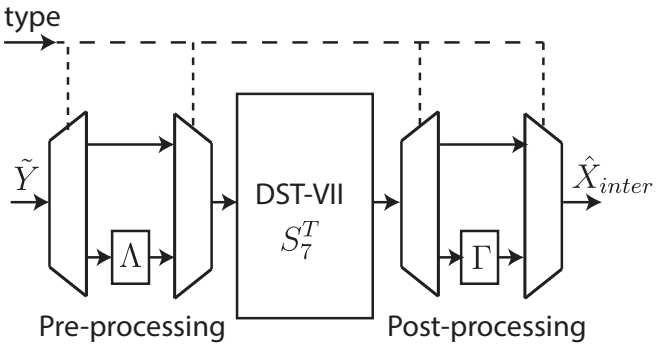

Fig. 1. N-point IDCT-VIII/IDST-VII hardware design using IDST-VII as kernel transformation, pre-processing and post-processing blocks are used when IDCT-VIII type is active.

Several IDCT-II hardware implementations have been proposed in the literature as it is the classic transform used in the previous video coding standards. However, none of the existing implementations supports the size of 64-point IDCT-II, until now this is the first proposed solution based on a constant multiplier architecture that reaches the size 64. Figure 2 shows the unified N-point hardware design for the IDCT-II. The proposed implementation relies on the state-of-the-art butterfly architectures and includes the IDCT-II of order 64.

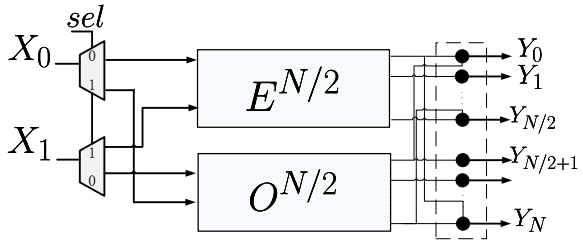

Fig. 2. General structure of the order-64 inverse DCT-II. Blocks $O^{N/2}$ corresponds to odd decomposition and $E^{N/2}$ to the even decomposition of $[C_2^N]^T$. X_0 and X_1 denote the input samples, while Y_i denote the outputs.

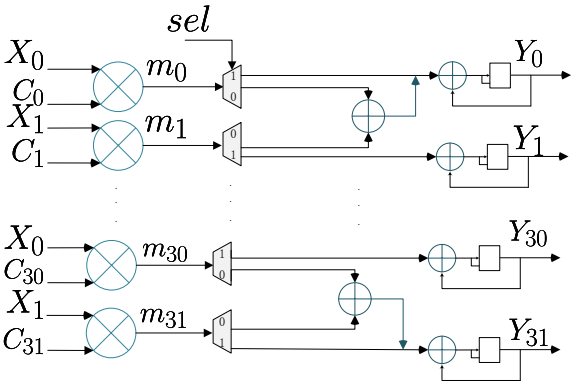

Fig. 3. RM architecture, X_0 and X_1 are the input samples, C_i is the transform coefficient, m_i is a multiplier and Y_i represents the output

Table 2. Synthesis results for both ASIC and FPGA platforms

		MCM arch.	RM arch.
ASIC	Frequency (Mhz)	600	600
	Combinational area	221659	59588
	Noncombinational area	27860	32413
	Buf/Inv area	16796	4848
	Total area	266315	96849
FPGA	Frequency (Mhz)	139	165
	ALMs (/427200)	51182 (12%)	9723 (2%)
	Registers	16924	14368
	DSP blocks(/1518)	0 (0%)	32 (2%)

3.2. Regular Multipliers Architecture

In this section, the RM architecture is investigated. The architecture of 4/8/16/32-point 1D IDCT-II/VIII, IDST-VII and 64-point 1D IDCT-II uses 32 shared multipliers. Thirty-two is the maximum number of multiplications needed to get an output rate of 2 pixels/cycle. This number is bounded by the odd decomposition of the

Table 3. Comparison of different hardware transform designs

Solutions	Fan *et al.* [11]	Garrido *et al.* [9]	Mert *et al.* [6]	Kammoun *et al.* [10]	Proposed RM architecture	
Technology	ASIC 65 nm	20 nm ME	ASIC 90 nm	ME 20 nm FPGA	ASIC 28 nm	ME FPGA
Gates/ALMs	496400	1312	417000	133017	96849	9723
Registers	–	3624	–	–	–	14368
DSPs	–	32	–	1561	–	32
Frequency (Mhz)	250	458.72	160	147	600	165
Throughput (fps)	–	3840 × 2160p23	7680 × 4320p39	1920 × 1080p50	3840×2160p30	1920×1080p50
Memory	–	41 × 21 Kbits	–	–	–	
Transform size	4, 8, 16, 32	4, 8, 16, 32	4, 8	4, 8, 16, 32	4, 8, 16, 32, 64	
Transform type	DCT-II/VIII, DST-VII	DCT-II/VIII, DST-VII	DCT-II/VIII, DST-VII	DCT-II/V/VIII, DST-I/VII	DCT-II/VIII, DST-VII	

64×64 IDCT-II transform matrix and the 32×32 IDST-VII/IDCT-VIII transform matrices. For large block sizes and by taking advantage of the zeroing, we can process one pixel/cycle at the input to get a 2 pixel/cycle at the output. In the case of 64-point IDCT-II, the size of the input vector is 32 and the output vector size is 64. For the 32-point IDCT-VIII/IDST-VII, the size of the input and output vectors are 16 and 32, respectively. In both cases, the output vector is twice the size of the input vector, and thanks to this, we can lower the input rate to one pixel per cycle.

Figure 3 shows the architecture of the hardware module using 32 RMs referred to m_i. X_0 and X_1 are the input samples. Using the zeroing for large block sizes, X_0 and X_1 interfaces will carry the same input sample and sel signal is disabled, otherwise they carry two different samples and sel is enabled. The input samples are then multiplied by the corresponding transform coefficients C_i ($i \in 1..N$). C represents a line from the transform matrix. Each X_i is multiplied by its corresponding C_i coefficient. The result is then accumulated at the output vector Y_i using the adders and the feedback lines as shown in Figure 3.

The transform coefficients are stored in a ROM. The total memory size is 17408-bits which corresponds to 68 columns of 256 bit-depth (68×256). The ROM stores the coefficients of the 64-point IDCT-II, 32-point, 16-point, 8-point and 4-point IDST-VII matrix coefficients. The 64-point IDCT-II is decomposed using its butterfly structure, and the resultant sub matrices are stored. In fact, one sub matrix is replicated to respect the output rate. The relationship between IDCT-VIII and IDST-VII enables us to compute both transforms using one kernel. Thus, we store only the IDST-VII transform matrices.

4. EXPERIMENTAL RESULTS

VHDL hardware description language is used to implement both proposed designs. A state-of-the-art logic simulator [12] is used to test the functionality of the 1D transform module. The test strategy is as follows. First a set of 10^5 pseudo-random input vectors have been generated and used as test patterns. Second, a software implementation of the inverse transforms has been developed, based on the transform procedures used in VTM 6.0 [1]. Using self-check techniques, the bit accurate test-bench compares the simulation results with those obtained using the reference software implementation.

The proposed design supports three different video standards including AVC/H.264, HEVC/H.265 and the emerging VVC/H.266 standard. The MCM 1-D architecture MTS core works at 600Mhz with 249K cell area, while the RM architecture operates at 600Mhz with 93K cell area and 17408 bits of ROM used to store transforms coefficients.

The 1D-MTS module has been Synthesised by DC with Taiwan Semiconductor Manufacturing Company (TSMC) 28nm standard cell library targeting ASIC platform. Table 2 gives the synthesis results for both architectures on ASIC and Arria 10 FPGA (ME) platforms. It shows that the RM architecture consumes 63% less gates than the MCM one.

The result given in Table 2 for FPGA platforms shows that the RM architecture consumes 80% less ALMs, 15% less registers compared to the MCM-based one. On the other hand, the total number of used DSP blocks is 32 which is 2% of the FPGA total DSP blocks. These implementations could be further optimized for FPGAs to enhance the system frequency. However, it is estimated that the RM architecture will always give better performance.

A fair comparison with other studies in the literature is quite difficult. Most studies focus on earlier version of VVC, and there are few ASIC-based designs for the VVC MTS. For comparison, Table 3 lists the key performance of state-of-the-art ASIC and FPGA-based works, including the VVC-MTS related works [11, 9, 6, 10]. Gate count is the logical calculation part and it can be seen from Table 3 that compared with implementations of Fan *et al.* [11] and Mert *et al.* [6], our solution has obvious advantages. We present a unified transform architecture that can realize IDCT-II/IDST-VII/IDCT-VIII for transform unit of order 4,8,16,32 and 64 with a fixed throughput of 2 pixels per cycle. Practically, up to 80.5% of area can be reduced compared to [11] and up to 76.7% compared to [6]. In term of ALMs, we provide 92.7% reduction compared to implementation proposed by Kammoun *et al.* in [10]. However, currently, no implementation have been found for the IDCT-II of order 64. Although, [11] and [10] supports 2D for all transform types, the transform could be achieved only up to the 32×32 block size.

5. CONCLUSION

In this paper, two hardware implementations of 1D VVC-MTS module of are proposed. The two architectures are able to implement the VVC inverse transforms including block sizes from 4×4 to 64×64. Up the best of our knowledge, this is the first implementation that supports all VVC-MTS sizes. For our VVC/HEVC/AVC ASIC decoder, we adopted the RM architecture since it enables a significant area saving.

In the future, we aim to extend this architecture to support two dimension transform and include the Low-Frequency Non Separable Transform LFNST and quantization block within a unified module.

6. REFERENCES

[1] F. Bossen, X. Li, A. Norkin, K, and Shring, "Jvet-o0003," July. 2019, JVET AHG report: Test model software development (AHG3.

[2] N. Sidaty, W. Hamidouche, P. Philippe, J. Fournier, and O. Dforges, "Compression Performance of the Versatile Video Coding: HD and UHD Visual Quality Monitoring," November 2019, Picture Coding Symposium (PCS).

[3] M. Pschel Y. Voronenko, "Multiplierless constant multiple multiplication," May. 2007, ACM Trans on Algorithms, vol. 3, no. 2.

[4] S. Shen, W. Shen, Y. Fan, and X. Zeng, "A Unified 4/8/16/32-Point Integer IDCT Architecture for Multiple Video Coding Standards," in *2012 IEEE International Conference on Multimedia and Expo*, July 2012, pp. 788–793.

[5] A. Kammoun, S. Ben Jdidia, F. Belghith, W. Hamidouche, J. F. Nezan, and N. Masmoudi, "An optimized hardware implementation of 4-point adaptive multiple transform design for post-HEVC," in *2018 4th International Conference on Advanced Technologies for Signal and Image Processing (ATSIP)*, March 2018, pp. 1–6.

[6] A.Can. Mert, E. Kalali, and I. Hamzaoglu, "High Performance 2D Transform Hardware for Future Video Coding," *IEEE Transactions on Consumer Electronics*, vol. 62, no. 2, May 2017.

[7] M.J. Garrido, F. Pescador, M. Chavarrias, P.J. Lobo, and C. Sanz, "A High Performance FPGA-based Architecture for Future Video Coding Adaptive Multiple Core Transform," *IEEE Transactions on Consumer Electronics*, March 2018.

[8] E. Alshina, A. Alshin, K. Choi, and M. Park, "Performance of JEM 1 tools analysis," in *Document JVET-B0022 3rd 2nd JVET Meeting: San Diego, CA, USA*, February 2016.

[9] M. J. Garrido, F. Pescador, M. Chavarras, P. J. Lobo, and C. Sanz, "A 2-d multiple transform processor for the versatile video coding standard," *IEEE Transactions on Consumer Electronics*, vol. 65, no. 3, pp. 274–283, Aug 2019.

[10] A. Kammoun, W. Hamidouche, F. Belghith, J. Nezan, and N. Masmoudi, "Hardware Design and Implementation of Adaptive Multiple Transforms for the Versatile Video Coding Standard," *IEEE Transactions on Consumer Electronics*, vol. 64, no. 4, October 2018.

[11] Y. Fan, Y. Zeng, H. Sun, J. Katto, and X. Zeng, "A pipelined 2d transform architecture supporting mixed block sizes for the vvc standard," *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 1–1, 2019.

[12] "RivieraPro-Aldec-Functional-Verification-Tool," [Online]:https://www.aldec.com/en/products/ functional_verification/riviera-pro.