AN EFFICIENT SUB-SAMPLE INTERPOLATOR HARDWARE FOR VP9-10 STANDARDS

Guilherme Paim[1], Wagner Penny[1], Jones Goebel[1], Vladimir Afonso[1-2],
Altamiro Susin[2], Marcelo Porto[1], Bruno Zatt[1], Luciano Agostini[1]

[1]Federal University of Pelotas (UFPel) – PPGC – Group of Architectures and Integrated Circuits(GACI)
[2]Federal University of Rio Grande do Sul (UFRGS) – Graduate Program in Microelectronics (PGMicro)

ABSTRACT

This paper presents a hardware design for the sub-sample interpolator used in FME (Fractional Motion Estimation) and MC (Motion Compensation) stages according to the VP9 and VP10 video coding standards. The proposed architecture is able to save hardware resources through an optimized filter organization whereas reaching high throughput and low-power dissipation. The hardware design was described in Verilog and synthesized for ASIC technology. The synthesis results were generated for 45nm Nangate standard cells and demonstrate that the developed architecture is able to process 2160p@60fps videos with a power dissipation of 2.34mW focusing on a VP9-10 decoder.

Index Terms**—**Hardware Design, Fractional Motion Estimation, Motion Compensation, VP9, VP10.

1. INTRODUCTION

In late 2011, Google started the development of a standard with focus on high-resolution digital videos, the VP9 [1][2], in parallel with the HEVC (High Efficiency Video Coding) development. The *profile 0* of the VP9 was finalized in June 2013. The main motivation was based on the fact that the HEVC is not a royalty-free video coding standard. This strategy was adopted by Google because the YouTube has one billion of users, with four billion of views per day on average, moving $4 billion per year (data from Feb. 2015) [3]. Still, YouTube has no profit [3] and the HEVC upgrade would bring financial losses to Google. The commercial use of the HEVC started charging percentages in broadcast on the income of each HEVC-coded video [4]. This royalty rule was not charged by MPEG LA (H.264/AVC intellectual property rights manager) considering broadcast.

Google recently announced the VP10 standard and launched its first reference software version. Initially, VP10 presented a set of new tools able to provide a coding gain of ten percent in compression efficiency over the VP9 [5]. However, the high-complexity 1/8-pel interpolation algorithm used in VP9 remains unchanged in VP10.

Many devices, as smartphones, tablets and camcorders, which have support to high-definition digital videos, are designed to be battery powered, which requires energy-aware hardware design. Furthermore, these devices require high throughput to obtain real-time processing of high-definition videos, which makes this task even harder. Based on that, the development of dedicated hardware architectures for video coders/decoders is mandatory. Thus, it is possible to obtain both energy efficiency and real-time processing of high-definition videos when these different modules are integrated in a System on a Chip. The challenges associated to such systems realization drives the interest from industry [6]-[8] and academia [9]-[16]. Currently, Ittiam [6], NVidia [7] and VeriSilicon [8] have announced interest to develop hardware support VP9.

The ME (Motion Estimation) is the most complex module in the current video coding standards, corresponding to 62-94% of the total encoding time in the HEVC [9]. This percentage, about 39% is due the FME (Fractional Motion Estimation) [10], and a big part of this complexity is related to the sub-sample interpolator used to generate the fractional samples. In the same way, the Motion Compensation (MC), presented in coders and decoders, also uses a sub-sample interpolator to generate luminance (luma) and chrominance (chroma) samples for those blocks coded with FME. Considering the decoder, the MC is responsible by more than 50% of the total encoding time in current standards like H.264/AVC [11].

This work presents an efficient hardware design for the sub-sample interpolator of the VP9 and VP10 standards. To the best of our knowledge, there is no other published work targeting the hardware design for the VP9-10 interpolation process. The proposed architecture is capable to interpolate samples for both FME and MC processes. However, the hardware design presented in this paper was developed to reach a throughput for real-time processing for Ultra High Definition (UHD) 2160p@60fps (3840x2160 pixels) videos with low-energy consumption for the VP9-10 decoder.

2. BACKGROUND AND RELATED WORKS

The VP9-10 standards support the use of 4x4, 4x8, 8x4, 8x8, 8x16, 16x8, 16x16, 16x32, 32x16, 32x32, 32x64, 64x32, and 64x64 block sizes [1][5]. The ME and MC are the most complex modules of the VP9-10 coders and decoders, respectively. A great part of this complexity is due to the sub-sample interpolation. In the ME, the interpolation is responsible to generate the samples at fractional positions for the block matching process. In the MC, the interpolation is used to generate the luminance and chrominance samples to reconstruct the blocks coded by the FME.

The interpolation process used on the VP9-10 standard is based on FIR (Finite Impulse Response) and bi-linear filters. The maximum precision in the FME is 1/8, so, the MC process can operate with precision of 1/8 and 1/16-pixel, for luminance and chrominance samples, respectively. This precision is higher when compared with other current video coding standards as H.264/AVC and HEVC, both with 1/4 precision. The higher prediction precision used in the VP9-10 increases the compression and image quality, however, also increases the computational-effort, which makes the real-time processing of high-definition videos a big challenge.

Three sets of 8-tap FIR filters, plus a bilinear set, can be selected at the frame or superblock levels in the VP9-10 standards. These sets of filters are: a) 8-tap Regular, which consists of a Lagrange-based interpolation filter; b) 8-tap Sharp, which consists on a DCT-based interpolation filter with sharper response to be used in sharper edges; c) 8-tap Smooth, which consists of a smoothing filter designed with a Hamming Window; d) bilinear, which is used in fast encode/decode scenarios. The sets of filters, as well as the coefficients used for each one, were summarized in Table I. It is important to notice that the coefficients used in the calculation of 5/8, 6/8, and 7/8 luma are similar to 1/8, 2/8, and 3/8 luma, but with rotated indexes. Similarly, the coefficients used in the calculation of 9/16 to 15/16 chroma filters are also the same used in 1/16 to 7/16 chroma filters, but with rotated coefficients order. Due lack of space the rotated filters are omitted in Table I.

There are important similarities between the four filtering processes defined by the VP9-10 standards that may be useful for an efficient hardware design. Thus, the three sets of 8-tap FIR filters and the bi-linear filters were factored, maintaining mathematical equivalence, in order to replace the constant multiplications by mux-shift-adders and sharing common sub-expressions. Thus, the hardware can be configured to operate as any type of filter according to a control signal. Other two important aspects in terms of hardware design are: (a) temporary buffers are needed to store interpolated samples to feedback the filters; and (b) samples beyond the current-block borders (from neighboring blocks) are required in the interpolation due to the width of the FIR filters. As both aspects are present in the different sets of the VP9-10 interpolation filters, the unified VP9-10 solution avoids the replication of hardware by sharing buffers and memory access.

Several works in the current literature propose hardware designs for the sample interpolator according different standards, as HEVC and H.264/AVC. Thus, some prominent works that present hardware solutions for the sub-sample interpolator targeting current video standards were used for comparisons.

Zatt [11] proposes an architecture for the sample interpolator targeting the MC of the H.264/AVC standard. The architecture proposed in [11] is unable to process UHD videos in real time. Liu [12] proposes a FME architecture for the H.264/AVC encoder able to process 2160p@30fps videos but it does not present a power-dissipation analysis. Lian [13] proposes an interpolation-filter algorithm for the HEVC FME. The developed architecture is able to process 2160p@47fps videos but also does not present a power dissipation analysis. Pastuszak [14] proposes an architecture for the HEVC interpolation. However, the solution [14] cannot achieve the processing of UHD videos in real time. Wang [15] also proposes an architecture for the HEVC interpolation targeting the MC step. The work [15] reaches throughput to process 4320@30fps videos but it does not present power dissipation results. Penny [16] proposes a real-time architecture for the HEVC MC interpolator able to process UHD videos. Maich [10] presents a multi-standard interpolator for both HEVC and H.264/AVC able to process 4320p@30fps videos. However, as previously mentioned, no hardware solution for the VP9-10 interpolator was found in the literature.

3. VP9-10 INTERPOLATOR ARCHITECTURE

This section presents the VP9-10 Interpolator Architecture. First, the definition of the architecture parallelism is presented, and then the interpolator and the filter designs are shown.

3.1. Memory-Aware Basic Processing Unit (BPU)

At the decoder side, all the prediction block sizes must be supported in the MC creating important challenges at memory behavior, control, and parallelism perspectives. To deal with it, each prediction block may be decomposed in Basic Processing Units (BPUs - smaller blocks), allowing a more regular processing and memory access [16]. Parameters as memory communication (amount of data required from memory); samples to process (amount of samples to be interpolated), including the overhead; and the line-level parallelism, must be evaluated to determine which is the best size of the BPU. These three parameters were evaluated in this work for the VP9 and VP10 standards, observing its behavior with the use of 4x4, 8x8 and 16x16 BPUs. The experiments were performed through the Common Test

Table I. Types of filters on VP9-10 standard and their respective coefficients.

Y	CbCr	Lagrange-Based	DCT-Based Sharp	Low-Pass Smooth	Bi-linear
-	1/16	{0, 1, -5, 126, 8, -3, 1, 0}	{-1, 3, -7, 127, 8, -3, 1, 0}	{-3, -1, 32, 64, 38, 1, -3, 0}	{0, 0, 0, 120, 8, 0, 0, 0}
1/8	2/16	{-1, 3, -10, 122, 18, -6, 2, 0}	{-2, 5, -13, 125, 17, -6, 3, -1}	{-2, -2, 29, 63, 41, 2, -3, 0}	{0, 0, 0, 112, 16, 0, 0, 0}
-	3/16	{-1, 4, -13, 118, 27, -9, 3, -1}	{-3, 7, -17, 121, 27, -10, 5, -2}	{-2, -2, 26, 63, 43, 4, -4, 0}	{0, 0, 0, 104, 24, 0, 0, 0}
2/8	4/16	{-1, 4, -16, 112, 37, -11, 4, -1}	{-4, 9, -20, 115, 37, -13, 6, -2}	{-2, -3, 24, 62, 46, 5, -4, 0}	{0, 0, 0, 96, 32, 0, 0, 0}
-	5/16	{-1, 5, -18, 105, 48, -14, 4, -1}	{-4, 10, -23, 108, 48, -16, 8, -3}	{-2, -3, 21, 60, 49, 7, -4, 0}	{0, 0, 0, 88,40, 0, 0, 0}
3/8	6/16	{-1, 5, -19, 97, 58, -16, 5, -1}	{-4, 10, -24, 100, 59, -19, 9, -3}	{-1, -4, 18, 59, 51, 9, -4, 0}	{0, 0, 0, 80, 48, 0, 0, 0}
-	7/16	{-1, 6, -19, 88, 68, -18, 5, -1}	{-4, 11, -24, 90, 70, -21, 10, -4}	{-1, -4, 16, 57, 53, 12, -4, -1}	{0, 0, 0, 72,56, 0, 0, 0}
4/8	8/16	{-1, 6, -19, 78, 78, -19, 6, -1}	{-4, 11, -23, 80, 80, -23, 11, -4}	{-1, -4, 14, 55, 55, 14, -4, -1}	{0, 0, 0, 64, 64, 0, 0, 0}

Conditions (CTCs) [17] provided by ITU/ISO (Note: there are no standard test conditions specifically defined for VP9-10) and show that the 8x8 BPU size presents the best results. The use of 8x8 BPU reduces 45.94% and 46.52% the memory communication, for VP9 and VP10 respectively (on average), when compared to the use of 4x4 BPUs. This choice introduces a small overhead of 18% and 16% in the amount of samples to process, for VP9 and VP10 respectively, when also compared to the use of 4x4 BPUs.

Since our architecture works with 8x8 BPUs, when the architecture processes blocks with 4x8, 8x4 or 4x4 sizes, half, half, and one-third of the samples that compose the 8x8 block are lost, respectively. This causes unnecessary processing that can be compensated by raising the frequency of operation by the same factors of the overhead, i.e., 18% and 16% for VP9 and VP10, respectively. The drawback, however, is compensated by the memory communication reduction. For example, considering an 8x8 block calculated using 4x4 BPUs, 484 samples would be required (11x11x4 samples), while using an 8x8 BPU, the number of samples is 225 (15x15 samples). This occurs because the 4x4 BPU requires 7-border samples around the block (as other BPU sizes) and four BPUs should be processed in this case. Larger BPUs, such 16x16, show a worse result in terms of overhead than 8x8 BPUs, with similar memory communication. The use of 16x16 BPU increases 141% and 131% the overhead, for VP9 and VP10 respectively (on average), when compared to the use of 4x4 BPUs.

3.2. Interpolator Architecture

The Fig. 1 shows the developed top-level interpolator architecture, which is composed by eight horizontal (H) filters, eight sets containing eight intermediate shift registers, and eight vertical (V) filters. Each black square on the architecture represents a sample (Luma or Chroma). The reference matrix is of 15x15 samples, to predict an 8x8 block. At each cycle, 15 samples (one line) are read from the reference matrix to feed the inputs of the eight H filters. Each H filter receives eight input samples, following the scheme proposed by the architecture. Next, each H filter is connected to eight intermediate shift registers and the calculated samples are sequentially stored in these registers one by one, cycle after cycle. In the sequence, after all registers are fed (eight clock cycles), their outputs are connected to the V filter inputs. While the registers are not completely fed, the Vertical Data Gating (DG_V) units block the signal to V filter inputs in order to reduce the propagation of spurious values. Finally, the outputs from the V filters are the interpolator outputs. Thus, interpolated samples are not delivered in the interpolator outputs between cycles 1 and 8, and eight samples of the block are delivered per cycle between cycles 9 and 15, composing an 8x8 block every 15 clock cycles.

The Data Gating is also used when the motion vector in X axis (MV_X) or the motion vector in Y axis (MV_Y) are equal to zero since in these cases the samples need not be interpolated through the H filter or V filter, respectively. Basically, the Horizontal Data Gating (DG_H) is used when MV_X is equal to zero and DG_V is used when MV_Y is equal to zero. In these cases a bypass path is used.

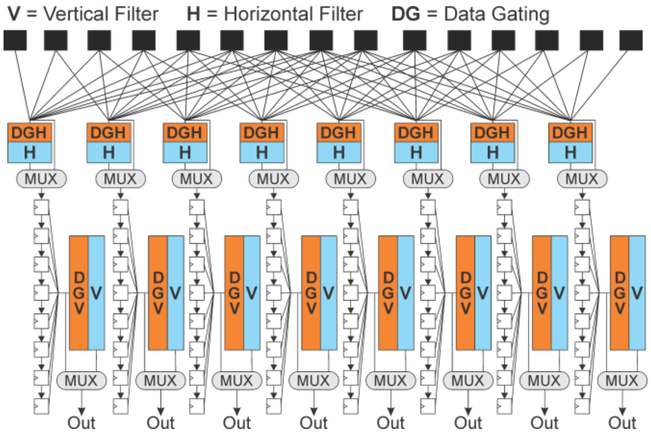

Fig.1. Luma/Chroma Interpolator architecture.

3.3. Luma/Chroma Unified Filter

The Fig. 2 shows the developed Luma/Chroma unified filter architecture. Each horizontal filter (H) has a unified filter, contemplating 15 Lagrange-based filters, 15 DCT-based filters, 15 Low-Pass filters, and 15 bilinear filters. As explained before, the filters are symmetric, i.e., the filter at 1/8 position has the same coefficients of the filter at 7/8 position, only changing its input sequence. That means it is possible to use the same architecture for both filters, by selecting 1/8 or 7/8 position. The information about the MV_X defines which filter must be used, or even if neither of

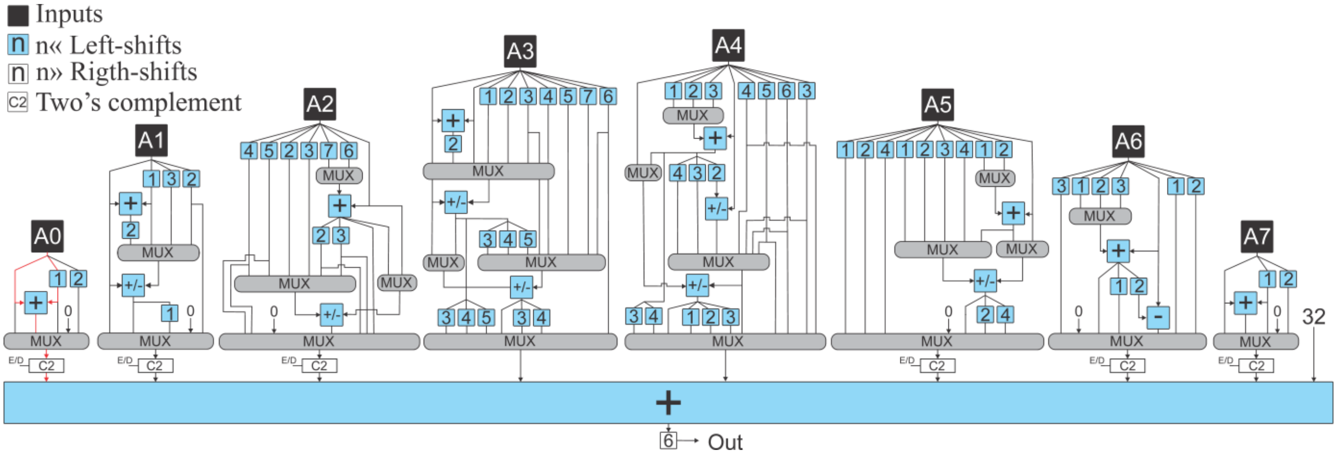

Fig. 2. VP9-10 Luma/Chroma Unified Filter hardware architecture.

them need to be used (in the case of MV_X points to an integer position). The same structure is used in the V filters, where the control is given by the MV_Y.

In the Fig. 2, A_0-A_7 represent the input samples of the filter. Each input-sample value can be multiplied by different constants according to the type of filter that is configured. For instance, the sample A_0 can be multiplied by the constants 0 (Lagrange and bilinear), 1 (Lagrange, DCT and Smooth), 2 (DCT and Smooth), 3 (DCT and Smooth), and 4 (DCT) (see Table I). Then, a multiplexer selects which operation result (shift-add) should be used according the type of filter: 0, A_0, A_0<<1, A_0+A_0<<1, or A_0<<2. An example is highlighted (red wires) in Fig.2 to the multiplication by 3, i.e., A_0+A_0<<1. As can also be observed in Fig.2, the multiplex output is connected to a 2-complement operation (negative coefficient), and after, to the inputs of the main adder. The main adder performs the sum of all samples and delivers the final result according the selected filter after a 6-bit shift to right.

4. RESULTS AND DISCUSSIONS

The developed VP9-10 Luma/Chroma interpolator architecture was described in Verilog HDL and synthesized using a 45nm (1.25V) Nangate standard-cell library with the Cadence Encounter RTL Compiler tool. The synthesis results are shown in Table II, where the gate count is calculated based on 2-input NANDs. The Table II also presents the results of seven related works, two for H.264/AVC, four for HEVC, and one solution for both.

The synthesis results shown that the maximum frequency is 500MHz, therefore, the developed architecture achieves real-time processing for UHD videos 4320p@30fps (at 413.9MHz). In terms of logic area, the architecture occupies 67.3k gates without Data Gating and 71.2k gates with Data Gating. Therefore, the Data Gating increases 5.79% the total logic area.

The power-dissipation analysis is presented in Table III, was performed with real input vectors extracted from the interpolation function of the VP9 and VP10 reference software [2]. The power dissipation was evaluated for the decoder with two high-resolution video sequences (CTCs) considering four QPs (Quantization Parameters). The interpolator architecture was stimulated with 750k samples at 206.7MHz, frequency of operation necessary to decode 2160p@60fps. Table III shows that when the Low-Power Data-Gating technique was applied to interpolator, a power-dissipation reduction of about 54.47% and 29.56% was provided for both VP9 and VP10 standards, respectively (on average). Table III also shows that the lower QP (higher video quality) increases up to 15.77% the power dissipation when compared with the higher QP.

It is not possible to provide a direct comparison with related works since no hardware solution for the VP9-10 interpolator was found in the literature. However, comparing with related works for H.264/AVC and HEVC standards, the achieved gate count is lower than the most architectures, due to the optimized design of unified Luma/Chroma filter and interpolator architecture. Despite our architecture being more complex than the others due to support more filter types, the power dissipation of this work tends (with and without Data Gating) to be lower than [10] and [16] because they process the samples for all supported filters at the same time, and then choose the results for only one. This work processes only the selected filter with the Unified Filter.

Table III. Power-dissipation results.

Arch. Version	Std.	Video Sequence	Power dissipation according to the QP value (mW)			
			QP22	**QP27**	**QP32**	**QP37**
Normal	VP9	PeopleOnStreet	4.82	4.56	4.40	4.06
		Traffic	5.55	5.41	5.28	5.15
	VP10	PeopleOnStreet	2.91	2.95	2.94	2.92
		Traffic	3.24	3.25	3.14	3.13
Data Gating	VP9	PeopleOnStreet	2.24	2.17	2.12	2.04
		Traffic	2.34	2.31	2.27	2.26
	VP10	PeopleOnStreet	2.21	2.18	2.09	1.93
		Traffic	2.25	2.22	2.21	2.14

5. CONCLUSIONS

This paper presented an efficient hardware design for Luma/Chroma sub-sample interpolation of both, VP9 and VP10 standards. Different types of interpolation filters are defined in VP9-10 standards and their similarities were found and explored in this work. The developed design has support for all filtering modes of the VP9-10, being fully compliant with MC/FME modules in both standards. This is a memory-efficient hardware architecture that decomposes the prediction blocks in 8x8 BPUs. The synthesis results for 45nm Nangate standard cells present power dissipation estimation with the use of real input vectors extracted from of the VP9-10 reference software. The results show that the developed design is able to process 2160p@60fps videos with a power dissipation of only 2.34mW.

Table II. Synthesis results and comparison with related works.

Related Work	**Zatt [11]**	**Liu [12]**	**Lian [13]**	**Pastuszak [14]**	**Wang [15]**	**Penny [16]**	**Maich [10]**	**Developed**
Standard	H.264	H.264	HEVC	HEVC	HEVC	HEVC	H.264/HEVC	**VP9/VP10**
Supported Tools	MC	FME	FME	MC/FME	MC	MC	MC/FME	**MC/FME**
Filters Supported	1 FIR + 51 bil.	1 FIR + 2 bil.	3 FIR	3 FIR	11 FIR	11 FIR	4 FIR + 2 bil.	**45 FIR + 15 bil.**
Solution	Luma & Chroma	Luma	Luma	Luma & Chroma	Luma	Luma & Chroma	Luma	**Luma & Chroma**
Technology (nm)	180	130	90	90	90	65	65	**45**
Frequency (MHz)	82	350	193	300	280	490.19	482.11	**413.9**
Gates (k)	12.345	75.74	64.5	211.69	90.3	98.68	166.8	**71.2**
Tot. Power for 2160p@60fps (mW)	-	-	-	-	-	8.10	39.15	**2.34**
Max. Throughput	1080p @30fps	2160p @30fps	2160p @47fps	1080p @30fps	4320p @30fps	4320p @60fps	4320p @30fps	**4320p@30fps**

6. REFERENCES

[1] Mukherjee, Debargha; Han, Jingning; Bankoski, Jim; Bultje, Ronald; Grange, Adrian; Koleszar, John; Wilkins, Paul; Xu, Yaowu, "A Technical Overview of VP9—The Latest Open-Source Video Codec," in Motion Imaging Journal, SMPTE , vol.124, no.1, pp.44-54, Jan. 2015

[2] Chromium open-source browser project, VP9-10 source code, Online: http://git.chromium.org/gitweb/?p=webm/libvpx.git;a=tree;f=vp9;hb=aaf61dfbcab414bfacc3171 501be17d191ff8506

[3] The Wallstret Journal,Tech: 1 Billion viewers - no profit. Online: http://www.wsj.com/articles/viewers-dont-add-up-to-profit-for-youtube-1424897967

[4] HEVC Advance, Royaltes rates summary. Online: http://www.hevcadvance.com/pdf/RoyaltyRatesSummary.pdf

[5] Mukherjee, Debargha; Su, Hui ; Bankoski, James; Converse, Alex ; Han, Jingning; Liu, Zoe ; Xu, Yaowu. "An overview of new video coding tools under consideration for VP10: the successor to VP9".. ProceedingsSPIE 9599, Applications of Digital Image Processing XXXVIII.

[6] ARM Community , Ittiam and ARM are the first to efficiently bring Google's VP9 to mobile devices. Online: http://www.ittiam.com/newsroom/news/2014-2/ittiam-to-publicly-showcase-arm-mali-gpu-compute-accelerated-hevc-and-vp9-at-mobile-world-congress-2014/

[7] Nvidia, TegraX1processor. Online: http://www.nvidia.com/object/tegra-x1-processor.html

[8] Business Wire, VeriSilicon Introduces a VP9 Decoder.Online:http://www.businesswire.com/news/home/20150301 005078/en/VeriSilicon-Introduces-Hantro-G2v2-Multi-format-Decoder-IP

[9] Afonso, V.; Maich, H; Audibert, L; Zatt, B; Porto M; Agostini, L. "Memory-Aware and High-Throughput Hardware Design for the HEVC Fractional Motion Estimation". in Proceedings of the 28th Symposium on Integrated Circuits and Systems Design (SBCCI), 2015.

[10] Maich, H.; Paim, G.; Afonso, V.; Agostini, L.; Zatt, B.; Porto, M., "A multi-standard interpolation hardware solution for H.264 and HEVC," in Image Processing (ICIP), 2015 IEEE International Conference on , vol., no., pp.2910-2914, 27-30 Sept. 2015.

[11] Zatt, B.; Susin, A.; Bampi, S; Agostini, L. 2008. High throughput architecture for H.264/AVC motion compensation sample interpolator for HDTV. In Proceedings of the 21st annual symposium on Integrated circuits and system design (SBCCI), 2008.

[12] J. Liu, X. Chen, Y. Fan and X. Zeng, "A full-mode FME VLSI architecture based on 8x8/4x4 adaptive hadamard transform for QFHD H.264/AVC encoder", 19th International Conference on VLSI and System-on-Chip (VLSI-SoC), 2011.

[13] X. Lian, W. Zhou, Z. Duab and R. Li, "An efficient interpolation filter VLSI architecture for HEVC standard," IEEE China Summit & International Conference onSignal and Information Processing (ChinaSIP), 2014.

[14] Pastuszak, G.; Trochimiuk, M., "Architecture Design and Efficiency Evaluation for the High-Throughput Interpolation in the HEVC Encoder," in Digital System Design (DSD), 2013 Euromicro Conference on , vol., no., pp.423-428, 4-6 Sept. 2013

[15] S. Wang, D. Zhou and S. Goto, "Motion compensation architecture for 8K UHDTV HEVC decoder," IEEE International Conference onMultimedia and Expo (ICME), 2014.

[16] Penny, W.;Paim, G.;Porto,M.;Agostini, L; Zatt, B. "Real-Time Architecture for HEVC Motion Compensation Sample Interpolator for UHD Videos". in Proceedings of the 28th Symposium on Integrated Circuits and Systems Design (SBCCI), 2015.

[17] F. Bossen, Common Test Conditions and Software Reference Configurations (JCTVC-L1100), 2013.