A LOW ENERGY HEVC SUB-PIXEL INTERPOLATION HARDWARE

Ercan Kalali, Ilker Hamzaoglu

Faculty of Engineering and Natural Sciences, Sabanci University
34956, Tuzla, Istanbul, Turkey
{ercankalali, hamzaoglu}@sabanciuniv.edu

ABSTRACT

Sub-pixel interpolation is one of the most computationally intensive parts of High Efficiency Video Coding (HEVC) video encoder and decoder. Therefore, in this paper, a low energy HEVC sub-pixel (half-pixel and quarter-pixel) interpolation hardware, which uses Hcub multiplierless constant multiplication algorithm, is proposed. The proposed HEVC sub-pixel interpolation hardware, in the worst case, can process 30 quad full HD (3840x2160) video frames per second. It has up to 48% less energy consumption than original HEVC sub-pixel interpolation hardware.

Index Terms**—** HEVC, Sub-Pixel Interpolation, Hardware Implementation, FPGA, Low Energy

1. INTRODUCTION

Joint collaborative team on video coding (JCT-VC) recently developed a new video compression standard called High Efficiency Video Coding (HEVC) [1]-[5]. HEVC provides 50% better coding efficiency than H.264 video compression standard [6]. In order to increase the performance of integer pixel motion estimation, sub-pixel (half and quarter) accurate variable block size motion estimation is performed in HEVC. Sub-pixel interpolation is one of the most computationally intensive parts of HEVC video encoder and decoder. On average, one fourth of the HEVC encoder complexity and 50% of the HEVC decoder complexity are caused by sub-pixel interpolation [6].

In H.264 standard, a 6-tap FIR filter is used for half-pixel interpolation and a bilinear filter is used for quarter-pixel interpolation [7, 8]. In HEVC standard, 3 different 8-tap FIR filters are used for both half-pixel and quarter-pixel interpolations. In H.264, 4x4 and 16x16 block sizes are used. However, in HEVC, prediction unit (PU) sizes can be from 4x4 to 64x64. Therefore, HEVC sub-pixel interpolation is more complex than H.264 sub-pixel interpolation.

Therefore, in this paper, a low energy HEVC sub-pixel (half-pixel and quarter-pixel) interpolation hardware for all PU sizes is proposed. The proposed hardware calculates common sub-expressions in different FIR filter equations in HEVC sub-pixel interpolation algorithm once. The proposed hardware also uses Hcub multiplierless constant multiplication (MCM) algorithm [9] in order to reduce number and size of the adders and to minimize the adder tree depth.

The proposed HEVC sub-pixel interpolation hardware is implemented in Verilog HDL. The Verilog RTL code is verified to work at 200 MHz in a Xilinx Virtex 6 FPGA. The proposed HEVC sub-pixel interpolation hardware**,** in the worst case, can process 30 quad full HD (3840x2160) video frames per second. It has up to 48% less energy consumption than original HEVC sub-pixel interpolation hardware.

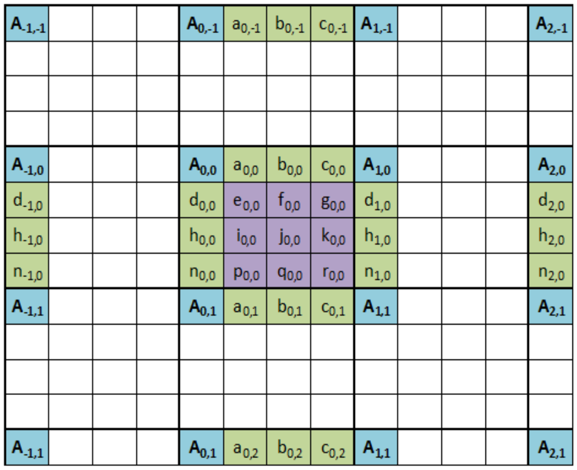
Several HEVC sub-pixel interpolation hardware are presented in the literature [10]-[13]. In [10], 3 different FIR filters are implemented using a reconfigurable datapath. Therefore, area and power consumption of HEVC sub-pixel interpolation hardware are reduced. However, this hardware is useful only for motion compensation. In [11], a sub-pixel interpolation hardware only for 4x4 PU size is proposed. This hardware is slower and has larger area than the hardware proposed in this paper, because it has restricted reconfigurability for filter datapaths. In [12], a coarse-grained reconfigurable datapath and adaptive scheduling are proposed for HEVC sub-pixel interpolation. These optimizations reduce the area and increase the throughput of HEVC sub-pixel interpolation hardware. But, its power consumption is not reported. In [13], a sub-pixel interpolation hardware is proposed for HEVC encoder. However, this sub-pixel interpolation hardware has much larger area and lower throughput than the other HEVC sub-pixel interpolation hardware.

The rest of the paper is organized as follows. In Section 2, HEVC sub-pixel interpolation algorithm is explained. In Section 3, the proposed HEVC sub-pixel interpolation hardware is presented, and its implementation results are given. Section 4 presents the conclusions.

2. HEVC SUB-PIXEL INTERPOLATION ALGORITHM

In HEVC standard, 3 different 8-tap FIR filters are used for both half-pixel and quarter-pixel interpolations. These 3 FIR filters type A, type B and type C are shown in (1), (2), and

978-1-4799-5751-4/14/$31.00 ©2014 IEEE



$A_{-1,-1}$				$A_{0,-1}$	$a_{0,-1}$	$b_{0,-1}$	$c_{0,-1}$	$A_{1,-1}$				$A_{2,-1}$
$A_{-1,0}$				$A_{0,0}$	$a_{0,0}$	$b_{0,0}$	$c_{0,0}$	$A_{1,0}$				$A_{2,0}$
$d_{-1,0}$				$d_{0,0}$	$e_{0,0}$	$f_{0,0}$	$g_{0,0}$	$d_{1,0}$				$d_{2,0}$
$h_{-1,0}$				$h_{0,0}$	$i_{0,0}$	$j_{0,0}$	$k_{0,0}$	$h_{1,0}$				$h_{2,0}$
$n_{-1,0}$				$n_{0,0}$	$p_{0,0}$	$q_{0,0}$	$r_{0,0}$	$n_{1,0}$				$n_{2,0}$
$A_{-1,1}$				$A_{0,1}$	$a_{0,1}$	$b_{0,1}$	$c_{0,1}$	$A_{1,1}$				$A_{2,1}$
$A_{-1,1}$				$A_{0,1}$	$a_{0,2}$	$b_{0,2}$	$c_{0,2}$	$A_{1,1}$				$A_{2,1}$

Fig. 1. Integer, Half and Quarter Pixels

$$a_{0,0} = (-A_{-3,0} + 4 * A_{-2,0} - 10 * A_{-1,0} + 58 * A_{0,0} + 17*A1,0-5*A2,0+A3,0 \gg shift1 \quad (1)$$

$$b_{0,0} = (-A_{-3,0} + 4 * A_{-2,0} - 11 * A_{-1,0} + 40 * A_{0,0} + 40*A1,0-11*A2,0+4*A3,0-A4,0 \gg shift1 \quad (2)$$

$$c_{0,0} = (-A_{-2,0} - 5 * A_{-1,0} + 17 * A_{-0,0} + 58 * A_{1,0} - 10*A2,0+4*A3,0-A4,0 \gg shift1 \quad (3)$$

(3), respectively. The shift1 value is determined based on bit depth of the pixel [1, 2].

Integer pixels ($A_{x,y}$), half pixels ($a_{x,y}$, $b_{x,y}$, $c_{x,y}$, $d_{x,y}$, $h_{x,y}$, $n_{x,y}$) and quarter pixels ($e_{x,y}$, $f_{x,y}$, $g_{x,y}$, $i_{x,y}$, $j_{x,y}$, $k_{x,y}$, $p_{x,y}$, $q_{x,y}$, $r_{x,y}$) in a PU are shown in Fig. 1. The half pixels a, b, c are interpolated from nearest integer pixels in horizontal direction, and the half-pixels d, h, n are interpolated from nearest integer pixels in vertical direction. The quarter pixels e, f, g are interpolated from the nearest half pixels a, b, c respectively in horizontal direction using type A filter. The quarter pixels i, j, k are interpolated similarly using type B filter, and the quarter pixels p, q, r are interpolated similarly using type C filter.

HEVC sub-pixel interpolation algorithm used in HEVC encoder calculates all the sub-pixels necessary for the sub-pixel accurate motion estimation.

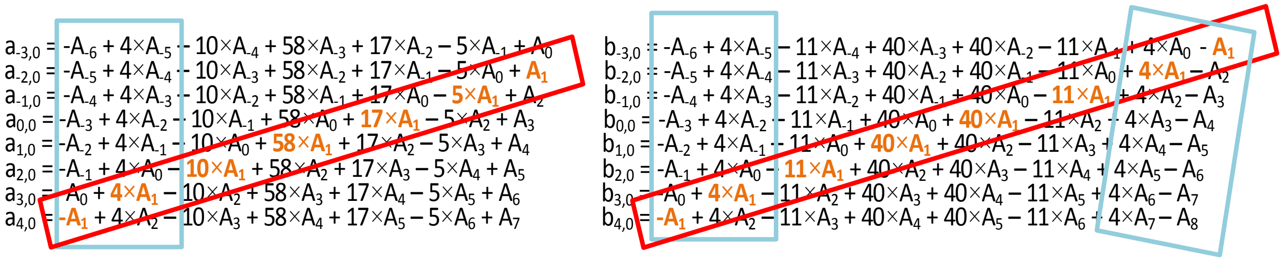
Table 1. Constant Coefficients of Input Pixels

Input Pixel	Coefficient	Datapath
A_{-6}	-1	C1
A_{-5}	-1,4	
A_{-4}	-1,4,-5,-10,-11	M1
A_{-3}	-1,4,-5,-10,-11,17,40,58	M2
A_{-2}	-1,4,-5,-10,-11,17,40,58	
A_{-1}	-1,4,-5,-10,-11,17,40,58	
A_{0}	-1,4,-5,-10,-11,17,40,58	
A_{1}	-1,4,-5,-10,-11,17,40,58	
A_{2}	-1,4,-5,-10,-11,17,40,58	
A_{3}	-1,4,-5,-10,-11,17,40,58	
A_{4}	-1,4,-5,-10,-11,17,40,58	
A_{5}	-1,4,-5,-10,-11,17,40,58	
A_{6}	-1,4,-5,-10,-11	M1
A_{7}	-1,4	C1
A_{8}	-1	

3. PROPOSED HEVC SUB-PIXEL INTERPOLATION HARDWARE

The type A and type B FIR filter equations for 8 half-pixels are shown in Fig. 2. As shown in Fig. 2, common sub-expressions are calculated in different equations and same integer pixel is multiplied with different constant coefficients in different equations. Therefore, in the proposed hardware, common sub-expressions in different equations are calculated once, and the result is used in all the equations. The proposed hardware also uses Hcub MCM algorithm in order to reduce number and size of the adders, and to minimize the adder tree depth [9].

Hcub algorithm tries to minimize number of adders, their bit size and adder tree depth in a multiplier block, which multiplies a single input with multiple constants. Hcub algorithm is used in this paper, because it produces better results than other MCM algorithms [9]. Multiplier block creation tool from Spiral implementing Hcub algorithm is used [14]. This tool takes constants to be multiplied as input and produces all necessary shift and add operations in a multiplier block as output. A multiplier block hardware has only one input, and it outputs results of multiplications with all the constants.



$$\begin{aligned}
a_{-3,0} &= -A_{-6} + 4\times A_{-5} - 10\times A_{-4} + 58\times A_{-3} + 17\times A_{-2} - 5\times A_{-1} + A_{0}\\
a_{-2,0} &= -A_{-5} + 4\times A_{-4} - 10\times A_{-3} + 58\times A_{-2} + 17\times A_{-1} - 5\times A_{0} + A_{1}\\
a_{-1,0} &= -A_{-4} + 4\times A_{-3} - 10\times A_{-2} + 58\times A_{-1} + 17\times A_{0} - 5\times A_{1} + A_{2}\\
a_{0,0} &= -A_{-3} + 4\times A_{-2} - 10\times A_{-1} + 58\times A_{0} + 17\times A_{1} - 5\times A_{2} + A_{3}\\
a_{1,0} &= -A_{-2} + 4\times A_{-1} - 10\times A_{0} + 58\times A_{1} + 17\times A_{2} - 5\times A_{3} + A_{4}\\
a_{2,0} &= -A_{-1} + 4\times A_{0} - 10\times A_{1} + 58\times A_{2} + 17\times A_{3} - 5\times A_{4} + A_{5}\\
a_{3,0} &= -A_{0} + 4\times A_{1} - 10\times A_{2} + 58\times A_{3} + 17\times A_{4} - 5\times A_{5} + A_{6}\\
a_{4,0} &= -A_{1} + 4\times A_{2} - 10\times A_{3} + 58\times A_{4} + 17\times A_{5} - 5\times A_{6} + A_{7}
\end{aligned}$$

(a) Type A FIR Filters

$$\begin{aligned}
b_{-3,0} &= -A_{-6} + 4\times A_{-5} - 11\times A_{-4} + 40\times A_{-3} + 40\times A_{-2} - 11\times A_{-1} + 4\times A_{0} - A_{1}\\
b_{-2,0} &= -A_{-5} + 4\times A_{-4} - 11\times A_{-3} + 40\times A_{-2} + 40\times A_{-1} - 11\times A_{0} + 4\times A_{1} - A_{2}\\
b_{-1,0} &= -A_{-4} + 4\times A_{-3} - 11\times A_{-2} + 40\times A_{-1} + 40\times A_{0} - 11\times A_{1} + 4\times A_{2} - A_{3}\\
b_{0,0} &= -A_{-3} + 4\times A_{-2} - 11\times A_{-1} + 40\times A_{0} + 40\times A_{1} - 11\times A_{2} + 4\times A_{3} - A_{4}\\
b_{1,0} &= -A_{-2} + 4\times A_{-1} - 11\times A_{0} + 40\times A_{1} + 40\times A_{2} - 11\times A_{3} + 4\times A_{4} - A_{5}\\
b_{2,0} &= -A_{-1} + 4\times A_{0} - 11\times A_{1} + 40\times A_{2} + 40\times A_{3} - 11\times A_{4} + 4\times A_{5} - A_{6}\\
b_{3,0} &= -A_{0} + 4\times A_{1} - 11\times A_{2} + 40\times A_{3} + 40\times A_{4} - 11\times A_{5} + 4\times A_{6} - A_{7}\\
b_{4,0} &= -A_{1} + 4\times A_{2} - 11\times A_{3} + 40\times A_{4} + 40\times A_{5} - 11\times A_{6} + 4\times A_{7} - A_{8}
\end{aligned}$$

(b) Type B FIR Filters

Fig. 2. Type A and Type B FIR Filters

The proposed HEVC sub-pixel (half-pixel and quarter-pixel) interpolation hardware for all PU sizes is shown in Fig. 3. The proposed hardware interpolates all the sub-pixels (half-pixels and quarter-pixels) for the luma component of a PU using integer or half pixels. Four buffers are used to store integer and half pixels necessary for interpolating the half and quarter pixels. The interpolated a, b, c half-pixels are stored in the filtered pixels buffers A, B, C. These on-chip buffers reduce the required off-chip memory bandwidth and power consumption.

8x3=24 sub-pixels are interpolated in parallel using type A, type B and type C FIR filter equations. Common 1 (C1) datapath calculates the common sub-expressions in the equations shown as blue boxes in Fig. 2. Multiplier 1 (M1) and Multiplier 2 (M2) datapaths calculate the multiplications with multiple constant coefficients shown as red boxes in Fig. 2. As shown in Table 1, since constant coefficients of input pixels (A_{-4}, A_6) and (A_{-3}-A_5) are different, two different multiplier block hardware are used. Then, sub-pixels are calculated using adder trees.

Since 15 sub-pixels should be interpolated for one integer pixel, 64x15 sub-pixels should be interpolated for an 8x8 PU. 8x7 extra a, b, c half-pixels are necessary for the interpolation of quarter pixels. Therefore, the proposed hardware, in the worst case, interpolates the sub-pixels for an 8x8 PU in 48 clock cycles.

First, integer pixels are loaded into integer pixels buffer in one clock cycle. Then, 8x8 d, h, n half-pixels are interpolated and stored in the output buffer. After that, 8x15 a, b and c half-pixels necessary for interpolating quarter pixels are interpolated in 15 clock cycles, and stored in the filtered pixel buffers A, B, and C. Finally, 9x8x8 quarter pixels are interpolated and stored in the output pixel buffers.

In this paper, an original HEVC sub-pixel interpolation hardware (SPIHW) is also designed for energy consumption comparison. This hardware computes type A, B and C filters separately same as the original hardware proposed in [10].

The proposed SPIHW and SPIHW+MCM HEVC sub-pixel interpolation hardware are implemented using Verilog HDL. The hardware implementations are verified with RTL simulations. The RTL simulation results matched the results of a software model of HEVC sub-pixel interpolation algorithm. The Verilog RTL codes are synthesized and mapped to a Xilinx XC6VLX130T FF1156 FPGA with speed grade 3 using Xilinx ISE 13.4. SPIHW FPGA implementation uses 4110 LUTs, 3448 DFFs and 6 BRAMs. SPIHW+MCM FPGA implementation uses 3929 LUTs, 3422 DFFs, and 6 BRAMs. Both FPGA implementations are verified to work at 200 MHz by post place and route simulations. Therefore, they can process 30 quad HD (3840x2160) video frames per second.

The power consumptions of both FPGA implementations are estimated using Xilinx XPower Analyzer tool for Tennis (1920x1080) and Kimono (1920x1080) videos [15]. The energy consumptions of SPIHW and SPIHW+MCM FPGA implementations are shown in Fig. 4 and Fig. 5. As shown in these figures, the proposed HEVC sub-pixel interpolation hardware has up to 48% less energy consumption than original HEVC sub-pixel interpolation hardware.

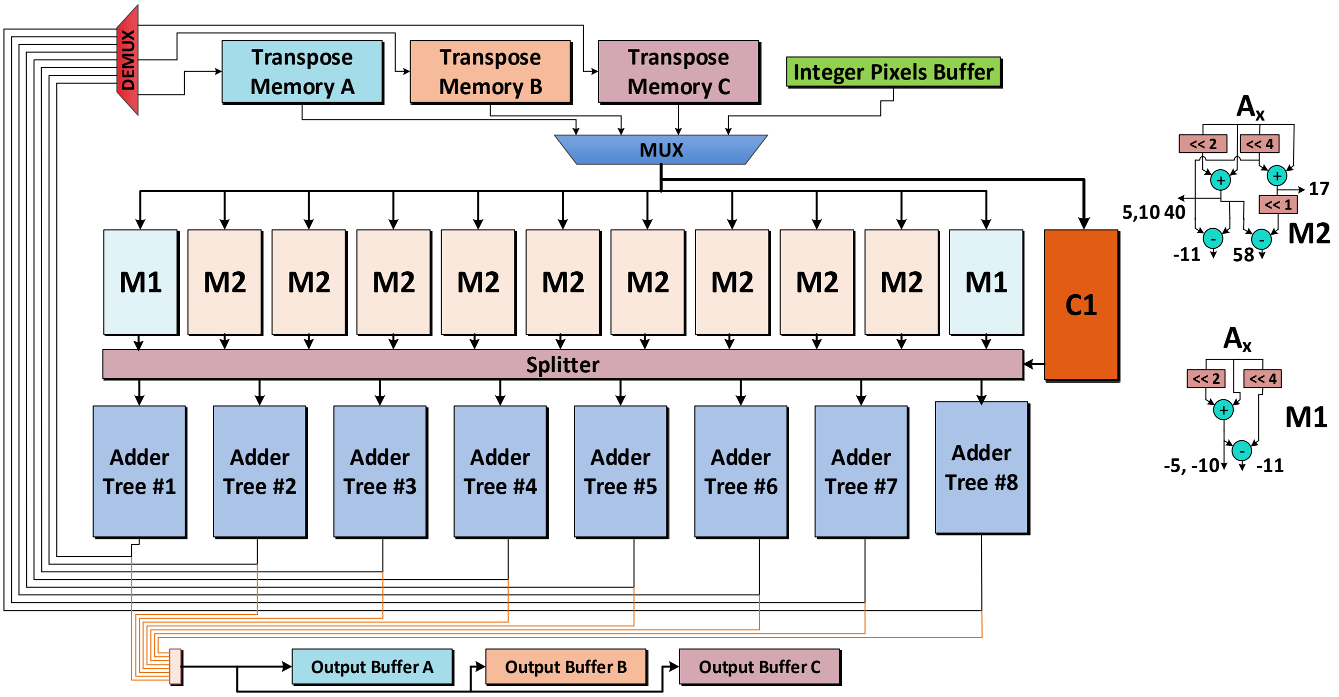


Fig. 3. Proposed HEVC Sub-pixel Interpolation Hardware

Table 2. HEVC Sub-pixel Interpolation Hardware Comparison

	[11]		[10]		[12]	[13]	Proposed	
Technology	90 nm	90 nm	90 nm	Xilinx Virtex 6	150 nm	90 nm	90 nm	Xilinx Virtex 6
Gate/Slice Count	19.6 K	32.5 K	10.5 K	883	30.2 K	224 K	28.5 K	1557
Max Speed (MHz)	85.5	171	100	100	312	333	200	200
Frames per Second	30 1920x1080	60 3840x2160	30 3840x2160	30 3840x2160	30 3840x2160	30 1920x1080	30 3840x2160	30 3840x2160
Design	Only MC	Only MC	Only MC	Only MC	ME + MC	ME + MC	ME + MC	ME + MC

In order to estimate the power consumption of a sub-pixel interpolation hardware, timing simulation of its placed and routed netlist is done at 100 MHz using Mentor Graphics Questa for encoding one frame of each video sequence. The signal activities of these timing simulations are stored in VCD files, and these VCD files are used for estimating the power consumption of that sub-pixel interpolation hardware using Xilinx XPower Analyzer tool. Since sub-pixel interpolation hardware will be used as part of a HEVC encoder or decoder, only internal power consumption is considered and input and output power consumptions are ignored.

The Verilog RTL code of the proposed HEVC sub-pixel interpolation hardware is also synthesized and place & routed to Synopsys 90nm standard cell library. The gate count of resulting ASIC implementation is calculated as 28.5k, excluding on-chip memories, based on NAND (2x1) gate area.

HEVC sub-pixel interpolation hardware comparison is shown in Table 2. Since the sub-pixel interpolation hardware proposed in [10] and [11] are designed only for motion compensation (MC), their areas are lower than the proposed hardware. Since these sub-pixel interpolation hardware should interpolate 15 sub-pixels instead of 1 sub-pixel for motion estimation, their areas will increase. The area of the sub-pixel interpolation hardware proposed in [12] is slightly larger than the proposed hardware. In addition, it requires higher clock frequency to achieve real time performance. The sub-pixel interpolation hardware proposed in [13] has much larger area and less performance than the proposed hardware. The power consumptions of sub-pixel interpolation hardware in [11]-[13] are not shown.

4. CONCLUSION

In this paper, a low energy HEVC sub-pixel (half-pixel and quarter-pixel) interpolation hardware for all PU sizes is proposed. The proposed hardware uses Hcub multiplierless constant multiplication algorithm. The proposed hardware is implemented in Verilog HDL. The Verilog RTL code is verified to work at 200 MHz in a Xilinx Virtex 6 FPGA. The proposed hardware, in the worst case, can process 30 quad full HD (3840x2160) video frames per second. It has up to 48% less energy consumption than original HEVC sub-pixel interpolation hardware.

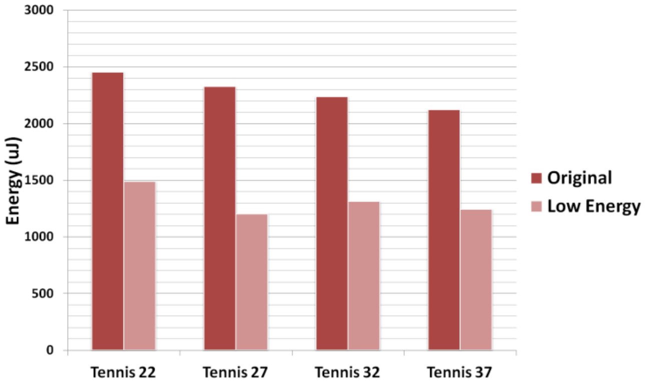


Fig. 4. Energy Consumption of HEVC Sub-Pixel Interpolation Hardware for Tennis (1920x1080) with Different QP Values

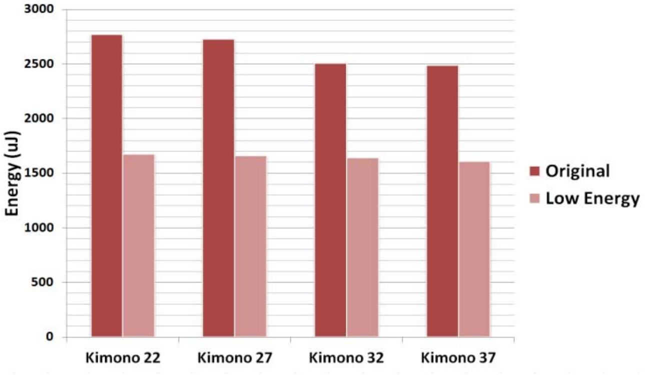


Fig. 5. Energy Consumption of HEVC Sub-Pixel Interpolation Hardware for Kimono (1920x1080) with Different QP Values

5. REFERENCES

[1] B. Bross, W.J. Han, J.R. Ohm, G.J. Sullivan, Y.K. Wang, T. Wiegand, “High Efficiency Video Coding (HEVC) Text Specification Draft 10”, *JCTVC-L1003*, Feb. 2013.

[2] G. J. Sullivan, J.R. Ohm, W.J. Han, T. Wiegand, “ Overview of the High Efficiency Video Coding (HEVC) Standard”, *IEEE Trans. on CAS for Video Technology*, vol.22, no.12, pp.1649-1668, Dec. 2012.

[3] E. Kalali, Y. Adibelli, I. Hamzaoglu, “A High Performance and Low Energy Intra Prediction Hardware for HEVC Video Decoding”, *Conference on Design and Architectures for Signal and Image Processing,* Oct. 2012.

[4] E. Kalali, E. Ozcan, O. M. Yalcinkaya, I Hamzaoglu, “A Low Energy HEVC Inverse DCT Hardware”, *IEEE Int. Conference on Consumer Electronics – Berlin*, Sept. 2013.

[5] M. Tikekar, C.-T. Huang, C. Juvekar, V. Sze, A. P. Chandrakasan, “A-249-Mpixel/s HEVC Video Decoder Chip for 4K Ultra-HD Applications”, *IEEE Journal of Solid-State Circuits*, vol. 49, no. 1, pp. 61-72, Jan. 2014.

[6] J. Vanne, M. Viitanen, T. D. Hamalainen, A. Hallapuro, “Comparative Rate-Distortion-Complexity Analysis of HEVC and AVC Video Codecs”, *IEEE Trans. on CAS for Video Technology*, vol.22, no.12, pp.1885-1898, Dec. 2012.

[7] S. Yalcin, I. Hamzaoglu, “A High Performance Hardware Architecture for Half-Pixel Accurate H.264 Motion Estimation”, *14th IFIP Int. Conference on VLSI-SoC*, Oct. 2006.

[8] S. Oktem, I. Hamzaoglu, “An Efficient Hardware Architecture for Quarter-Pixel Accurate H.264 Motion Estimation”, *10th Euromicro Conference on Digital System Design*, Aug. 2007.

[9] Y. Voronenko, M. Püschel, "Multiplierless Constant Multiple Multiplication", *ACM Transactions on Algorithms*, vol. 3, no. 2, May 2007.

[10] E. Kalali, Y. Adibelli, I. Hamzaoglu, “A Reconfigurable HEVC Sub-Pixel Interpolation Hardware”, *IEEE Int. Conference on Consumer Electronics - Berlin*, Sept. 2013.

[11] Z. Guo, D. Zhou, S. Goto, “An Optimized MC Interpolation Architecture for HEVC”, *IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, March 2012.

[12] C. M. Diniz, M. Shafique, S. Bampi, J. Henkel, “High-Throughput Interpolation Hardware Architecture with Coarse-Grained Reconfigurable Datapaths for HEVC”, *IEEE Int. Conference on Image Processing*, Sept. 2013.

[13] G. Pastuszak, M. Trochimiuk, “Architecture Design and Efficiency Evaluation for The High-Throughput Interpolation in the HEVC Encoder”, *16th Euromicro Conference on Digital System Design*, Sept. 2013.

[14] Spiral Multiplier Block Generator, http://spiral.ece.cmu.edu/mcm/gen.html.

[15] F. Bossen, “Common HM Test Conditions and Software Reference Configurations”, *JCTVC-L1100*, March 2013.

978-1-4799-5751-4/14/$31.00 ©2014 IEEE