POLITECNICO DI TORINO
Repository ISTITUZIONALE

Multiplierless, Folded 9/7 - 5/3 Wavelet VLSI Architecture

Original
Multiplierless, Folded 9/7 - 5/3 Wavelet VLSI Architecture / Martina, M., Masera, G.. - In: IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS. II, EXPRESS BRIEFS. - ISSN 1549-7747. - STAMPA. - 54:9(2007), pp. 770-774. [10.1109/TCSII.2007.900354]

Availability:
This version is available at: 11583/1788971 since:

Publisher:
IEEE

Published
DOI:10.1109/TCSII.2007.900354

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright

(Article begins on next page)

19 November 2026

Multiplierless, folded 9/7 - 5/3 wavelet VLSI architecture

Maurizio Martina, *Member IEEE*, Guido Masera, *Member IEEE*

***Abstract*—This paper proposes a multiplierless VLSI architecture for the famous 9/7 wavelet filters. The novelty of this architecture is the possibility to compute the 5/3 wavelet results into the 9/7 data-path with a reduced number of adders compared to other solutions. The multiplierless architecture has been characterized in terms of performance through simulations into a JPEG2000 environment and compared to other solutions. Implementation on a 0.13 μm standard cell technology shows that the proposed architecture compared to other multiplierless architectures requires a reduced amount of logic with excellent performance.**

***Index Terms*—JPEG2000, Wavelet, Multiplierless Implementation, Filter Bank, VLSI**

I. Introduction

The Discrete Wavelet Transform (DWT) is widely employed in many image and video compression systems due to its excellent decorrelation properties [1]. In particular, many famous coders have been proposed to effectively compress images or frames processed via the DWT [2]. Besides, the DWT is employed in JPEG2000, the new standard for image compression, where the 9/7 [3] and the 5/3 [4] wavelet filters are employed as the default filters for lossy and lossless compression respectively [5].

Several recent publications describe efficient implementations of JPEG2000 encoders and decoders [6]. Moreover, many research works have faced the problem of reducing the DWT complexity. This issue has been investigated mainly from two perspectives: i) reducing the memory access overhead [7], [8], [9]; ii) reducing the DWT computational complexity [9], [10]. Recently, reduced complexity solutions, involving multiplierless implementations either through filter banks [11], [12], [13], [14] or lifting scheme [15], [16], [17] have been proposed.

The aim of this paper is to embed the 5/3 wavelet computation into the 9/7, in order to exploit as much as possible the 5/3 results to achieve the 9/7 ones. To the best of our knowledge, this is the first work where not only hardware resources, but also the output values obtained via the 5/3 wavelet filters are used to compute the 9/7 results. In [11], [12] and [14] the 9/7 filters symmetry is exploited by adding input samples (x_i) together to obtain $w_0 = x_i$, $w_1 = x_{i+1} + x_{i-1}$, $w_2 = x_{i+2} + x_{i-2}$ and so on. Some multiplierless solutions (e.g. [11], [12]) are based on butterfly structures where the w_i values are first added together, then partial results are shifted and combined to obtain the output values. On the other hand in [14] w_i are first partially shifted and then added. To obtain the 5/3 wavelet results different shift amounts have to be applied to each w_i, as detailed in section II. As a consequence, the solution proposed in [14] is more suited to embed the 5/3 calculation and reuse partial results instead of adding resource to support both the 9/7 and the 5/3 wavelet filters. The proposed architecture, that is derived from the multiplierless 9/7 filter bank (FB) proposed in [14], embodies the 5/3 FB and further reduces the number of adders required from 21 to 19, granting the same performance. It is worth pointing out that, even if it is not a goal of this work, the proposed architecture allows on–the–fly switching between the 5/3 and 9/7 wavelet filters.

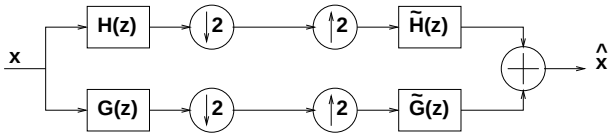


Figure 1. Filter bank block scheme

II. Theoretical Derivation

Let's consider the filter bank shown in Fig. 1, where $H(z) = \sum_i^k h[i]z^{-i}$ and $G(z) = \sum_i^l g[i]z^{-i}$ are the low pass and high pass analysis filters with length k and l respectively, and $\tilde{H}(z) = \sum_i^{\tilde{k}} \tilde{h}[i]z^{-i}$ and $\tilde{G}(z) = \sum_i^{\tilde{l}} \tilde{g}[i]z^{-i}$ the low pass and high pass synthesis ones with length $\tilde{k}$ and $\tilde{l}$.

In [14] it has been proved that the 9/7 wavelet filters can be decomposed into two stages. The first stage involves only rational factors, whereas the second one requires multiplications with 5 constants: $K1 = (a+b/2+3/8)/2a$, $K2 = (b+1)/8a$, $K3 = 1/32a$, $J1 = (r+1/2)/2r$ and $J2 = 1/8r$, where r is the real solution of the third order equation

$$1/20 + 4/20x + 10/20x^2 + x^3 = 0 \qquad (1)$$

and a and b are the product and the sum of the two complex conjugate solutions respectively.

As suggested in [14], accepting a small loss in terms of Peak Signal to Noise Ratio (PSNR) the 5 constants can be approximated with few power of 2 values: $K1 \simeq 2 + 1/16$, $K2 \simeq 1$, $K3 \simeq 1/8 + 1/16 + 1/32$, $J1 \simeq 1 + 1/4$ and $J2 \simeq 1/4 + 1/8$. Considering the two analysis filters $h^{9,7}[n]$ and $g^{9,7}[n]$ as vectors ($\underline{h}^{9,7}$ and $\underline{g}^{9,7}$), we can represent them as the product of a matrix and a vector, where the matrix contains the rational factors and the vector contains the constants - see (2) and (3). Besides, $h^{9,7}[n]$ and $g^{9,7}[n]$ symmetry suggests, for the sake of simplicity, to concentrate only on taps with index

The authors are with CERCOM (Center for Multimedia Radio Communications) - Dipartimento di Elettronica - Politecnico di Torino. Copyright (c) 2006 IEEE. Personal use of this material is permitted. However, permission to use this material for any other purposes must be obtained from the IEEE by sending an email to pubs-permissions@ieee.org.

Table I
COEFFICIENTS FOR THE 9/7 AND 5/3 FILTERS

n	$h^{9,7}[n]$			$g^{9,7}[n]$			$h^{5,3}[n]$	$g^{5,3}[n]$
0	$\frac{6}{8}K1-\frac{8}{8}K2+\frac{2}{8}K3$	=	0.60294901823636	$\frac{6}{8}J1-\frac{8}{8}J2$	=	0.55754352622850	$\frac{6}{8}$	$\frac{8}{8}$
±1	$\frac{4}{8}K1-\frac{7}{8}K2+\frac{4}{8}K3$	=	0.26686411844287	$-\frac{4}{8}J1+\frac{7}{8}J2$	=	−0.29563588155713	$\frac{2}{8}$	$-\frac{4}{8}$
±2	$\frac{1}{8}K1-\frac{4}{8}K2+\frac{6}{8}K3$	=	−0.07822326652899	$\frac{1}{8}J1-\frac{4}{8}J2$	=	−0.02877176311425	$-\frac{1}{8}$	0
±3	$-\frac{1}{8}K2+\frac{4}{8}K3$	=	−0.01686411844287	$\frac{1}{8}J2$	=	0.04563588155713	0	0
±4	$\frac{1}{8}K3$	=	0.02674875741081	0			0	0

$n \geq 0$ (see Table I). Thus $\underline{h}^{9,7} = \underline{\underline{M}} \cdot \underline{K}$ and $\underline{g}^{9,7} = \underline{\underline{N}}^{R_5,C_3} \cdot \underline{J}$ where

$$\underline{\underline{M}} = \begin{pmatrix} \frac{6}{8} & -\frac{8}{8} & \frac{2}{8} \\ \frac{4}{8} & -\frac{7}{8} & \frac{4}{8} \\ \frac{1}{8} & -\frac{4}{8} & \frac{6}{8} \\ 0 & -\frac{1}{8} & \frac{4}{8} \\ 0 & 0 & \frac{1}{8} \end{pmatrix}, \ \underline{K} = \begin{bmatrix} K1 \\ K2 \\ K3 \end{bmatrix}, \ \underline{J} = \begin{bmatrix} J1 \\ J2 \end{bmatrix} \quad (2)$$

and

$$\underline{\underline{N}} = \begin{pmatrix} \frac{6}{8} & -\frac{8}{8} & \frac{2}{8} \\ -\frac{4}{8} & \frac{7}{8} & -\frac{4}{8} \\ \frac{1}{8} & -\frac{4}{8} & \frac{6}{8} \\ 0 & \frac{1}{8} & -\frac{4}{8} \\ 0 & 0 & \frac{1}{8} \end{pmatrix}, \ \underline{\underline{N}}^{R_5,C_3} = \begin{pmatrix} \frac{6}{8} & -\frac{8}{8} \\ -\frac{4}{8} & \frac{7}{8} \\ \frac{1}{8} & -\frac{4}{8} \\ 0 & \frac{1}{8} \end{pmatrix} \quad (3)$$

Since $G(z) = \tilde{H}(-z)$, odd terms have different sign and $\underline{\underline{N}}$ can be directly derived from $\underline{\underline{M}}$ changing the sign of odd terms (see Table I $n \in \{1,3\}$). As it can be observed $\underline{\underline{N}}^{R_5,C_3}$ is obtained removing the 5th row and the 3rd column of $\underline{\underline{N}}$.

Let's focus on the first stage defined by $\underline{\underline{M}}$ and $\underline{\underline{N}}^{R_5,C_3}$: grouping together some of the first stage elements we obtain the 5/3 filter. For the sake of simplicity, we exploit the 9/7 and the 5/3 filters symmetry to group together samples that ought to be multiplied by the same tap h_i. So that we have $w_0 = x_i$, $w_1 = x_{i+1} + x_{i-1}$, $w_2 = x_{i+2} + x_{i-2}$ and so on. For the low-pass coefficients we have (see Table I):

$$\begin{aligned} y_l^{9,7} &= h^{9,7}[0]x_i + h^{9,7}[1](x_{i+1} + x_{i-1}) + \cdots \\ &= \left(\frac{6}{8}K1 - \frac{8}{8}K2 + \frac{2}{8}K3\right) w_0 + \cdots \\ &= \left(\frac{6}{8}w_0 + \frac{4}{8}w_1 + \frac{1}{8}w_2\right) K1 + \cdots \\ &= p1K1 - p2K2 + p3K3 \end{aligned} \quad (4)$$

where

$$p1 = \frac{6}{8}w_0 + \frac{4}{8}w_1 + \frac{1}{8}w_2 \quad (5)$$

$$p2 = \frac{8}{8}w_0 + \frac{7}{8}w_1 + \frac{4}{8}w_2 + \frac{1}{8}w_3 \quad (6)$$

$$p3 = \frac{2}{8}w_0 + \frac{4}{8}w_1 + \frac{6}{8}w_2 + \frac{4}{8}w_3 + \frac{1}{8}w_4 \quad (7)$$

Since for the 5/3 low-pass coefficient holds true:

$$y_l^{5,3} = \frac{6}{8}w_0 + \frac{2}{8}w_1 - \frac{1}{8}w_2 \quad (8)$$

we can rewrite $p1$ as

$$\begin{aligned} p1 &= \frac{6}{8}w_0 + \frac{2}{8}w_1 + \frac{2}{8}w_1 + \frac{2}{8}w_2 - \frac{1}{8}w_2 \\ &= \frac{6}{8}w_0 + \frac{2}{8}w_1 - \frac{1}{8}w_2 + \frac{2}{8}w_1 + \frac{2}{8}w_2 \\ &= y_l^{5,3} + \frac{2}{8}w_1 + \frac{2}{8}w_2 \end{aligned} \quad (9)$$

With similar considerations we derive for the high-pass coefficient:

$$y_h^{9,7} = q1J1 - q2J2 \quad (10)$$

where

$$q1 = \frac{6}{8}w_0 - \frac{4}{8}w_1 + \frac{1}{8}w_2 \quad (11)$$

$$q2 = \frac{8}{8}w_0 - \frac{7}{8}w_1 + \frac{4}{8}w_2 - \frac{1}{8}w_3 \quad (12)$$

Then

$$y_h^{5,3} = \frac{8}{8}w_0 - \frac{4}{8}w_1 \quad (13)$$

so

$$\begin{aligned} q2 &= \frac{8}{8}w_0 - \frac{4}{8}w_1 - \frac{3}{8}w_1 + \frac{4}{8}w_2 - \frac{1}{8}w_3 \\ &= y_h^{5,3} - \frac{3}{8}w_1 + \frac{4}{8}w_2 - \frac{1}{8}w_3 \end{aligned} \quad (14)$$

So that we can build a folded architecture that exploits the 5/3 coefficients to obtain the 9/7 ones rewriting $p2$ and $q1$ as

$$p2 = z_l^{5,3} + \frac{3}{8}w_1 + \frac{4}{8}w_2 + \frac{1}{8}w_3 \quad (15)$$

$$q1 = z_h^{5,3} - \frac{2}{8}w_1 + \frac{2}{8}w_2 \quad (16)$$

with

$$z_l^{5,3} = \frac{8}{8}w_0 + \frac{4}{8}w_1 \quad (17)$$

$$z_h^{5,3} = \frac{6}{8}w_0 - \frac{2}{8}w_1 - \frac{1}{8}w_2 \quad (18)$$

III. PERFORMANCE AND EXPERIMENTAL RESULTS

The multiplierless FB described in section II allows to obtain lossless compression when the 5/3 wavelet is selected. On the other hand when the 9/7 is employed the performance detailed in [14] are achieved. In order to compare the proposed FB architecture with the multiplierless LS 9/7 solution proposed in [17], simulations inside the JPEG2000 image coding standard [5] framework have been performed.

A free implementation of a JPEG2000 codec written in C language, *openjpeg* [18], that is Class-1 Profile-1 compliant with the standard, has been employed for our tests. Five standard images have been used for the test: 'Lenna' 256×256 (I=1), 'Barbara' 512×512 (I=2), 'Boat' 512×512 (I=3), 'Goldhill' 512×512 (I=4) and 'Fingerprint' 512×512 (I=5) [19]. The number of wavelet decomposition levels (L) has been varied from 1 to 3 for 256×256 images and from 1 to 4 for 512×512 images. Different compression rates have been imposed, namely 1, 0.5, 0.25 and 0.125 bit per pixel (bpp), precinct and code-block size are the encoder default values [5].

Table II
PERFORMANCE COMPARISON AMONG DIFFERENT 9/7 WAVELET IMPLEMENTATIONS FOR MULTIPLE IMAGES (I), DECOMPOSITION LEVELS (L) AND COMPRESSION RATES (1, 0.5, 0.25, 0.125 BPP) INTO JPEG2000: A - ORIGINAL *openjpeg*, B - MULTIPLIERLESS FB [14], C - LS [17]

I	L	A [dB]				B [dB]				C [dB]			
		1	0.5	0.25	0.125	1	0.5	0.25	0.125	1	0.5	0.25	0.125
1	1	37.08	30.59	25.22	19.36	37.11	30.62	25.22	19.33	37.16	30.62	25.22	19.35
	2	38.50	33.29	28.72	24.61	38.50	33.30	28.74	24.60	38.46	33.28	28.72	25.01
	3	38.75	33.44	29.11	25.82	38.72	33.48	29.13	25.79	38.60	33.43	29.08	25.77
2	1	35.77	29.48	24.05	20.63	35.75	29.52	24.06	20.61	35.83	29.56	24.22	20.61
	2	37.56	32.16	27.69	24.17	37.56	32.08	27.70	24.18	37.57	32.16	27.88	24.26
	3	37.87	32.78	28.75	25.31	37.78	32.74	28.75	25.32	37.78	32.80	28.74	25.60
	4	37.89	32.78	28.83	25.79	37.77	32.82	28.86	25.79	37.76	32.83	28.85	25.81
3	1	37.66	32.64	28.22	23.39	37.61	32.63	28.28	23.39	37.62	32.64	28.28	23.40
	2	38.87	33.96	30.14	26.96	38.84	34.01	30.10	26.96	38.84	33.97	30.11	27.13
	3	39.04	34.46	30.88	27.86	39.03	34.47	30.91	27.96	38.99	34.44	30.91	27.86
	4	39.08	34.55	30.99	28.06	38.99	34.55	31.01	28.00	38.95	34.51	30.99	28.05
4	1	35.77	31.68	27.55	23.14	35.79	31.70	27.57	23.13	35.80	31.77	27.57	23.13
	2	36.32	32.87	30.10	27.37	36.33	32.94	30.16	27.37	36.35	32.94	30.17	27.38
	3	36.45	33.15	30.53	28.43	36.40	33.16	30.52	28.43	36.45	33.18	30.52	28.45
	4	36.45	33.19	30.52	28.48	36.44	33.19	30.52	28.45	36.43	33.19	30.52	28.46
5	1	35.80	31.73	27.76	17.72	35.78	31.73	27.95	17.72	35.82	31.73	27.95	17.72
	2	36.18	32.36	29.14	25.99	36.17	32.36	29.13	25.98	36.17	32.35	29.12	25.98
	3	36.26	32.45	29.48	26.79	36.24	32.46	29.47	26.78	36.19	32.50	29.44	26.74
	4	36.25	32.48	29.52	26.88	36.24	32.49	29.52	26.88	36.24	32.52	29.49	26.85

The results of our experiments are summarized in Table II where in column A the PSNR values obtained with the original *openjpeg* implementation are shown. In column B we give the results of the multiplierless 9/7 filter bank described in [14]. These results are obtained implementing the multiplierless 9/7 FB described in [14] at the encoder side and decoding the bitstream with the standard 9/7 LS decoder. In column C the results obtained with the multiplierless 9/7 LS described in [17] are shown. These results are obtained implementing the multiplierless 9/7 LS described in [17] at the encoder side and decoding the bitstream with the standard 9/7 LS decoder. We can observe that B and C grant performance very closed to the ones available with the original *openjpeg* implementation (A). As detailed in [14], [15] and [17] this relevant figure is achieved thanks to the filters zeros position that is extremely closed to the original 9/7 filters one. Furthermore, in some cases non linearities caused by the quantization and the optimal truncation performed by EBCOT produces slightly better results in terms of PSNR with B or C than with A. Since in some other cases A is better than B or C, we can conclude that the difference among the three models is very limited (from few cents to some fractions of dB). This confirms that the hardware simplification achieved through [14] and [17] multiplierless solutions does not worsen the DWT performance.

IV. PROPOSED ARCHITECTURE

The proposed architecture is based on the multiplierless data-path (Fig. 2) that implements (4) and (10), embedding the 5/3 calculation. The main idea is to consider that the downsampling required by filter bank implementation (see Fig. 1) allows to alternatively generate a low pass output sample and a high pass output sample. Considering (8) and (18), we can observe that $y_l^{5,3}$ and $z_h^{5,3}$ differ only in the sign of the w_1 term. The same consideration can be extended to $y_h^{5,3}$ and $z_l^{5,3}$ observing (13) and (17). As a consequence, exploiting these properties, during the low pass cycle both $y_l^{5,3}$ and $z_l^{5,3}$ are produced, whereas during the high pass cycle $y_h^{5,3}$ and $z_h^{5,3}$ are generated.

In order to reduce the complexity of the proposed architecture to generate the 9/7 results, we can rewrite (4) and (10) as a function of $y_l^{5,3}$, $z_l^{5,3}$, $y_h^{5,3}$, $z_h^{5,3}$ and w_i with $i \in \{0,1,2,3,4\}$:

$$y_l^{9,7} = \left(2+\frac{1}{16}\right) y_l^{5,3} - z_l^{5,3} + \\ + \frac{1}{4}\left\{\frac{1}{4}\left[\left(1-\frac{1}{8}\right)(w_0+w_2) + \left(\frac{1}{2}-\frac{1}{16}\right) w_4\right] + \right. \\ \left. + \left(w_1 - \frac{1}{16} w_3\right) + \frac{1}{2} w_2 \right\} \tag{19}$$

$$y_h^{9,7} = \left(1+\frac{1}{4}\right) z_h^{5,3} - \left(\frac{1}{4}+\frac{1}{8}\right) y_h^{5,3} + \\ + \frac{1}{4}\left\{\frac{1}{4}\left[\frac{1}{2} w_3 + \left(1+\frac{1}{4}\right) w_1\right] + \right. \\ \left. - \left(w_1 - \frac{1}{16} w_3\right) + \frac{1}{2} w_2 \right\} \tag{20}$$

In (19) and (20), the terms required to generate $y_l^{9,7}$ and $y_h^{9,7}$ are split in three main contributions written on three different lines. As an example the last contributions of $y_l^{9,7}$ and $y_h^{9,7}$ (third line) differ only in the sign of the first term. Thus the proposed architecture can be obtained by combining the three contributions and adding few multiplexers to account for $y_l^{9,7}$ and $y_h^{9,7}$. In Fig. 2, the proposed architecture is shown where solid lines represent 16 bits wide data and dashed lines represent control signals. The gray box on the top of Fig. 2 contains the adders required to generate the w_i terms. The light–gray box labeled with 5/3 contains the programmable adders/subtracters required to compute $y_l^{5,3}$, $z_l^{5,3}$, $y_h^{5,3}$ and $z_h^{5,3}$ as described in (8), (17), (13) and (18). The three gray boxes in the center of the figure implement the three contributions of (19) and (20) employing three, four and two adders respectively and few multiplexers to select the low pass or high pass contributions. In particular, the lo_hin signal

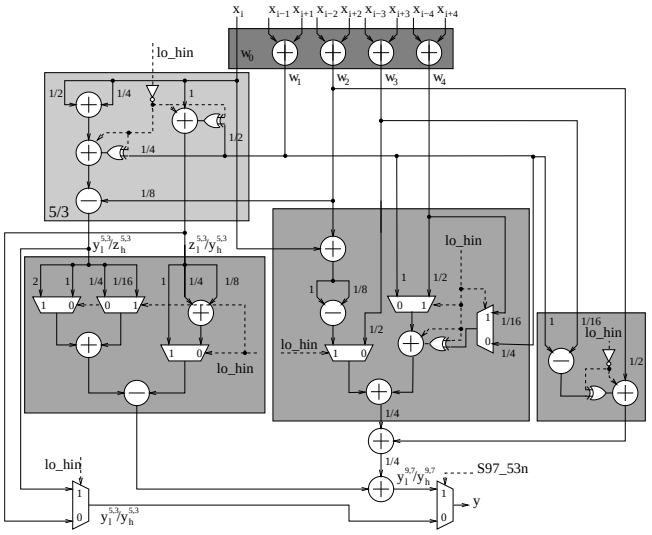


Figure 2. Proposed folded 9/7 and 5/3 FB architecture – 16 bits wide data-path

drives the multiplexers to select low pass or high pass output values whereas the $S97_53n$ signal allows to select between the 9/7 and the 5/3 results.

V. Implementation results and comparison

Table III
Comparisons of multiplierless architectures for the 9/7 filter in terms of PSNR, multipliers (M), adders (A), registers (R), kgates, post place & route average power consumption (P) and throughput (T) per cycle

Arch.	Type [dB]	PSNR	Complexity M	A	R	kgates	P [mW]	T
[10]-I	FB	A	9	14	9	14.65	35.66	1
[13]	FB	B	0	32	9	-	-	1
[11]	FB	A	0	43	9	5.39	33.90	1
[12]	FB	A	0	27	9	4.17	15.15	1
[14]	FB	B	0	21	9	2.81	12.88	1
[10]-II	LS	A	4	8	6	12.45	29.43	2
[16]	LS	A	4	8	4	10.10	-	2
[9]	LS	A	2	4	20	-	-	1
[15]	LS	C	0	19	14	7.43	11.36	2
[17]	LS	C	0	15	14	3.79	-	1
Our	**FB**	$\boldsymbol{B}$	0	**19**	**9**	**2.69**	**9.74**	**1**

The architecture described in section IV has been described in VHDL resorting to only 19 adders. The logical synthesis results on a 0.13 μm standard cell technology with a target clock frequency of 200 MHz confirmed the reduced amount of logic required (2.69 kgates) and its low power consumption (9.74 mW). In Table III, several FB and LS architectures are considered and their performance (PSNR and throughput), complexity (kgates) and power consumption summarized. If we consider multiplierless implementations [11] requires 43 adders, [13] requires 32 adders, [12] requires 27 adders, [14] requires 21 adders, whereas the proposed architecture requires only 19 adders. As far as multiplierless FB architectures are concerned, logical synthesis results summarized in Table III take into account the combinational logic (including the adders) and the register placed on the data-path output (y). On the other hand, even if the multiplierless LS architectures require only 19 [15] or 15 [17] adders, the LS serial nature imposes to use a greater number of registers compared to FB architectures, leading to an increase in terms of complexity. As a consequence, we can observe that the proposed multiplierless FB architecture with embedded 5/3 computation shows a computational complexity reduction with respect to other multiplierless solutions, together with excellent performance in terms of PSNR and a reduced power consumption. Moreover, the inherent flexibility of the proposed architecture, that supports both the 9/7 and 5/3 wavelet filters, has been obtained with no penalties in terms of complexity or performance.

It is known that several systematic methods to design complexity-aware multiplierless filters have been proposed in the literature (e.g. [20], [21], [22], [23]). Even if this work does not make use of these techniques, it is interesting to compare the number of adders required by the proposed architecture (19 adders) to the number of adders achieved with a systematic method. In [22] a heuristic search algorithm based on a genetic algorithm, called CSDC, is developed. However, said N_t the number of taps, CSDC shows to achieve best results on long filters, $N_t \geq 60$. Moreover [23], where the KMSD algorithm is proposed, proves that the n-Dimensional Reduced Adder Graph (RAG-n) algorithm [21] is best for short filters $N_t < 12$. Since this is the case of the proposed architecture, we investigate the number of adders required to implement (19) and (20) resorting to the Canonic Signed Digit (CSD) representation [20] and to the RAG-n algorithm [21]. In order to make the comparison as fair as possible we consider the 5/3 block outputs and the w_i values as inputs ($\hat{x}$) and we apply the CSD and the RAG-n methodologies to the equivalent low pass ($\hat{h}$) and high pass ($\hat{g}$) filters $y_l^{9,7} = \hat{h} \cdot \hat{x}$ and $y_h^{9,7} = \hat{g} \cdot \hat{x}$ with $\hat{h} = [33/16 \ -1 \ 7/128 \ 1/4 \ 23/128 \ -1/64 \ 7/256]$ ($\hat{h}$ length is 7) and $\hat{g} = [5/4 \ -3/8 \ 0 \ -11/64 \ 1/8 \ 3/64 \ 0]$ ($\hat{g}$ length is 5). This allows to obtain a multiplierless architecture able to support both the 9/7 and the 5/3 wavelet filters. To simplify the comparison we split the number of required adders in four contributions: 1) 4 adders to generate the w_i values, 2) 4 adders required by the 5/3 block, 3) 6 adders to combine the 7 elements produced by $\hat{h}$, 4) m adders required by the filters coefficients when CSD or RAG-n is employed (m_{CSD} or m_{RAG-n}). Since the low pass and the high pass outputs are alternatively generated, the adders required by $\hat{h}$ and $\hat{g}$ can be shared introducing few multiplexers. It can be easily obtained that $m_{CSD}^{\hat{h}} = 6$ and $m_{CSD}^{\hat{g}} = 7$, if adders sharing is employed $m_{CSD} = 7$, leading to 21 adders. On the other hand exploiting RAG-n precomputed table [24], we obtain $m_{RAG-n}^{\hat{h}} = 5$ and $m_{RAG-n}^{\hat{g}} = 5$, thus $m_{RAG-n} = 5$, leading to 19 adders. It is worth observing that $\hat{h}$ and $\hat{g}$ can be represented as cost 0, cost 1 and cost 2 RAG-n elements, so the heuristic part of the RAG-n algorithm is not employed. As a consequence the result obtained with the RAG-n algorithm is optimal. Since the proposed architecture requires the same number of adders required by the RAG-n solution, the proposed architecture is Pareto-optimal.

In order to have an accurate estimation of the power consumption figure of the proposed architecture, switching activity values are required. To this purpose, the proposed

architecture has been interfaced to an external memory and a micro controller through a FIFO and a simple control unit as depicted in Fig. 3. *LINE_LEN* is the number of processed samples (i.e the current row or column length) and *start* allows to start the elaboration, whereas a *done* is generated when *LINE_LEN* pixels have been elaborated. Besides, the *S97_53n* signal allows to switch between the 9/7 and the 5/3 filters. Finally, the architecture validates every correct output value (y) through a *valid* signal.

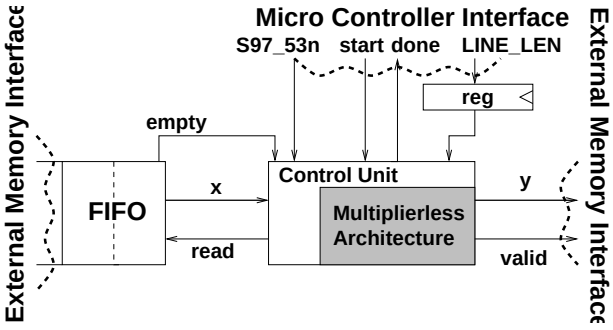


Figure 3. Proposed architecture interface and block scheme

The complete design flow (including place and route) has been performed for the proposed architecture with its control unit. Annotating the design activity on 256 samples of the standard image 'Lenna' 256×256 into `Cadence-Encounter` power estimation environment with f_{clk}=200MHz, we obtained an average power consumption of 10.27 mW.

Through VHDL simulation we observe that the proposed architecture requires 264 clock cycles to elaborate 256 samples and to generate 128 low-pass and 128 high-pass results at full speed (FIFO always ready). This means 1 sample per clock cycle while 8 additional clock cycles are required for managing the boundary extensions ($n_{lat} = 8$). Thus the proposed architecture can compute the 2D-DWT of an $R \times C$ image in $2RC + n_{lat}(R + C)$ clock cycles. We can evaluate the number of $R \times C$ frames per second (YUV 4:2:0 format) that can be elaborated by the proposed architecture as:

$$n_{frames} = \frac{f_{clk}}{\frac{3}{2}(\frac{4}{3}2RC + 2n_{lat}(R + C))} \qquad (21)$$

where f_{clk} is the clock frequency, the factor $\frac{4}{3}2RC$ and $2n_{lat}(R + C)$ account for an unlimited number of wavelet decomposition levels, the term $3/2$ for the YUV 4:2:0 format and n_{lat} for the architecture latency ($n_{lat} = 8$). Thus the proposed architecture can sustain 30 frames per seconds for HD video applications (1440×1080).

VI. Conclusions

In this paper, a novel multiplierless 9/7 wavelet VLSI architecture has been presented. The novelty of the paper stems from the use of the 5/3 wavelet results to decrease the 9/7 architecture complexity. The proposed architecture has been compared in terms of performance and complexity with other multiplierless 9/7 filter banks and lifting scheme solutions showing lower complexity with comparable performance. Finally, it is noticeable that the proposed architecture can run at 200 MHz, being able to sustain 30 frames per second of HD video sequences (1440×1080) with an average power consumption of 10.27 mW.

References

[1] G. Strang and T. Q. Nguyen, *Wavelets and Filter Banks*. Wellesley-Cambridge, MA: Wellesley, 1996.

[2] D. Taubman, "High performance scalable image compression with EBCOT," *IEEE Tran. on Image Processing*, vol. 9, no. 7, pp. 1158–1170, Jul. 2000.

[3] M. Antonini et al "Image coding using the wavelet transform," *IEEE Tran. on Image Processing*, vol. 1, no. 2, pp. 205–220, Apr. 1992.

[4] D. L. Gall and A. Tabatai, "Sub-band coding of digital images using symmetric short kernel filters and arithmetic coding techniques," in *IEEE Int. Conference on Acoustics, Speech and Signal Processing*, 1988, pp. 761–764.

[5] M. Boliek, "JPEG 2000 Final Committee Draft," 2000.

[6] H. Yamauchi et al. "1440x1080 pixel, 30 frames per second motion-JPEG 2000 codec for HD-movie transmission," *IEEE Jour. of Solid-State Circuits*, vol. 40, no. 1, pp. 331–341, Jan. 2005.

[7] O. Fatemi and S. Bolouki, "Pipeline, memory-efficient and programmable architecture for 2D discrete wavelet transform using lifting scheme," *IEE Proc. on Circuits Devices and Systems*, vol. 152, no. 6, pp. 703–708, Dec. 2005.

[8] C. T. Huang, P. C. Tseng, and L. G. Chen, "Generic RAM-based architectures for two-dimensional discrete wavelet transform with line-based method," *IEEE Tran. on Circuits and Systems for Video Technology*, vol. 15, no. 7, pp. 910–920, Jul. 2005.

[9] B. F. Wu and C. F. Lin, "A high-performance and memory-efficient pipeline architecture for the 5/3 and 9/7 discrete wavelet transform of JPEG2000 codec," *IEEE Tran. on Circuits and Systems for Video Technology*, vol. 15, no. 12, pp. 1615–1628, Dec. 2005.

[10] J. M. Jou, Y. H. Shiau, and C. C. Liu, "Efficient VLSI architectures for the biorthogonal wavelet transform by filter bank and lifting scheme," in *IEEE Int. Symposium on Circuits and Systems*, 2001, pp. 529–533.

[11] M. Alam et al. "Efficient distributed arithmetic based DWT architecture for multimedia applications," in *IEEE Int. Workshop on SoC for Real-Time Applications*, 2003.

[12] X. Cao et al. "An efficient VLSI implementation of distributed architecture for DWT," in *IEEE Workshop on Multimedia Signal Processing*, 2006, pp. 364–367.

[13] K. A. Kotteri, A. E. Bell, and J. E. Carletta, "Design of multiplierless, high-performace, wavelet filter banks with image compression applications," *IEEE Tran. on Circuits and Systems-I*, vol. 51, no. 3, pp. 483–494, Mar. 2004.

[14] M. Martina and G. Masera, "Low-complexity, efficient 9/7 wavelet filters VLSI implementation," *IEEE Tran. on Circuits and Systems-II*, vol. 53, no. 11, pp. 1289–1293, Nov. 2006.

[15] D. B. H. Tay, "A class of lifting based integer wavelet transform," in *IEEE Int. Conference on Image Processing*, 2001, pp. 602–605.

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

[17] K. A. Kotteri et al. "A comparison of hardware implementations of the biorthogonal 9/7 DWT: Convolution versus lifting," *IEEE Tran. on Circuits and Systems-II*, vol. 52, no. 5, pp. 256–260, May 2005.

[18] "http://www.openjpeg.org."

[19] M. Martina, "Low Complexity 9/7 Wavelet: Modified OpenJPEG model," downloadable at www.vlsilab.polito.it/∼martina.

[20] R. Hartley, "Optimization of canonic signed digit multipliers for filter design," in *IEEE Int. Symposium on Circuits and Systems*, 1991, pp. 1992–1995.

[21] A. G. Dempster and M. D. Macleod, "Use of minimum-adder multiplier blocks in FIR digital filters," *IEEE Tran. on Circuits and Systems-II*, vol. 42, no. 9, pp. 569–577, Sep. 1995.

[22] W. Yongtao and K. Roy, "CSDC: a new complexity reduction technique for multiplierless implementation of digital FIR filters," *IEEE Tran. on Circuits and Systems-I*, vol. 52, no. 9, pp. 1845–1853, Sep. 2005.

[23] M. D. Macleod and A. G. Dempster, "Multiplierless FIR filter design algorithms," *IEEE Signal Processing Letters*, vol. 12, no. 3, pp. 186–189, Mar. 2005.

[24] A. G. Dempster and M. D. Macleod, "Constant integer multiplication using minimum adders," *IEE Proc. on Circuits Devices and Systems*, vol. 141, no. 5, pp. 407–413, Oct. 1994.