

IMPLEMENTATION OF THE DECORRELATING TRANSFORMATION FOR LOW POWER FIR FILTERS

A.T. Erdogan, T. Arslan, and R. Lai

School of Engineering and Electronics, The University of Edinburgh
The King's Buildings, Edinburgh EH9 3JL, United Kingdom.

{ate,arslan}@ee.ed.ac.uk

ABSTRACT

This paper presents the implementation of the decorrelating (DECOR) transformation technique for low power FIR filtering cores. The technique was introduced in the past, but was not fully evaluated for its area, delay and power performance. Early evaluations did not consider the whole implementation and were merely based on either some analytical methods or high level simulation models. This paper presents the complete VLSI implementation of the technique and a study of its area, delay and power performance with different order of coefficient differences and various multiplier types. We show that although the technique achieves up to 47% power saving in the multiplier unit, the overall power saving is up to 25% with up to 24% increase in area.

I. INTRODUCTION

Due to the popularity of the portable battery-powered devices in recent years, there is a high demand for low power and high performance devices, especially in wireless communication systems such as cellular phones, pagers, and wireless modems. These devices require efficient and flexible cores for the implementation of complex algorithms which in turn involve recursive and repetitive implementation of various FIR filters on these cores with very constrained power and area demands. Intuitively, there are two approaches, sequential and parallel, to implement FIR filters. The sequential implementation can be cost and area-effective in hardware but its bottleneck is low throughput; hence it is not suitable for high performance applications. On the other hand, the parallel implementation can maximize the throughput with some additional hardware, such as multipliers and adders.

Furthermore, power dissipation is becoming a crucial factor in the efficient realization of FIR filters. There is

increasing number of published techniques to reduce power consumption of FIR filters. The authors in [1] optimize word-lengths of input/output data samples and coefficient values. This involves using a general search based methodology, which is based on statistical precision analysis and the incorporation of cost/performance/power measures into an objective function through word-length parameterisation. In [2], Mehendale et al. presents an algorithm for optimizing the coefficients of an FIR filter, so as to reduce power consumption in its implementation on a programmable digital signal processor. The use of coefficient segmentation, block processing and combined segmentation and block processing algorithms for low power FIR filter implementations has been shown in [3]. The authors in [4] and [5] have introduced high throughput FIR filter implementations.

Most realizations of FIR filters use the coefficients directly to compute the convolution with the input data. The differential coefficients method (DCM) introduced in [6] uses various orders of differences between the coefficients along with stored intermediate results rather than the actual coefficients themselves to compute the convolution. If fewer bits are required to represent the differences compared to the actual coefficients, the size of the arithmetic units in the filter realization can be reduced, thereby reducing the power consumption. However, the method comes with the overheads of $N-1$ additional latches (to store the intermediate results) and $N-1$ additional adders (to add those intermediate results) for an N tap filter. Although, greater orders of differences have smaller magnitudes, the overheads required by DCM increase as the order of differences increased. Therefore, there is a point beyond which the gains (due to smaller magnitudes) are less than the net cost of overheads [6]. To minimize the overhead while retaining the benefit of DCM, differential coefficient and input method (DCIM) [7] and decorrelating (DECOR) transformations [8] have been proposed. Some of the advantages of DECOR over DCM were listed as (a) lower overheads for a given filter order, (b) overheads being independent of the filter order, and (c) power savings over a wider range of filter bandwidths.

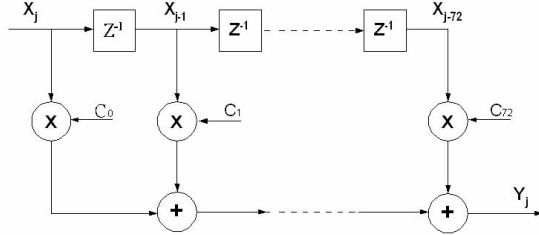


Fig. 1. 73-tap direct form FIR filter structure.

However, the effectiveness (in terms of power, speed and area) of the above methods was evaluated analytically using some high level models of multipliers, adders and storage units. In [8], in addition to the analytical results, simulation based results were also presented. However, these results were also not based on real VLSI implementation of the filters, but based on some C simulation models, assuming zero and/or unit delays for the circuit gates. Therefore, for a more realistic evaluation, in this work we have implemented DECOR in VLSI, targeting a 0.18 micron standard cell CMOS technology. The performance analysis in term of power, speed and area was made based on full implementation of DECOR, for different multiplier types and order of coefficient differences.

II. IMPLEMENTATION

An N-tap FIR filter performs the following convolution:

$$Y_j = \sum_{k=0}^{N-1} C_k X_{j-k} \quad (1)$$

where C_k 's are the coefficients of the filter, X_j and Y_j are the j th terms of the input and output sequences, respectively. According to (1), the direct form (DF) structure of FIR filter is shown in Fig. 1.

The z-transfer of (1) is given below:

$$Y(z) = H(z)X(z) \quad (2)$$

where $Y(z)$, $H(z)$, and $X(z)$ are the z-transforms of the output, filter, and input respectively. In DECOR, the transfer function $H(z)$ is multiplied and divided by the polynomial [8]:

$$T(z) = (1 + \alpha z^{-\beta})^m \quad (3)$$

where m represents the order of coefficient difference, α and β are parameters chosen depending on the type of FIR

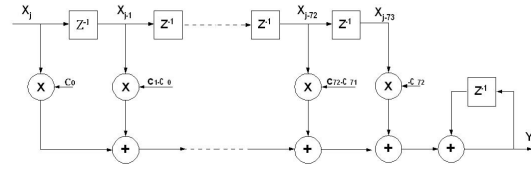


Fig. 2. DECOR FIR filter structure using first order differential coefficients.

filter. The frequency response is not altered by multiplying and dividing the transfer function $H(z)$ by this polynomial. For example, the z-transfer of the first order lowpass FIR filter is given by ($\alpha = -1$, $\beta = 1$, $m = 1$):

$$Y(z) = \frac{(1 - z^{-1})H(z)}{1 - z^{-1}} X(z) \quad (4)$$

$$Y(z) - Y(z)z^{-1} = H(z)X(z) - H(z)X(z)z^{-1} \quad (5)$$

According to Eq.1 and Eq.5 the transformed filter can be expressed as:

$$Y_j - Y_{j-1} = \sum_{k=0}^{N-1} C_k X_{j-k} - \sum_{k=0}^{N-1} C_k X_{j-k-1} \quad (6)$$

Re-arranging Eq.6 we can obtain the following equation for first order ($m=1$) differential coefficients:

$$Y_j = C_0 X_j + \sum_{k=1}^{N-1} (C_k - C_{k-1}) X_{j-k} - C_{N-1} X_{j-N} + Y_{j-1} \quad (7)$$

Clearly, as Eq.7 shows, for first order differential coefficients, the filter outputs can be obtained using the differences between adjacent coefficients (except for first and last coefficients) and the previous filter output. Also note that the transformed filter requires one additional multiplication and subtraction operation in order to realize the term $(-C_{N-1} X_{j-N})$ in Eq.7. Therefore, this together with adding the previous filter output represents the overhead for DECOR using first order differential coefficients. Fig. 2. illustrates the first order DECOR FIR filter structure for a 73-tap lowpass FIR filter.

Similarly, it can be shown that DECOR filters for 2nd and 3rd order differential coefficients can be expressed with Eq.8 and Eq.9 respectively.

$$Y_j = C_0 X_j + (C_1 - 2C_0) X_{j-1} + \sum_{k=2}^{N-1} (C_k - 2C_{k-1} + C_{k-2}) X_{j-k} + (C_{N-2} - 2C_{N-1}) X_{j-N} + C_{N-1} X_{j-N-1} + 2Y_{j-1} - Y_{j-2} \quad (8)$$

$$\begin{aligned}
Y_j = & C_0 X_j + (C_1 - 3C_0) X_{j-1} + (C_2 - 3C_1 + 3C_0) X_{j-2} \\
& + \sum_{k=3}^{N-1} (C_k - 3C_{k-1} + 3C_{k-2} - C_{k-3}) X_{j-k} \\
& + (-3C_{N-1} + 3C_{N-2} - C_{N-3}) X_{j-N} + (3C_{N-1} - C_{N-2}) X_{j-N-1} \\
& + C_{N-1} X_{j-N-2} + 3Y_{j-1} - 3Y_{j-2} + Y_{j-3}
\end{aligned} \tag{9}$$

Clearly, as the order of coefficient differences increased the overhead in terms of additional multiplications and additions is also increased.

A. Conventional FIR Filter Core

In order to evaluate the performance of the DECOR FIR filter we have first implemented a conventional FIR filter core based on Eq.1. There are eight blocks in this conventional FIR filter core implementation, as shown in Fig. 3. It consists of two memory blocks for storing the input data (X_RAM) and the coefficients (B_COEFF_ROM), two registers for holding the input data (BETA_MEM) and coefficient (GAMMA_MEM), the control block (CONTROL), the rounding block (ROUND) for rounding the filter output in order to fit the output word length, the output register (OUT_STORE) for storing the output data, and the main arithmetic block (MAC). A brief description of these blocks is given below:

- **CONTROL:** The controller is based on a counter and it is responsible for generating every control signal to synchronize the activity of each block in the filter.
- **X_RAM:** This is the RAM memory for the storage of the input data. There are three parts for the RAM memory. They are memory bank, demultiplexer and multiplexer. Memory bank is a bank of 16-bit registers implemented in the form of a latch based circular buffer to reduce its power consumption.
- **B_COEFF_ROM:** This is the ROM memory for the storage of the coefficients of the filter.
- **GAMMA_MEM:** This is a 16-bit register implemented in the form of a flip-flop based circuit for synchronization with clock.
- **BETA_MEM:** This is a 16-bit register implemented in the form of a flip-flop based circuit for synchronization with clock.
- **MAC:** It consists of a multiplier (mult), an adder (add), an accumulator (acc) and a clearing logic (clacc) in the form of a multiplexer. The clacc unit

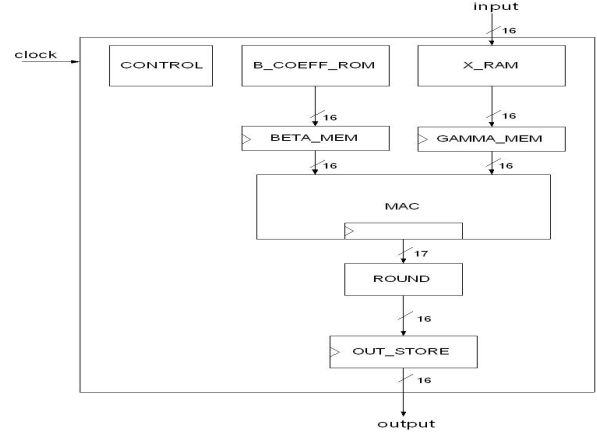


Fig. 3. Block diagram of the conventional FIR filter core.

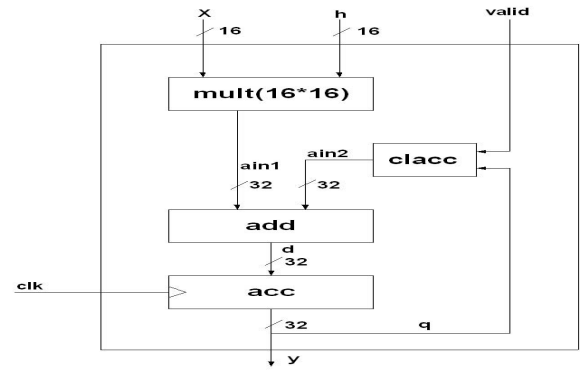


Fig. 4. MAC block diagram for the conventional FIR filter core.

carries out the dual operation of feeding the accumulated values back to the adder and also for clearing the accumulator when the valid input to the MAC is asserted high. The valid input is asserted high after the generation of a filter output. The MAC is used to multiply an input data with a coefficient and adding the previous stored accumulator register value to the product at the same clock period. Data samples and the filter coefficients are represented as 16-bit two's complement numbers whereas the MAC output is 32 bits. A block diagram of MAC is shown in Fig. 3.

- **ROUND:** It is used to round the MAC output to fit the filter output word length.
- **OUT_STORE:** This is a 16-bit register implemented in the form of a flip-flop based circuit for synchronization with the clock.

B. DECOR FIR Filter Core

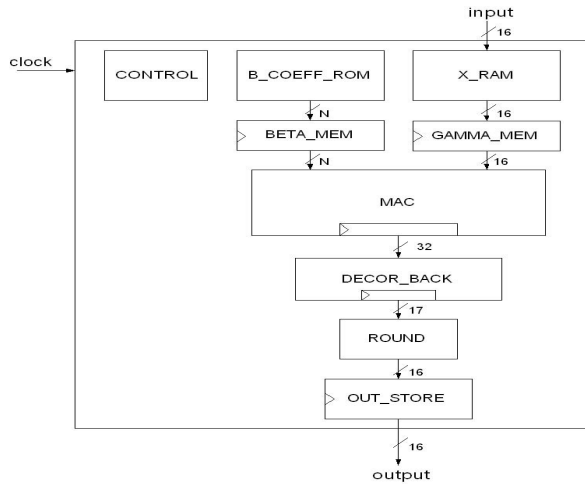


Fig. 5. Block diagram of DECOR FIR filter core.

DECOR FIR filter implementation is based on the conventional filter implementation, as shown in Fig. 4. It uses the same blocks of the conventional one with little modifications. The DECOR FIR has only one extra block, DECOR_BACK, for implementing the additional terms in the DECOR FIR filter equations 7, 8 and 9. This represents part of the overhead. The other part of the overhead is in the MAC block due to the extra multiplications and additions required by DECOR. The DECOR_BACK block consists of a number of registers for storing the previous filter outputs (Y_{j-1} , Y_{j-2} , etc.) and adder/subtractor units for adding a combination of previous filter outputs ($[2Y_{j-1}-Y_{j-2}]$, $[3Y_{j-1}-3Y_{j-2}+Y_{j-3}]$, etc.) to the current filter output, depending on the value of m (order of coefficient difference) used. Note that, $m=1$ presents a special case in which only the previous filter output (Y_{j-1}) is required to be added to the current filter output. Therefore, this can be realized more efficiently by removing the accumulator clearing logic (*clacc*) in the MAC block. Thus, first order DECOR does not need the DECOR_BACK block and uses a simpler MAC block.

There is a slight difference in the multiplier unit in the MAC block. Due to the smaller coefficient range, a smaller port size for the coefficient input of the multiplier is used. Therefore, the multiplier size is 16 by N, where N indicates the word length of the coefficients and the data word length is 16-bits. Therefore, a smaller size multiplier is used compared to the multiplier used for the conventional filter core.

III. RESULTS

Two different FIR cores namely conventional (CON), and DECOR have been implemented and analyzed with regard to area usage, delay and power consumption. DECOR has been implemented for 1st, 2nd, 3rd and 4th order coefficient differences in order to study the impact of coefficient difference order on area, delay and power parameters. The cores were designed using Verilog HDL and then synthesized using Synopsys Design Compiler™ targeting the UMC 0.18μ standard cell CMOS library. The requirements for the synthesis were identical for all the cores. This was necessary in order to allow for a consistent delay, power consumption and area usage comparisons. A maximum circuit delay of 10ns has been defined for all the cores. A layout for each core was generated using the Envisia™ Silicon Ensemble™ place-and-route software. This was followed by extracting RC information and then performing RC back-annotated post-layout gate-level netlist simulations for a uniformly distributed random input data sample set equal to 1000 using Verilog-XL™ simulator. The resulting data, including switching activity of the circuit nets and the capacitive load information extracted from the layouts, was then used by Synopsys Design Power™ to compute power consumption for the different FIR cores. In all of the above stages a clock rate of 100 MHz and a supply voltage of 1.62 Volts were used.

Results are shown in Tables 1 and 2 for a 73-tap low-pass FIR filter with a cutoff frequency of $\pi/10$. Performance analysis was carried out using three different multiplier types from Synopsys DesignWare namely *csa*, *nbw* and *wall*. Table 1 shows that in general for all three multiplier types the percentage saving in power of the multiplier

Table 1: Power consumption analysis for different multipliers

Algorithm	csa		nbw		wall	
	[mW]	[%]	[mW]	[%]	[mW]	[%]
CON	7.35	-	3.59	-	3.55	-
DECOR_1st	6.74	8.3	3.36	6.2	3.07	13.3
DECOR_2nd	5.25	28.5	2.36	34.2	2.65	25.2
DECOR_3rd	4.34	40.9	2.10	41.3	2.31	35.0
DECOR_4th	4.67	36.5	2.22	38.1	1.89	46.8

Table 2: Power consumption analysis for FIR cores

Algorithm	csa		nbw		wall	
	[mW]	[%]	[mW]	[%]	[mW]	[%]
CON	14.48	-	9.87	-	9.96	-
DECOR_1st	14.06	2.9	9.61	2.6	9.41	5.5
DECOR_2nd	11.90	17.8	8.18	17.1	8.61	13.5
DECOR_3rd	10.80	25.4	7.61	22.9	7.96	20.1
DECOR_4th	11.46	20.8	7.76	21.4	7.48	25.0

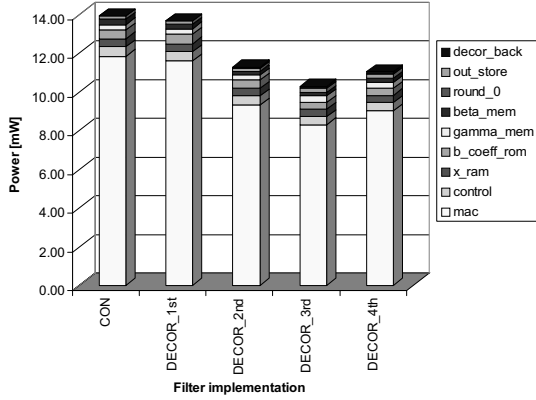


Fig. 6. Power consumption with *csa* multiplier.

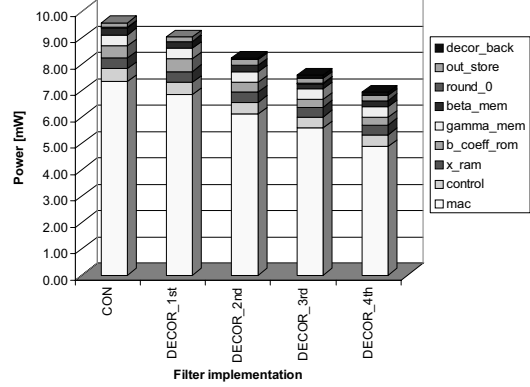


Fig. 8. Power consumption with *wall* multiplier.

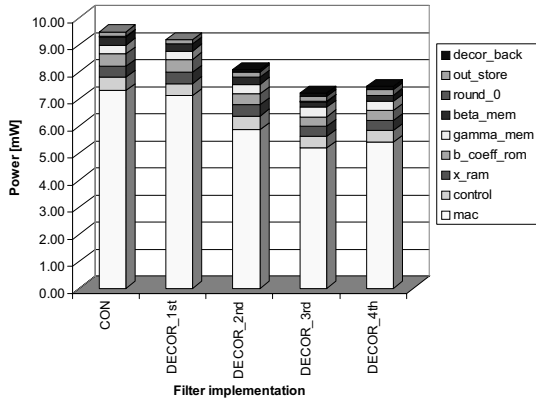


Fig. 7. Power consumption with *nbw* multiplier.

From Fig.6 it is clear that the MAC contributes most to the power consumption in all filter cases, with a maximum of 82% for CON. The remaining 18% is consumed by control (3%), x_ram (3%), coeff_rom (3%), beta_mem (2%), gamma_mem (2%), and the rest circuit (5%). Similarly, MAC consumes 83%, 79%, 78%, and 78% of the whole power for DECOR_1st, DECOR_2nd, DECOR_3rd, and DECOR_4th respectively. DECOR_BACK is responsible only for 1% of the power in all cases. As was explained in Section II.B there is no DECOR_BACK for DECOR_1st case. Power consumption contributions of the filter blocks for different filter cores using *nbw* and *wall* multiplier types are also shown in Fig.7. and Fig.8. respectively.

Fig.9. illustrates the area overhead for different filter cores and multiplier types. In general, the profile is similar for all three multipliers. DECOR_1st achieves 1-2% area

increases with the increase of coefficient difference order ($m=1, 2,$ and 3), except for $m=4$. In this case DECOR cannot reduce the multiplier power any more for *csa* and *nbw* type multipliers. This trend can also be observed for overall FIR core implementations, as shown in Table 2. For example, the best power saving (25.4%) with *csa* is achieved using DECOR_3rd, followed by DECOR_4th (20.8%), DECOR_2nd (17.8%), and DECOR_1st (2.9%). Fig. 6 illustrates the power consumption of all the blocks for different filter cores using a *csa* multiplier. For *nbw* multiplier case, a similar trend is observed; the best saving is achieved with DECOR_3rd (22.9%), followed by DECOR_4th (21.4%), DECOR_2nd (17.1%), and DECOR_1st (2.6%). However, for *wall* multiplier case the best power saving is achieved by DECOR_4th (25%), indicating a possible further power saving with the increase of coefficient difference order for this case.

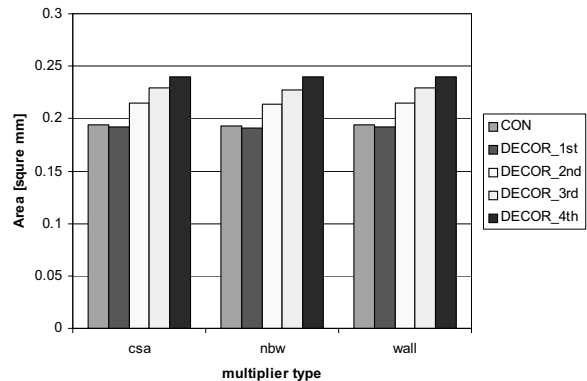


Fig. 9. Area comparisons.

saving due to reduced multiplier size and its architecture not requiring the DECOR_BACK block. However, area increases by 10%, 18% and 24% for DECOR_2nd, DECOR_3rd, and DECOR_4th cases respectively. This is mainly due to the DECOR_BACK block.

Delay analysis for all filter cores using different multipliers is given in Table 3. It is clear from the table that, in general, the delay reduces slightly with DECOR_1st (except for *csa*) and increases for other cases, again due to the DECOR_BACK block.

Table 3: Delay comparisons for FIR cores [ns]

Algorithm	<i>csa</i>	<i>nbw</i>	<i>wall</i>
CON	8.25	6.70	6.71
DECOR_1st	8.41	6.27	6.68
DECOR_2nd	9.09	7.49	7.24
DECOR_3rd	9.11	7.37	7.49
DECOR_4th	8.71	7.69	7.69

IV. CONCLUSION

This paper has presented the complete VLSI implementation of the DECOR transformation technique for low power FIR filtering cores. The technique was implemented for different order of coefficient differences using various multiplier types. We have shown that although the technique achieved up to 47% power saving in the multiplier unit, the overall power saving is up to 25%. In general, the area increased up to 24% with the increase of coefficient difference order, except for first order differences where 1-2% area saving was achieved. In most cases, the circuit delays were also slightly increased due to the overheads. The overall power saving was dependent on the multiplier type employed.

REFERENCES

[1]. H. Choi and W.P. Burleson: "Search-Based Wordlength

Optimisation for VLSI/DSP Synthesis", VLSI Signal Processing, vol. 7, pp. 198-207, 1994.

[2]. M. Mehendale, S. D. Sherlekar, G. Venkatesh: "Low Power Realization of FIR filters on Programmable DSP's", IEEE Trans. on VLSI Systems, Vol. 6, No. 4, pp. 546-553, Dec. 1998.

[3]. A.T. Erdogan, M. Hasan and T. Arslan: "Algorithmic Low Power FIR Cores", IEE Proceedings - Circuits, Devices and Systems, Volume 150, Number 3, June 2003, pp. 155-160.

[4]. A.T. Erdogan, T. Arslan: "Low power implementation of high throughput FIR filters", IEEE Int. Conf. on Circuits and Systems, pp. 373-376, May 2002.

[5]. D.A. Parker, K.K. Parhi: "Area-efficient parallel FIR digital filter implementations", Application Specific Systems, Architectures and Processors, pp.93-111, Aug. 1996.

[6]. N. Sankarayya, K. Roy, and D. Bhattacharya: "Algorithms for Low Power and High Speed FIR Filter Realisation Using Differential Coefficients", IEEE Trans. on Circuits and Systems-II: Analog and Digital Signal Processing, vol. 44, pp. 488-497, June, 1997.

[7]. T-S Chang, Y-H Chu, and C-W Jen: "Low Power FIR Filter Realization with Differential Coefficients and Inputs", IEEE Trans. on Circuits and Systems-II: Analog and Digital Signal Processing, vol. 47, no. 2, pp. 137-145, Feb., 2000.

[8]. S. Ramprasad, N.R. Shanbhag, and I.H. Hajj: "Decorrelating (DECOR) Transformations for Low Power Digital Filters", IEEE Trans. on Circuits and Systems-II: Analog and Digital Signal Processing, vol. 46, no. 6, pp. 776-788, June, 1999.