AN ORDER BASED SEGMENTATION ALGORITHM FOR LOW POWER IMPLEMENTATION OF DIGITAL FILTERS

A. T. Erdogan and T. Arslan

University of Edinburgh,
Department of Electronics & Electrical Engineering,
Edinburgh EH9 3JL, Scotland, United Kingdom.

Ahmet.Erdogan@ee.ed.ac.uk   Tughrul.Arslan@ee.ed.ac.uk

ABSTRACT

The paper presents a new algorithm for low power implementation of digital filters. The algorithm reduces power consumption through a two phased strategy, which targets reducing switched capacitance within the multiplier circuit. The first phase involves the segmentation of coefficients into more primitive components which could in turn be processed through a single shift and a more *primitive* mutiplication operations. The second phase exploits the correlation among the new set of coefficients at the coefficient input of the multiplier for more reduction in switched capacitance. The paper describes the algorithm and its evaluation environment, and provides results with a number of filter examples demonstrating up to 65% reduction in power compared to conventional filtering.

1. INTRODUCTION

Power is increasingly becoming a dominant factor in the implementation of multiplication intensive DSP functions. The principal reasons for this are maximum operating temperature and, for portable applications, battery life.

It can be shown that the main source of power dissipation, in a typical CMOS logic gate, is due to switching power, P_{sw}, given by [1]:

$$P_{sw} = \tfrac{1}{2} k . C_{load} . V_{dd}^2 . f \qquad (1)$$

where V_{dd} is the supply voltage, f is the clock frequency, C_{load} is the load capacitance of the gate, and k is the switching activity factor which is defined as the average number of times the gate makes an active transition in a single clock cycle. Therefore, for achieving low-power in CMOS circuits one must target minimising one or more of the parameters V_{dd}, C_{load}, and k.

Techniques and methodologies for low power implementation of digital filters are emerging in the literature. These include the following: use of various orders of differences between coefficients along with stored intermediate results rather than using the coefficients themselves directly for computing the partial products in the FIR equation [2], wordlength optimisation [3], block processing [4], and the introduction of dynamic programming to assist in the search for an optimal ternary coefficient set [5].

The authors have proposed coefficient ordering [6] and coefficient segmentation [7] algorithms for low power implementation of FIR filters. Both algorithms reduce power by reducing the amount of switched capacitance within the DSP hardware platform utilised for filter implementation. Coefficient ordering reduces switched capacitance within the multiplier unit. Whereas, coefficient segmentation reduces switched capacitance within the multiplier unit in addition to a reduction in the effective wordlength of the coefficient input to the multiplier. Such techniques have also been extended to other applications, such as the DCT [8].

In this paper, we propose a new algorithm, which utilises coefficient segmentation and ordering in a two phase framework which combines their respective advantages for more reduction in power. The first phase segments coefficients into more primitive components which could be processed through a single shift and a more primitive multiplication operations. This reduces the switching activity at the coefficient input of the multiplier. The second phase exploits correlation among the new set of coefficients (produced after segmentation at the first phase above) at the coefficient input of the multiplier for more reduction in switching activity. This is achieved by re-ordering coefficients such that the Hamming distance between consecutive coefficients is minimised. The use of a transposed direct form (TDF) realisation structure in conjunction with the above algorithm can results in further reduction in power. It is well known that this realisation structure results in less switching activity at the data inputs of the multiplier [9]. The accumulative effect of using coefficient segmentation together with coefficient ordering and TDF structure results in up to 65% reduction in power consumption compared to conventional filtering. The algorithm can be used for both programmable and custom DSP hardware implementations.

2. IMPLEMENTATION

The algorithm commences by processing the coefficient set through a segmentation algorithm [7]. The algorithm segments the coefficients into two primitive parts. The first part, s_k, is processed through a barrel shifter and the remaining part, m_k, is applied to the hardware multiplier unit. The algorithm performs the segmentation through selecting a value for s_k which leaves m_k to be the smallest possible positive number. This has an effect in eliminating switching between positive and negative number values and reducing effective wordlength of coefficients. As a result, the switching activity at the coefficient input of the multiplier is significantly reduced. Details of this algorithm can be found in [7].

A further reduction in switching activity at the coefficient input of the multiplier can be achieved by implementing the filter such that respective data samples are multiplied with filter coefficients in a non-sequential order of their respective filter stages. The ability to re-order multiplication operations is desirable since it provides the potential for the switching activity at the coefficient input of the multiplier to be reduced further by minimising the *Hamming distance* between those filter coefficients used in successive multiplication operations [9]. It must be noted that the ordering of coefficients is performed only once prior to the commencement of filtering. Subsequent use of the filter will utilise the same order of coefficients. For this reason, coefficient ordering has no implications on the speed of the filtering procedure.

The sequence of steps for the implementation of this algorithm can be summarised as follows:

(1) segment coefficients into two parts; one for shifter, s(k), and one for multiplier, m(k).

(2) apply the coefficient ordering algorithm to the set of m(k) values in order to produce the re-ordered set of m(k)'s, (we will name this as m'(k)).

(3) clear the accumulator and initialise index k to 0.

(4) get the data sample, x(n-k), and apply this to data inputs of the multiplier and shifter units,

(5) get the multiplier part, m'(k), and apply this to the coefficient input of the multiplier (this will multiply x(n-k) and m'(k)),

(6) get the corresponding shifter part, s'(k), and apply this to the control inputs of the shifter (this will shift x(n-k) by the amount specified by s'(k)),

(7) sum the results of (5) and (6) above,

(8) add the result of (7) to the accumulator,

(9) increment k and repeat steps (4) to (8) for the remaining coefficients,

(10) get the filter output, y(n), from the accumulator,

(11) increment n and repeat steps (3) to (10) for the next filter output, y(n+1).

The switching activity at the data input of the multiplier can also be reduced by utilising a TDF structure implementation, since the data input will remain unchanged for a substantial number of multiplication operations, corresponding to the filter length. This in turn results in reducing the switching activity at both multiplier inputs. Using the TDF realisation, however, requires a number of intermediate values, which we term *Pre-Calculated Values* (PCVs), corresponding to the outputs of the respective filter stages. The PCV for a filter stage k at time instant n is given by:

$$PCV_k = m'(k)x(n) + s'(k)x(n) + PCV_{k+1} \qquad (2)$$

When coefficients are processed in non-sequential order two possible situations may arise:

(a) the previous filter stage, *stage k-1*, has been evaluated in a previous cycle, in which case a single PCV is required to keep the output of stage k as shown in equation (2).

(b) *stage k-1* has not been evaluated yet, in which case two PCVs will be required, one to preserve the previous output value of *stage k* (see equation (3)) for the evaluation of *stage k-1*, and another to keep the new output value of *stage k* (see equation (4))

$$PCV_k = PCV_k' \qquad (3)$$

$$PCV_k' = m'(k)x(n) + s'(k)x(n) + PCV_{k+1} \qquad (4)$$

The sequence of steps for the implementation of the above algorithm with TDF realisation can be summarised as follows:

(1) segment coefficients into two parts; one for shifter, s(k), and one for multiplier, m(k).

(2) apply the coefficient ordering algorithm to the set of m(k) values in order to produce m'(k).

(3) get the data sample, x(n), and apply this to the data inputs of the multiplier and shifter units,

(4) initialise index k to 0,

(5) get the multiplier part, m'(k), and apply this to the coefficient input of the multiplier,

(6) get the corresponding shifter part, s'(k), and apply this to the control inputs of the shifter,

(7) get the output of the next filter stage for the previous data sample, PCV_{k+1},

(8) sum the results of (5), (6), and (7) above,

3256

(9) if situation (a), see above, arises then save the result of (8) as PCV_k. Otherwise, situation (b), first save PCV'_k as PCV_k and then save the result of (8) as PCV'_k.

(10) increment k and repeat steps (5) to (9) for the remaining coefficients,

(11) get the filter output, $y(n)$, from PCV_0,

(12) increment n and repeat steps (3) to (11) for the next filter output, $y(n+1)$.

3. SIMULATIONS AND RESULTS

To demonstrate our results, we have implemented a two's complement (Baugh-Wooley) array multiplier which was selected as an example of a commonly used multiplier in DSP implementation and research. 8x8, 16x16, and 24x24-bit multipliers were implemented using Cadence VLSI suite with 0.7 μm CMOS technology. The multipliers were constructed using *AND, OR, XOR* and *INVERTER* gates. Coefficient sets were obtained by designing eight practical FIR filters with the Remez exchange algorithm developed by Parks and McClellan [10]. These are: (1) A low-pass filter with a filter order of N=24. (2) A band-pass filter with N=32. (3) A band-pass filter with N=50, in which unequal weighting is used in the two stop-bands. Thus the peak error in the upper stop-band is ten times smaller than the peak error in the lower stop-band. (4) A band-stop filter for N=31 and with equal weighting in both pass-bands. (5) A five-band filter for N=55 with three stop-bands and two pass-bands. The weighting in each of the stop-bands is different, making the peak approximation error differ in each of these bands. (6) A full band differentiator for N=32 and the peak approximation error = 0.0062. (7) A Hilbert transformer for N=20 and the peak approximation error = 0.02, where the upper cutoff frequency is 0.5 and the lower cutoff frequency is 0.05. (8) A band-pass filter with an arbitrary weighting function and for N=128. These benchmark examples were obtained from [10]. The coefficient sets were quantised to 8, 16, and 24-bits. This was followed by generating zero mean uniformly distributed data samples for each filter. Next the coefficient sets were processed by the segmentation algorithm in order to produce s_k and m_k values for each coefficient. This was followed by generating input simulation files, in which the generated input data samples were associated with the corresponding m_k values for a given filter, for the Cadence's Verilog-XL^{TM} digital simulator. Verilog-XL uses a hardware description language (Verilog HDL) form of the multiplier circuit for the simulation procedure. For each simulation the number of signal transitions for each gate was monitored. Capacitive information (wiring and loading capacitances) for each gate was extracted by performing a layout of the multiplier circuit. Both capacitive information and the switching activity figure were used to obtain the switched capacitance of each gate. This was then accumulated to give an overall figure for the switched capacitance of the multiplier.

The average results for the eight filter examples are shown in Table 1 for different multiplier sizes. The fourth column in the table shows the switched capacitance of the multiplier. Overhead estimation [7] mainly due to shifter, PCVs, data and coefficient memory bus activities are shown in the fifth column of the table. Column six is the sum of the fourth and the fifth columns, indicating the total switched capacitance. Power reduction figures are obtained by comparing switched capacitance values in the sixth column of the table to that of conventional filtering (C), where data and coefficient values are directly applied to the multiplier inputs, and a DF structure realisation is used. The table illustrates the amount of switched capacitance (pF/sample) and the percentage reduction in switched capacitance for the following cases:

(a) Conventional filtering with a DF structure,
(b) Coefficient ordering (O) with a DF structure,
(c) Coefficient segmentation (S) with a DF structure,
(d) Coefficient ordering and coefficient segmentation with a DF structure,
(e) Conventional filtering with a TDF structure,
(f) Coefficient ordering with a TDF structure,
(g) Coefficient segmentation with a TDF structure, and
(h) Coefficient ordering and coefficient segmentation with a TDF structure.

In the case of an 8x8-bit multiplier and the DF structure realisation, coefficient ordering and coefficient segmentation algorithms achieve reductions of 20.40% and 59.659% respectively. When they are used together, however, the reduction is increased to 62.31%. On the other hand, with the TDF structure implementation the conventional filtering, coefficient ordering, and coefficient segmentation each achieve reductions of 17.96%, 45.68%, and 65.63% respectively. When coefficient segmentation is used together with coefficient ordering, although the switched capacitance of the multiplier is reduced from 45 to 29 the overall switched capacitance is increased from 155 to 177 due to the increase in overhead (from 110 to 148), resulting in 60.75% reduction. Therefore, for 8x8-bit multiplier the best power reduction is achieved when coefficient segmentation is used alone with the TDF structure realisation. For 16x16-bit and 24x24-bit multiplier cases, however, the best reductions (63.82% and 55.10% respectively) are achieved when coefficient segmentation is used together with coefficient ordering and the TDF realisation.

4. CONCLUSIONS

In this paper we have presented an algorithm for low power implementation of FIR filters. The algorithm reduces power through reducing the effective switched capacitance due to multiplication operations. Results have been produced using a number of practical filter examples with various realisation structures and wordlength values. Up to 65% power reduction has been reported including effects

associated with the algorithm. The algorithm provides the potential for more significant reductions in power since it can be exploited with more powerful segmentation and multiplication sequencing algorithms.

5. REFERENCES

[1] Chandrakasan, A.P. and Brodersen, R.W.: "Minimising Power Consumption in Digital CMOS Circuits", *Proceedings of IEEE*, 1995, vol. 83, no. 4, pp. 498-523.

[2] Sankarayya, N., Roy, K. and Bhattacharya, D.: "Algorithms for Low Power and High Speed FIR Filter Realisation Using Differential Coefficients", *IEEE Transactions on Circuits and Systems-II: Analog and Digital Signal Processing*, 1997, vol. 44, no. 6, pp. 488-497.

[3] Choi, H. and Burleson, W.P.: "Search-Based Wordlength Optimisation for VLSI/DSP Synthesis", *VLSI Signal Processing VII*, IEEE Press, 1994, pp. 198-207.

[4] Arslan, T. and Erdogan, A.T.: "Data Block Processing for Low Power Implementation of Direct Form FIR Filters on Single Multiplier CMOS DSPs", *IEEE Int. Symposium on Circuits and Systems (ISCAS'98)*, May 1998, Monterey, California, USA, pp. D441-D444.

[5] Hezar, R. and Maisetti, V:K.: "Low-Power Digital Filter Implementations Using Ternary Coefficients", *VLSI Signal Processing IX*, IEEE Press, 1996, pp.179-188.

[6] Erdogan, A.T. and Arslan, T.: "Low power multiplication scheme for FIR filter implementation on single multiplier CMOS DSP processors", *IEE Electronics Letters*, 1996, vol. 32, no. 21, pp. 1959-60.

[7] Erdogan, A.T. and Arslan, T.: "A Coefficient Segmentation Algorithm for Low Power Implementation of FIR Filters", *IEEE Int. Symposium on Circuits and Systems (ISCAS'99)*, May 1999, Orlando, Florida, USA, pp.359-362.

[8] Masupe, S. and Arslan, T.: "A Low Power Implementation Approach of the DCT on VLSI DSP Processors", *IEEE Int. Symposium on Circuits and Systems (ISCAS-99)*, May 1999, Orlando, Florida, USA.

[9] Erdogan, A.T. and Arslan, T.: "Low Power Implementation of Linear Phase FIR Filters for Single Multiplier CMOS Based DSPs", *IEEE Int. Symposium on Circuits and Systems (ISCASS'98)*, May 1998, California, USA, pp. D425-D428.

[10] Mcclellan, J.H., Parks, T.W., and Rabiner, L.R.: "A Computer Program for Designing Optimum FIR Linear Phase Digital Filters", *IEEE Trans. on Audio and Electroaccoustics*, 1973, vol. AU-21, no. 6, pp. 506-526.

Table 1: Power reduction results including overheads

Multiplier size	Algorithm	Structure	multiplier	overhead	Total	Reduction (%)
8x8-bit	C	DF	380	71	451	
	O	DF	307	52	359	20.40
	S	DF	130	52	182	59.65
	O&S	DF	121	49	170	62.31
	C	TDF	238	132	370	17.96
	O	TDF	91	154	245	45.68
	S	TDF	45	110	155	65.63
	O&S	TDF	29	148	177	60.75
16x16-bit	C	DF	2722	155	2877	
	O	DF	2328	121	2449	14.88
	S	DF	1294	153	1447	49.70
	O&S	DF	1123	127	1250	56.55
	C	TDF	1996	282	2278	20.82
	O	TDF	1078	392	1470	48.91
	S	TDF	961	273	1234	57.11
	O&S	TDF	650	391	1041	63.82
24x24-bit	C	DF	9756	238	9994	
	O	DF	8766	198	8964	10.31
	S	DF	6384	273	6657	33.39
	O&S	DF	5876	237	6113	38.83
	C	TDF	7427	425	7852	21.43
	O	TDF	5190	576	5766	42.31
	S	TDF	4959	446	5405	45.92
	O&S	TDF	3889	598	4487	55.10