

Segmentation Strategies for Low Power Implementation of Digital Filters

T. Arslan and A. T. Erdogan

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

E-mail: arslan@ee.ed.ac.uk ate@ee.ed.ac.uk

Abstract

This paper describes a generic algorithm for low power implementation of digital filters. The algorithm reduces power through a reduction in the amount of switched capacitance within the multiplier section of the filter. This reduction in turn is achieved through the segmentation of a given coefficient into two primitive sub-components. The algorithm is suitable for prototyping as an Intellectual Property core with high degree of parameterisability. Hence making the core extremely valuable for rapid system-on-chip implementation for multi-media/wireless applications. The paper describes the algorithm, its adaptability to different representation formats, and provides results which show significant power saving with different wordlengths and filter orders. The paper also indicates overheads due to the additional shift operations. Our results indicate power savings in the range of 27-85% even after including the effects of these overheads.

1. Introduction

Multimedia applications demand real-time signal processing which consists of intensive multiply and multiply-accumulate operations unique to digital signal processing (DSP) algorithms, such as filtering, fast Fourier transforms, and discrete cosine transforms. Manipulation of the multiplication procedure plays a key role in achieving low power implementation of these algorithms. In fact, in many DSP algorithms, such as filtering, the multiplier lies in the critical delay path and ultimately determines the performance of the algorithm. These DSP algorithms are implemented mainly on CMOS-based VLSI devices which could be dedicated ASICs or DSP processors. Such devices typically integrate a parallel multiplier as the central unit to handle the computational burden [1].

Example techniques for low power implementation of digital filters in the literature 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], coefficient ordering [4], and block processing [5].

This paper describes a generic algorithm for low power implementation of digital filters. The algorithm reduces power through a reduction in the amount of switched capacitance within the multiplier section of the filter. This reduction in turn is achieved through the segmentation of a given coefficient into two primitive sub-components. The algorithm is suitable for prototyping as an Intellectual Property (IP) core with high degree of parameterisability. Hence making the IP extremely valuable for rapid system-on-chip implementation for multi-media/wireless applications. The paper describes the algorithm, its adaptability to different representation formats, and provides results which show significant power saving with different wordlengths and filter orders. The paper also indicates overheads due to the additional shift operations. Our results indicate power savings in the range of 27-85% even after including the effects of these overheads.

Data and coefficient representations have an effect on the power consumption of the multiplier circuit. The choice of data/coefficient representation also influences the type of the hardware multiplier used within the DSP system. It is currently possible to design IP cores with a parameterisable multiplier type, typically from a rich vendor's library [6]. For this reason, in this paper the effect of the proposed algorithm on power saving is investigated with three data/coefficient representation scenarios. These are as follows:

1. Both data samples and coefficient values two's complement encoded and presented to a two's complement multiplier hardware.

- Both data samples and coefficient values sign-magnitude encoded and presented to a sign-magnitude multiplier hardware.
- Two's complement encoded data samples and unsigned encoded coefficient values are presented to a two's complement multiplier hardware.

2. Implementation

2.1 Coefficient segmentation algorithm with two's complement representation

Two's complement representation is most commonly used in DSP applications. This is due to ease of performing arithmetic operations such as additions and subtractions. Given the coefficient set $H = (h_0, h_1, \dots, h_{N-1})$, where N is the filter order, the algorithm proceeds through the coefficients sequentially. For a given coefficient h_k , the algorithm targets dividing it such that $h_k = s_k + m_k$, where s_k is the component to be implemented using a shift operation and m_k is the data to be applied to the hardware multiplier. In order to reduce the switched capacitance of the multiplier consecutive values of m_k must be of the same polarity, to minimise switching activity at the multiplier inputs, and have the smallest value possible, to minimise effective wordlength. This criteria can be met by careful selection of s_k and consequently m_k values. This selection procedure could be summarised in a number of stages which form the algorithm [7]. For a small positive m_k , s_k must be the largest power-of-two number closest to h_k . For this reason, stage 1 is an iterative procedure which aims to find the largest power-of-two number greater than or equal to $|h_k|$. Stage 2 deals with coefficients which are already power-of-two numbers, in which case the complete coefficient is realised using a single shift operation (i.e., $s_k = h_k$ and $m_k = 0$). In stage 3, the polarity of h_k is monitored. If h_k is a positive number then s_k is chosen as the largest power-of-two number smaller than h_k (i.e., $s_k = 2^{i-1}$). On the other hand if h_k is negative, s_k is chosen to be the smallest power-of-two number larger than $|h_k|$ (i.e., $s_k = -2^i$). In both cases m_k is equal to $h_k - s_k$.

2.2 Coefficient segmentation algorithm with sign-magnitude representation

A major drawback of two's complement representation is the sign extension, which causes the MSB sign-bits to switch when signals transition from positive to negative or vice-versa, hence increasing its power overhead. Another common method of representing data and coefficients in DSPs is the sign-magnitude representation [8]. The use of sign-magnitude avoids extra switching due to sign extension and hence can result in low power implementation. Therefore, the

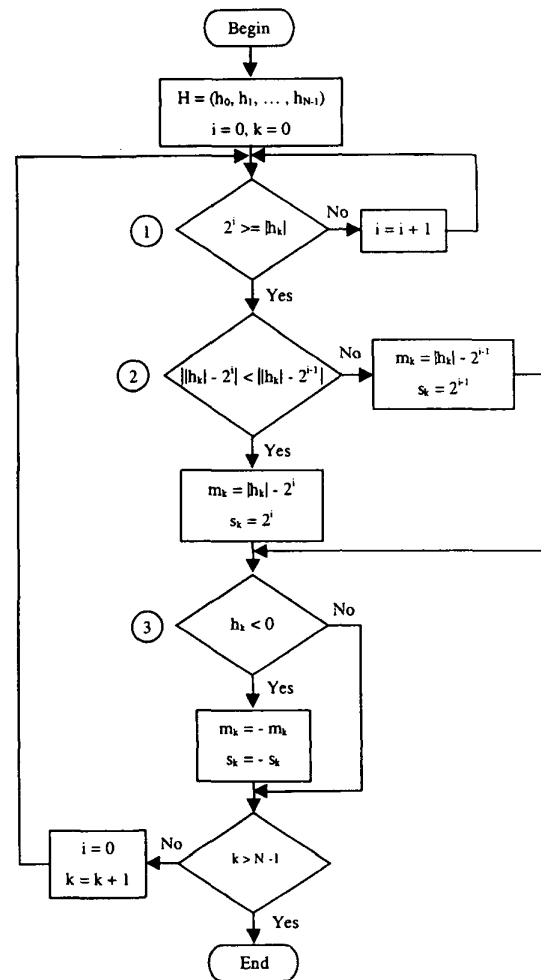


Figure 1: Flowchart of the coefficient segmentation algorithm for sign-magnitude representation.

segmentation algorithm described in the previous section can be tailored for the cases when both data samples and coefficient values are encoded with sign-magnitude representation. Figure 1 illustrates the flow chart of the modified algorithm. As with the previous algorithm, the main target of this algorithm is to decompose a given coefficient, h_k , into two parts such that $h_k = s_k + m_k$. However, the main difference is in the selection of m_k values. With two's complement representation, one of the requirements for m_k values was to make all of these positive numbers, in order to prevent switching between positive and negative numbers. However, using sign-magnitude representation m_k values can be reduced further by allowing them to be negative for some of the coefficients. For example, with two's complement representation $h_9 (= 127)$ is decomposed as $h_9 = s_9 + m_9 = 64 + 63$. Whereas, for sign-magnitude representation decomposition is performed as $h_9 = 128 + (-1)$.

The smallest possible value for m_k can be selected by comparing the distance of h_k to the nearest power-of-two number smaller than h_k and to the nearest power-of-two number larger than h_k . m_k is the minimum of these two distances (see stage 2 in Figure 1). In stage 3, the polarity of h_k is monitored. If h_k is negative, then s_k and m_k values are negated.

2.3 Coefficient segmentation algorithm with mixed-mode representation

Although the switched capacitance of the multiplier circuit is significantly reduced by using sign-magnitude representation in comparison to two's complement representation, one of the drawbacks of sign-magnitude representation is the fact that addition and subtraction operations are difficult to perform. Therefore, sign-magnitude encoded data is usually converted to two's complement prior to any addition/subtraction operations. Furthermore, most DSP processors incorporate a two's complement multiplier in their architecture. For this reason, in this section we propose the use of a mixed-mode data representation (two's complement for data samples and sign-magnitude for coefficients) using a two's complement multiplier. When using a two's complement multiplier with mixed-mode representation, however, the following cases may arise:

- (a) the coefficient is a positive number and the result at the multiplier output is correct,
- (b) the coefficient is a negative number and the result at the multiplier output requires to be negated.

In hardware implementation these cases can be satisfied by using an adder-subtractor circuit after the multiplier. MSB of the coefficient can be used to select between an addition operation (case (a)) and a subtraction operation (case (b)), as shown in Figure 2. This can also be realised in software since most DSPs incorporate instructions for performing an addition/subtraction after a multiplication (for example *MAC* and *MSUB* instructions for TMS320c54x processor from Texas Instruments). Therefore, for positive coefficients a *MAC* instruction and for negative coefficients a *MSUB* instruction can be used in order to obtain the correct filter output. For coefficient segmentation the same algorithm described in section 2.2 can be used.

3. Simulations and Results

For our results, we implemented both a sign-magnitude and a two's complement (Baugh-Wooley) array multipliers [8]. Both were selected as examples of commonly used

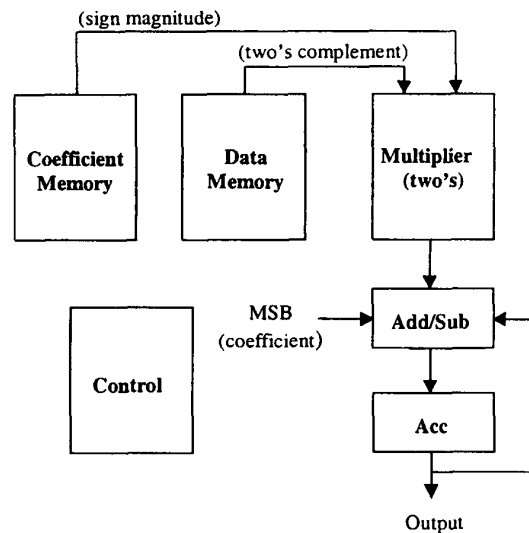


Figure 2: Simplified filter architecture for mixed-mode multiplication.

multipliers in DSP implementations and research. 8x8, 16x16, and 24x24-bit multipliers were implemented using Cadence VLSI suite with ES2 0.7 μm CMOS technology. The multipliers were constructed using *AND*, *OR*, *XOR* and *INVERTER* gates. Coefficient sets were obtained by designing ten practical FIR filters, five lowpass and five bandpass, for different filter specifications with filter lengths varying from 32 to 89 as shown in tables 1 and 2. Matlab™ was used to design the FIR filters with the Remez exchange algorithm developed by Parks and McClellan. The coefficient sets were then 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™ 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, C_i , was extracted by performing a layout of the multiplier circuit. Both capacitive information and the switching activity figure, k_i , 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. Table 3 provides an overall comparison between the conventional and coefficient segmentation algorithms

Table 1: Lowpass filter specifications

Filter	Passband (kHz)	Stopband (kHz)	Passband ripple(dB)	Stopband attenuation(dB)	Window function	Filter length
1	0 - 1.5	2 - 4	0.1	50	Hamming	53
2	0 - 1.2	1.7 - 5	0.01	40	Kaiser	71
3	0 - 3.375	5.625 - 10	0.002	90	-	42
4	0 - 1	1.5 - 5	0.0135	56	-	61
5	0 - 1.5	2 - 4	0.1	50	Blackman	89

Table 2: Bandpass filter specifications

Filter	Stopband (kHz)	Passband (kHz)	Stopband (kHz)	Passband ripple(dB)	Stopband attenuation(dB)	Window function	Filter length
1	0 - 0.1	0.15 - 0.25	0.3 - 0.5	0.1	60	Kaiser	73
2	0 - 0.45	0.9 - 1.1	1.55 - 7.5	0.8	30	-	34
3	0 - 5	8 - 12	15 - 44.14	0.00868	60	Kaiser	54
4	0 - 1	2 - 3.5	4.25 - 5	0.13	56.4	-	32
5	0 - 0.1	1.375-3.625	4 - 5	0.1	68.4	-	80

Table 3: Power reduction

Multiplier size	Algorithm/Coding	swcap/sample (pF)	Reduction (%)
8x8-bit	conventional/two's	998	-
	segmentation/two's	342	65.73
	conventional/mixed mode	439	56.01
	segmentation/mixed mode	321	67.84
	conventional/sign magnitude	251	74.85
	segmentation/sign-magnitude	136	86.37
16x16-bit	conventional/two's	7493	-
	segmentation/two's	3204	57.24
	conventional/mixed mode	4060	45.82
	segmentation/mixed mode	2775	62.97
	conventional/sign magnitude	3714	50.43
	segmentation/sign-magnitude	2427	67.61
24x24-bit	conventional/two's	22607	-
	segmentation/two's	14538	35.69
	conventional/mixed mode	16456	27.21
	segmentation/mixed mode	13229	41.48
	conventional/sign magnitude	15802	30.10
	segmentation/sign-magnitude	12518	44.63

with two's complement, sign-magnitude and mixed-mode representations for 8x8, 16x16, and 24x24-bit multipliers. In

the table, all results are compared to the conventional multiplication algorithm with two's complement representation. For example, using the coefficient segmentation algorithm with two's complement provides 65.73% reduction in average switched capacitance. On the other hand, mixed-mode representation alone reduces the average switched capacitance by 56.01%, and sign-magnitude by 74.85 for an 8x8-bit multiplier. However, when mixed-mode and sign-magnitude representations are used together with the coefficient segmentation algorithm then the reductions are increased to 67.84% and 86.37% respectively. Other comparisons for 16x16 and 24x24-bit multipliers can also be found in the same table. From these comparisons, it is clear that the best reductions are obtained using the coefficient segmentation with sign-magnitude representation.

In order to estimate the overheads caused by shift operations, high-level models of the multiplier and the shifter circuits were used as shown in Table 4 [9]. Analysis based on these models revealed that an 8-bit shifter consumes 97% less capacitance than an 8x8-bit multiplier. Similarly, 16 and 24-bit shifters consume 98% and 99% less capacitance than 16x16 and 24x24-bit multipliers respectively. These were used with our simulation results for estimating overheads. For instance, our simulations with two's complement and 8x8-bit multiplier show that the average switched capacitance is 998 pF and 342 pF using conventional and segmentation algorithms respectively. Hence, the switched capacitance of the shifter could be estimated as $998 * 0.03 = 30$ pF. Therefore, the total switched capacitance, including the

shifter overhead, with coefficient segmentation algorithm is $342+30 = 372$ pF. For this reason, the net reduction is 62.73%. The above procedure is repeated for 16x16 and 24x24-bit multipliers and the results are reported in Table 5. These results clearly indicate that overheads are less than 5%.

4. Conclusions

We have presented an algorithm for low power implementation of FIR filters on CMOS DSP systems. The algorithm targets reducing the amount of switched capacitance within the multiplier section of the DSP system through fragmentation of the coefficients into two primitive parts, which can be processed with a significant reduction in the amount of switched capacitance. Results are presented, using a range of practical FIR filter examples, which indicate significant power savings. The algorithm maintains power reduction with different filter orders and wordlengths. In addition, it has been demonstrated with a number of data representation schemes. This flexibility makes it suitable for prototyping as an IP for System-on-Chip applications.

5. References

- [1] K. Nadehara, I. Kuroda, M. Daito, and T. Nakayama: "Low-Power Multimedia RISC", IEEE Micro, 1995, 15 (6), pp. 20-28.
- [2] 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, 1997, 44 (6), pp. 488-497.
- [3] H. Choi and W.P. Burlison: "Search-Based Wordlength Optimisation for VLSI/DSP Synthesis", VLSI Signal Processing VII, 1994, pp. 198-207.
- [4] A.T. Erdogan and T. Arslan: "Low Power Implementation of Linear Phase FIR Filters for Single Multiplier CMOS Based DSPs", IEEE Int. Symp. on Circuits and Systems, 1998, Monterey, CA, USA, pp. D425-D428.
- [5] T. Arslan and A.T. Erdogan: "Data Block Processing for Low Power Implementation of Direct Form FIR Filters on Single Multiplier CMOS DSPs", IEEE Int. Symp. on Circuits and Systems, 1998, Monterey, CA, USA, pp. D441-D444.
- [6] R. Zimmermann: "VHDL Library of Arithmetic Units", Forum on Design Languages (FDL'98), September 1998, (http://www.iis.ee.ethz.ch/~zimmi/publications/arith_lib_fdl.pgz)
- [7] A.T. Erdogan and T. Arslan: "Low Power Coefficient Segmentation Algorithm for FIR Filter Implementation", IEE Electronics Letters, 1998, 34 (19), pp. 1817-1819.
- [8] A.R. Omondi: "Computer Arithmetic Systems: Algorithms, Architecture and Implementation", (Prentice Hall, 1994).
- [9] A.P. Chandrakasan, M. Potkonjak, R. Mehra, J. Rabaey, and R.W. Brodersen: "Optimizing Power Using Transformations", IEEE Trans. on CAD, 1995, 14 (1), pp. 12-31.

Table 4: Capacitance models for a 1.2 μ m CMOS technology

Component	Capacitance Model (in fF)	Parameters
<i>Multiplier</i>	$253 * N1 * N2$	N1,N2: input bitwidths
<i>Shifter</i>	$N*28.7+\log(M+1)*$ $(43.8+12.2*N+0.06*N^2+0.24*M*N-0.18*S*N)$	M: maximum shift allowed N: input bitwidth S: shift

Table 5: Average reductions including overheads

Algorithm/Coding	Multiplier Size					
	8x8-bit		16x16-bit		24x24-bit	
	swcap/sample (pF)	Reduction (%)	swcap/sample (pF)	Reduction (%)	swcap/sample (pF)	Reduction (%)
conventional/two's	998	-	7493	-	22607	-
segmentation/two's	372	62.73	3354	55.24	14764	34.69
conventional/mixed mode	439	56.01	4060	45.82	16456	27.21
segmentation/mixed mode	334	66.53	2856	61.88	13394	40.75
conventional/sign magnitude	251	74.85	3714	50.43	15802	30.10
segmentation/sign-magnitude	144	85.57	2501	66.62	12676	43.93