

A LOW POWER FIR FILTERING CORE

A. T. Erdogan, M. Hasan and T. Arslan

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

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

ABSTRACT

The authors present a DSP core for low power implementation of FIR filters. The core is based on direct form realisation of FIR digital filters and processes filter coefficients in a non-conventional order. The paper describes the architecture of the core and the implementation of its key components. An overall power reduction of up to 17% is obtained as compared to the conventional filtering cores. A power profile of the main contributing blocks in the core is also provided.

I. INTRODUCTION

One of the fastest growing areas in the computing industry is the provision of high throughput DSP systems in a portable form [1]. With the advent of SoC technology, DSP algorithms such as FIR filters are being prototyped as parameterisable cores which could be embedded within the SoC platform. For high performance low power applications, there is a continuous demand for DSP cores, which provide high throughput while minimising power consumption.

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]:

$$P_{sw} = \frac{1}{2} k \cdot C_{load} \cdot V_{dd}^2 \cdot f \quad (1)$$

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 .

FIR filters are widely used in DSP systems and are characterised by the extensive sequence of multiplication operations. Implementation of FIR filters could be either sequential or parallel. The sequential approach aims to minimise area requirements through the re-use of as much of the hardware as possible. Each product of sum in the FIR filter equation is computed sequentially and added to the accumulated sum in the accumulator. On the other hand, a parallel implementation aims to maximise the sample rate of an FIR filter by performing all or some of the multiplications in parallel with the cost of additional multiplier/adder hardware.

In recent years a number of techniques have been proposed for low power implementation of FIR filters. These include the following: use of differential coefficients [2], wordlength optimisation [3], multirate architectures [4], and dynamic adjustment of filter order [5].

A number of researchers including the authors have investigated the use of coefficient ordering for low power implementation of FIR filters [6][7][8]. However, *all reported work considers power reduction at multiplier inputs or within the multiplier unit alone*. To our best knowledge no work exists which consider the impact on the overall FIR core.

In this paper, the authors present a DSP core for low power implementation of FIR filters. The core is based on direct form realisation of FIR digital filters and processes filter coefficients in a non-conventional order. The paper describes the architecture of the core and the implementation of its key components. An overall power reduction of up to 17% is obtained as compared to the conventional filtering cores. A power profile of the main contributing blocks in the core is also provided.

II. IMPLEMENTATION

A conventional implementation of a direct form (DF) FIR filter is executed such that at each clock cycle a new data sample, $x(n)$, and the corresponding filter coefficient, $h(k)$, are fetched from the memory simultaneously and applied to the multiplier inputs. Therefore, for each multiplication both inputs of the multiplier receive new data. Due to this continuous change at both inputs, there will be a high level of switching activity within the multiplier leading to a higher power consumption [6].

The multiplier is a major bottleneck governing the performance of a DSP algorithm. In addition, the power dissipated within the multiplier represents a significant proportion of the overall power dissipated by the DSP device [1]. A reduction in the switching activity within the multiplier block can be achieved by implementing the filter such that respective data samples are multiplied with filter coefficients in a non-conventional order [6]. This order can be obtained by minimising the *Hamming distance* between those filter coefficients used in successive multiplication operations. 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

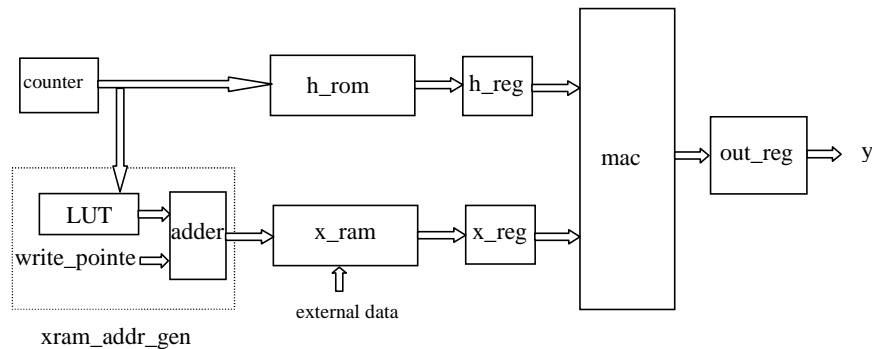


Figure 1. A block diagram of the FIR filter core.

reason, coefficient ordering has no implications on the speed of the filtering process.

Filtering commences by fetching a coefficient and the corresponding data sample. These are then presented to the multiplier inputs and the result is added to the accumulator. The rest of the coefficients are processed in a similar manner. Once all the coefficients are processed the filter output is obtained from the accumulator. For the next filter output, the accumulator is cleared, a new data sample is read into the data memory (replacing the oldest data in the memory), and the above steps are repeated.

A block diagram of the FIR core is illustrated in Figure 1. It consists of a coefficient memory (h_rom), data memory (x_ram), multiply-accumulate (mac) unit, and control circuitry. The control circuitry governs the non-conventional processing of coefficients. It consists of a counter and an address generator ($xram_addr_gen$) which in turn consists of a look-up table (LUT) and an adder. The x_ram is realised in the form of a circular buffer for reducing its power consumption. The x_ram position to be currently written is tracked by the write_pointer. The write_pointer always points towards the most recently entered data value $x(0)$. In a conventional implementation, the access of the first coefficient $h(0)$ in h_rom and the data sample $x(0)$ in x_ram must only be aligned and all other combinations of coefficients and data will automatically fall in place. In an ordered implementation, the coefficients in the h_rom can be in any order depending upon the ordering algorithm and the original coefficient set. Let us assume that the first h_rom location stores $h(5)$ instead of $h(0)$ as per the ordered list. In order to generate the address of the correct data corresponding to $h(5)$, the $x_ram_addr_gen$ should be able to generate the address of the corresponding data $x(5)$ which is always located five positions away from $x(0)$ in an x_ram . The $x(0)$ position is always pointed at by the write_pointer. Hence, the first offset in the LUT, for the write_pointer corresponding to the first entry of h_rom , must be 5. The remaining entries of the LUT can be obtained by storing appropriate offsets with respect to the write_pointer's position by examining the corresponding coefficients in the h_rom . The h_rom and LUT are both addressed by the same counter. The final x_ram read address for a given counter value is obtained by adding the corresponding offset to the write_pointer. The write pointer is always decremented by one after the generation of every output as the x_ram receives a new data sample value only after every output generation.

III. SIMULATIONS AND RESULTS

A number of cores were designed and implemented based on the conventional and the new architecture described in section II. In the conventional implementation, the coefficients are in their normal sequence (*Norm*) whereas in the new architecture they are sequenced for minimum Hamming distance (*Ham*). The cores were synthesized using Cadence BuildGates and Alcatel 0.35u CMOS technology for 16-bit data/coefficient wordlength and using two different multiplier types (*csa*: carry-save array multiplier and *wall*: Wallace-tree Booth multiplier).

Two example coefficient sets corresponding to linear phase bandpass FIR filters were obtained using the Remez exchange algorithm developed by Parks and McClellan, see Table 1.

In order to evaluate the performance of the designed cores, netlist simulations were performed using Cadence's Verilog-XL simulator for 1000 zero mean uniformly distributed random data samples. The switching activity information obtained from the netlist simulations was then fed into Synopsys DesignPower for power analysis. For this, a clock frequency of 10 MHz and a supply voltage of 3 V were used.

The power analysis results for the designed cores are presented in Table 2. The table illustrates the total dynamic power for both filter examples using two different multiplier types. The percentage reduction in the power consumption is obtained by comparing *Ham* and *Norm* in each case. It can be seen that the *Ham* based filter implementations achieve 8% and 17% reduction for BPF_1 using *csa* and *wall* multipliers respectively at an expense of 2% increase in area, see Table 3. Whereas BPF_2 achieves a reduction of 16% for both multiplier types with an area overhead of only 1%. The contributions of the individual blocks within the *Ham* based core to the overall area and power consumption are illustrated in Figure 2 (a) and (b) respectively for BPF_2 example using a *csa* multiplier. It is clear that although the *mac* block consumes only 20% of the area but its power consumption is substantially higher of the order of 73%. Whereas the x_ram occupies 69% of the area it consumes only 9% of the overall power. The contributions of the rest of the blocks towards the area and power are not significant.

Table 1: Bandpass filter specifications

Filter #	Stopband (kHz)	Passband (kHz)	Stopband (kHz)	Passband ripple (dB)	Stopband attenuation (dB)	Window function	Sampling Freq. (kHz)	Filter length
BPF_1	0 - 0.1	0.15 - 0.25	0.3 - 0.5	0.1	60	Kaiser	1	73
BPF_2	0 - 0.1	1.375-3.625	4 - 5	0.1	68.4	-	10	80

Figure 3 depicts the power consumption in the various blocks of the FIR filter core for the *Norm* and *Ham* based architectures using a *csa* multiplier and BPF_2 example. It is clear that the power savings are maximum in the *mac* block for the *Ham* approach because of the reduction in the switching activity at the inputs of its most power consuming multiplier block. The power consumption in the data memory *x_ram* for the *Ham* approach slightly goes up on account of the higher switching activity on its address bus. This higher switching activity is due to the non-conventional data access in the *Ham* approach. The *h_reg* and *h_rom* power consumption go down as a direct consequence of low switching activity of ordered coefficients in the *Ham* approach over the *Norm* approach. The *counter* power consumption slightly goes up in the *Ham* approach due to the higher fanout in the form of a more complex *xram_addr_gen* block. The *Ham's* *xram_addr_gen* block consumes more power because it is primarily responsible for keeping track of the non-conventional data addresses corresponding to the ordered coefficient set. The *out_reg* and *write_pointer* almost consume the same power in both the approaches.

Figure 4 depicts the power savings obtained by the *Ham* approach at the data bus, coefficient bus, multiplier, mac and filter levels. The percentage reduction on the coefficient bus is around 80% due to the high correlation among the coefficients after ordering. Whereas, data samples are uncorrelated and hence the change in their order may yield a slight positive or negative reduction in

power. It happens that in our case the data bus power is going up by a small amount of 5% but it may go down as well for some other data sets. The percentage reduction in power in the multiplier is directly related to the switching activity of its inputs. In our case, the multiplier power reduction is 29%. The mac power reduction is around 25% and is directly dictated by its multiplier sub-block. The overall FIR filter power saving is around 16%. It is evident that the percentage reduction in power goes down as one moves from the multiplier to the filter level on account of the overheads associated with the realization of the *Ham* approach and also due to the size of the respective reference blocks.

IV. CONCLUSIONS

We have presented a DSP core for low power implementation of FIR filters. The core is characterised by processing coefficients in a non-conventional order with the aim of reducing the effective switched capacitance within the multiplier circuit. Until now researchers in the literature have studied the effect of ordering coefficients focusing on the multiplier without considering the overall core. In this paper we have designed a number of cores based on the ordered coefficients demonstrating an overall power reduction of up to 17%. It is also shown that the overall reduction in power depends on the filter characteristics and the multiplier type used in the implementation.

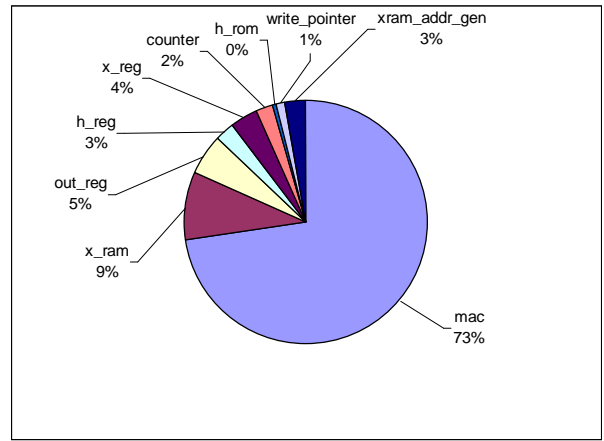
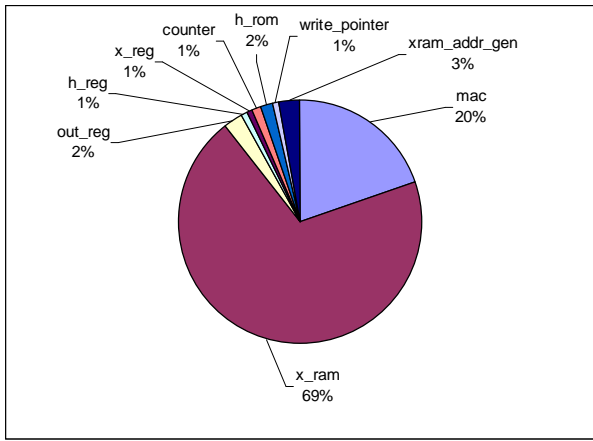
Table 2: Power consumption analysis for FIR cores

Filter	Algorithm	csa		wall	
		[mW]	[%]	[mW]	[%]
BPF_1	Norm	9.21	-	9.10	-
	Ham	8.46	8	7.54	17
BPF_2	Norm	10.10	-	8.97	-
	Ham	8.49	16	7.57	16

Table 3: Area comparisons for FIR cores

Filter	Algorithm	csa		wall	
		[u]	[%]	[u]	[%]
BPF_1	Norm	9093	-	9215	-
	Ham	9306	2	9428	2
BPF_2	Norm	9579	-	9701	-
	Ham	9672	1	9794	1

u represents equivalent NAND gates



(a) Area (b) Power
Figure 2. Distribution of area and power for the Ham based FIR filter core.

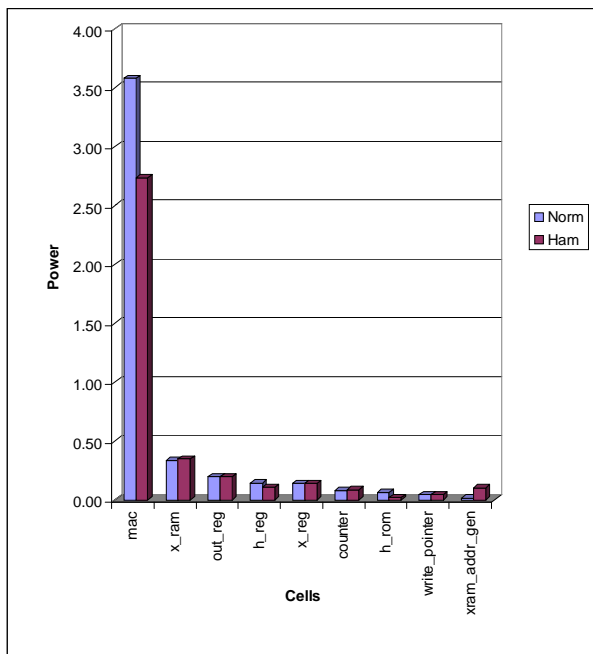


Figure 3. Block power distributions

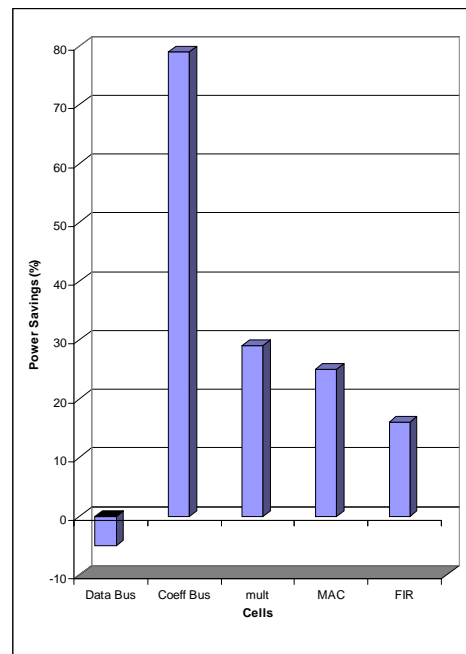


Figure 4. Power savings in various blocks

V. REFERENCES

- [1] I.S. Abu-Khater, A. Bellaouar and M.I. Elmasry: "Circuit Techniques for CMOS Low-Power High-Performance Multipliers", IEEE Journal of Solid-State Circuits, vol. 31, no. 10, pp. 1535-1546, Oct. 1996.
- [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, vol. 44, no. 6, pp. 488-497.
- [3] H. Choi and W.P. Bursleson: "Search-Based Wordlength Optimisation for VLSI/DSP Synthesis", VLSI Signal Processing VII, IEEE Press, 1994, pp. 198-207.
- [4] M. Mehendale, S.D. Sherlekar and G. Venkatesh: "Low Power Realisation of FIR Filters Using Multirate Architectures", 9th Int. Conf. on VLSI Design, pp. 370-375, Jan. 1996.
- [5] J.T. Ludwig, S.H. Nawab, and A.P. Chandrakasan: "Low Power Digital Filtering Using Approximate Processing", IEEE Journal of Solid-State Circuits, vol. 31, no. 3, pp. 395-399, Mar. 1996.
- [6] A.T. Erdogan and T. Arslan: "Low Power Implementation of Linear Phase FIR Filters for Single Multiplier CMOS Based DSPs", IEEE ISCAS'98, May 1998, California, USA, pp. D425-D428.
- [7] M. Mehendale, S.D. Sherlekar and 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.
- [8] K.Masselos, S. Theoharis, P.K. Merakos, T. Stouraitis, and C.E. Goutis: "Low Power Synthesis of Sum-of-Products Computation", ISLPED'00, 2000, Italy, pp. 234-237.