

# Algorithmic Implementation of Low-Power High Performance FIR Filtering IP Cores

C.H.Wang<sup>1</sup>, A.T.Erdogan<sup>1,2</sup> and T.Arslan<sup>1,2</sup>

<sup>1</sup>University of Edinburgh, School of Engineering and Electronics  
Edinburgh, EH9 3JL, Scotland, United Kingdom

<sup>2</sup>Institute of System Level Integration, The ALBA campus  
Livingston, EH54 7EG, Scotland, United Kingdom

[C.Wang@sms.ed.ac.uk](mailto:C.Wang@sms.ed.ac.uk), [Ahmet.Erdogan@ee.ed.ac.uk](mailto:Ahmet.Erdogan@ee.ed.ac.uk), [Tughrul.Arslan@ee.ed.ac.uk](mailto:Tughrul.Arslan@ee.ed.ac.uk)

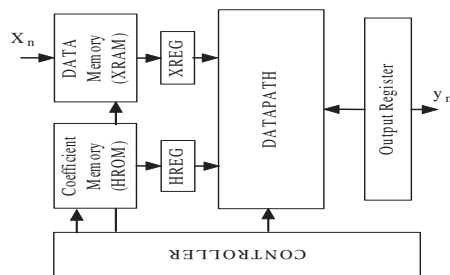
## Abstract

This paper presents two schemes for the implementation of high performance and low power FIR filtering Intellectual Property (IP) cores. Low power is achieved through the utilization of algorithms such as coefficient segmentation, block processing and combined segmentation and block processing algorithms. On the other hand, multiple data paths are utilized for achieving high performance. The paper presents the complete architectural implementation of these algorithms for high performance applications. The paper describes the design methodology, evaluation environment, and provides results which show up to 40% reduction in power consumption.

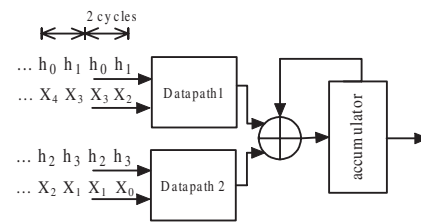
## 1. Introduction

The demand for more functionality in portable application such as portable PCs and camera-enabled mobile phones has dictated the need for the provision of high data processing capability together with efficiency in terms of battery life. The availability of high performance IP cores for System on Chip (SoC) devices which make up the above systems is important segment of the electronic market and has attracted significant research interest.

Finite impulse response (FIR) filtering is one of the most widely used operations in Digital Signal Processing (DSP) devices and is characterized by the extensive



**Figure 1. The generic block diagram of FIR filtering implementation**



**Figure 2. A example 2 datapaths architecture for 4-tap filter**

sequence of multiplication operations. Two implementation approaches are sequential and parallel. Despite of the former being low-cost and area-efficient in hardware, its low throughput can not satisfy the requirement of advanced applications. However, using the later approach requires careful design effort in incorporating the added hardware in terms of adders and multipliers into the design when targeting low power applications. Therefore, the focus of the paper is to implement a variant of architectures for high-throughput FIR filtering IP cores with low-power consumption.

Recently, various techniques for high performance realization of FIR filtering with low power dissipation have been presented in the literature. High-performance FIR filters by exploiting computation sharing multiplier (CSHM) have been presented in [2]. The authors utilise the alphabet set, consisted of smaller bit sequence, to compute the result of multiplication in advance. CSHM comprises precomputation, select and adder (S&A) units. In the precomputation unit, the multiplication of alphabet with input  $x$  is calculated and then the final result can be obtained through S&A operation. The main advantage of CSHM is that the data of precomputation can be shared by the S&A units. The operations can be performed in parallel. Hence, computation reuse and parallelism can be targeted for FIR filtering achieving up to 52% speedup in comparison to FIR filters based on carry-save for multiplier. Another approach used in [3] is differential coefficients method (DCM) which involves using various orders of differences between coeffi-

cients along with stored intermediate results rather than using the coefficients themselves directly for computing the partial products in the FIR equation. To minimize the overhead while retaining the benefit of DCM, differential coefficient and input method (DCIM) [5] has been proposed.

This paper presents the implementation of high throughput and low power FIR filtering Intellectual Property (IP) cores. With the consideration of increased number of datapaths, two new efficient architectures are also implemented and compared.

## 2 Proposed FIR filtering architectures

### 2.1 Multiple Datapath Architectures

In direct form (DF) realization of a  $N$ -tap FIR filter, there are  $N$  multipliers,  $N-1$  delay operations and  $N-1$  adders for multiplying and adding  $N$  coefficients with the correct and  $N-1$  previous data samples. Figure 1 illustrates a generic block diagram of a single datapath FIR filtering implementation. It comprises a controller block, two memory blocks, a datapath, two input registers and output register. The datapath consists of a multiplier, an adder, and an accumulator register. For high throughput applications, intuitively, the number of datapaths must be increased. This presents a trade-off between area and throughput as well as power consumption. For example, Figure 2 illustrates an architecture for a 4-tap filter with 2 datapaths for doubling the throughput. In the example, coefficients are divided into 2 parts,  $(h_1, h_2)$  and  $(h_3, h_4)$ , and are processed through corresponding of datapaths. In general, The latency of FIR filter can be reduced to  $\lceil M/N \rceil$  cycles, where  $M$  is the number of filter taps,  $N$  is the number of exploited datapaths and  $\lceil \cdot \rceil$  presents an integer no less than  $M$ .

## 3 Low Power Architectures

### 3.1 Coefficient Segmentation Algorithm

Two's complement representation is the most commonly used in DSP applications. Nevertheless, its main drawback is the sign extension and causes significant switch activities of consecutive operations between the positive and negative data. Hence, we have presented the coefficient Segmentation algorithm in [4]. It segments the coefficients  $h_k$  to two parts. One is  $m_k$  for multipliers and another is  $s_k$  for SEG components consisting of hard-wiring shifter, converter and multiplexer. By segmentation,  $m_k$  can be performed to be the smallest positive value for minimizing the switch activities at input of multipliers. On the other hand, the  $s_k$  is a power of two number and could be positive or negative value depending on the initial coefficients. The MSB of  $s_k$  acts as the sign bits and remainders are the measure of shift. For instance, if coefficient is 11100101, its  $m_k$  and  $s_k$  will be 00000101 and 10101, respectively.

### 3.2 Data Block-Processing

Owing to the successive change of both input samples and coefficients at each clock cycle, there are significant switch activities within the datapath. This high switching activity can be reduced significantly, if the coefficient input of the multiplier is kept unchanged and multiplied with a block of data samples [7]. Once a block of data samples are processed, a following coefficient is multiplied with a new block of input samples. However, additional sets of accumulators are needed in this algorithm, depending on the size of the block. Previous research has shown that a block size of 2 provides the best power saving for single datapath implementations [7]. Therefore, a block size of 2 has also been adopted for multiple datapath implementations in this work.

### 3.3 Combinational Algorithm

The architectures of coefficient segmentation and data block-processing can be merged to achieve better power saving with a slight increase in area. The authors in [6] demonstrated the use of combined algorithm for single datapath implementations.

## 4 Two Schemes Implementation

Two schemes are implemented in this work with the consideration of critical path, area and power consumption.

### 4.1 Scheme I

In scheme I, the multipliers are replaced with their counterparts using DNR technique[2] in order to speed up the multiplication process as well as save power due to reduced logic. Similarly, the multiple input carry ripple adder is replaced with a Wallace Tree Compressor, namely  $WTC_m$ , followed by a carry-look-ahead,  $CLA$ , adder. An example architecture of this scheme for the combined algorithm with two datapaths is illustrated in Figure 3. In this architecture, BP Dual-Accumulator and SEG are used to implement the block processing and coefficient segmentation algorithms respectively. Clearly, each datapath consists of a multiplier and a SEG module and their three outputs, together with the multiplexer output, are sent to  $WTC_m$  for reducing the number of adder inputs to two, before a final output is obtained through the  $CLA$ .

### 4.2 Scheme II

As the number of datapaths increases a more efficient architecture could be devised, as shown in Figure 4. The critical path, intuitively, is located in the multipliers rather than shifters so the difference in arrival times prior to  $WTC_m$  will result in a significant number of glitches. Therefore, we sum the shifter outputs and the multiplexer output through a separated Wallace Tree Compressor, namely  $WTC_s$ , prior to  $WTC_m$ . This reduces the glitches

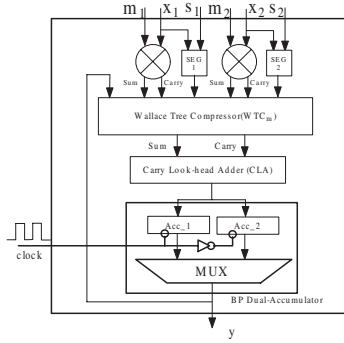


Figure 3. An example of scheme I with 2 datapaths

Table 1. Power analysis for two schemes

# of Datapath	Algorithms	Scheme 1		Scheme 2	
		Power (mw)	Saving (%)	Power (mw)	Saving (%)
1 Datapath	CON	1.432	-	1.365	-
	CSEG	1.344	6.1	1.230	10.2
	BP	1.049	26.7	0.998	26.8
	COMB	1.043	27.1	0.908	33.4
2 Datapath	CON	2.695	-	2.479	-
	CSEG	2.526	6.2	2.239	9.6
	BP	1.910	29.1	1.756	32.6
	COMB	1.785	33.7	1.596	35.6
4 Datapath	CON	5.039	-	4.747	-
	CSEG	4.750	5.7	4.157	12.4
	BP	3.350	33.5	3.099	34.7
	COMB	3.215	36.2	2.869	39.5
8 Datapath	CON	8.993	-	8.839	-
	CSEG	8.851	1.58	7.738	12.4
	BP	6.079	32.4	5.694	35.5
	COMB	6.064	32.5	5.319	39.8

by balancing the arrival times between the multiplier and SEG outputs, and the number of inputs to  $WTC_m$  decreases as well. Moreover, in scheme I the CLA is located at the end of the critical path and therefore it contributes to the long critical path time and consumes significant amount of power due to the accumulated glitches activity from prior processing elements (multipliers, shifters and  $WTC_m$ ). In scheme II, we relocate the CLA after the accumulator registers. However, this requires a modification to the BP Dual-Accumulator, as shown in Figure 4. Each BP Dual-Accumulator consists of 2 accumulators and needs a CLA to compute the final result.

## 5 Simulations and Results

We have analyzed throughput, area and power consumption of 2 schemes for different FIR filter algorithms, namely conventional (CON), coefficient segmentation (CSEG), block processing (BP), and combined CSEG and BP (COMB). In order to evaluate their performance for different number of datapaths, we have implemented these with 1, 2, 4 and 8 datapath cases. All FIR IP cores were implemented for an example of 73-tap band-pass filter, using an 16x16-bit Booth multipliers in their datapaths. The cores were designed using Verilog HDL and then synthesized using Synopsys Design Compiler, target-

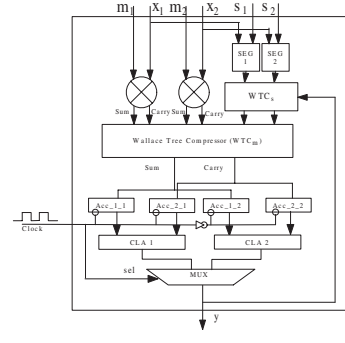


Figure 4. An example of scheme II with 2 datapaths

Table 2. Area analysis for two schemes

# of Datapath	Algorithms	Scheme 1		Scheme 2	
		Area (mm <sup>2</sup> )	Increase (%)	Area (mm <sup>2</sup> )	Increase (%)
1 Datapath	CON	0.151	-	0.153	-
	CSEG	0.156	3.7	0.159	3.7
	BP	0.157	3.7	0.168	9.2
	COMB	0.163	7.9	0.173	12.7
2 Datapath	CON	0.213	-	0.216	-
	CSEG	0.226	5.9	0.228	5.6
	BP	0.223	4.5	0.233	7.9
	COMB	0.234	9.6	0.246	13.7
4 Datapath	CON	0.340	-	0.343	-
	CSEG	0.367	7.7	0.370	7.8
	BP	0.352	3.3	0.361	5.2
	COMB	0.377	10.7	0.386	12.5
8 Datapath	CON	0.598	-	0.601	-
	CSEG	0.648	8.34	0.652	8.4
	BP	0.619	2.3	0.628	4.4
	COMB	0.669	10.6	0.678	12.8

ing a 0.18 micron standard cell CMOS library. Simulations were performed for 1000 randomly generated data samples using Verilog-XL simulator. This was followed by computing their power consumption with Synopsys Design Power. In all cases, a clock rate of 10MHz and 1.8V were assumed.

As mentioned in II.A, the throughput will be slightly lower than the expected throughput. For example, for a 73-tap filter, a filter output can be generated in 73, 37, 19, and 10 clock cycles using 1, 2, 4, and 8 datapaths respectively. Therefore, the throughput will be 137, 270, 526 and 1000 K samples/sec for 1, 2, 4, and 8 datapaths respectively.

The results of power consumption and area usage are given in Table 1 and Table 2 for the two schemes with different number of datapaths. Clearly, for scheme I COMB achieves the best power saving of up to 27%-36% with an 8%-11% increase in area. The second best power saving is achieved by BP algorithm with only 2-5 % increase in area. However, for 8 datapath case COMB and BP achieve similar power savings. This can be explained in respect to CSEG's performance in power saving. As the number of datapaths increases the power saving of CSEG decreases significantly. The main reason for this decrease in power efficiency is the increase in power consumption of  $WTC_m$  due to the increased number of shifters employed by this algorithm. For example, there are 25 inputs in the

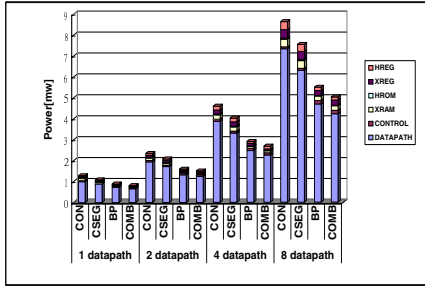


Figure 5. Overall power consumption of Scheme II

$WTC_m$  in 8-datapath case for CSEG compared to only 17 inputs for CON. Although, the increased number of datapaths reduces the power of the multipliers, this reduction becomes less than the overheads due to the additional shifters and increased complexity of the  $WTC_m$ . Therefore, this degradation in power saving of CSEG directly contributes to COMB's power efficiency as well.

Similar to scheme I, for scheme II the best power saving of up to 33%-40% is also achieved by COMB with a 12%-13% increase in area. The second best power saving (26%-36%) is achieved by BP algorithm with 4%-10% increase in area. On the other hand, CSEG provides the least power savings (9%-13%) with the least area increase (3%-9%).

Figure 5 illustrates the power contribution of the main blocks for scheme II. Clearly, as expected most of the power is consumed by the datapaths. Furthermore, the power contributions of the main blocks within the datapath units are depicted in Figure 6 for scheme II. In all cases the multiplier block is responsible for most of the power consumption. In overall, CSEG reduces multiplier power by 27%-35%, BP by 41%-44% and COMB by 58%-61%. Clearly, the power consumption is reduced significantly in all cases. Moreover, the relocation of CLA reduces its power consumption significantly. For example, in scheme II the CLA power consumption is reduced by 87% with COMB for 8-datapath case compared to scheme I.

The area usage analysis for all algorithms with different number of datapaths is given in Table II. For scheme I, BP results in least area increase (2-5%), followed by CSEG (3-9%), and COMB (8-11%) compared to CON area. Similarly, for scheme II BP results in (4-9%), CSEG (3-9%), and COMB (12-14%) area increase. Moreover, in overall the area increase of both schemes only increases by about 1.5, 2.4, and 4.4 folds for 2, 4, and 8-datapath cases compared to 1-datapath case.

In overall, scheme II provides better power saving results (4%-13%) compared to scheme I. CSEG and COMB achieve the best improvements. As described in Section IV.B, the utilization of  $WTC_s$  helps balancing the arrival times among the  $WTC_m$  inputs. It also reduces the number of  $WTC_m$  inputs. A further power saving for all algorithms is achieved by the relocation of CLA as was also described in Section IV.B. However, these improvements

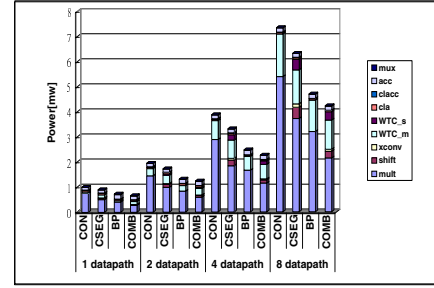


Figure 6. Datapath power consumption of Scheme II

result in up to 7% area increase. As expected, the largest increase in area is for BP and COMB algorithms, they both employ one additional dual-accumulator and CLA unit. However, for CON and CSEG algorithms, there is only slight increase in area.

## 6 Summary and Conclusions

In this paper, we have presented the complete architectural implementations of a number of low power algorithms based on two different schemes for high performance applications. The paper has demonstrated the impact of parameterisation in terms of datapath parallelisation on the power, area and speed metrics for FIR IP cores. Results have been provided which show up to 40% reduction in power consumption with less than 14% increase in area depending on the number of datapaths employed.

## References

- [1] H. Choo, K. Muhammad, and K. Roy, "Two's complement computation sharing multiplier and its applications to high performance dfe," *IEEE Trans. Signal Processing*, vol. 51, pp. 458-469, Feb. 2003.
- [2] J. Park, K. Muhammad, K. Roy, "High Performance FIR filters design based on sharing multiplication," *VLSI systems*, IEEE transactions on, 2003, Page(s): 244-253.
- [3] N. Sankaraya, K. Roy, D. Bhattacharya, "Algorithm for Low Power and High Speed FIR Filter Realization Using Differential Coefficients," *Circuits and Systems II, Analog and Digital Signal Processing*, IEEE Transaction on, 1997, Page(s): 488-497.
- [4] A. T. Erdogan, T. Arslan, "A Coefficient Segmentation Algorithm for Low Power Implementation of FIR Filters," *Circuit and System*, 1999. IEEE International Symposium on, 1999, Page(s): 359-362.
- [5] T-S Chang, Y-H Chu, and C-W Jen, "Low Power FIR Filter Realization with Differential Coefficients and Inputs," *Circuits and Systems-II*, IEEE transactions on, 2000, Page(s): 137-145.
- [6] A. T. Erdogan, M. Hasan, T. Arslan, "Algorithm Low Power FIR Cores," *Circuits, Devices and Systems*, IEE Proceedings, 2003, Pages(s): 155-160.
- [7] T. Arslan and A. T. Erdogan "Data Block Processing for Low Power Implementation of Direct Form FIR Filters on Single Multiplier CMOS DSPs," *Circuits and Systems*, 1998. IEEE International Symposium on, 1998., Page(s): 441-444.