Low Power Synthesis of Sum-Of-Products Computation

K. Masselos S. Theoharis P. K. Merakos T. Stouraitis C. E. Goutis
VLSI Design Laboratory
Department of Electrical and Computer Engineering
University of Patras, Rio 26500, Greece
Tel: (+) 30 61 997324
fax: (+) 30 61 994798
E-mail: {masselos theoharis merakos thanos goutis}@ee.upatras.gr

ABSTRACT

Novel techniques for the power efficient synthesis of sum-of-product computations are presented. Simple and efficient heuristics for scheduling and assignment are described. Different partly static cost functions are proposed to drive the synthesis tasks. The proposed cost functions target the power consumption either in the buses connecting the functional units with the storage elements or inside the functional units. The partly static nature of the proposed cost functions reduces the time of the synthesis procedure. Experimental results from different relevant digital signal processing algorithmic kernels prove that the proposed synthesis techniques lead to significant power savings.

1. INTRODUCTION

The recent advances in the areas of wireless communications and multimedia technology made available a large number of portable systems that make extensive use of digital signal processing (DSP). For this reason the development of methodologies for low power implementation of digital signal processing algorithms is very important.

The switching component of power dissipation can be optimized in the high levels of abstraction where the most significant power savings can be achieved [1]. Several data path synthesis techniques for power optimization have been described [2, 3]. Power optimization techniques for linear computations in digital signal processing algorithms have been proposed in [4, 5]. Techniques for power optimization of FIR filters have been proposed in [6, 7].

Sum-of-products computation forms an important part of the total power budget of a DSP system [7]. Two main categories of DSP algorithmic kernels requiring sum-of-products computation are Convolutional Algorithms and Transformational Algorithms.

The basic computation of the Convolutional Algorithms is the convolution between data and coefficient vectors and is described by Equation 1.

$$Y_n = \sum_{i=0}^{N-1} C_i X_{n-i} \qquad (1)$$

where C_i's are constant coefficients and X_n, Y_n are the n_{th} terms of the input and output sequences respectively.

ISLPED '00, Rapallo, Italy.

The basic computation of the Transformational Algorithms is the matrix-vector multiplication between a coefficient matrix and a data vector and is described by Equation 2.

$$\begin{bmatrix} Y_0 \\ Y_1 \\ \\ Y_{N-1} \end{bmatrix} = \begin{bmatrix} C_{00}, & C_{01}, & \ldots, C_{0M-1} \\ C_{10}, & C_{11}, & \ldots, C_{1M-1} \\ & & \\ C_{(N-1)0}, & C_{(N-1)1}, & \ldots, C_{(N-1)(M-1)} \end{bmatrix} \times \begin{bmatrix} X_0 \\ X_1 \\ \\ X_{M-1} \end{bmatrix} \qquad (2)$$

For the algorithms of this categiry there are two options: I) Sequential output computaton and II) Parallel output computation. The rest of the paper is organized as follows: In section 2 the target architecture is described. The proposed synthesis techniques are described in detail in section 3. In section 4 different cost functions are proposed. Experimental results for several digital signal processing algorithms are presented in section 5. Finally in section 6 conclusions are offered.

2. TARGET ARCHITECTURE

The proposed architecture can be used for both convolutional and transformational kernels. Foreground registers store the data terms required for the computation of each output term, which allows exploitation of the data re-use in both convolutional and transformational basic computations, favoring power consumption reduction. The computations are performed in a multiply/accumulator based data path, while bit-parallel implementations are assumed. A cental and a localized architecture are proposed in Figures 1 and 2 respectively.

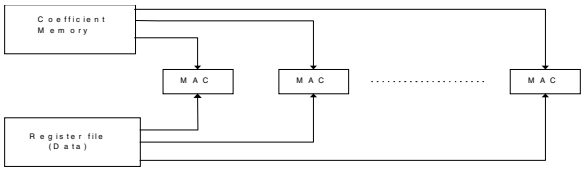

Figure 1. Centralized memory and bus organization

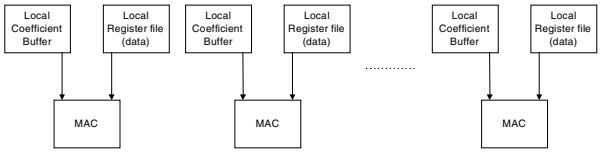

Figure 2. Localized memory and bus organization

3. PROPOSED SYNTHESIS TECHNIQUES

3.1 Synthesis Techniques for Independent Data Path Operations

The synthesis of these operations on a single functional unit requires only the scheduling of the operations.

A power related cost $PC_{i,j}$ is assigned to each pair (i,j) of operations. This cost is related to the power consumed when operations i, j are successively executed on the same functional unit. In the next step the graph of the operations is built with the vertices representing the operations and the edges the unconstrained transitions between operations. To each edge (i, j) the cost $PC_{i,j}$ corresponding to the operations i, j connected by the edge is assigned. The power efficient scheduling problem can be formulated as a Travelling Salesman's Problem (TSP) on the graph described above. The cost that is minimized during the scheduling procedure is given by the following equation:

$$cost\_function\_1 = \sum_{i=0}^{A-1} PC_{i,i+1} + PC_{A-1,0} \quad (3)$$

A large number of solutions have been proposed for TSP [8]. In the prototype implementation of the proposed synthesis techniques an exact algorithm is used for the solution of the TSP problem when the number of the graph nodes is smaller than 12. When the number of nodes is larger than or equal to 12 a fast non-exact heuristic based on genetic algorithms is used.

3.2 Synthesis of Sum-of-Products Computation for Single Functional Unit

3.2.1 Convolutional Algorithms

The convolution operation performed by the algorithms of this category for the evaluation of an output term is described by the following equation:

$$y_n = \sum_{f(k)=0}^{N-1} C_{f(k)} X_{n-f(k)} \quad (4)$$

The scheduling function f(k), k=0,…,N-1, generates for each partial product k a relative order of computation and it is one-by-one and onto. In the original case f(k)=k.

The total power cost (PC), of the sequence of evaluation of the partial products on a single functional unit as determined by the scheduling function f(k) is given by the following equation:

$$PC(f(k)) = \sum_{f(k)=0}^{N-1} PC(p_{f(k)}, p_{f(k)+1}) + PC(p_0, p_{N-1}) \quad (5)$$

where $p_{f(k)}$ is the partial product $c_{f(k)}x_{n-f(k)}$. The power cost for a pair of partial products $PC(p_k,p_l)$ is given by the power related cost function selected to drive the synthesis procedure. The aim of the proposed technique is to derive the ordering function f(k).

The scheduling problem for the convolutional algorithms is formulated as a TSP. The graph G (V, E) of the problem consists of the set V of the N vertices, which are the partial products for the computation of an output term, and the set E of edges which model the transition from one partial product to another. To each edge of the graph the power cost PC between the two partial products that the edge connects is assigned.

3.2.2 Transformational Algorithms

The sequential computation of the output points of a transformation of length N×M is described by the following equation:

$$y_{f(i)} = \sum_{g(i,j)=0}^{M-1} c_{f(i),g(i,j)} x_{g(i,j)} \quad (6)$$

for f(i)=0,…,N-1.

The computation is performed according to the scheduling functions f(i), g(i,j), i=0,1,2,…N-1, j=0,1,2,…M-1. The f(i) function determines the computation order of inner products while the g(i,j) function determine the order of partial products that form the inner products. The total power cost (PC) of the sequence of evaluation of the partial products that constitute the basic computation is given by the following equation:

$$PC(f,g) = \sum_{f(i)=0}^{N-1} \{ [\sum_{g(i,j)=0}^{M-1} PC(p_{f(i),g(i,j)}, p_{f(i),g(i,j)+1})] + PC(p_{f(i),N-1}, p_{f(i)+1,0}) \} \quad (7)$$

where $p_{f(i),g(i,j)}$ is the partial product $c_{f(i)g(i,j)}x_{g(i,j)}$. The power cost for a pair of partial products is given by the power related cost function selected to drive the synthesis procedure. The proposed techniques aim at deriving the scheduling functions f(i), g(i,j).

The formulation of the scheduling problem is more complex for the transformational algorithms. In a first step the minimum cost scheduling of the inner products IP_i of the basic computation is determined. In a second step the minimum cost scheduling of the partial products p_{ij} (j=0,1,…M-1) that constitute each inner product, must be found. The problem is tackled in the following way: For all possible pairs of inner products IP_i, IP_j, (i=0,1,…N-1, j=0,1,…N-1) the minimum power cost connection is determined. The minimum power cost connection is defined as the pair of partial products that minimizes the power cost between all possible pairs of partial products. The complexity of this procedure is $M^2 N (N-1)$.

Since the minimum power cost connection has been determined for all possible pairs of inner products the graph G (V, E) of the problem is constructed. where V is the set of N vertices corresponding to the N inner products while E is the set of edges that model the unconstrained transitions from one inner product to another. To each edge the power cost of the minimum cost connection corresponding to the inner products connected by the edge is assigned as a cost. The determination of a minimum cost ordering of the inner products is formulated as a TSP on this graph. As soon as the minimum power cost scheduling of the inner products is determined, the partial products that constitute each inner product must be scheduled to minimize the power cost required for its computation. Each vertex of the initial problem's graph is considered as a graph with vertices corresponding to the partial products that constitute the inner product. However the scheduling of the inner products performed in the first step results in the determination of the partial products that will be evaluated first and last for each inner product. Thus this problem can be formulated as a restricted minimum cost open path (over the vertices of the inner product graph).

The formulation of the scheduling problem in the parallel output computation case is similar to the formulation of the sequential output computation case. The inner products of the sequential output computation case are replaced by the sets of partial products that use the same input data term.

3.3 Synthesis of Sum-of-Products Computation for Multiple Functional Units

3.3.1 Convolutional Algorithms

In this case the basic computation is scheduled first assuming one functional unit implementation, according to the tehnique described in section 3.2. The resulted partial products are clustered sequentially. The number of clusters equals the number

of available functional units. The partial products are divided equivalently to each cluster for load balancing purposes and each cluster is assigned to a functional unit.

3.3.2 Transformational Algorithms

In case the number of available functional units equals to the number of inner products, the assignment is straightforward and one inner product is assigned to each functional unit. The scheduling of the partial products in each functional unit is performed in the same way as for the convolutional algorithms in the single functional unit case.

In case the inner products are more than the functional units, they are first scheduled assuming single functional unit implementation. The resulting sequence is then divided sequentially to the available functional units. The schedule of the partial products within each inner product assigned to the same functional unit, is achieved by using the same approach as for the single functional unit implementation of transformational algorithms.

4. POWER COST FUNCTIONS

The synthesis techniques presented earlier require a power related cost function (PC) that drives the optimization procedure. Two main power-related cost functions have been used:

4.1 Interconnect Oriented Power Cost Function

This cost function captures the power consumed in the buses connecting the storage elements to the functional units. Assuming two partial products $c_i d_i$ and $c_j d_j$ the cost function is given by the following equation:

$$PC_B(c_i d_i, c_j d_j) = CW_c \times HD(c_i, c_j) + CW_d \times AHD(d_i, d_j) \qquad (8)$$

where CW_c , CW_d are weights related to the capacitances of the coefficient and data buses, $HD(c_i, c_j)$ represents the Hamming distance between the coefficients of the partial products and can be directly evaluated since coefficients are known before realization and $AHD(d_i, d_j)$ is the Average Hamming distance between the data terms of the partial products and is determined through simulation.

The previously mentioned cost function includes static and dynamic components. It may become fully static by taking into consideration only the coefficients.

It must be noted that in deep submicron technologies the power consumed in the interconnect buses becomes more significant than the power consumed in the functional units.

4.2 Functional Unit Internal Oriented Power Cost Function

This cost function aims at reducing the power consumed in the functional units internally. Assuming two partial products $c_i d_i$ and $c_j d_j$ the cost function is given by the following equation:

$$PC_I(c_i d_i, c_j d_j) = \sum_{\forall k} a_k \times c_k \qquad (9)$$

where a_k is the switching activity in the kth node of the functional unit and c_k is the node capacitance.

In order to calculate the functional units related power weights, the method presented in [9] is used. This method is an incremental static (or probabilistic) power estimation approach, at gate level of abstraction.

For each weight the input statistics are extracted from the corresponding vector set of the input samples and the algorithm's coefficients. These statistics are propagated through the circuit nodes, using the method [9] and hence the corresponding power weight is estimated. The mean error in the power consumption that estimated is in the range of 5-10 % compared with the power consumption that is calculated using QuickSim II gate level simulator of Mentor Graphics.

5. EXPERIMENTAL RESULTS

Prototype software in ANSI C has been developed to implement the proposed synthesis techniques. Bit true simulations of ANSI C codes of the DSP kernels (using special instrumentation code as well) have been used to monitor the switching activities at the input buses of realizations of the DSP kernels. The data that have been used for the simulation of the algorithms are either typical image data or random data or outputs of intermediate computations like, for example, the output of horizontal part of level three of wavelet transform.

The input buses switching activity savings achieved for different coefficient bitwidths are shown in Figure 3.

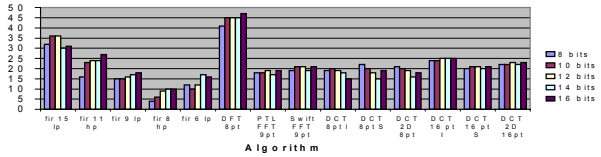

Figure 3. Single functional Unit Interconnect power savings (%) for different coefficient bitwidths.

In most cases interconnect power savings of more than 20% are achieved. An important observation is that the savings achieved do not increase proportionally to the coefficient bitwidth.

The effect of the presence of multiple functional units is evaluated in Figure 4. The main conclusion that can be drawn in this case is that the interconnect power savings achieved by the proposed techniques do not decrease as the number of functional units increase.

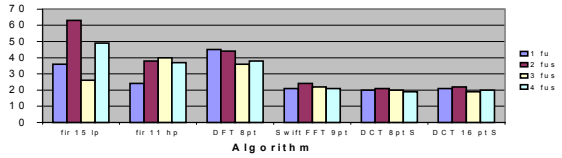

Figure 4. Interconnect power savings (%) for multiple functional units realizations

A comparison of the results achieved by taking into account the coefficients or not, are presented in Figure 5. In all cases the differences in the savings achieved by the two different cost functions are small while in some cases the savings are identical. This is important since no simulation effort is required for the evaluation of the fully static cost function speeding up significantly the synthesis process.

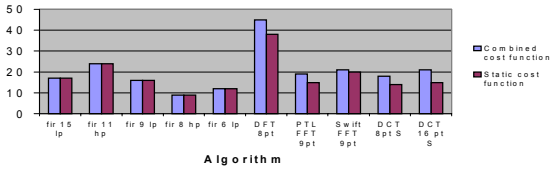

Figure 5. Comparison of the two cost functions

Results for the parallel output computation case of the transformational algorithms are presented in Figure 6.

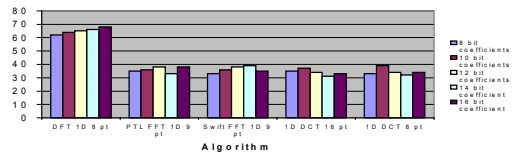

Figure 6. Savings for the parallel output computation case. Single functional Unit Implementation

The savings achieved for the parallel output computation case are larger in comparison to those achieved for the sequential output computation case. This is because no penalties in the data switching activity are introduced in comparison to the reference case.

The effect of the techniques when the target is the power consumed in the functional units internally has been evaluated as well. Bit parallel multiply-accumulators based on two different types of multipliers (Carry Save (CSA) array and a Booth encoded-Wallace multiplier) have been developed. Internal power has been estimated through gate level simulations using a target technology of 0.7 μm.

The results are presented in Table 1. Power figures are in μWs and have been obtained assuming a frequency of operation of 20 MHz, and a supply voltage of 5 volts. Larger reductions are achieved in the case of the carry save array multiplier.

Table 1. Effect of the proposed techniques on the functional units power consumption

Algorithm	Initial power Array-CSA	% reduction Array-CSA	Initial power Wallace-Booth	% reduction Wallace-Booth
Fir 11 hp	10505	22,9	12604	27,9
fir 9 lp	9225	19,2	12095	22,5
fir 8 lp	10897	2,0	10065	20,9
PTL FFT 9	17312	12,8	14267	6,3
Swift FFT 9	19300	33,7	14982	7,3
DCT 8 (1)	12517	11,1	14114	6,6
DCT 8 (2)	4620	15,4	6026	7,7
DCT 8 (3)	9129	12,7	11734	7,9
DCT 8 (4)	45481	5,8	33762	5,8
DCT 8 (5)	17474	16	16064	6,9
DCT 8 (6)	34953	32,8	31886	43,6
Average	22428	16,8	17493	14.9

6. CONCLUSIONS

Novel data path synthesis techniques for sum-of-products computation have been presented. The proposed techniques take advantage of special characteristics of sum-of-products computation. Simple and efficient scheduling and assignment heuristics based on the Travelling Salesman's Problem have been described. Cost functions targeting different components of the total power dissipation that can be partly statically evaluated have been proposed. Experimental results have proven that the proposed techniques lead to 25% savings in terms of the interconect power and to 15% savings in terms of functional units power on average.

7. REFERENCES

[1] J. M. Rabaey, M. Pedram, “Low Power Design Methodologies”, Kluwer Academic Publishers 1995.

[2] A. Raghunathan, N. Jha, "An ILP Formulation for Low Power based on Minimizing Switched Capacitance during Data Path Allocation", Proceedings 1995 IEEE 1995 International Symposium on Circuits and Systems (ISCAS'95).

[3] E. Musoll, J. Cortadella, "High-Level Synthesis Techniques for Reducing the Activity of Functional Units" proc. of the 1995 Intl. Symposium on Low Power Design.

[4] A. Chatterjee, R. Roy, “Synthesis of Low Power Linear DSP Circuits using Activity Metrics”, proc. of the 7th Intl. Conference on VLSI Design-January 1994, pp. 265-270.

[5] H. Nguyen, A. Chatterjee, R. Roy, "Activity Measures for Fast Relative Power Estimation in Numerical Transformation for Low Power DSP Sythesis", Journal of VLSI Signal Processing 18, pp. 25-38, Kluwer Academic Publishers 1998.

[6] M. Mehendale, S. D. Sherlekar, and G. Venkatesh, “Coefficient optimization for low power realization of FIR filters”, in IEEE Workshop on VLSI Signal Processing, Japan, 1995, pp.352-361.

[7] M. Tien-Chien Lee, V. Tiwari, S. Malik, and M. Fujita, “Power Analysis and Minimization Techniques for Embedded DSP Software”, IEEE Transactions on VLSI Systems, Vol. 5, No. 1, pp. 123-135, March 1997.

[8] C. H. Papadimitriou, K. Steiglitz, “Combinatorial Optimisation: Algorithms and Complexity”, Prentice-Hall, Inc., Englewood Cliffs, 1982.

[9] S. Theoharis, G. Theodoridis, N. Zervas and C. Goutis, “Accurate And Fast Power Estimation Of Large Combinational Circuits”, 9th Int. Workshop PATMOS’ 99, pp 199-208.