PowerShake: A Low Power Driven Clustering and Factoring Methodology for Boolean Expressions

Sumit Roy Harm Arts Prithviraj Banerjee†
Ambit Design Systems Center for Parallel & Distributed Computing
Santa Clara, USA Northwestern University, USA
sroy, harm@ambit.com banerjee@ece.nwu.edu

Abstract

This paper describes algebraic techniques that target low power consumption. A unique power cost function based on decomposed factored form *representation of a Boolean expression is introduced to guide the structural transformations. Circuits synthesized by the SIS [5] and POSE [1] consume* **54.5%** *and* **10.4%** *more power than that obtained by our tool respectively.*

1 Introduction

Recent trends in digital circuit technology have made power optimization an important factor in the design of digital systems. Although, power optimization can be applied at different levels of the design, in this work, we focus on algebraic restructuring techniques, namely algebraic factorization and collapsing, to reduce power at the combinational logic level.

Previous research efforts have been in identifying suitable power cost function for guiding the logic synthesis. In [4], the power gain for extracting a subexpression was defined as the difference between the power cost of the expression (based on the same factored form) before and after the extraction. One shortcoming of [4] is that once the sub-expressions are extracted, the expression will not necessarily have the assumed factorized form. In [1], a cost function based on the sum-of-products(**SOP**) form of a expression was introduced to alleviate the above inaccuracy. Although it performs well for two-level implementation of circuits, it can be potentially inaccurate for multi-level implementation since the cost function in [1] can not distinguish between difference in the power consumed by the several factored forms of a given expression. In this paper, a new cost function is defined that estimates accurately the average power consumption of the mapped circuit. We propose a new methodology based on iterative clustering and algebraic factoring.

2 Cost function

In this section, we introduce an alternative approach for computing the power cost based on the factored form representation [7] of a Boolean expression since we believe that it resembles the structure of a multi-level circuit implementation.

2.1 Decomposed Factored Form

A factored form Boolean expression can be represented by an unique *F-tree* [7] $T(V, E)$ with vertex set V and edge set E. Edges represent the direction of the signal flow. Each vertex of V represents a part of the Boolean expression and would be refered to as an internal node of the F-tree or simply *node* henceforth. Given a factored form of a Boolean expression, x, the F-tree $T_x(V_x, E_x)$ can be computed in linear time by simply parsing the factored form and merging adjacent nodes of similar operator type.

The power cost of x based on the given *F-tree*, $T_x(V_x, E_x)$ is defined as the sum of the estimated *switching capacitance*, $\rho(v)$, (defined later) over all the vertices, v, of the tree T_x. Since a node v can represent a multi-input AND/OR gate, we estimate the power consumed by the internal nodes of v by *implicitly* decomposing it into a tree of two input gates only. We used the modified Huffman algorithm presented in [3] to decompose an n-input OR/AND gate into a binary tree. Based on the above decomposition, the average *switching capacitance* of a node v of the *F-tree* is given by:

$$\rho(v) = \sum_{n \in dec(v)} [C_o.p_t(n) + C_i.(p_t(lchild(n)) + p_t(rchild(n)))] \quad (1)$$

where C_i and C_o are the input and output estimated capacitance of a 2 input AND/OR gate, $p_t(x)$ is the transition probability [2], $dec(v)$ is the set of nodes of the decomposed binary tree of v and *lchild* and *rchild* are the left and the right child of n. Note that the power cost is defined for a Boolean expression whereas switching capacitance is defined for every node of the decomposed F-tree.

The above power cost function is more accurate than that used in [1] in the two ways: it uses the factored form of the current expression which correlates to the multi-level implementation instead of the sum of product representation; it accounts for the power consumed by the internal nodes of all multi-input nodes of an F-tree by performing the optimal Huffman decomposition.

3 Methodology

In this section, we will describe our synthesis methodology using algebraic techniques. Given a multi-level circuit(SOP form is a special case), we perform clustering and algebraic factorization alternately while there is any improvement in the circuit.

This research was supported in part by the Semiconductor Research Corporation under Contract SRC 95-DP-109 and the Advanced Research Projects Agency under contract DAA-H04-94-G-0273 administered by the Army Research Office.

3.1 Algebraic Factorization

Algebraic factorization identifies common expressions and introduces them as new nodes into a network in order to optimize a given cost function. A recent algebraic factorization technique extracts only two-cube divisors and two-literal single-cube divisors both in normal and complement form [6]. It uses the weight of a divisor to measure the magnitude of area saving it brings about. We have extended it for power optimization by introducing our power cost as the weight of the divisors.

We estimate the gain in power of extracting a divisor by taking the difference of the power cost of the Boolean expression before and after the extraction. Note that we refactorize the extracted expression to obtain its power cost unlike the approach in [4].

3.2 Clustering

Clustering is the technique of selectively collapsing a set of Boolean expressions into a single expression such that it provides a better structure for the circuit. It tries to cluster nodes with high switching activity into a single expression such that the latter can be implicitly decomposed into nodes with smaller switching activity.

Example 3.1 *Let F = K*L, G=L+e, H=L+f, K=a*b, L=c*d be a given circuit with* $p_s(a) = 0.6, p_s(b) = 0.8, p_s(c) = 0.2, p_s(d) = 0.95, p_s(e) = 0.5$ *and* $p_s(f) = 0.5$. *($p_s(x)$ = signal probability of x [2])*

The initial circuit has expressions K and L with high switching activity($p_t(K)$=0.499, $p_t(L)$=0.308). Clustering identifies them, collapses into F and finally implicitly decomposes it into *F=m+d, m=n+b, n=a+c* by introducing less active nodes m and n($p_t(m)$=0.174, $p_t(n)$=0.211). Clustering proceeds as follows: each expression x has all its fanout $y_1, y_2, \ldots, y_k, k > 0$ as clustering candidates. It evaluates the transformation of collapsing into y_i by estimating the new value of $P(y_i')$. As part of the estimation, y_i' gets implicitly decomposed and its power cost calculated. If the new power cost of y_i' is better than the old cost of y_i, x is collapsed into y. If x gets collapsed into all its fanouts, the power cost associated with it gets added to the power weight. In [1], a similar technique known as *selective collapse* has been reported where it selectively eliminating nodes in a network in order to reduce the power cost of the network. Our clustering algorithm is superior to the published algorithm for selective collapse [1] in two ways: it restructures circuits to provide a better power structure using implicit decomposition; it can partially collapse a node into some of its fanouts which the approach in [1] cannot.

4 Experimental Results

We developed a tool **PowerShake** which takes in a multi-level network and the input switching activity and performs algebraic restructuring. Initially the switching activity of all the internal nodes are computed based on the input switching activity. Then it iteratively performs clustering and algebraic factorization based on the techniques discussed in Section 3 on the expressions in the circuit till the average power consumption of the circuit cannot be reduced.

Table 1: Comparison between SIS, POSE and PowerShake(a*=area, p*=power (μW), -= timeout after 8 hours.)

Circuit	SIS		POSE		PowerShake		% Imp	
	a1	p1	a2	p2	a3	p3	p31	p32
squar5	191	442	201	424	81	251	76	69
sct	195	677	145	384	105	255	165	51
sqrt8ml	191	643	144	417	122	374	72	12
unreg	147	725	155	465	155	465	56	0
comp	217	791	251	758	210	504	57	51
i3	178	903	178	903	206	679	33	33
o64	186	835	-	-	209	618	35	-
9symml	331	1227	344	1057	403	1023	20	3

We compared our tool, PowerShake, with SIS [5] (*script.algebraic*) and POSE [1] (*script.power.s*) on the first 35 circuits(sorted according to initial size) from the MCNC benchmark. In each case, we optimized the circuit using each of the tool(SIS, POSE, PowerShake), mapped it to *mcnc.genlib* using *map -m 0* in SIS and estimated the zero-delay power using *power_estimate* in SIS at 20MHz. We noticed that SIS optimized circuits consume **54.5%** more power than that obtained by PowerShake on an average over the 35 circuits. Also, POSE optimized circuits on an average consume **10.4%** more power than our approach. In particular, it can consume upto **77.19%** more power than obtained using our tool. Due to shortage of space, we tabulate the results obtained by the 8 largest circuits amongst the given 35 MCNC circuits in Table 1. (*p31=(p3/p1-1)*100, p32=(p3/p2-1)*100*)

In conclusion, we have proposed a novel and accurate power cost function based on *decomposed factored form* representation of a Boolean expression. We also developed a synthesis methodology based on clustering and factorization. Finally, our experimental results show that our approach is quite promising.

References

[1] S. Imam and M. Pedram. Logic extraction and factorization for low power. In *Design Automation Conference*, June 1995.

[2] F. Najm. A survey of power estimation techniques in vlsi circuits. *IEEE Transactions on VLSI systems*, 2(4):446–455, Dec. 1994.

[3] W. Noth and R. Kolla. Decomposition in low power technology mapping. In *International Symposium on Low-Power Electronics and Design*, Monterey, Aug. 1997.

[4] K. Roy and S. Prasad. Circuit activity based logic synthesis for low power reliable operations. *IEEE Transactions on VLSI systems*, pages 503–513, Dec. 1993.

[5] R. K. Brayton, A. Sangiovanni-Vincentelli, et. al. SIS: A system for sequential circuit synthesis. Technical Report UCB/ERL M92/41, Department of Electrical Engineering and Computer Science, University of California, Berkeley, CA, May 1992.

[6] J. Vasudevamurthy and J. Rajski. A method for concurrent decomposition and factorization of boolean expressions. In *Proceedings of the Design Automation Conference*, San Diego, CA, June 1990.

[7] A. R. Wang. Algorithms for multi-level logic optimization. Ph.D. Dissertation, University of California, Berkeley, 1989.