PLA Minimization for Low Power VLSI Designs

Sasan Iman, Massoud Pedram, and Chi-ying Tsui

CENG Technical Report 95-27

Department of Electrical Engineering - Systems
University of Southern
Los Angeles, California 90089-2562
(213)740-4458

December 1995

PLA Minimization for Low Power VLSI Designs

Sasan Iman, Massoud Pedram

Department of Electrical Engineering - Systems
University of Southern California

Chi-ying Tsui

Department of Electrical and Electronics Engineering
Hong Kong University of Science and Technology

Abstract

In this paper we study the problem of optimizing the two-level representation of a Boolean function in order to minimize power consumption in PLAs. We first give power models used to estimate the power consumption in pseudo-NMOS and dynamic PLAs. Using these power cost functions we then prove that a minimum power solution for dynamic PLAs consists only of prime implicants of the function. For pseudo-NMOS PLAs we show that for incompletely specified multiple output functions, the minimal solution may contain non-prime implicants. We then describe exact and heuristic solutions for minimizing the power consumption of a Boolean function implemented using PLAs. We finally use the results of our experiments to draw conclusions on the effectiveness of low power two-level function minimization for PLAs.

1 Introduction

Recent advances and trends in the electronics industry has made power consumption an important factor in the design of digital systems. Increasing popularity of portable devices has resulted in a demand for longer battery life and therefore lower power consumption. At the same time, power dissipation is becoming one of the limiting factors in the amount of logic that can be placed on a chip. Therefore by reducing power, chip density can be increased. Environmental and economical factors are also contributing to the demand for low power systems.

A power conscious design methodology will have to consider power at all stages of the design process. The problems for estimating and optimizing power consumption at all levels of the design hierarchy has been addressed by researchers in the past few years [3][4][7][8][11].

Two-level logic minimization for area is an important part of logic synthesis algorithms both for minimizing area of programmable logic devices and for minimizing area of multi-level circuits. This problem has been addressed by many researchers. ESPRESSO[1] presents a heuristic approach where novel techniques are used to efficiently produce good area solutions. ESPRESSO EXACT [9] presents an exact method for solving the minimum area solution. The problem of exact two level logic minimization for minimum area is stated as follows:

Problem: *Given a Boolean function f with input set* $V = (v_1,...v_N)$*, find a two-level implementation of the function f such that the number of product terms and then the number of literals in the sum-of-products form is minimum.*

This problem is solved by first generating the set of all prime implicants of the function[1]. The minimum area solution is then obtained by solving an exact minimum covering formulation of the problem where a subset of the primes are selected to cover the function.

Minimizing the number of product terms and literals of a logic function is independent of the technology being used to implement the function. This means that the same optimized implementation may be used for static and dynamic logic circuits. The cost functions used for measuring power however are different in static and dynamic circuits[5]. This means that different strategies need to be developed for different technologies. The goal of this paper is to provide function minimization techniques to be used in optimizing power consumption in PLAs.

Recent work on minimizing the power consumption of Boolean functions have concentrated on heuristic approaches. In [10] a method was presented where the power cost function of the primes were used to solve an ESPRESSO-like procedure in order to minimize the contribution of the node implementation to the power cost of a multi-level network. This procedure does not consider including non-prime implicants in the cover of the function although these implicants can lead to more power reduction. More recently, a heuristic approach for minimizing the power consumption of a function in CMOS circuits was presented in [12]. The major shortcoming of this

approach is that it requires the same signal probability for all inputs of the function which in general does not represent realistic conditions.

In this paper, we address the problem of minimizing power consumption in two-level logic circuits implemented as PLAs. The loading information in PLAs are known for a specific technology/implementation and hence the switched capacitance can be minimized directly. In particular, we show how logic minimization for area is modified to obtain a minimum power solution.

The rest of this paper is organized as follows: In section 2 we discuss the power model used in minimizing the power consumption. In section 3 we show how prime implicants correspond to implicants needed for power optimization for both pseudo-NMOS and dynamic PLA implementations. In section 4 we use the results in section 3 to minimize power consumption in PLAs. Results and conclusions are presented in sections 5 and 6.

2 Power cost function

Normally, high speed PLAs are built by transforming the SOP representation of a two level logic to the NOR-NOR structure with inverting inputs and outputs and implementing it with two NOR arrays (Figure 1). Two common types of implementing NOR arrays are pseudo-NMOS NOR gates and dynamic CMOS NOR gate. In this paper we assume that the functions are imple-

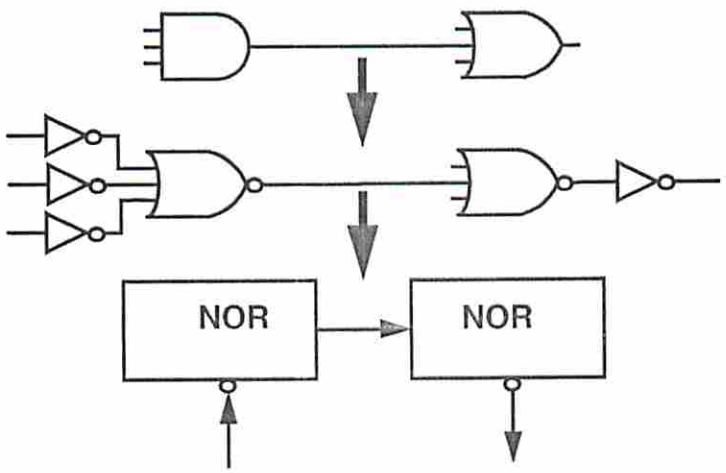

Figure 1: NOR-NOR PLA

mented using a PLA that is driven using static CMOS drivers. This means that the cost function for an implicant in the final cover of the function assumes that the AND gate is implemented using either pseudo-NMOS or dynamic technology while the input power consumption follows the power cost model for static CMOS structures.

In static CMOS circuits, dynamic power required to charge and discharge the load capacitance at the output of a gate g is given by:

$$Power_{dyn} = \frac{V_{dd}^2 \cdot f}{2} c_{load}^{g} E_g(switching) \quad (1)$$

where c^g_{load} is the load capacitance at the output of the gate, V_{dd} is the supply voltage, f is the clock frequency, and $E_g(switching)$ is the expected number of transitions per clock cycle. Each input literal of AND term in the final implementation of the function creates additional load on the input CMOS drivers. Equation 1 will be used to measure the power contribution of the PLA input literals on the total power consumption.

The primary source of power consumption for a pseudo-NMOS NOR gate is the static power dissipation (Figure 2). When a pseudo-NMOS NOR gate evaluates to zero, both the PMOS and NMOS parts of the gate are conducting and there exists a direct current path. The charging and discharging energy is small compared with that dissipated by the direct current when the frequency of operation is not extremely large. Furthermore, the direct current I_{dc} is relatively constant irrespective of the number of NMOS transistors that are on. In this paper we assume that static power dominates in pseudo-NMOS PLAs. The power cost at the output of a product term p is given by:

$$OutPower(p) = V_{dd} \cdot I_{dc} \cdot sp_p^0 \quad (2)$$

where sp^0_p is the probability that the product term p evaluates to 0. The total power cost of a product term which accounts for power consumptions at its inputs and output is then given by:

$$Power(p) = \frac{V_{dd}^2 \cdot f}{2} \cdot \sum_{i=1}^{k} c_i E_i(switching) + V_{dd} \cdot I_{dc} \cdot sp_p^0 \quad (3)$$

where c_i gives the load due to product term p on input i.

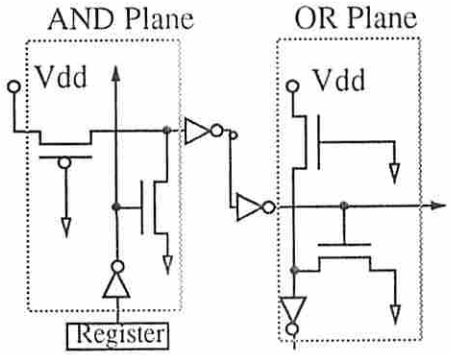

Figure 2: Pseudo-NMOS PLA

In a dynamic PLA circuits (Figure 3), dynamic power consumption is the major source of power dissipation. The output of the product term is pre-charged to 1 and switches when it evaluates to 0. Therefore the power consumption at the output of a product term p is given by:

$$OutPower(p) = \frac{V_{dd}^2 \cdot f}{2} \cdot c_p \cdot sp_p^0 \tag{4}$$

where c_p is the load seen at the output of the product term and is estimated as the number of gates in the OR-plane that p fans out to.

The total power cost due to a product term p is obtained by taking into account the power consumption on the inputs and clocks and is given by:

$$Power(p) = \frac{V_{dd}^2 \cdot f}{2} \cdot \left(\sum_{i=1}^{k} c_i E_i(switching) + C_p SP_p^0 + c_{clock} \times 2 \right) \tag{5}$$

where c_{clock} is the load capacitance of the pre-charge and evaluate transistors that the clock drives.

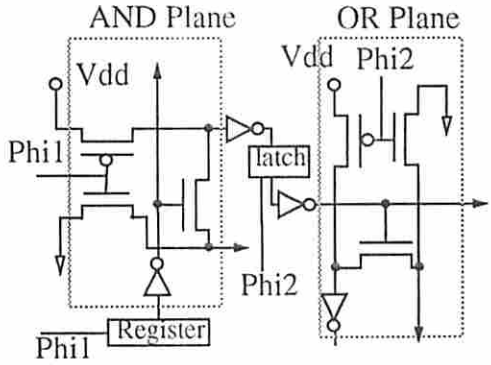

Figure 3: Dynamic PLA

The power due to the OR plane of the PLA is measured using equations 2 and 4 presented for the product terms since both the input and output planes of the PLA are implemented using NOR gates.

3 Prime implicants and Power Optimization

The two level logic minimization problem for low power is equivalent to finding a cover *C(F)* such that the following objective function is minimized:

$$\sum_{p \in C(F)} Power(p) + \sum_{q \in O(F)} OutPower(q) \tag{6}$$

where O(F) is the set of gates in the OR plane.

For area minimization, it has been shown that it is sufficient to consider the set of prime implicants of a function based on the assumption that the area cost of a prime implicant is always lower than or equal to that of any implicant it contains. In the following, we examine whether this assumption holds for power minimization. We discuss the cases for pseudo-NMOS and dynamic

PLAs separately.

3.1 Pseudo-NMOS PLA

We first examine the effect of using don't cares on the power cost of a pseudo-NMOS PLA.

The first level NOR gate in a pseudo-NMOS PLA will draw direct current when the AND term evaluates to 0. Similarly the second level NOR gate will draw direct current when the gate output is 0 (i.e., the function output evaluates to 1). Including don't care in the cover will reduce the probability of the AND term evaluating to 0 and hence reduces power consumption at the output of the AND term. At the same time, it will increase the probability of the output evaluating to 1 which increases the power consumption.

The example in Figure 3 shows how don't cares may be used to make power trade-offs at the output of the AND plane and OR plane. Assuming each variable has 0.5 signal probability and $V_{dd}I_{dc(AND)}=1$, the Outpower cost of implicant P_1 (given by equation 2) is 0.75 while that of implicant P_{11} is 0.875. Therefore Outpower cost is reduced when P_1 is used instead of P_{11}. A Similar statement holds for P_2 and P_{21}. However, if P_1 and P_2 are used instead of P_{11} and P_{12}, the Outpower cost at the output of the OR plane will be increased from 0.25 to 0.375.

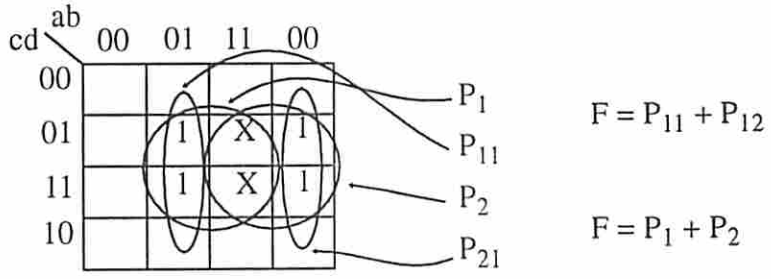

Figure 4: Including Don't Cares in the cover

PLA is a regular structure and the transistor sizes for the OR plane and the AND plane are identical. Therefore $I_{dc(AND)}$ and $I_{dc(OR)}$ are the same. Based on this observation, the following theorem shows that including don't cares in the cover of a single output function, in the worst case, will keep the power of the PLA unchanged, while in most cases, it will reduce the power consumption.

Theorem.1: *Including don't cares in the logic cover of a single output Boolean function will not increase the power cost of the final implementation of a pseudo-NMOS PLA.*

Proof: Let P be the set of minterms covered by the product term AND, and D be the subset of don't care minterms in P. When we subtract D from P, in the best case, the number of product terms remains unchanged. The minimum increase in power cost (due to the increase in $sp_{AND}{}^{0}$) is given by:

$$Power_{inc} = V_{dd} \cdot I_{dc} \cdot \sum_{dc \in D} sp_{dc}^{1} \tag{7}$$

The best reduction in power cost at the output when D is removed from the cover, occurs when D is only covered by p but no other product term. Hence, the maximum decrease in power is given by:

$$Power_{red} = V_{dd} \cdot I_{dc} \cdot \sum_{dc \in D} sp_{dc}^{1} \tag{8}$$

If some of the minterms in D are covered by other product terms in the AND plane, P_{red} will be smaller. From equations 7 and 8, removing don't care from the cover cannot increase the power cost.

■

Note also that when don't cares are included in a product term, in general the number of inputs to the product term is also reduced. This will in turn reduce the power due to the input literals. The preceding analysis shows that without considering the power at the input, including don't cares at worst will not increase the power. Considering the reduction in power on the inputs, including don't cares will always result in reduction in power for a single output function.

The statement of theorem 1 does not in general hold for incompletely specified multiple output functions. This is because the power cost of a product term in the AND plane of a pseudo-NMOS PLA is independent of the load at its output. Consider a product term p and OR-terms $(o_1,..., o_n)$ that p fans out to. Removing don't care points from p will result in an increase in the power cost of p. Note however that this increase is independent of the number of fanouts. Now removing don't care from p will in turn result in a decrease in the power cost of OR-plane terms $(o_1,..., o_n)$ if the don't cares are true don't cares for more than one OR-terms and at the same time are not covered by other product terms. In general it is possible for the reduction in power at the OR plane to be larger than the increase in power of p and therefore reducing the overall power cost. We will discuss our approach for dealing with multiple output functions in section 4.2.

The following theorem shows that power cost of a product term in a single output Boolean function is no more than any product term which it contains.

Theorem.2: *Let power cost of a product term P for a single output Boolean function be given as:*

$$V_{dd} I_{dc} sp_{P}^{0} \tag{9}$$

Let P_1 be a product term that is contained in P, then $Power(P) \leq Power(P_1)$.

Proof: Let M and M_1 be the sets of minterms in P and P_1 respectively. M_1 is a subset of M and let D be the difference of them. If D does not have any don't care in it, then in any case D must be included in the function output and hence including D in the prime implicant P will not increase

power cost at the NOR gate of the output plane. Thus we have

$$\begin{aligned} PowerCost(P) - PowerCost(P_1) &= V_{dd} \cdot I_{dc} \cdot \left(\sum_{i \in M} sp_i^0 - \sum_{j \in M_1} sp_j^0 \right) \\ &= V_{dd} \cdot I_{dc} \cdot \left(\sum_{j \in M_1} sp_j^1 - \sum_{i \in M} sp_i^1 \right) \\ &= -V_{dd} \cdot I_{dc} \cdot \sum_{k \in D} sp_k^1 \leq 0 \end{aligned} \tag{10}$$

In the case that D includes don't care points, let D_1 be the sets of don't care in D. From theorem 1, including don't care will not increase the power cost of a product term. Therefore in the worst case equation 10 becomes:

$$PowerCost(P) - PowerCost(P_1) = -V_{dd} \cdot I_{dc} \cdot \sum_{k \in D_s} sp_k^1 \tag{11}$$

where D_s is the difference of D and D_1 and D_1 is only covered by P. In the worst case, D_1 is equal to D and equation 11 is equal to zero.

■

3.2 Dynamic PLA

We assume that NOR gates in the AND and OR planes drive the same capacitive loading. In this case, the effect of using don't care in the logic cover will be the same as the case for pseudo-NMOS PLAs. Also note that for dynamic PLA implementations, this results also holds for multiple output functions. This is easily proved by noting that AND plane power for dynamic CMOS PLAs is a function of its load. This load is proportional to the number of OR-terms that this product terms fans out to. Removing don't cares from a product term p will result in even more increase in power as the number of fanouts of a p increases. This is in contrast with pseudo-NMOS circuits where the power in the AND plane is independent of its load. We consider equation 5 for the power cost of a product term for dynamic PLA. The final term in this equation ($2*c_{clock}$) is a constant term and is dropped in the optimization.

Theorem.3: *Let Power cost of a product term P be given as:*

$$\frac{V_{dd}^2 \cdot f}{2} \cdot \left(\sum_{i=1}^{k} c_i E_{x_i}(switching) + c_p sp_p^0 \right) \tag{12}$$

Let P_1 be a product term that is contained in P, then Power(P) ≤ Power(P_1).

Proof: Let p and p_1 be the set of literals used in P and P_1. p is a subset of p_1 since P contains P_1. Therefore:

$$\sum_{x_i \in P} c_i \cdot E_{x_i}(switching) - \sum_{x_j \in P_1} c_j \cdot E_{x_j}(switching) \leq 0 \tag{13}$$

Also from equation 12, it is shown that:

$$sp_P^0 \leq sp_{P_1}^0 \tag{14}$$

If we replace P by P_1 in a cover, the number of product terms in the sum of product form in the logic cover will, at the best case, unchanged. Therefore:

$$C_P \leq C_{P_1} \tag{15}$$

Combining equations 12, 13 and 14 proves the claim of this theorem.

■

This means that only prime implicants need to be considered in search for the minimum power solution in PLA optimization.

4 Two level function minimization

Traditionally two level logic minimization for PLA targets for minimum area. PLA is a regular structure and its area is proportional to (number of inputs + number of outputs) × (number of product terms). Therefore the problem of minimizing the PLA area is equivalent to the problem of finding the minimum number of product terms to implement the Boolean function. For power minimization, the objectives is to minimize the power consumption at both the AND plane and the OR plane. In the following sections, we show the relationship between optimizing area and optimizing power for PLA minimization and then describe how algorithm for area minimization are modified to minimize power consumption.

4.1 Area optimization

Since the area of a PLA is proportional to the sum of inputs and outputs multiplied by the number of AND terms (or product terms), hence minimizing area is equivalent to minimizing the number of product terms. Also PLA is usually synthesized by tiling characteristic cells according to the personality matrix. The PLA area is relatively independent of the implementation technology. Therefore the minimization algorithm that target minimum area work for different types of technologies. It is easy to see that if the cost of a product term is not larger than any product term which it contains, then a minimum cost solution exists for the two level minimization problem which consists only of prime implicants. For PLA area minimization, the cost of a product term is equal to the area for adding a row to the PLA and is constant for every product term, therefore the above condition is satisfied and we only need to consider prime implicants during the minimization process. The classic solution of finding a minimum cover for a function is the Quine-McClus-

key algorithm which consists of the following steps:

1-Generate all of the prime implicants.
2-Form the prime implicant table.
3-Derive a minimum cover of this table.

For generating all prime implicants, some algorithms resort to explicit enumeration of the minterms which in worst case is exponential in complexity. Even if the prime implicants are derived without enumerating minterms, there exist functions with an exponential number of prime implicants as a function of the number of implicants in a minimum cover. Also the minimum-cover problem itself is NP-complete.

Rudell [9] proposed an exact algorithm to solve the two-level minimization problem based on the motivation that the problems faced in reality do not have the worst case behavior. Also an exact solution can be used to indicate the quality of the heuristic methods. Heuristic methods such as ESPRESSO[1] were proposed to solve problems with large number of inputs and product terms. ESPRESSO uses an iterative improvement to achieve confidence in the optimality of the final result. Instead of generating all prime implicants and finding a minimum cost cover, implicants are expanded to prime implicants and then an irredundant cover is generated from the expanded prime implicants. This approach eliminates the basic problem of explicit generation of prime implicants. To avoid local minima, after the expansion and removal of covered implicants, the remaining implicants are maximally reduced while still maintaining a cover. Then the expansion process is repeated and the entire procedure is iterated until no improvement is obtained. The sequence of operations carried out by Espresso is as follows:

1-Compute the *Off-set* and the *DC-set.*
2-*Expand* each implicant into a prime and remove redundant primes. During the expansion process, "blocking matrix" and "covering matrix" are formed to guide the choosing of the *lowering set* and the *raising set* which are the sets of literals that are going to be lowered and raised in the expanded prime implicants respectively.
3-Extract the *essential primes* and put them in the *DC-set.*
4-Find an *irredundant cover* such that the cardinality of the cover is minimum.
5-*Reduce* each prime implicant in the irredundant cover to a minimum essential implicant.
6-Iterate steps 2, 4, and 5 until no further improvement is made.
7-Try steps 5, 2, and 4 one last time using a different strategy. If successful, continue the iteration.
8-Include the essential primes back into the cover and make the PLA structure as sparse as possible.

4.2 Power Optimization

Only prime implicants need to be considered for completely specified single output functions and incompletely specified multiple-output functions which target dynamic CMOS and pseudo-NMOS implementations. In these cases, standard Quine-McCluskey procedure is used to find the minimum power solution. In building the prime implicant table, each column representing a

prime implicant is tagged with the power cost given by equation 2 or 5.

For an exact solution, the algorithm described in Espresso-Exact [9] is modified by using appropriate weight functions during the branch-and-bound process that is used to find a minimum cost cover. Instead of using the number of prime implicants in the partial solution as the current cost, we use the sum of the power costs of the corresponding prime implicants. In case of incompletely specified single output function, we also have to add the power cost incurred at the output functions which is caused by including don't cares in the cover. Therefore the power cost of a partial cover is given by:

$$\sum_{p \in PartialCover} PowerCost(p) + \sum_{d \in DC_{PartialCover}} V_{dd} \cdot I_{dc} \cdot sp_d^1 \tag{16}$$

where $DC_{partialCover}$ is the set of don't care that is included in the partial cover. $DC_{PartialCover}$ can be found by augmenting the prime implicant table by adding a row for each element in the DC_set of the function. If a row is covered by one of the prime implicants already in the partial cover, then the corresponding don't care point is in $DC_{PartialCover}$.

For incompletely specified multiple output functions that target pseudo NMOS PLAs, it is possible to obtain a lower power implementation by removing don't cares from product terms. We define a *pseudo prime implicant* as follows:

Definition.1 *Given a Boolean function f and its don't care set, a pseudo-prime implicant is defined as an implicant p such that if it is possible to expand p in any direction, then the new points in the on-set of p will only include don't care points.*

Note that for a function with no don't care, the set of pseudo-prime implicants are identically the same as the set of prime implicants of the function. Also note that each pseudo-prime implicant P_p corresponds to a prime implicant P of the function where P_p and P contain the same on-set points of the function. We solve the incompletely specified multiple output pseudo-NMOS problem by using the minimum covering problem to select the set of pseudo-prime implicants which result in minimum total power cost. Once a minimum power solution is obtained, we expand each pseudo-prime implicant in the cover after checking to make sure this expansion does not increase the total power. Note that this approach does not guarantee exact optimal solution. Another approach for solving the problem for incompletely specified multiple-output pseudo-NMOS PLA is to solve the exact covering problem using the prime implicants of the function and then reduce each prime implicant in the final cover if this reduction results in a decrease in total power. Special cases may also be considered during this step. For example theorem 1 states that if a don't care set is only included in one output function, then that don't care set may freely be included in the on-set.

For a heuristic solution, Espresso can be used with the following modification. Let $C(F)$ be a

cover of the function F. If every implicant in $C(F)$ is used in the final implementation, the power cost is given by:

$$\sum_{I \in C(F)} PowerCost(I) \tag{17}$$

If an implicant is expanded to a prime implicant PI which covers implicants in $C_1 \subset C(F)$, then including PI in the cover will reduce the power cost by:

$$\sum_{I \in C_1} PowerCost(I) - PowerCost(PI) - \sum_{dc \in DC} PowerCost(dc) \tag{18}$$

where DC is the set of additional don't care introduced in the output when PI is included in the cover. During the expansion step, each implicant is expanded to prime such that the reduction in power cost specified in equation 18 is maximized.

While finding an irredundant cover, the cover $C(F)$ that minimizes the following cost function is generated:

$$\sum_{I \in C} PowerCost(I) + \sum_{dc \in DC} PowerCost(dc) \tag{19}$$

For the reduction step, the prime implicant is reduced such that after reduction, the cover $C(F)$ has minimum power cost given by equation 19.

5 Results

The procedure for ESPRESSO-exact was modified to obtain low power implementations of pseudo-NMOS and dynamic CMOS PLAs. Table 1 gives the results of these experiments for dynamic CMOS PLAs. Columns 1, 2 and 3 give the number of product terms, number of literals in the SOP form and the power of the resulting implementation when the examples are minimized for minimum area. Columns 4, 5 and 6 gives the same values when the circuits are optimized for low power. Note that the values in the last three columns are normalized with respect to their corresponding values in the first three columns. It can be seen that optimization for power has resulted in a 5% increase in the number of cubes of the function while reducing the power by an average of 11%. In general optimizing the functions for power has also resulted in functions with fewer literals in the SOP form. For pseudo NMOS PLAs, the power reduction is not as significant as that of dynamic CMOS PLA. The reason is that the power due to input literals is small compared to the static power drawn in the NOR gates. Therefore the effect of the number of product term used on power consumption is more significant than the input literal power. The minimum area solution which gives a minimum number of product terms will hence provide a good solution for low power. The results of the exact solutions show that in general a minimum literal PLA also provides a good low power solution. Based on the results of our exact solutions we do not expect

ex	1	2	3	4	5	6
Z5xp1	63	419	175.74	1.03	0.90	0.82
amd	66	738	348.87	1.03	0.98	0.96
b10	100	1220	485.24	1.02	0.98	0.95
dekoder	9	57	28.43	1.11	0.91	0.82
dk48	21	301	56.78	1.14	0.40	0.59
f51m	76	447	164.17	1.00	0.98	0.96
gary	107	1226	431.98	1.02	0.99	0.96
in0	107	1226	446.42	1.02	0.99	0.96
in2	134	1532	472.58	1.02	0.98	0.96
inc	29	241	116.50	1.07	0.93	0.89
log8mod	38	243	86.23	1.08	1.00	0.95
m181	41	288	129.65	1.05	0.86	0.81
newcpla1	38	350	168.19	1.05	0.97	0.95
opa	77	1357	864.90	1.06	0.97	0.94
root	57	412	142.84	1.00	0.95	0.91
sqr6	47	331	155.79	1.09	0.91	0.83
wim	9	60	37.08	1.11	0.92	0.90
AVG				1.05	0.92	0.89

Table 1.

heuristic solutions for low power to provide significant improvements in power over the current minimal area solutions obtained using programs such as ESPRESSO.

6 Conclusion

This paper we presented a solution to the problem of minimizing power consumption in PLAs. In doing so, we showed that for dynamic PLA circuits, as with a minimum area solution, a minimum power solution also consists only of prime implicants of the function. Therefore approaches used for minimizing area of PLAs may be used to also minimize the PLA power consumption. The same results was also proved for pseudo NMOS implementations of incompletely specified single output functions and completely specified multiple output functions. A heuristic approach was also presented for pseudo-NMOS implementations of incompletely specified multiple output functions. The results in the paper also show that during function minimization for dynamic CMOS PLA, it is still possible to make area-power trade-offs by using implicants with fewer literals and also literals with input statistics that result in a lower power consumption. This reduction in power however, has been at the expense of increasing the number of implicants in the final cover of the function which will in turn results in an increase in the number of rows of the PLA and therefore an increase in area.

7 References

[1] R.K. Brayton, G.D. Hachtel, C. McMullen and A. Sangiovanni-Vincentelli. "Logic Minimization Algorithms for VLSI Synthesis," Kluwer Academic Publishers, Boston, 1984.

[2] E. Sentovich, K. Singh, C. Moon, H. Savoj, R. Brayton and A. Sangiovanni-Vintentelli. "Sequential circuit design using synthesis and optimization," In proceedings of the International Conference on COmputer Design, pp328-333, October 1992.

[3] A. Chandrakasan, M. Potkonjak, J. Rabaey, and R. W. Brodersen.``HYPER-LP: A system for power minimization using architectural transformations," In Proc. of the IEEE Int'l. Conf. on Computer Aided Design, pages 300--303, November 1992.

[4] A. P. Chandrakasan, S. S. Scheng, and R. W. Broderson. "Low power CMOS digital design," IEEE Journal of Solid State Circuits, 27(4):473–483, April 1992.

[5] S. Devadas, K. Keutzer and J. White. "Estimation of Power Dissipation in CMOS Combinational Circuits using Boolean Manipulations," IEEE Transactions on CAD, vol 11, no 3, March 1992.

[6] S. Iman and M. Pedram. ``Two-Level Logic Minimization for Low Power," To appear In Proc. of the IEEE Int'l. Conf. on Computer Aided Design, November 1995.

[7] S. Iman and M. Pedram. ``Multi-level network optimization for low power," In Proc. of the IEEE Int'l. Conf. on Computer Aided Design, pages~372-377, November 1994.

[8] D. Liu and C. Svensson. "Trading speed for low power by choice of supply and threshold voltages," IEEE Journal of Solid State Circuits, 28(1):10–17, January 1993.

[9] R. Rudell. "Logic Synthesis for VLSI Design," Ph.D. thesis, University of California, Berkeley, 1989.

[10] A. Shen and S. Devadas and A. Ghosh and K. Keutzer. ``On average power dissipation and random pattern testability of combinational logic circuits," In Proc. of the Int'l Conference on Computer-Aided Design, pages 402--407,November 1992.

[11] [30] C-Y. Tsui, M. Pedram and A. M. Despain. ``Power efficient technology decomposition and mapping under an extended power consumption model," In IEEE Trans.~on Computer-Aided Design, Vol.~13, No.~9 (1994), pages 1110--1122.

[12] S.B.K. Vrudhula, H.Y. Xie. "Techniques for CMOS Power Estimation and Logic Synthesis for Low Power," Proceedings of the International Workshop on Low Power Design, pp. 21-26, 1994.