

# Gate-Level Synthesis for Low-Power Using New Transformations \*

Dhiraj K. Pradhan    Mitrajit Chatterjee    Madhu V. Swarna    Wolfgang Kunz  
Laboratory for Computers and Digital Systems Research  
Dept. of Computer Science  
Texas A&M University, College Station, TX - 77843

## Abstract

A new logic optimization method of multi-level combinational CMOS circuits is presented, which minimizes both power as well as power dissipation per unit area. The method described here uses Boolean transformations which exploit implications at the gate-level, based on both controllability and observability relationships. New transformations which form the basis of our synthesis method are presented. The emphasis is on power consumption rather than on area. Experimental results demonstrate that circuits synthesized by our method consume less power with a comparable area than those synthesized by state-of-the-art tools.

## 1 Introduction

Increased emphasis on portability and mobility has placed new constraints on VLSI design. Power consumption is one important example. Power consumption can be reduced at various stages of the design process; power optimization at the logic design stage has considerably less investment cost since it does not affect the IC fabrication process.

A logic synthesis tool is presented here which operates on unmapped, multi-level combinational circuits. The optimization criterion incorporates power consumption considerations. We begin with an unoptimized circuit as it offers more degrees of freedom during the optimization process; hence it can lead to a minimal power cost. The netlist transformations presented here are tailored for power and area of the circuit; area is sacrificed for power if the transformations reduce only one. Present tools which optimize combinational circuits for low power consumption concern themselves with circuit representations like Boolean networks, factored forms, etc. [6, 8]. The procedure described here possesses the added advantage of working with a level which is much closer to the physical design of the circuits than the previous approaches. Also, a power estimation technique which is applicable on the gate-level design is used. Such a cost function, combined with the estimate of the area, is used during the synthesis procedure. Boolean transformations mentioned in [2, 3] are modified to favor power-cost reduction. New transformations which reduce power consumption are proposed, as well.

\*Research reported is supported partly by the NSF grant MIP 9406946. W. Kunz has a joint appointment with Texas A&M University and Max-Planck-Society, Fault Tolerant Computing Group, Potsdam, Germany.

## 2 Prior Work

The focus here is on technology-independent power optimization for multi-level circuits. All earlier methods [7, 9, 8] work with a *Boolean Network* representation of the circuit. All such synthesis methods contain two essential components - power estimation, and optimization guided by the cost function.

Most of the previous approaches to power optimization during synthesis were based on functional methods. The reader can refer to [7, 9, 8] for details of the optimization methods. Logic factorization for low-power proposed in [8] is guided by the 'power cost' of kernels extracted during optimization, rather than using area cost of the kernels (as done in SIS [6]). Optimization using don't-cares [7, 9] is performed by examining the power cost of compatible don't-cares in a Boolean network.

## 3 Power Estimation Model

In a digital CMOS circuit, dynamic power consumption is the dominant source of power consumption. The amount of energy dissipated in a CMOS circuit is the energy required to charge or discharge the load capacitances at various nodes in the Boolean network [9]. For optimization, the following power-cost can be used:

$$P = \sum_{\forall \text{ nodes } i} p_i(1 - p_i) \times C_i \quad (1)$$

where  $p_i$  = signal probability of node  $i$   
 $C_i$  = load capacitance of node  $i$

The product  $p_i(1 - p_i)$  is called the *switching activity*  $E_i$  of node  $i$ . The sum,  $P$ , is the total *switched capacitance* of the network. The power is directly proportional to the value of the total switched capacitance in the circuit. The switched capacitance will be used as the cost function during circuit optimization. (It should be noted that both the power consumed due to short circuit power dissipation, as well as power dissipation due to glitching, are not incorporated into this estimation). Given a multi-level gate-description of a circuit, the proposed power estimation method first estimates the signal probabilities at each line in the circuit. Signal probabilities are used to calculate the switching activity at each line. The *switched capacitance* in the circuit is calculated, based on Equation 1. The signal probabilities at each line are determined by random-pattern simulation. Random-pattern simulation can also incorporate pre-defined signal probabilities at the primary inputs of the circuit.

*Capacitance Calculation:* The load capacitance calculation is related to the implementation of the gate as a standard CMOS cell. Load capacitance calculation

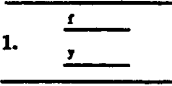
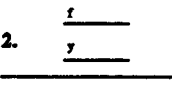
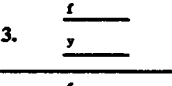
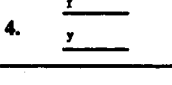
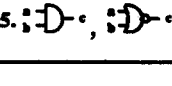
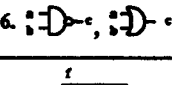
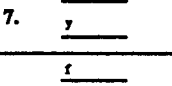
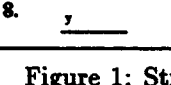
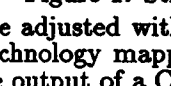
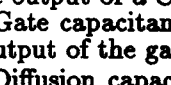
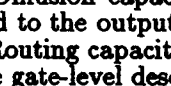
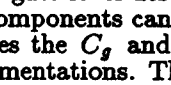
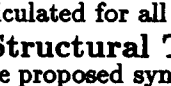
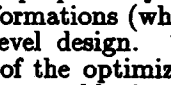
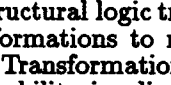
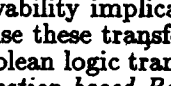
INITIAL STRUCTURE	CONDITION	TRANSFORMATION
1. 	$y=0 \Rightarrow f=1$	
2. 	$y=0 \Rightarrow f=0$	
3. 	$y=1 \Rightarrow f=1$	
4. 	$y=1 \Rightarrow f=0$	
5. 	$c=0 \Rightarrow c \oplus y=1$	
6. 	$c=1 \Rightarrow c \oplus y=1$	
7. 	$y=0 \Rightarrow f=0$ $y=1 \Rightarrow f=0$	
8. 	$y=0 \Rightarrow f=1$ $y=1 \Rightarrow f=1$	

Figure 1: Structural Transformations.

can be adjusted with any other implementation used for technology mapping. The total load capacitance at the output of a CMOS gate [10] is the sum of:

1. Gate capacitance (of other inputs connected to the output of the gate) -  $C_g$ .
2. Diffusion capacitance (of the drain regions connected to the output) -  $C_{ds}$ .
3. Routing capacitance.

In the gate-level description of a circuit, only the first two components can be estimated. Our procedure estimates the  $C_g$  and  $C_{ds}$ , assuming standard CMOS implementations. The values of load capacitance can be calculated for all other implementations, as well.

#### 4 Structural Transformations

The proposed synthesis is based on structural logic transformations (which preserve functionality) to the gate-level design. This Section defines the framework of the optimization process, reviews the existing structural logic transformations, and proposes new transformations to reduce dynamic power consumption. Transformations use certain controllability and observability implications among lines in the circuit. Because these transformations can cover a wide range of Boolean logic transformations, they are denoted as *Implication-based Boolean* (IB) transformations. The implications used here have been described in [2, 3].

##### 4.1 IB Transformations - Overview

IB transformations, and their derivatives, have been applied to gate-level circuits, either for minimizing area of technology-independent gate designs [5, 2], minimizing area after technology mapping, or even to enhance testability [3].

The transformations use *orthogonal expansion* to make gate-level connections, based on the above implications. Transformations applied using *controllability implications* exploit the controllability don't cares in

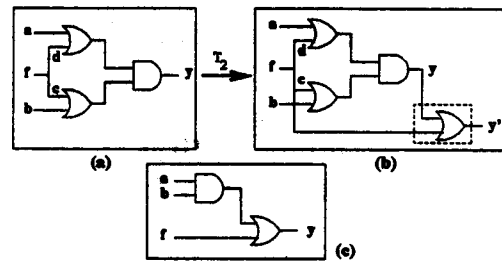


Figure 2: An example using Transformation 2. the network, whereas those with *observability implications* exploit the observability don't cares [6]. The methodology can be viewed as a three-step process:

1. Identifying candidates by making implications.
2. Transformation: using one candidate.
3. Reduction: Redundancy elimination.

Different types of transformations at the second step can result in different kinds of logic operations. Eight types of such transformations, *Transformations 1 - 8*, have been presented in Figure 1 [2, 4, 3]. These transformations alter the structure of the circuit without changing the functionality of the circuit. The first four transformations introduce AND, OR and NOT gates in the circuit, whereas the next four transformations introduce EX-OR or EX-NOR gates in the circuit. Each transformation is followed by *redundancy elimination*, which may result in a circuit of reduced cost. In the context of power optimization, these transformations can be viewed as a three-step process, which includes selection of candidates, addition of some active area in one part of the circuit (transformation), and, removal of some active area in another part of the circuit (redundancy removal).

*Example 4.1:* Consider the circuit shown in Fig. 2(a). It can be observed that  $y = 0 \Rightarrow f = 0$ .  $f$  is referred to as a *candidate* for making a transformation involving node  $y$ . Many such *candidates* might exist in practical situations. Transformation  $T_2$  is now applied to the circuit. The resultant circuit is shown in Fig. 2(b). On processing the resultant circuit, lines 'c' and 'd' can be found to be redundant s-a-1 lines, and hence, can be eliminated. The final circuit after redundancy elimination is shown in Fig. 2(c). □

##### 4.2 New Transformations Reducing Power

This Section includes proposed transformations which tend to reduce the switched capacitance in the circuit. The transformations may, sometimes, increase the area of the circuit.

###### A. Power Transformations

In *power transformations*, the steps involved are geared towards reducing the switching capacitance of the lines in the circuit. The three steps involved in *Power Transformations* are:

1. Identify implications and order the candidates obtained, according to the cost criterion (*Sorting*).
2. Making connections, as in Transformations 1-8.
3. Eliminate redundancy selectively, using certain heuristics (*Selective Redundancy Elimination*).

*Sorting:* Several promising candidates can be identified for making transformations. For example, in Fig. 2(a), the candidate is  $f$ . It is used to make the new connection shown in Fig. 2(b). In a scenario where a

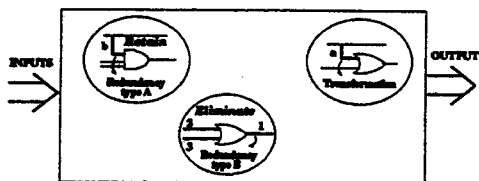


Figure 3: Selective redundancy elimination

candidate used to make a connection becomes a permanent one in the circuit, it is necessary that a proper candidate (which favors minimum activity) be chosen. The candidates can be sub-divided based on two types of implications, *controllability implications* and *observability implications*. In the former case, the candidate with minimum switching activity is selected. In the latter case, where the implication is observability-based, the switching activity of the new line formed changes. How the candidate influences the switching capacitance of the new line depends on what type of gate is being introduced (refer to Fig. 1). The choice of candidate, therefore, is made by taking into account the switching activity of the new line being formed and that of the candidate.

**Selective Redundancy Elimination:** During redundancy removal, two types of redundancies are identified. The first type is *redundancy-type A*, whose elimination results in removal of the whole gate and its fanins until a fanout stem is reached (as in line 1 in Fig. 3). Removal of these lines will reduce the switching capacitance in the circuit as it leads to removal of gates. Also, lines with signal probability, in the vicinity of 0.5, are good candidates for removal. The other redundancy type is *redundancy-type B*, and its elimination results in removal of just the one fanout stem (as in line *b* in Fig. 3). Removal of these lines might not reduce the power consumption of the circuit. A line which is a fanout line from a gate and has a low switching activity can be retained, as it does not consume much power. This line may, on the contrary, aid in suppressing the activity of the gate at its output. *Selective redundancy elimination* removes only *redundancy-type A* lines and retains *redundancy-type B* lines. After additional transformations, the *redundancy-type B* lines may not remain redundant in the circuit. However, all redundancies can be removed at the end of the synthesis process.

#### B. Transformations Based on Trend

Transition probability in the circuit can be minimized by having more lines whose signal probabilities are lop-sided (close to 0 or 1). Successive transformations might change the signal probabilities of a line in an increasing, as well as decreasing, way. In a low-power synthesis procedure, *transformations based on trend*, should move the signal probability of the lines in a fixed direction. Here, a transformation is accepted only when most of the lines, when weighted by their load capacitances, change their signal probabilities in the desired direction. If the signal probabilities of  $m$  lines change in favorable directions and  $n$  of them change in the opposite direction, the cost calculated is given by the difference:

$$\text{Trend\_cost} = \sum_{v_m} (\Delta p)_i \times \text{fanout} - \sum_{v_n} (\Delta p)_i \times \text{fanout}$$

where  $(\Delta p)_i$  is the change in signal prob. at line  $i$ .

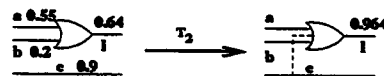


Figure 4: Dampening activity of lines

#### C. Dampening Switching Activity in Gates

The signal probabilities of certain lines in the circuit can be changed by adding appropriate lines to their fanins. This requires that the lines added do not alter the functionality of the circuit, and that the lines have skewed signal probability after the transformation. It has been observed that transformations based on *observability implications* can pull in lines at a gate that can change its switching activity, and yet maintain functionality. *Controllability implications* would not be able to change the signal probability of the gate to which the line is pulled in. Though redundancy elimination following the transformation may change the signal probability of the gate, *observability implications* make an immediate impact on the switching activity. Transformations of this type can be used to target specific lines with high switching activity, and can be very effective. For example, in Fig. 4, the signal probability of an OR gate may be changed from 0.64 to 0.964, thus dampening its switching activity.

#### D. Reducing EX-OR Gates

EX-OR gates are prone to high switching activity because EX-OR gates are not biased towards a particular value, and the output of an EX-OR gate is sensitive to all its inputs. Thus, transformations which reduce the number of EX-OR gates in a circuit will reduce the power consumption of a circuit. Don't-care conditions at the inputs to the EX-OR gates and at the outputs of the EX-OR gates can be used to change them into other gate-types [3, 4, 6].

### 5 Synthesis for Low Power

This Section describes a flow of the optimization process, with the primary objective of reducing power consumption. All transformations mentioned in previous Sections are used, guided by the power-cost (in terms of the switched capacitance in the circuit). The two main features include a synthesis guidance procedure, based on the cost value, and a method for efficient estimation of the switched capacitance in the circuit. The switched capacitance is measured, based on the proposed model. However, the power consumed by the circuit has to be estimated at every iteration, to provide a proper guidance to the synthesis process. A detailed description of the incremental signal probability calculation can be found in [11].

It has been observed that synthesis for area (or other goals) does not necessarily result in a circuit with lower power. As an example, in Fig. 5, we show two small circuits, *A* and *B*, with almost the same area. The two circuits can be obtained from one and other, by means of the implications shown. Assuming the inputs have a signal probability of 0.5, the power estimated in circuit *A* is less than in circuit *B*. This example shows that there can be two (or more) instances of the same circuit with the same area and different power consumptions. A synthesis method geared towards a lesser area may transform circuit *A* to *B*, as

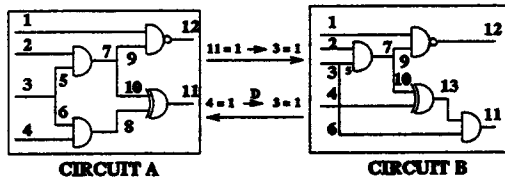


Figure 5: Transformation changing power-cost.

```

Optimization For Low Power (Iter,circuit)
1. For i = 1 to Iter do
2.   rand_simulate(circuit) /* Estimate sig. prob. */;
3.   power_estimate(circuit) /* Proposed model */;
4.   if (i is even) then
5.     For all lines in the circuit do
6.       Make Transformations 1 - 8;
7.       selective_red_eliminate(circuit);
8.       Remove unnecessary EX-OR gates;
9.     else
10.    For all lines in the circuit do
11.      Make Transformations for Trend;
12.    For lines with high switching activity do
13.      Dampen signal probabilities;
14.  Redundancy elimination; /* Final step */

```

Figure 6: Algorithm for Power Optimization in HANNIBAL [2]. However, during power optimization, circuit A is a better choice; a synthesis procedure encounters a large number of such situations, where a correct decision leads to a better circuit.

There can be a number of such algorithms based on the proposed transformations. One such procedure is outlined in Fig. 6. The heuristics and other details of the algorithm are treated in [11].

## 6 Experimental Results

The algorithm outlined in the previous Section was implemented, to demonstrate the effectiveness of the transformations to minimize power and area. The circuits were power-optimized, based on an input signal probability distribution. Results, comparing the performance of the tool with SIS-1.2 [6] on the ISCAS85 circuits, have been presented here. Further details and comparisons with HANNIBAL [2] and POSE [8] can be found in [11].

**Comparison with SIS-1.2:** The area and the power of the resultant circuits are compared with those by SIS-1.2 in Table 2. For SIS-1.2, the scripts considered were *script.rugged* and *script.algebraic*. Both of these scripts optimize the area of the circuit, using optimization techniques based on the Boolean network. The proposed method optimizes both the area and the power of the circuit at the gate-level. The resultant circuits were compared, based on the power estimated and the area. The power reported here (Pwr) has been measured by the proposed 'zero-delay' model. The area estimated (Ar) is based on the equivalent number of 2-input gates. For *script.rugged*, the average reduction in area and power is 10% and 20%, respectively, whereas for the *script.algebraic*, the area and power reduction is 16% and 22%, respectively. Also, note that the reduction in power is significantly greater than the reduction in area.

Circuit Name	Proposed		Prop/SIS_R		Prop/SIS_A	
	Ar	Pwr	Ar	Pwr	Ar	Pwr
c432	103	683.8	0.85	0.77	0.65	0.68
c499	333	1997.5	0.86	0.62	0.86	0.62
c880	285	1891.6	0.95	0.85	0.89	0.80
c1355	348	2219.9	0.91	0.71	0.76	0.71
c1908	323	2083.7	0.92	0.80	0.89	0.83
c2670	508	3538.0	0.95	0.84	0.88	0.82
c3540	836	4572.9	0.94	0.92	0.92	0.82
c5315	1344	10205.8	1.05	0.88	0.97	0.84
c6288	1873	21186.8	0.86	0.94	0.87	1.11
c7552	1281	10089.0	0.75	0.65	0.71	0.60

Table 1: Comparison with SIS 1.2.

## 7 Conclusions

A technology-independent multi-level synthesis method to optimize area and power has been presented. Based on the gate-level representation of the circuit, this method uses structural transformations. Several new structural transformations have been proposed to reduce power, and a new synthesis methodology is provided to guide the circuit towards less power without considerable loss of area. Experimental results demonstrate that the proposed transformations are capable of synthesizing circuits with a lesser cost function, compared to state-of-the-art synthesis tools. Furthermore, these transformations can be extended to sequential circuits, and technology mapping.

## References

- [1] W. Kunz, D.K. Pradhan, "Recursive Learning: A New Implication Technique for Efficient Solution to CAD Problems - Test, Verification and Optimization," IEEE Trans. on CAD, Sept., 1994.
- [2] W. Kunz, P.R. Menon, "Multi Level Logic Optimization by Implication Analysis," ICCAD 1994.
- [3] M. Chatterjee, D. Pradhan and W. Kunz, "LOT: Logic optimization with testability - New Transformations using recursive learning," ICCAD 1995.
- [4] S.C. Chang *et al.*, "Perturb and Simplify: Multi-Level Boolean Network Optimizer," ICCAD, 1994.
- [5] S. Muroga *et al.*, "The Transduction Method - Design of Logic Networks Based on Permissible Functions," IEEE Trans Compt., Oct 1989.
- [6] G. De. Micheli, "Synthesis and Optimization of Digital Circuits," McGraw-Hill, Inc. 1994.
- [7] S. Iman, M. Pedram, "Multi-level network optimization for low power," ICCAD, 1994.
- [8] S. Iman, M. Pedram, "POSE: Power Optimization and Synthesis Environment," DAC, 1996.
- [9] A. Shen, *et al.*, "On average power dissipation and random pattern testability of CMOS combinational logic networks," ICCAD, 1992.
- [10] N. Weste, K. Eshraghian, "Principles of CMOS VLSI design - A systems perspective," Addison Wesley Publishing Co.
- [11] D.K. Pradhan, *et al.*, "Implication-Based Gate-level Synthesis for Low-Power," Tech. Rep. 95-044, Dept. of Computer Sc., Texas A&M University.
- [12] D.K. Pradhan, W.Kunz, "Method of Circuit Verification and Multi-Level Circuit Optimization Based on Structural Implications," US Patent # 5,526,514, June '96.